Tijani Chahed   Bruno Tuffin (Eds.)

Network Control and Optimization

First EuroFGI International Conference, NET-COOP 2007
Avignon, France, June 5-7, 2007
Proceedings

Springer

Volume Editors

Tijani Chahed
GET/Institut National des Telecommunications
91011 Evry Cedex, France
E-mail: tijani.chahed@int-evry.fr

Bruno Tuffin
IRISA-INRIA, Campus universitaire de Beaulieu
35042 Rennes Cedex, France
E-mail: btuffin@irisa.fr

Library of Congress Control Number: 2007927054

CR Subject Classification (1998): C.2.4, C.2, F.2, D.1.3, D.2, D.4.8, H.4

LNCS Sublibrary: SL 5 – Computer Communication Networks and Telecommunications

ISSN 0302-9743
ISBN-10 3-540-72708-6 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-72708-8 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12068746 06/3180 5 4 3 2 1 0

Preface

This volume 4465 of the *Lecture Notes in Computer Science series* is a collection of the papers of the NET-COOP 2007 conference, a first-of-a-series Euro-NGI/FGI Conference on Network Control and Optimization. The event took place in the beautiful city of Avignon, France, June 5–7, 2007, was jointly organized by INRIA and the University of Avignon and was hosted by the latter.

Internet communications and services are experiencing an increase in volume and diversity both in their capacity and in their demand. This comes at the cost of an increase in the complexity of their control and optimization, mainly due to the heterogeneity in architecture as well as usage. The need for new ways of effectively and fairly allocating resources belonging to a wide set of not necessarily cooperative networks to a collection of possibly competing users is urgent and is the aim of this conference.

Specifically, this conference aims at developing research on control and optimization of the Internet, ranging from performance evaluation and optimization of general stochastic networks to more specific targets such as lower-layer functionalities in mobile networks, routing for computational grids, game theoretic approaches to access control, cooperation, competition and adversary capacities in diverse environments.

As stated earlier, the event is the first of a series initiated by Euro-NGI/FGI Network of Excellence, a European consortium composed of 55 universities, research laboratories and industrial partners, whose goal is to create, lead and maintain the most prominent European center of excellence on future generation Internet design and engineering. In this context, NET-COOP, initiated and led by Eitan Altman from the eponymous working group, aims to bring together researchers, both from Euro-NGI/FGI and outside, working in the areas of network optimization and control.

During the conference, three keynote talks were given: by Thomas Bonald from France Telecom R&D ("Resource Allocation in Data Networks"), Moshe Haviv from the Hebrew University of Jerusalem (title: "To queue or not to queue: The cases of partially and of fully observable M/G/1 queues"), and Ravi Mazumdar from the University of Waterloo (title: "Distributed congestion control in networks: Solution concepts, distributed algorithms, and some recent results"). There were also 31 paper presentations, 22 issued from regularly submitted papers (out of 46) and 9 invited ones, both groups having undergone a review process. The final program shows a real international scope, with authors from Europe, Asia and the Americas.

The success of NET-COOP 2007 was largely due to the General Chair, Jorma Virtamo, and the Technical Program Committee, whose members devoted much of their time and effort to provide a highly qualified technical program. To them we express our many thanks and deepest gratitude.

We would also like to very sincerely thank our technical co-sponsors: IEEE Control Systems Society (CSS) and IFIP TC6, as well as our financial co-sponsors, Euro-NGI/FGI, INRIA, GET - Groupe des Ecoles des Télécommunications, Région Provence-Alpes-Côte d'Azur through their Conseil Régional and France Telecom. We thank them very much for their trust and support.

Many warm thanks also go to the Springer LNCS team particularly Alfred Hofmann, the Editorial Director, and Ursula Barth, for their confidence, kindness and continuing help.

None of this would have been possible without the work and devotion of the organizers and hosts, Eitan Altman, Tania Jimenez, Rachid El-Azouzi, Yezekael Hayel, Ephie Deriche, Dany Sergeant and Monique Simonetti, from INRIA and the University of Avignon. We tried to thank them all the way through but if we did not, let us do it here: Thank you!

We would like to extend our thanks to all authors who submitted very high quality papers to our conference, many of which were not accepted solely because of room, congratulate once again those whose papers were accepted, thank all presenters and attendees for their effort and time and wish all the prospective readers to take as much pleasure in reading this manuscript as we took in editing it.

June 2007

Tijani Chahed
Bruno Tuffin

Organization

NET-COOP 2007 was organized jointly by INRIA Sophia Antipolis and the University of Avignon, who also hosted the event.

Executive Committee

General Chair	Jorma Virtamo (HUT, Finland)
Program Committee Co-chairs	Tijani Chahed (GET/INT, France)
	Bruno Tuffin (IRISA/INRIA Rennes, France)
Organizing Chair	Tania Jimenez (University of Avignon, France)
Responsible for invited talks	Eitan Altman (INRIA Sophia Antipolis, France)
	Yezekael Hayel (University of Avignon, France)

Steering Committee

Eitan Altman	(INRIA, France)
Konstantin Avrachenkov	(INRIA, France)
Onno Boxma	(UTE, The Netherlands)
Tijani Chahed	(GET/INT, France)
Klaus Hackbarth	(UC, Spain)
Mikael Johansson	(KTH, Sweden)
Daniel Kofman	(GET/ENST, France)
Isi Mitrani	(Newcastle University, UK)
Pascale Primet	(INRIA, France)
Alexandre Proutière	(FT/ENS, France)
Bruno Tuffin	(IRISA/INRIA, France)
Jorma Virtamo	(HUT, Finland)

Organizing Committee

Eitan Altman	(INRIA Sophia Antipolis, France)
Ephie Deriche	(INRIA Sophia Antipolis, France)
Rachid El-Azouzi	(University of Avignon, France)
Yezekael Hayel	(University of Avignon, France)
Tania Jimenez	(University of Avignon, France)
Dany Sergeant	(INRIA Sophia Antipolis, France)
Monique Simonetti	(INRIA Sophia Antipolis, France)

Program Committee

Ivo Adan	(Eindhoven University of Technology)
Konstantin Avratchenkov	(INRIA)
Tamer Basar	(University of Illinois)
Thomas Bonald	(France Telecom R&D)
Vivek Borkar	(TIFR)
Onno Boxma	(Eindhoven University of Technology)
Antonio Capone	(Politecnico di Milano)
Rajarathnam Chandramouli	(Stevens Institute of Technology)
Costas Courcoubetis	(Athens University of Economics and Business)
Serguei Foss	(Heriot-Watt University)
Nidhi Hegde	(France Telecom R&D)
Brigitte Jaumard	(Concordia University)
Hisao Kameda	(University of Tsukuba)
Peter Key	(Microsoft Research)
Arzad Kherani	(Indian Institute of Technology)
Anurag Kumar	(Indian Institute of Science)
Harold J. Kushner	(Brown University)
Steven Low	(California Institute of Technology)
John Lui	(The Chinese University of Hong Kong)
Patrick Maillé	(GET/ENST-Bretagne)
Michel Mandjes	(CWI)
Peter Marbach	(University of Toronto)
Richard Marquez	(University of Los Andes Venezuela)
Ravi Mazumdar	(University Waterloo)
Boris Miller	(Russian Academy of Sciences)
Isi Mitrani	(University of Newcastle)
Jose Nino-Mora	(Universidad Carlos III de Madrid)
Fernando Paganini	(University of ORT)
Balakishna Prabhu	(CWI)
Pascale Primet	(ENS-Lyon)
Guy Pujolle	(University of Paris 6)
Rudesindo Núñez Queija	(CWI)
Zari Rachev	(University of California at Santa Barbara)
Peter Reichl	(FTW)
Jacques Resing	(Eindhoven University of Technology)
Ulrich Rieder	(University of Ulm)
David Ros	(GET/ENST-Bretagne)
Sanjay Shakkottai	(University of Texas at Austin)
George Stamoulis	(Athens University of Economics and Business)
Nicolas Stier	(Columbia University)
Darryl Veitch	(University of Melbourne)
Jorma Virtamo	(Helsinki University of Technology)

Referees

Ivo Adan
Eitan Altman
Konstantin Avratchenkov
Tamer Basar
Walid Ben Ameur
Marina Bistaki
Thomas Bonald
Vivek Borkar
Onno Boxma
Antonio Capone
Tijani Chahed
Rajarathnam Chandramouli
Costas Courcoubetis
Manos Dramitinos
Rachid El-Azouzi
Serguei Foss
Pranava Goundan
Yezekael Hayel
Nidhi Hegde
Brigitte Jaumard
Hisao Kameda
Peter Key
Arzad Kherani
Anurag Kumar
Harold J. Kushner
Steven Low
John Lui
Patrick Maillé
Enrico Malaguti
Michel Mandjes
Peter Marbach
Richard Marquez
Ravi Mazumdar
Boris Miller
Isi Mitrani
Jose Nino-Mora
Fernando Paganini
Thanasis Papaioannou
Balakishna Prabhu
Pascale Primet
Guy Pujolle
Rudesindo Núñez Queija
Zari Rachev
Peter Reichl
Jacques Resing
Ulrich Rieder
David Ros
Stauros Routzounis
Sanjay Shakkottai
Sergios Soursos
George Stamoulis
Nicolas Stier
Corinne Touati
Bruno Tuffin
Darryl Veitch
Jorma Virtamo

Technical Sponsoring Institutions

IEEE Control Systems Society (CSS)
IFIP TC6

Financial Sponsoring Institutions

Euro-NGI/FGI NoE
INRIA
GET - Groupe des Ecoles des Télécommunications
Conseil Régional Provence-Alpes-Côte d'Azur
France Telecom

euro ngi
NETWORK OF EXCELLENCE

INRIA
SOPHIA ANTIPOLIS

UNIVERSITÉ D'AVIGNON

get

INT

IEEE CSS

ifip tc6

Région
Provence-Alpes-Côte d'Azur

france telecom

Table of Contents

A Jamming Game in Wireless Networks with Transmission Cost*

E. Altman[1], K. Avrachenkov[1], and A. Garnaev[2]

[1] INRIA Sophia Antipolis, France
{altman,k.avrachenkov}@sophia.inria.fr
[2] St. Petersburg State University, Russia
agarnaev@rambler.ru

Abstract. We consider jamming in wireless networks with transmission cost for both transmitter and jammer. We use the framework of non-zero-sum games. In particular, we prove the existence and uniqueness of Nash equilibrium. It turns out that it is possible to provide analytical expressions for the equilibrium strategies. These expressions is a generalization of the standard water-filling. In fact, since we take into account the cost of transmission, we obtain even a generalization of the water-filling in the case of one player game. The present framework allows us to study both water-filling in time and water-filling in frequency. By means of numerical examples we study an important particular case of jamming of the OFDM system when the jammer is situated close to the base station.

Keywords: Wireless networks, Jamming, Non-zero-sum games, Nash Equilibrium, Water-filling.

1 Introduction and Problem Formulation

Power control in wireless networks became an important research area. Since the technology in the current state cannot provide batteries which have small weight and large energy capacity, the design of algorithms for efficient power control is crucial. For a comprehensive survey of recent results on power control in wireless networks an interested reader can consult [15]. It turns out that game theory provides a convenient framework for approaching the power control problem see for instance [9] and references therein. Most of the work on application of game theory to power control considers mobile terminals as players of the same type. Here we consider the jamming problem with two types of players. The first type of players are regular users of the wireless mobile network who want to use the available wireless channels in the most efficient way. The second type of players are jammers who want to prevent or to jam the communication of the regular users. The study of jamming in wireless networks is important in the context of military actions or fighting against terrorist activity. On a battlefield, it is very

* The work was partially supported by the European Research Grant BIONETS and by RFBR and NNSF Grant no.06-01-39005.

likely that one side will try to prevent the wireless communication of the other side. Thus, one side is interested in the best usage of power to overcome the artificial noise emitted by the other side. And conversely, the other side tries to use power to harm the communication in the most efficient way.

In [2] the authors have studied the application of dynamic stochastic zero sum game to the jamming problem in wireless networks. In the model of [2] the transmission power can be chosen from a discrete set. Here we suppose that the power level can be chosen from a continuous set. This allows us not only to prove the existence of the Nash equilibrium (NE) but also to show its uniqueness. Here, in addition to the power constraint we introduce the cost of power usage. This makes the problem a non-zero game. Furthermore, the current continuous model allows us to study not only temporal power distribution for one channel but also the distribution of power among different sub-channels.

In the works [7] and [14] the authors have analyzed the worst case wireless channel capacity when the noise variances are fixed (possibly unknown at the transmitter) and the carrier gains are allowed to vary while verifying a certain constraint. In that case, transmission at the worst rate guarantees error free communication under any possible conditions of the channel, although it might give a pessimistic result. This formulation leads to a minimax problem. In the works [7] and [14] as well as in [2] the cost of transmission is not taken into account. Other problem formulations involving jamming in which one wireless terminal wishes to maximize the mutual information and the other tries to minimize it, can be found at [3]. For other related work, see [5].

Let us specify the present model formulation. We consider two mobile terminals and one base station. Since we use the framework of game theory, we shall use the terms mobiles and players interchangably. Player 1 seeks to transmit information to the base station. We shall refer to it as "Transmitter". Player 2 has an antagonistic objective: to prevent or to jam the transmissions of Player 1 to the base station. Thus, we shall call Player 2 "Jammer". Both players have in addition a transmission cost (see below) which prevents us from using zero-sum games to model our problem.

We assume that there are n independent resources, each of which can be used simultaneously by both players. We further assume that resource i has a "weight" of π_i.

Possible interpretations

(i) The resources may correspond to capacity available at different time slots; we assume that there is a varying environment whose state changes among a finite set of states $i \in [1, n]$, according to some ergodic stochastic process with stationary distribution $\{\pi_i\}_{i=1}^n$. We assume that both players have perfect knowledge of the environment state at the beginning of each time slot.

(ii) The resources may correspond to frequency bands (e.g. as in OFDM) where one should assign different power levels for different sub-carriers [15]. In that case we may take $\pi_i = 1/n$ for all i.

The pure strategy of Transmitter is $T = (T_1, \dots, T_n)$ where $T_i \geq 0$ for $i \in [1, n]$ and $\sum_{i=1}^n \pi_i T_i \leq \bar{T}$ where $\bar{T} > 0$, $\pi_i > 0$ for $i \in [1, n]$. The component T_i can be interpreted as the power level dedicated to resource of type i. If the resource i is the available capacity when the environment state is i, then T_i is the power level that is chosen whenever we visit state i, and $\bar{T}$ is a bound on the power averaged over time.

If the resources correspond to frequency bands, then T_i is the average power to be transmitted at the ith band. $\bar{T}$ is then the maximal average power level that can be used by Transmitter.

The pure strategy of Jammer is $N = (N_1, \dots, N_n)$ where $N_i \geq 0$ for $i \in [1, n]$ and $\sum_{i=1}^n \pi_i N_i \leq \bar{N}$ where $\bar{N} > 0$. The payoffs to Transmitter and Jammer are given as follows

$$\begin{aligned} v_T(T, N) &= \sum_{i=1}^n \pi_i \ln\left(1 + \frac{g_i T_i}{h_i N_i + N_i^0}\right) - c_T \sum_{i=1}^n \pi_i T_i, \\ v_N(T, N) &= -\sum_{i=1}^n \pi_i \ln\left(1 + \frac{g_i T_i}{h_i N_i + N_i^0}\right) - c_N \sum_{i=1}^n \pi_i N_i \end{aligned} \tag{1}$$

where N_i^0 is the power level of the uncontrolled noise of the environement at state i, $c_T > 0$ and $c_N > 0$ are the costs of power usage for Transmiter and Jammer, and $g_i > 0$ and $h_i > 0$ are fading channel gains for Transmiter and Jammer when the environement is in state i. The first sum in payoff is the expected value of the Shanon capacity [6,10,15] and the second sum is the average cost of transmission.

We shall look for a NE, that is, we want to find $(T^*, N^*) \in A \times B$ such that

$$\begin{aligned} v_T(T, N^*) \leq v_T(T^*, N^*) \text{ for any } T \in A, \\ v_N(T^*, N) \leq v_N(T^*, N^*) \text{ for any } N \in B, \end{aligned}$$

where A and B are the sets of all the strategies of Transmitter and Jammer, respectively. In particular, we shall prove that the NE exists and is unique and we shall provide closed form analytic expressions for its calculation.

In the special case when c_T and c_N are zero in (1), the game is zero-sum. As v_T is convex in T_i and concave in N_i, we can apply Sion's minimax Theorem to conclude that it has a saddle point.

The structure of the paper is as follows: To complete the picture and to introduce notations, in Section 2 we consider single player water-filling game with the environment when the transmission cost is taken into account. Section 3 is the main part of the paper where we study the structure of the NE in the jamming game. Then, in Section 4, based on theoretical results of Section 3, we provide an algorithm for determination of the NE. We study some numerical examples in Section 5 and make conclusions in Section 6.

2 Water-Filling with Transmission Cost

In this section we consider the following single person game with the environment. There is one player named Transmitter. He/she wants to send information through a channel which state depends on the state of the environment or through n sub-channels. The goal of Transmitter is to maximize the sending rate of the transmitted information and to minimize the transmission cost. The pure strategy of Transmitter is $T = (T_1, \dots, T_n)$ where $T_i \geq 0$ for $i \in [1, n]$ and $\sum_{i=1}^{n} \pi_i T_i \leq \bar{T}$ where $\bar{T} > 0$ and $\pi_i > 0$ for $i \in [1, n]$. The payoff to Transmitter is given as follows

$$v(T) = \sum_{i=1}^{n} \pi_i \ln\left(1 + \frac{T_i}{N_i^0}\right) - c_T \sum_{i=1}^{n} \pi_i T_i,$$

where $N_i^0 > 0$ is the noise level when the environment is in state i, $i \in [1, n]$ and c_T is a cost for power usage. We would like to emphasize that this is a generalization of the standard water-filling scheme, see e.g., [8,13,15]. Following the standard water-filling approach we can get the following result.

Theorem 1. *Let* $1/N_1^0 = \max_{i \in [1,n]} 1/N_i^0$ *and* $T_i(\omega) = \left[1/(c_T + \omega) - N_i^0\right]_+$ *for* $i \in [1, n]$ *and* $H_T(\omega) = \sum_{i=1}^{n} \pi_i T_i(\omega)$.

If $c_T \geq 1/N_1^0$ *then* $T^* = (0, \dots, 0)$ *is the unique optimal strategy and its payoff is 0. If* $c_T < 1/N_1^0$ *then* $T(\omega^*) = (T_1(\omega^*), \dots, T_n(\omega^*))$ *is the unique optimal strategy and its payoff is* $v(T(\omega^*))$ *where for* $H_T(0) \leq \bar{T}$ $\omega^* = 0$ *and for* $H_T(0) > \bar{T}$ ω^* *is the unique root of the equation* $H_T(\omega) = \bar{T}$.

3 Jamming Game

In this section we consider a non-zero-sum game between Transmitter and Jammer with payoff functions defined by (1). We shall study the NE of this game, that is, we want to find $(T^*, N^*) \in A \times B$ such that

$$v_T(T, N^*) \leq v_T(T^*, N^*) \text{ for any } T \in A,$$
$$v_N(T^*, N) \leq v_N(T^*, N^*) \text{ for any } N \in B,$$

where A and B are the sets of all the strategies of Transmitter and Jammer, respectively.

Note that

$$\frac{\partial^2 v_T(T, N)}{\partial T_i^2} = -\frac{\pi_i g_i^2}{(g_i T_i + h_i N_i + N_i^0)^2 (h_i N_i + N_i^0)^2} < 0$$

and

$$\frac{\partial^2 v_N(T, N)}{\partial N_i^2} = -\frac{\pi_i T_i g_i h_i^2 (g_i T_i + 2h_i N_i + 2N_i^0)}{(g_i T_i + h_i N_i + N_i^0)^2 (h_i N_i + N_i^0)^2} < 0.$$

Thus, v_T and v_N are concave in T and N respectively. So, we can apply the Kuhn-Tucker Theorem to find the form that the NE has, namely, we will show in Theorem 3 that each NE is of the form $(T(\omega, \nu), N(\omega, \nu))$ for some nonnegative ω and ν where $T(\omega, \nu)$ and $N(\omega, \nu)$ are given in closed form in (5) and (6). These functions have a nice monotonous properties established in Lemma 1, namely, $N(\omega, \nu)$ is decreasing in ω and ν and $T(\omega, \nu)$ is decreasing in ω and is increasing in ν. This properties allow us to prove in Theorem 4 that there is at most one NE. Then, based on the monotonous properties of $T(\omega,\nu)$ and $N(\omega,\nu)$, we produce a NE in Theorems 5 and 6 in a way where the original two parametric problems in ω and ν reduces to one parametric problem either in ω or in ν where the optimal values of ω and ν can be found from solution of an equation with monotonous function. This in turn allows us in Section 4 to produce an effective algorithm based on the bisection method for numerical determination of NE.

Now we can pass on to our analysis. As it was noticed v_T and v_N are concave in T and N, thus, the Kuhn - Tucker Theorem implies the following theorem.

Theorem 2. *(T^*,N^*) is a NE if and only if there are non - negative ω and ν such that*

$$\frac{\partial}{\partial T_i} v_T(T^*,N^*) = \frac{g_i}{g_iT_i^* + h_iN_i^* + N_i^0} - c_T \begin{cases} = \omega & \textit{for } T_i^* > 0, \\ \le \omega & \textit{for } T_i^* = 0, \end{cases} \tag{2}$$

$$\begin{aligned} \frac{\partial}{\partial N_i} v_N(T^*,N^*) = & \frac{g_ih_iT_i^*}{(g_iT_i^* + h_iN_i^* + N_i^0)(h_iN_i^* + N_i^0)} \\ & - c_N \begin{cases} = \nu & \textit{for } N_i^* > 0, \\ \le \nu & \textit{for } N_i^* = 0, \end{cases} \end{aligned} \tag{3}$$

where

$$\omega \begin{cases} \ge 0 & \textit{for } \sum_{i=1}^n \pi_iT_i^* = \bar{T}, \\ = 0 & \textit{for } \sum_{i=1}^n \pi_iT_i^* < \bar{T} \end{cases} \quad \textit{and} \quad \nu \begin{cases} \ge 0 & \textit{for } \sum_{i=1}^n \pi_iN_i^* = \bar{N}, \\ = 0 & \textit{for } \sum_{i=1}^n \pi_iN_i^* < \bar{N}. \end{cases} \tag{4}$$

For non-negative ω and ν let

$$\begin{aligned} I_{00}(\omega,\nu) &= I_{00}(\omega) = \big\{i \in [1,n] : \ h_ig_i/N_i^0 \le h_i(\omega + c_T)\big\}, \\ I_{10}(\omega,\nu) &= \big\{i \in [1,n] : \ h_i(\omega + c_T) < h_ig_i/N_i^0 \le h_i(\omega + c_T) + g_i(\nu + c_N)\big\}, \\ I_{11}(\omega,\nu) &= \big\{i \in [1,n] : \ h_i(\omega + c_T) + g_i(\nu + c_N) < h_ig_i/N_i^0\big\}, \end{aligned}$$

$$T_i(\omega,\nu) = \begin{cases} \dfrac{g_i}{(\omega + c_T)h_i + (\nu + c_N)g_i} \times \dfrac{\nu + c_N}{\omega + c_T} & \text{for } i \in I_{11}(\omega,\nu), \\ \dfrac{1}{c_T + \omega} - \dfrac{N_i^0}{g_i} & \text{for } i \in I_{10}(\omega,\nu), \\ 0 & \text{for } i \in I_{00}(\omega,\nu), \end{cases} \tag{5}$$

$$N_i(\omega,\nu) = \begin{cases} \dfrac{g_i}{(\omega + c_T)h_i + (\nu + c_N)g_i} - \dfrac{N_i^0}{h_i} & \text{for } i \in I_{11}(\omega,\nu), \\ 0 & \text{for } i \in I_{00}(\omega,\nu). \end{cases} \tag{6}$$

Theorem 3. *Each NE is of the form $(T(\omega,\nu),N(\omega,\nu))$ for some nonnegative ω and ν.*

Now we go on to finding optimal ω and ν. Let

$$H_T(\omega,\nu) = \sum_{i=1}^n \pi_iT_i(\omega,\nu), \quad H_N(\omega,\nu) = \sum_{i=1}^n \pi_iN_i(\omega,\nu).$$

Then Theorem 3 implies that

$$H_T(\omega,\nu) = \sum_{i\in I_{10}} \pi_i \left(\frac{1}{c_T+\omega} - \frac{N_i^0}{g_i}\right) + \frac{\nu + c_N}{\omega + c_T} \sum_{i \in I_{11}} \frac{\pi_i g_i}{(\omega + c_T)h_i + (\nu + c_N)g_i},$$

$$H_N(\omega,\nu) = \sum_{i\in I_{11}} \pi_i \Big(\frac{g_i}{(\omega + c_T)h_i + (\nu + c_N)g_i} - \frac{N_i^0}{h_i}\Big).$$

In the next lemma some monotonous properties of $T_i(\omega,\nu)$ and $N_i(\omega,\nu)$, $H_T(\omega,\nu)$ and $H_N(\omega,\nu)$ are obtained.

Lemma 1. (i) *For fixed $\omega > 0$ and $0 \le \nu_1 < \nu_2$ we have: (1) $T_i(\omega,\nu_1) \le T_i(\omega,\nu_2)$ where strict inequality holds if and only if $i \in I_{10}(\omega,\nu_1)$, (2) $N_i(\omega,\nu_1) \ge N_i(\omega,\nu_2)$ where strict inequality holds if and only if $i \in I_{10}(\omega,\nu_1)$, (3) $H_T(\omega,\nu_1) \le H_T(\omega,\nu_2)$ where equality holds if and only if $I_{10}(\omega,\nu_1) = \emptyset$, (4) $H_N(\omega,\nu_1) \ge H_N(\omega,\nu_2)$ where equality holds if and only if $I_{10}(\omega,\nu_1) = \emptyset$.*

(ii) *For fixed $\nu > 0$ and $0 \le \omega_1 < \omega_2$ we have: (1) $T_i(\omega_1,\nu) \le T_i(\omega_2,\nu)$ where equality holds if and only if $i \in I_{00}(\omega_1,\nu)$, (2) $N_i(\omega_1,\nu) \ge N_i(\omega_2,\nu)$ where equality holds if and only if $i \notin I_{10}(\omega_1,\nu)$, (3) $H_T(\omega_1,\nu_1) \ge H_T(\omega_2,\nu)$ where equality holds if and only if $I_{00}(\omega,\nu_1) = [1,n]$, (4) $H_N(\omega_1,\nu) \ge H_N(\omega_2,\nu)$ where equality holds if and only if $I_{10}(\omega_1,\nu) = \emptyset$.*

(iii) *$H_T(\omega,\nu)$ and $H_N(\omega,\nu)$ are non-negative and continuous in $[0,\infty) \times [0,\infty)$.*

(iv) *If $H_N(0,0) \le \bar{N}$ then $H_N(\omega,\nu) < \bar{N}$ for $\omega > 0$ and $\nu > 0$.*

Based on monotonous properties described in Lemma 1 we can establish the following result about the number of NE the game can have.

Theorem 4. *There is at most one NE.*

Note that

$$\begin{aligned} H_T(\omega,0) = &\sum_{i\in[1,n]:\, h_i(\omega+c_T)<h_i g_i/N_i^0 \le h_i(\omega+c_T)+g_i c_N} \pi_i \left(\frac{1}{c_T+\omega} - \frac{N_i^0}{g_i}\right) \\ &+ \frac{c_N}{\omega+c_T} \times \sum_{i\in[1,n]:\, h_i(\omega+c_T)+g_i c_N < h_i g_i/N_i^0} \pi_i \frac{g_i}{(\omega+c_T)h_i + c_N g_i}. \end{aligned} \tag{7}$$

The following lemma supplying some properties of $H_T(\omega,0)$ follows straighforward from (7) and Lemma 1.

Lemma 2. *(i) $H_T(\cdot,0)$ is non-negative and continuous in $(0,\infty)$,*
(ii) $H_T(\omega,0) = 0$ for enough big ω, namely, for $\omega \ge \max_i\{g_i/N_i^0 - g_i c_N/h_i\} - c_T$,
(iii) $H_T(\omega,0)$ is strictly decreasing on ω while $H_T(\omega,0) > 0$,

Lemma 2 implies that if $H_T(0,0) > \bar{T}$ that there exists the unique positive ω_{10}^* such that $H_T(\omega_{10}^*,0) = \bar{N}$ (indexes 10 mean that in this moment we look for the optimal solution where $\omega > 0$ and $\nu = 0$). If $H_T(0,0) \le \bar{T}$ then $H_T(\tau,0) < \bar{T}$ for $\tau > 0$. Then, from Theorems 2 and 3 and Lemmas 1(iv) and 2 we have the following theorem.

Theorem 5. *Let $H_N(0,0) \le \bar{N}$ then*
(a) if $H_T(0,0) \le \bar{T}$ then $(T(0,0), N(0,0))$ is NE,
(b) if $H_T(0,0) > \bar{T}$ then $(T(\omega^*_{10},0), N(\omega^*_{10},0)$ is NE.

By Lemma 1 the following Lemma holds

Lemma 3. *If $H_N(0,0) > \bar{N}$ then there is ν^*_{01} such that $H_N(0,\nu^*_{01}) = \bar{N}$ (subscript 01 signifies that we look for the optimal solution where $\omega = 0$ and $\nu > 0$) and there is $\hat{\omega}$ such that $H_N(\hat{\omega},0) = \bar{N}$. Thus, $H_N(\omega,\nu) < \bar{N}$ for each $\omega > \hat{\omega}$ and each non-negative ν. For each $\omega \in (0,\hat{\omega}]$ there is unique nonnegative $\nu(\omega)$ such that $H_N(\omega,\nu(\omega)) = \bar{N}$. $\nu(\omega)$ is continuous and strictly decreasing on ω, $\nu(0) = \nu^*_{01}$ and $\nu(\hat{\omega}) = 0$.*

Thus, by Lemma 3 we can introduce the following notation:

$$\bar{H}_T(\omega) = H_T(\omega,\nu(\omega)) = \sum_{i \in I_{10}(\omega,\nu(\omega))} \pi_i \left(\frac{1}{c_T+\omega} - N_i^0\right)$$
$$+ \frac{\nu(\omega)+c_N}{\omega+c_T} \times \sum_{i \in I_{11}(\omega,\nu(\omega))} \pi_i \frac{g_i}{(\omega+c_T)h_i + (\nu(\omega)+c_N)g_i}.$$

Then by Lemma 1 $\bar{H}_T$ is continuous and strictly decreasing in $(0,\hat{\omega})$. Thus, if $\bar{H}_T(0) \le \bar{T}$ then $\bar{H}_T(\omega) < \bar{T}$ for $\omega \in (0,\hat{\omega})$. If $\bar{H}_T(\hat{\omega}) > \bar{T}$ then $\bar{H}_T(\omega) > \bar{T}$ for $\omega \in (0,\hat{\omega})$. If $\bar{H}_T(\hat{\omega}) < \bar{T}$ and $\bar{H}_T(0) > \bar{T}$ then there is unique $\omega^*_{11} \in (0,\hat{\omega})$ such that $\bar{H}_T(\omega^*_{11}) = \bar{T}$ (subscript 11 signifies that we look for the optimal solution where $\omega, \nu > 0$). Then, from Theorems 2 and 3 we have the following theorem.

Theorem 6. *Let $H_N(0,0) > \bar{N}$ then*
(a) if $\bar{H}_T(0) = H_T(0,\nu^*_{01}) \le \bar{T}$ then $(T(0,\nu^*_{01}), N(0,\nu^*_{01}))$ is NE,
(b) if $\bar{H}_T(0) = H_T(0,\nu^*_{01}) > \bar{T}$ and $\bar{H}_T(\hat{\omega}) = H_T(\hat{\omega},0) > \bar{T}$ then $(T(\omega^*_{10},0), N(\omega^*_{10},0))$ is NE,
(c) if $\bar{H}_T(0) = H_T(0,\nu^*_{01}) > \bar{T}$ and $\bar{H}_T(\hat{\omega}) = H_T(\hat{\omega},0) \le \bar{T}$ then $(T(\omega^*_{11},\nu(\omega^*_{11})), N(\omega^*_{11},\nu(\omega^*_{11}))$ is NE.

Theorems 4 – 6 imply the following main result.

Theorem 7. *There is unique NE given by Theorems 5 and 6.*

The case where there are no the costs of power usage for Transmiter and Jammer, namely, $c_T = c_N = 0$, is an important particular case of our model. For this case our model from non-zero sum game turns into zero-sum game. Then, it is clear, that $H_N(0+,0+) = \infty$ and $\bar{H}_T(0+) = \infty$ and we come under conditions of Theorem 6 (b) and (c). Thus, if $H_T(\hat{\omega},0) \le \bar{T}$ (where $\hat{\omega}$ is defined by equation $H_N(\hat{\omega},0) = \bar{N}$, see Lemma 3), then $(T(\omega^*_{11},\nu(\omega^*_{11})), N(\omega^*_{11},\nu(\omega^*_{11}))$ is the equilibrium. If $H_T(\hat{\omega},0) > \bar{T}$ then $(T(\omega^*_{10},0), N(\omega^*_{10},0))$ is the equilibrium.

4 Algorithm

In this section we present an algorithm based on the bisection method and Theorems 5 and 6 and Lemmas 2 and 3 to find the optimal values of ω and ν and the corresponding optimal solution.

Algorithm

```
Step 1. If $H_N(0,0) \le \bar{N}$ and $H_T(0,0) \le \bar{T}$ then $\omega = \nu = 0$ and $(T(0,0), N(0,0))$ is NE and the algorithm is terminated.
Step 2. If $H_N(0,0) \le \bar{N}$ and $H_T(0,0) > \bar{T}$. Then call $\omega_{10}^* = BS_T^1(0)$, $(T(\omega_{10}^*, 0), N(\omega_{10}^*, 0))$ is NE and the algorithm is terminated.
Step 3. If $H_N(0,0) > \bar{N}$ then $BS_N^1(\hat{\omega}, 0)$ and $\nu_{01}^* = BS_N^2(0)$.
Step 4. If $H_T(0, \nu_{01}^*) \le \bar{T}$ then $(T(0, \nu_{01}^*), N(0, \nu_{01}^*))$ is NE and the algorithm is terminated.
Step 5. If $H_T(0, \nu_{01}^*) > \bar{T}$ and $H_T(\hat{\omega}, 0) > \bar{T}$ then $(T(\omega_{10}^*, 0), N(\omega_{10}^*, 0))$ is NE and the algorithm is terminated.
Step 6. If $H_T(0, \nu_{01}^*) > \bar{T}$ and $H_T(\hat{\omega}, 0) \le \bar{T}$ then $\omega^0 = 0$, $\omega^1 = \hat{\omega}$.
    Step 6a. $\nu^0 = BS_N^2(\omega^0)$, $\nu^1 = BS_N^2(\omega^1)$.
    Step 6b. Set $\bar{\omega} = (\omega^1 + \omega^0)/2$.
    Step 6c. $\bar{\nu} = BS_N^2(\bar{\omega})$.
    Step 6d. If $\omega^1 - \omega^0 \le \epsilon$, then $\omega_{11}^* = (\omega^1 + \omega^0)/2$, $\nu_{11}^* = BS_N^2(\omega_{11}^*)$ and $(T(\omega_{11}^*, \nu_{11}^*), N(\omega_{11}^*, \nu_{11}^*))$ is NE and the algorithm is terminated.
    Step 6e. If $\omega^1 - \omega^0 > \epsilon$, then, if $H_T(\bar{\omega}, \bar{\nu}) < \bar{N}$ then $\omega^0 = \bar{\omega}$, if $H_N(\bar{\omega}, \bar{\nu}) > \bar{N}$ then $\omega^1 = \bar{\omega}$ and go to Step 6b.
    Step 6f. Let $\omega^1 - \omega^0 > \epsilon$ and $H_N(\bar{\omega}, \bar{\nu}) = \bar{N}$ then $\omega_{11}^* = \bar{\omega}$, $\nu_{11}^* = \bar{\nu}$ and $(T(\omega_{11}^*, \nu_{11}^*), N(\omega_{11}^*, \nu_{11}^*))$ is NE and the algorithm is terminated.
```

Function $\omega = BS_T^1(\nu)$

```
Step 1. Let $\omega^0 = 0$, $\omega^1 = max_i\{g_i/N_i^0 - g_i c_N/h_i\} - c_T$
Step 2. Set $\bar{\omega} = (\omega^1 + \omega^0)/2$.
Step 3. If $\omega^1 - \omega^0 \le \epsilon$, then return $\omega = (\omega^1 + \omega^0)/2$.
Step 4. If $\omega^1 - \omega^0 > \epsilon$ then, if $H_T(\bar{\omega}, \nu) < \bar{T}$ set $\omega^0 = \bar{\omega}$, if $H_T(\bar{\omega}, \nu) > \bar{T}$ set $\omega^1 = \bar{\omega}$ and go to Step 2.
Step 5. Let $\omega^1 - \omega^0 > \epsilon$ and $H_T(\bar{\omega}, \nu) = \bar{N}$ then return $\bar{\omega}$.
```

Function $\omega = BS_N^1(\nu)$

```
Step 1. Let $\omega^0 = 0$, $\omega^1 = max_i\{g_i/N_i^0 - g_i c_N/h_i\} - c_T$
Step 2. Set $\bar{\omega} = (\omega^1 + \omega^0)/2$.
Step 3. If $\omega^1 - \omega^0 \le \epsilon$ then return $\omega = (\omega^1 + \omega^0)/2$.
Step 4. If $\omega^1 - \omega^0 > \epsilon$ then, if $H_N(\bar{\omega}, \nu) < \bar{N}$ set $\omega^0 = \bar{\omega}$, if $H_N(\bar{\omega}, \nu) > \bar{N}$ set $\omega^1 = \bar{\omega}$ and go to Step 2.
Step 5. Let $\omega^1 - \omega^0 > \epsilon$ and $H_N(\bar{\omega}, \nu) = \bar{N}$ then return $\bar{\omega}$.
```

Function $\nu = BS_N^2(\omega)$

```
Step 1. Let $\nu^0 = 0$, $\nu^1 = max_i\{h_i/N_i^0 - h_i c_T/g_i\} - c_N$
```

Step 2. Set $\bar{\nu} = (\nu^1 + \nu^0)/2$.
Step 3. If $\nu^1 - \nu^0 \le \epsilon$ then return $\nu = (\nu^1 + \nu^0)/2$.
Step 4. If $\nu^1 - \nu^0 > \epsilon$ then, if $H_N(\omega, \bar{\nu}) < \bar{N}$ set $\nu^0 = \bar{\nu}$, if $H_N(\omega, \bar{\nu}) > \bar{N}$ set $\nu^1 = \bar{\nu}$ and go to Step 2.
Step 5. Let $\nu^1 - \nu^0 > \epsilon$ and $H_N(\omega, \bar{\nu}) = \bar{N}$ then return $\bar{\nu}$.

5 Numerical Examples

In this section we consider a few numerical examples. The numerical examples correspond to the OFDM scheme with five sub-channels ($n = 5$). Consequently, we take $\pi_i = 1/5$. Let us consider an important particular case of jamming in the OFDM system when the jammer is near the base station. In this scenario $h_i = 1$ for all $i \in [1, 5]$. First, we take $g_i = \kappa^{i-1}$ for $i \in [1, 5]$ where $\kappa \in (0, 1)$. This corresponds to Rayleigh fading. Also we set $N_i^0 = 0.1$, $i \in [1, 5]$, $\bar{N} = \bar{T} = 1$ and $c_T = c_N = 0.1$. The payoffs of the players as functions of κ is shown in Figure 1. As an example, we depict the optimal strategies of the players in Figure 2 for the case $\kappa = 1/2$. It is interesting to observe that Jammer spends more energy in the sub-channels with good quality and Transmitter tries to use the resources of the bad quality sub-channels. In other words, Jammer pays less attention to the sub-channel with bad quality and Transmitter takes an opportunity to send some part of information over bad quality sub-channels.

In the second example, we consider that the background noise is different in each sub-channel. Specifically, we take $N_i^0 = i/10$ for $i \in [1, 5]$ and $\kappa = 0.2$. Then, we obtain the optimal strategies $T^* = (3.66, 1.18, 0.16, 0, 0)$ and $N^* = (3.90, 1.10, 0, 0, 0)$ with the payoffs 0.07 and -0.27. This example illustrates the possibility of the situation when Transmitter uses more sub-channels than Jammer, see formulae (5) and (6).

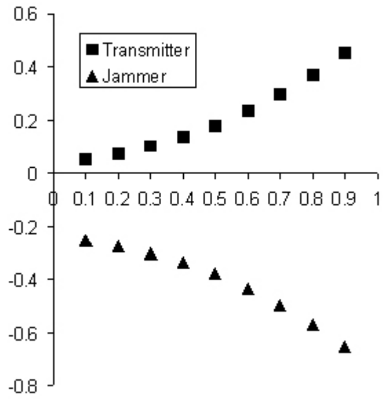


Fig. 1. The payoffs as functions of κ

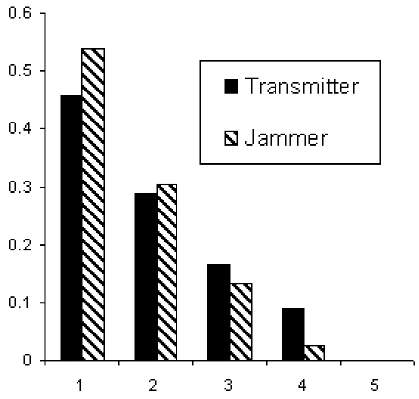


Fig. 2. The optimal strategies for $\kappa = 1/2$

6 Conclusions

In this paper we considered jamming in wireless networks with transmission cost for both transmitter and jammer from a game theoretical point of view. We proved the existence and uniqueness of NE. It turned out that it is possible to provide analytical expressions for the equilibrium strategies which depend on two parameters. We propose an efficient algorithm for finding these parameters, and hence, the optimal strategies. The presented jamming game is a generalization of the standard water-filling problem. In fact, since we take into account the cost of transmission, for the case of the single player, we obtain even the generalization of the water-filling optimization problem. The present framework allows us to study both water-filling in time and water-filling in frequency. By means of numerical examples we study an important particular case of jamming of the OFDM system when the jammer is situated close to the base station. These examples showed that Jammer pays less attention to the sub-channel with bad quality and Transmitter takes an opportunity to send some part of information over bad quality sub-channels.

Acknowledgements

We thank M. Debbah and A. Suarez-Real for insightful discussions and remarks on the early version of the manuscript.

References

1. S. Alpern, and S. Gal, *The Theory of Search Games and Rendezvous.* International Series in Operations Research and Management Science, v. 55. Kluwer Academic Publishers, 2003.
2. E. Altman, K. Avrachenkov, R. Marquez, and G. Miller, “Zero-sum constrained stochastic games with independent state processes”, *Math. Meth. Oper. Res.*, v.62, pp. 375-386, 2005.
3. Kashyap A, Basar T, Srikant R, ”Correlated jamming on MIMO Gaussian fading channels”, Information Theory, IEEE Transactions on, Vol. 50, No. 9. (2004), pp. 2119-2123.
4. V.J. Baston and A.Y. Garnaev, “A Search Game with a Protector”, *Naval Research Logistics*, v. 47, pp. 85-96, 2000.
5. M. H. Brady amd J. M. .Cioffi, ”The worst-case interference in DSL systems employing dynamic spectrum management”, Eurasip Journal on Applied Signal Processing, Vol. 2006, pages 1-11.
6. T. Cover and J. Thomas, *Elements of Information Theory*, Wiley, 1991.
7. E.A. Jorswieck and H. Boche, “Performance analysis of capacity of MIMO systems under multiuser interference based on worst case noise behavior”, EURASIP Journal on Wireless Communications and Networking, v.2, pp.273-285, 2004.
8. S. Kasturia, J.T. Aslanis and J.M. Cioffi, “Vector coding for partial response channels”, *IEEE Trans. Information Theory*, v.36(4), pp.741-762, 1990.

9. L. Lai and H. El Gamal, "The water-filling game in fading multiple access channels", submitted to *IEEE Trans. Information Theory*, November 2005, available at `http://www.ece.osu.edu/ helgamal/`.
10. R.G. Gallager, *Information Theory and Reliable Communication*, Wiley, 1968.
11. A. Garnaev, *Search Games and Other Applications of Game Theory*, Springer, 2000.
12. A. Garnaev, "Find a"Hidden" Treasure", *Naval Research Logistics*, 2006 (to apprear).
13. A.J. Goldsmith and P.P. Varaiya, "Capacity of fading channels with channel side information", *IEEE Trans. Information Theory*, v.43(6), pp.1986-1992, 1997.
14. A. Suarez-Real, *Robust Waterfilling strategies for the fading channel*, INRIA. Master SICOM Thesis. 2006.
15. D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*, Cambridge University Press, 2005.

Appendix

Proof of Theorem 3. Let (T^*, N^*) be a NE. Then for each $i \in [1, n]$ the following four cases are possible: (a) $T_i^* = N_i^* = 0$, (b) $T_i^* = 0$, $N_i^* > 0$, (c) $T_i^* > 0$, $N_i^* > 0$ and (d) $T_i^* > 0$, $N_i^* = 0$.

(a) Let $T_i^* = 0$ and $N_i^* = 0$ then by (2) we have that $g_i/N_i^0 - c_T \le \omega^*$. Thus, $i \in I_{00}(\omega^*, \nu^*)$ and $T^* = T(\omega^*, \nu^*)$, $N^* = N(\omega^*, \nu^*)$.

(b) Let $T_i^* > 0$ and $N_i^* = 0$ then by (2) we have that

$$\frac{g_i}{g_i T_i^* + N_i^0} - c_T = \omega^*.$$

Thus, $\dfrac{g_i}{N_i^0} > \omega^* + c_T$ and $T_i^* = \dfrac{1}{\omega^* + c_T} - \dfrac{N_i^0}{g_i}$. Then, by (3) we have that

$$\begin{aligned}\nu^* &\ge \frac{g_i h_i T_i^*}{(h_i T_i^* + N_i^0) N_i^0} - c_N = \left(\frac{1}{\omega^* + c_T} - \frac{N_i^0}{g_i}\right) \frac{h_i}{N_i^0}(\omega^* + c_T) - c_N \\ &= \frac{h_i}{N_i^0} - \frac{h_i}{g_i}(\omega^* + c_T) - c_N.\end{aligned}$$

Thus, $i \in I_{10}(\omega^*, \nu^*)$ and $T^* = T(\omega^*, \nu^*)$, $N^* = N(\omega^*, \nu^*)$.

(c) Let $T_i^* > 0$ and $N_i^* > 0$ then by (2) and (3) we have that

$$\begin{aligned}\omega^* &= \frac{g_i}{g_i T_i^* + h_i N_i^* + N_i^0} - c_T, \\ \nu^* &= \frac{g_i h_i T_i}{(g_i T_i^* + h_i N_i^* + N_i^0)(h_i N_i^* + N_i^0)} - c_N.\end{aligned}$$

(d) Let $T_i^* = 0$ and $N_i^* > 0$ then by (3) $\nu^* = -c_N < 0$. This contradiction proves that the assumption that $T_i^* = 0$ and $N_i^* > 0$ cannot take place and the result follows.

Proof of Lemma 1. (i1) For fixed $\omega > 0$ and $0 \le \nu_1 < \nu_2$ we have $I_{10}(\omega, \nu_1) \subseteq I_{10}(\omega, \nu_2)$ and $I_{11}(\omega, \nu_1) \supseteq I_{11}(\omega, \nu_2)$. Since for any ν $I_{10}(\omega, \nu) \cup I_{11}(\omega, \nu) =$

$[1,n]\backslash I_{00}(\omega)$ does not depend on ν we have to consider separately the cases $i \in I_{00}(\omega,\nu_1)$, $i \in I_{10}(\omega,\nu_1)$, $i \in I_{11}(\omega,\nu_2)$ and $i \in I_{11}(\omega,\nu_1) \cap I_{10}(\omega,\nu_2)$, and then (i1) now follows easily from the definitions.

Proof of Theorem 4. Suppose there are at least two NE, say $(T(\omega_1,\nu_1), N(\omega_1,\nu_1))$ and $(T(\omega_2,\nu_2), N(\omega_2,\nu_2))$.

Suppose that $\nu_1 = \nu_2 = \nu$. We can assume that $0 \le \omega_1 < \omega_2$. Thus, by Theorem 2, $H_T(\omega_2,\nu) = \bar{T}$. Thus, by Lemma 1 (ii3) $H_T(\omega_2,\nu) \le H_T(\omega_1,\nu)$ So, $H_T(\omega_1,\nu) = \bar{T} = H_T(\omega_2,\nu)$ and by Lemma 1 (ii3) $I_{00} = [1,n]$. Thus, $H_T(\omega_2,\nu) = 0$. This contradictions shows ω_1 has to be equal to ω_2.

Suppose that $0 \le \nu_1 < \nu_2 = \nu$. Thus, by Theorem 2, $H_T(\omega_2,\nu_2) = \bar{N}$. So, $I_{11}(\omega_2,\nu_2) \ne \emptyset$.

Assume that $\omega_1 \le \omega_2$. Then $I_{11}(\omega_2,\nu_2) \subseteq I_{11}(\omega_1,\nu_1)$ and $N_i(\omega_1,\nu_1) > N_i(\omega_2,\nu_2)$ for $i \in I_{11}(\omega_2,\nu_2)$. Thus, $H_N(\omega_1,\nu_1) > H_N(\omega_2,\nu_2) = \bar{N}$. This contradiction shows that the inequality $\omega_1 > \omega_2$ has to be held.

So, let $\omega_1 > \omega_2$. Thus, $I_{00}(\omega_2) \subseteq I_{00}(\omega_1)$. We can assume that $I_{00}(\omega_2) \ne [1,n]$ since otherwise the equilibrium coincides with each other, namely $T_i(\omega_k,\nu_k) = N_i(\omega_k,\nu_k) = 0$ for $k = 1,2$. So, $I_{00}(\omega_2) \ne [1,n]$. Thus, by Lemma $H_T(\omega_2,\nu_2) \ge H_T(\omega_2,\nu_1) > H_T(\omega_1,\nu_1) = \bar{T}$. This contradiction completes the proof of theorem.

A Network Formation Game Approach to Study BitTorrent Tit-for-Tat

Giovanni Neglia[1], Giuseppe Lo Presti[2], Honggang Zhang[3], and Don Towsley[4]

[1] D.I.E.E.T., Università degli Studi di Palermo, Italy
INRIA Sophia Antipolis, France
giovanni.neglia@ieee.org
[2] IT Dept., C.E.R.N., Switzerland
giuseppe.lopresti@cern.ch
[3] Math and Computer Science, Dept. Suffolk University
hzhang@ieee.org
[4] Computer Science Dept., University of Massachusetts Amherst
towsley@cs.umass.edu

Abstract. The Tit-for-Tat strategy implemented in BitTorrent (BT) clients is generally considered robust to selfish behaviours. The authors of [1] support this belief studying how Tit-for-Tat can affect selfish peers who are able to set their upload bandwidth. They show that there is a "good" Nash Equilibrium at which each peer uploads at the maximum rate. In this paper we consider a different game where BT clients can change the number of connections to open in order to improve their performance. We study this game using the analytical framework of network formation games [2]. In particular we characterize the set of *pairwise stable networks* the peers can form and how the peers can dynamically reach such configurations. We also evaluate the loss of efficiency peers experience because of their lack of coordination: we find that the loss of efficiency is in general unbounded despite the utilization of the Tit-for-Tat strategy.

1 Introduction

Recently peer-to-peer applications (e.g., BitTorrent [3], Kazaa, eDonkey, and Gnutella [4]) have become very popular. CacheLogic [5] estimates that peer-to-peer generated 60% of all US Internet traffic at the end of 2004 and in particular BitTorrent (BT in what follows), constituted about 30% of Internet backbone traffic in June 2004.

One of the reason of BT success is its ability to enforce cooperation among the peers contrasting the well know problem of free-ride. In fact all the peers interested in a specific file have to announce themselves to a central server, called *tracker*. The tracker maintains the set of active peers, also called the *swarm*, interested in that content and communicates a small random subset of peers from the swarm to each new peer. Peers use this subset to connect to other peers and exchange missing pieces of the file. In general a peer receives many requests for different pieces. In order to decide which requests should be satisfied, the peer uses the Tit-for-Tat strategy: it uploads to the n_u peers (the default value is 4) from which it can download at the highest rate, i.e., its best uploaders. This strategy is clearly intended to benefit the peers who contribute more to the system.

Tit-for-Tat is generally considered robust to selfish behaviour. To the best of our knowledge, the only analytical support to this belief is in [1]. The authors of [1] study how Tit-for-Tat can affect selfish peers who can change their upload bandwidth in order to try to maximize their downloading rate. We refer to their model as *the rate game*. Under several assumptions, they show that there is a good NE at which each peer uploads at the maximum rate (their model and their results are discussed in more detail in Section 2). However, we observe that BT clients can also change the number of connections to open in order to improve their performance and achieve better performance.

In order to study this aspect we have introduced a new model, which we refer to as *the connection game*. This model captures Tit-for-Tat reciprocation feature by considering that two peers set up a connection between themselves only when they both find it beneficial. We study this game using the analytical framework of network formation games [2]. In Section 3 we characterize the topologies of some *pairwise stable networks* peers can form both in homogeneous scenarios and in heterogenous scenarios (i.e. respectively when all the links have the same or different capacity values). In Section 4 we evaluate the loss of efficiency peers experience because of their lack of coordination: we find that the loss of efficiency is in general unbounded despite the utilization of the Tit-for-Tat strategy. Finally in Section 5 we propose a simple dynamics for this game. We prove that when connection costs are linear functions of the number of links, this dynamics converges to a pairwise stable network. We also quantify by simulations the convergence time and show that as the network size increases the dynamics leads to networks near to the equilibria described in Section 3.

The results about pairwise stable networks in the homogeneous scenario have already been presented in [6]. This paper presents a new result about the heterogeneous scenario and illustrates extensively the results about the dynamics which were just listed in [6]. To the best of our knowledge only three other papers [7,8,9] use game theory to study the overlay structure arising from the interaction among selfish peers . In these papers peers build an overlay to provide connectivity. The cost of each peer is the sum of the lengths of the shortest paths to all the other peers plus the cost of the links created to connect to the neighbours.

Due to space constraints, proofs are in [10].

2 The Rate Game

Before illustrating the results about Tit-for-Tat we describe in detail the network model considered in [1], because we adopt the same. According to this model every peer has two asymmetric access links to the Internet: one downstream link and one upstream link. Besides it is assumed that bottlenecks can occur only at upstream links. These assumptions are supported by measurement studies

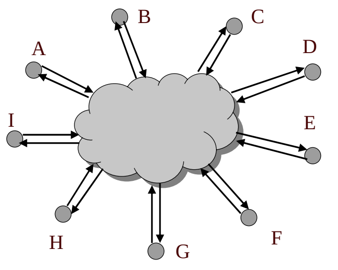


Fig. 1. An example of star network topology

(e.g. [11]): most peers in current peer-to-peer networks use cable modem or ADSL to get connected to the Internet and usually the data throughput is limited by the "last mile" and the downstream link has higher capacity than the upstream link [11]. Thus, in the star network shown in Figure 1, the Internet cloud can be represented simply as a central node.

A peer r uses its downstream link to get data from other peers. The downstream link of peer r is a "private" link in the sense that this link is only used by peer r itself. On the other hand, the upstream link of peer r is equally shared by all other peers that are downloading files from peer r. We can think of the upstream link of peer r as a "public" link from the point of view of peer r. The model ignore the content dynamics, because it assumes that every peer has potentially interesting data for every other peer and all possible connections can be established.

The authors of [1] study how Tit-for-Tat can affect selfish peers who are able to set their uploading bandwidth in a BT network. Due to Tit-for-Tat reciprocation mechanism, the downloading rate each peer gets is an increasing function of its uploading bandwidth. The authors assume that each peer sets its uploading bandwidth at the minimum level which guarantees them the maximum downloading rate they can achieve, i.e., the downloading rate they would get by uploading at their physical uploading bandwidth.[1] They also assume that the network has a finite number of groups of peers, each of them characterized by a different physical uploading bandwidth and with at least $n_u + 2$ peers. Under these assumptions the authors show that there is a single *good* Nash equilibrium point at which each peer uploads at the maximum rate. Note that in [1], for a given peer, the total number of other peers to set up a connection with is fixed. However, we observe that BT clients can benefit from changing their number of connections (an example is shown in [6]).

3 The Connection Game

In this section, we first formally introduce our game then we study the network equilibria arising in this game using the analytical framework of network formation games [2].

[1] In reality they need to assume that each peer sets its bandwidth to a value slightly larger than such minimum, otherwise there would be multiple Nash equilibria.

Assumptions are detailed in the previous section. We refer to peers as players and to connections as links. Let $\mathbf{R} = \{1, 2, \cdots, R\}$ denote the set of players. The strategy of a player i is the set of intended connections player i wants to establish, which is denoted by $s_i = \{s_{i,j} | \, j \in \mathbf{R} \backslash \{i\}\}$, where $s_{i,j} = 1$ means that player i intends to create a link (open a connection) with player j and $s_{i,j} = 0$ means that player i does not intend to create such a link. With the *Tit-for-Tat strategy*, both players have to agree in order to create a link, hence a link between players i and j is formed if and only if $s_{i,j} = s_{j,i} = 1$. A strategy profile $s = \{s_1, s_2, \cdots, s_R\}$ therefore induces a network $g(s) = \{g_{i,j}, i, j \in \mathbf{R}\}$, where $g_{i,j} = 1$ denotes the existence of link (i, j) and $g_{i,j} = 0$ denotes the absence of link (i, j). Given a network g, we use $g + g_{i,j}$ or $g - g_{i,j}$ to denote the network obtained by adding or severing the link (i, j). We also let $N_i(g) = \{j \in \mathbf{R} : j \neq i, g_{i,j} = 1\}$ be the set of player i's neighbors in graph g, and let $n_i(g) = |N_i(g)|$. A network is symmetric if $n_i(g) = n, \forall i \in \mathbf{R}$, i.e. its topology is a regular graph (all players have the same number of connections).

The payoff or benefit of player i is given by its download rate minus the cost of opening connections: $B_i = G_i - \Phi_i(n_i) = \sum_{j \in N_i(g)} C_j/n_j - \Phi_i(n_i)$, where C_j is the uploading capacity of peer j. We assume that Φ_i is a convex function of n_i (a linear function is a particular case). The marginal benefit for player i to open a new connection with player j is:

$$b_i(n_i(g), n_j(g)) = B_i(g + g_{i,j}) - B_i(g) = \frac{C_j}{n_j(g) + 1} - \Phi_i(n_i(g) + 1) + \Phi_i(n_i(g)).$$

A connection between two players can be set up only when both of them find this connection beneficial. This coordination requirement makes the concept of Nash equilibrium (NE) *partially inadequate*. To address this issue, the idea of NE has been supplemented with the requirement of pairwise stability [12], described below.

Definition 1. *A network g is a pairwise equilibrium network (PEN) if the following conditions hold: 1) there is a NE strategy profile which supports g; 2) for $g_{i,j} = 0$, $B_i(g + g_{i,j}) > B_i(g) \Rightarrow B_j(g + g_{i,j}) < B_j(g)$.*

3.1 Equilibria in Homogeneous Networks

In this section we consider homogeneous networks in which all peers have the same upload capacity and payoff function.

Based on the previous assumptions, our game is the local spillovers game with strategic substitutes properties studied in [13]. Some of the following results (Theorems 1, 2 and 4) can be derived from [13]. Please see Appendix IV in [10] for details.

Theorem 1. *If the number of players is even, a symmetric PEN always exists. Specifically, if $b(0, 0) \leq 0$, the empty network is a PEN; if $b(r - 2, r - 2) \geq 0$, the complete network is a PEN; if $b(k, k) \leq 0 \leq b(k - 1, k - 1)$, the regular graph with degree k is a PEN. When the previous inequalities are strict, the degree of the PEN is unique.*

Remark. Even when a symmetric network can arise from player interaction according to Theorem 1, the degree of the network is in general different from the default value used in current BitTorrent implementation ($n_u = 4$). This means that the symmetric network created by compliant peers in BitTorrent networks is not in general a PEN for our overlay formation game.

Besides symmetric PENs discussed in the above, we have the following theorem addressing asymmetric PENs.

Theorem 2. *There can be at most one player not connected to any other players in a PEN and the rest of the network is a symmetric network of a unique degree. In asymmetric networks with a single component, if two players with the same number of connections k (i.e. two nodes with the same degree k) are connected to each other, then any two players with fewer number of links than k (or two nodes with lower degrees than k) must be mutually connected.*

Theorem 3. *In a scenario where a unique degree -h- is possible for the symmetric PENs, there can be at most h players with degree smaller than h. Say l the number of players with degree smaller than h, there can be at most $(h-l)l$ players with degree bigger than h, each of them with degree at most $h+l$. If the cost function is linear then there are no players with degree bigger than h.*

Remarks. The two theorems above rule out many possible asymmetric networks, like those with two or more isolated players or interlinked stars.[2] Note that the degree of symmetric PENs h depends only on the cost function $\Phi()$ and the capacity C, and is independent from the number of players R. Because of these theorems the *distance* between a PEN and a symmetric PEN is bounded and becomes less significant as the number of players R increases. Formally:

$$\lim_{R\to\infty} \frac{1}{R} E\left\{\sum_{i=1}^{R} |n_i(g_{PEN}) - h|\right\} = 0.$$

Similarly the average payoff per player in a PEN converges to that of a symmetric PEN.

The following result shows that players having more connections gain higher payoffs than other players.

Theorem 4. *Let g be a pairwise equilibrium network in which $n_i(g) < n_j(g)$. If $\forall u \in N_i(g)$, $\exists v \in N_j(g)$ s.t. $n_u = n_v$, then $B_i(g) < B_j(g)$.*

Note that if player i's neighborhood is included in player j's neighborhood ($N_i \subset N_j$), the condition "$\forall u \in N_i(g)$, $\exists v \in N_j(g)$, s.t. $n_u = n_v$" is satisfied.

[2] An interlinked star network has a maximally connected group and a minimally connected group of players. In addition, the maximally connected players are connected to all players while the minimally connected group has links only with the players in the maximally connected set.

3.2 Equilibria in Heterogeneous Networks

In this section we consider that peers can have different uploading capacities. Given C_i the capacity of node i, let us indicate k_i a possible node degree in a symmetric PEN when all players have capacity C_i. The following result holds.

Theorem 5. *Under linear costs ($\Phi(n_i) = \alpha n_i$) if in network g each player i has degree k_i, then the network is a PEN.*

Sketch of the proof. We only consider the case $0 < k_i < R-1$. Let us consider $g_{i,j}$. If $g_{i,j} = 1$ then both $b_i(k_i - 1, k_j - 1) \geq 0$ and $b_j(k_j - 1, k_i - 1) \geq 0$ have to be satisfied in order to g be a PEN. $b_i(k_i - 1, k_j - 1) = C_j/k_j - \alpha \geq 0$ because k_j is the degree of a symmetric PEN when all the peers have capacity C_i (see Theorem 1), similarly for $b_j(k_j - 1, k_i - 1)$. If $g_{i,j} = 0$ then we just observe that no player wants to create the link. For example $b_j(k_j, k_i) \leq 0$ again because of the result in Theorem 1.

Remarks. *First*, note that this theorem, differently from Theorem 1 does not state the existence of a PEN. Depending on the values of the capacities, it could be impossible to create a network where all the players open k_i connections (for example if the number of players and the values k_i are odd). *Second*, the result does not hold in general under different cost functions (it is possible to show examples), here it is fundamental that the marginal benefit depends only on the number of connections of the other player. *Third*, in this PEN the distribution of the number of connections in the network mainly reflects the distribution of the capacities. In Section 5 we will show how the PEN selected by the dynamic process we will introduce is "near" to the PEN described in this theorem, and hence the distribution of the number of links resembles the distribution of the capacities in the network.

4 Loss of Efficiency of Symmetric Equilibria

In our game, given the number of players, the number of possible overlays players can create is finite. Hence there is one network g_{opt} with the highest total payoff $\sum_{i \in \mathbf{R}} B_i(g_{opt})$. We define the efficiency loss of a PEN g as the ratio of the highest total payoff over the total payoff of the PEN:

$$L_{eff}(r, C, \Phi) = \frac{\sum_{i \in \mathbf{R}} B_i(g_{opt})}{\sum_{i \in \mathbf{R}} B_i(g)}.$$

We note that L_{eff} depends in general on the number of players, and the upload capacities and cost functions of those players. The following theorem states that L_{eff} is unbounded even for the class of linear connection cost functions ($\Phi(n) = \alpha n$). Therefore, the price of anarchy (the worst efficiency loss of all NEs) is infinite.[3] Please see Appendix VI in [10] for a detailed proof.

[3] This is different from what happens for selfish routing, where the price of anarchy is finite, and independent from the network topology for networks in which edge latency does not depend in a highly nonlinear fashion on the edge congestion [14].

Theorem 6. *For the class of linear connection cost functions, the loss of efficiency is unbounded: given an even number of players and an upload capacity* C, $\forall M \in \mathbb{R}, \exists \alpha^* \in \mathbb{R}^+$ *s.t.* $L_{eff}(r, C, \Phi^*) > M$, *where* $\Phi^*(n) = \alpha^* n$.

5 Dynamic Models

We investigate in this section how peers can dynamically reach a PEN. Here we consider linear costs ($\Phi(n_i) = \alpha n_i$). We consider the following dynamic discrete-time process. Starting from an empty network, at each time a player pair (i,j) is randomly chosen. Link (i, j) is created (or kept) if both players find it beneficial. An existing link is removed if at least one of the two players of that link does not find it useful. We are going to show that this dynamic process always reaches a PEN.

Let us introduce some terminology according to [2]. A network g' is *adjacent* to a network g if $g' = g + g_{i,j}$ or $g' = g - g_{i,j}$ for some pair (i, j). A network g' *defeats* another network g if either $g' = g - g_{i,j}$ and $B_i(g') > B_i(g)$, or $g' = g + g_{i,j}$ with $B_i(g') \geq B_i(g)$ and $B_j(g') \geq B_j(g)$ with at least one inequality holding strictly. A network game exhibits *no indifference* if for any two adjacent networks, one defeats the other.

According to this terminology in the dynamic process we described above, the current network is altered if and only if the addition or deletion of a link would defeat the current network. The process leads to an *improving path*, i.e. a sequence of networks $g_1, g_2, ..., g_K$ where each network g_k is defeated by the subsequent (adjacent) network g_{k+1}. There are two kind of improving paths: those exhibiting cycles (which have infinite length) and those terminating with a PEN (called *stable state*). The following lemma (a theorem in [15]) characterizes when there are no cycles and pairwise stable networks exist.

Lemma 1. *Given* G *the set of all the possible networks* g, *if there exists a real valued function* $w : G \to \mathbb{R}$ *such that "*g' *defeats* g*" if and only if "*$w(g') > w(g)$ *and* g' *and* g *are adjacent", then there are no cycles. Conversely, if the network game exhibits no indifference, then there are no cycles only if there exists a function* $w : G \to \mathbb{R}$ *such that "*g' *defeats* g*" if and only if "*$w(g') > w(g)$ *and* g' *and* g *are adjacent".*

Based on this lemma, we have the following result.

Theorem 7. *If the connection cost function is a linear function* $\Phi(n) = \alpha n$, *the dynamic process introduced in this section always reaches a PEN.*

Sketch of the proof. If $h \in \{0, 1, \cdots, R-1\}$ is the degree of a symmetric equilibrium according to Theorem 1 and $b(h, h) < 0$ for $h \neq R-1$, the following function $w : G \to \mathbb{R}$, $w(g) = -\sum_{i=1}^{R} f(n_i)$, where $f(n_i) = h - n_i$, if $h \geq n_i$, and $f(n_i) = R(n_i - h)$ otherwise, satisfies the relation in Lemma 1 for our overlay formation game, hence the dynamic process always reaches a PEN. If $h \neq R-1$ and $b(h, h) = 0$, it is possible to define another function satisfying the relation in Lemma 1. The details of the proof are in Appendix VIII in [10].

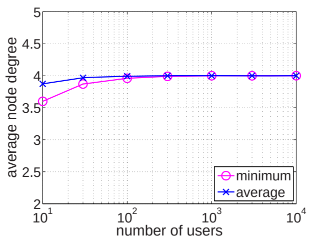


Fig. 2. Average node degree

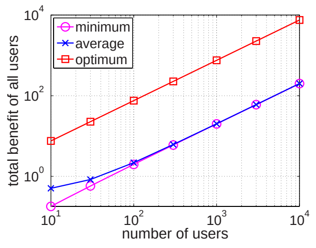


Fig. 3. Total benefit

5.1 Simulation Results

We present some simulation results. We considered a number of players ranging from 100 to 10000, having the same capacity, and $\alpha = 0.245$, for which the degree of a symmetric PEN is 4. For each setting we simulated 5000 runs of the above dynamic process. Each run terminates with a PEN. For this PEN we denote the average degree over all players as d_{avg}.

Figure 2 shows the minimum and the mean of d_{avg} over all the runs. We see that as R increases both the mean and the minimum converge to 4. This result confirms Theorem 3: as R increases the PENs *converge* to a symmetric one.

In Figure 3, the mean and the minimum of the total benefit are compared with the highest total benefit, which can be directly evaluated from the results in Appendix VII of [10]. This figure shows also the convergence of the payoffs of all PENs to the payoff of the symmetric PEN when R increases.

In addition, we present the number of iterations per peer in Figure 4. We observe that the average number of iterations to reach a PEN is of the order of R^2 and hence the number of iteration per peer is of the order of R. Let us consider this number of iterations in the context of BitTorrent (BT) [3]. Each peer in a BT network tries to replace an existing connection with a new, better connection every 10 seconds. All peers do such replacement simultaneously, unlike the sequential replacement in our simulations. So R^2 iterations in our simulations corresponds to $10R$ seconds in a BT network. For a population of 100 peers, the time needed to reach a PEN is of the order of 17 minutes, which is faster than the typical average time between changes in the population of peers (due to arrivals or departures). Figure 5 shows how the average and the

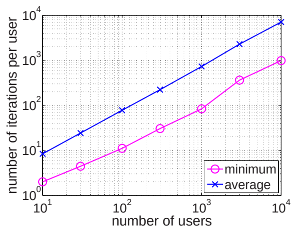


Fig. 4. Number of iterations per peer

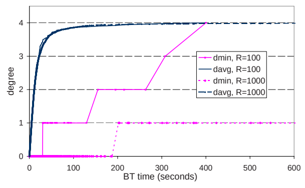


Fig. 5. Convergence to the PEN: the degree

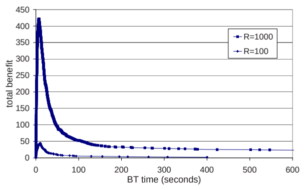


Fig. 6. Convergence to the PEN: the benefit

minimum degrees change during two simulation runs respectively for $R = 100$ and for $R = 1000$. The initial values are equal to 0 and converge to 4. The time scale represents time in a BT network; namely, R iterations are represented by 10s. We can observe that: 1) with this time scale the evolution of the average degree seems independent from the number of players; 2) the network converges quite rapidly to the PEN. In particular, the average degree reaches 3.8, i.e. 95% of the final value, after less than 80 seconds in both cases, or, equivalently, after less than 800 iterations for $R = 100$ and less than 8000 for $R = 1000$. Figure 6 shows the time evolution of the process as regards the total benefit. We note that for both runs, as the process begins the total benefit grows because of the high benefit of the initial connections, while it falls down to the expected value when the network approaches the equilibrium.

Finally, we considered heterogeneous scenarios where players have different upload capacities. The simulations show that at the final equilibrium almost all the players open a number of connections equal to that indicated by Theorem 5. For example we considered a network with 1000 players and $\alpha = 0.245$, where 50% of the nodes have capacity equal to 1, 30% have capacity equal to 2, and 20% have capacity equal to 4. The corresponding degrees in symmetric networks (k_i in Section 3.2) are respectively 4, 8, 16. In the PEN the fraction of nodes having a degree smaller than the corresponding k_i is on average equal to 0.15% of the total number of nodes, hence the degree distribution closely resembles the capacity distribution.

6 Conclusions

We studied the Tit-for-Tat strategy (built in BitTorrent [3]) through network formation games framework. We proved the existence of equilibrium overlays, and demonstrated the convergence of a simple game dynamics. Although the general belief is that the Tit-for-Tat can prevent selfish behavior, we showed that it can still lead to an unbounded loss of efficiency.

Acknowledgements

This research has been supported in part by Italian MIUR project Famous and by NSF under grant awards ANI-0085848, CNS-0519998, CNS-0519922, and

EIA-0080119. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. Qiu, D., Srikant, R.: Modeling and performance analysis of bittorrent-like peer-to-peer networks. In: Proc. of SIGCOMM '04. (2004) 367–378
2. Jackson, M.O.: A survey of models of network formation: Stability and efficiency. In: Group Formation in Economics: Networks, Clubs and Coalitions, New York, Cambridge University Press (2004)
3. Cohen, B.: Bittorrent, http://www.bittorrent.com
4. Wikipedia: http://en.wikipedia.org/wiki/peer-to-peer
5. CacheLogic: http://www.cachelogic.com
6. Zhang, H., Neglia, G., Towsley, D., LoPresti, G.: On Unstructured File Sharing Networks. In: Proc. of IEEE INFOCOM '07. (2007)
7. Fabrikant, A., Luthra, A., Maneva, E., Papadimitriou, C.H., Shenker, S.: On a network creation game. In: Proc. of the 22nd annual symposium on Principles of distributed computing, New York, NY, USA, ACM Press (2003) 347–351
8. Chun, B., Fonseca, R., Stoica, I., Kubiatowicz, J.: Characterizing selfishly constructed overlay networks. In: Proc. of IEEE INFOCOM'04, Hong Kong
9. Corbo, J., Parkes, D.C.: The price of selfish behavior in bilateral network formation. In: Proc. 24rd ACM Symp. on Principles of Distributed Computing (PODC'05), Las Vegas, Nevada, USA (2005) 99–107
10. Zhang, H., Neglia, G., Towsley, D., LoPresti, G.: On Unstructured File Sharing Networks. Technical Report 06-40, UMass (2006) `http://www-sop.inria.fr/maestro/personnel/Giovanni.Neglia/publications/Zhang06_UFS.pdf`.
11. Saroiu, S., Gummadi, P., Gribble, S.: A measurement study of peer-to-peer file sharing systems. In: Proceedings of Multimedia Computing and Networking. (2002)
12. Jackson, M., Wolinsky, A.: A strategic model of economic and social networks. Journal of Economic Theory **71**(1) (1996) 44–74
13. Goyal, S., Joshi, S.: Unequal connections. Forthcoming in International Journal of Game Theory (2006)
14. Roughgarden, T.: The price of anarchy is independent of the network topology. In: ACM Symposium on Theory of Computing. (2002)
15. Jackson, M., Watts, A.: The existence of pairwise stablenetworks. Seoul Journal of Economics **14(3)** (2001)

Fixed-Rate Equilibrium in Wireless Collision Channels

Ishai Menache and Nahum Shimkin

Department of Electrical Engineering
Technion, Israel Institute of Technology
Haifa 32000, Israel
{imenache@tx,shimkin@ee}.technion.ac.il

Abstract. We consider a collision channel, shared by a finite number of self-interested users with heterogenous throughput demands. It is assumed that each user transmits with a fixed probability at each time slot, and the transmission is successful if no other user transmits simultaneously. Each user is interested in adjusting its transmission rate so that its throughput demand is met. When throughput requirements are feasible, we show that there exist two equilibrium points where users satisfy their respective demands. In one equilibrium all users transmit at lower rates, compared to their transmission rates at the other equilibrium. This fact is meaningful in wireless systems, where lower transmission rates translate to power savings. Subsequently, we propose a distributed scheme that ensures convergence to the lower-rate equilibrium point. We also provide some lower bounds on the channel throughput that is obtained with self-interested users, both in the symmetric and non-symmetric case.

1 Introduction

1.1 Background and Motivation

As wireless networks become larger, it may be impractical to have a central authority (such as a base station) coordinate between wireless stations (which share the same communication medium) for better network utilization. Thus, random access, ALOHA-like protocols are often used (for example in the 802.11x standards). The incorporation of such protocols in wireless systems raises some novel challenges, as these protocols should consider additional wireless-specific items such as power control and varying channel conditions (an effect known as channel fading [1]). Hence, a major research challenge is to examine whether the distributed nature of random access protocols may lead the network to reasonable working points, thereby coping with additional complexity of wireless systems.

Since wireless nodes usually do not coordinate in establishing their transmission policies, non-cooperative game theory becomes a natural framework for analyzing their interaction. Game theoretic tools have been recently applied for analyzing selfish behavior of users (a "user" stands for a single node or station) in Aloha-like random access networks [2,3,4,5,6]. A common ground of most of these papers, is that users are identical (or symmetric), both in their physical parameters (such as the arrival rate of packets [4]) and also in their underlying objective (such as maximizing throughput [4], or minimizing the number of attempted transmissions before success [3]). In practice, however, network users are heterogenous in nature. For example, video, voice, ftp and e-mail applications, all have fundamentally diverse QoS requirements. A paper by Jin and Kesidis [2] does incorporate user heterogeneity, by studying an Aloha-like network with users who have fixed (and different) throughput demands. Users dynamically adapt their transmission rates in order to obtain their required throughput demands. It was shown by means of an example that the equilibrium point may not be unique. Additionally, the authors suggested a dynamic scheme that could lead to an equilibrium point.

1.2 Paper Organization and Contribution

In this paper we reconsider the model suggested in [2]. A description of the model is given in Section 2. A detailed equilibrium analysis (complementing missing analysis in [2]) is provided in Section 3. Our equilibrium analysis reveals, in particular, that when the rate requirements are within the sustainable region, there exist exactly two equilibrium points in the resulting game, and that one of them is strictly better than the other, in the sense that all users transmit at lower rates (in comparison to their transmission rates at the other equilibrium). This fact is meaningful in terms of power consumption in wireless systems. We also show that the equilibrium points can be computed in polynomial time. In the context of wireless systems, we examine in Section 4 how self-optimizing behavior affects the network performance. Specifically, we show that the performance gap (in terms of the total power consumption) between the equilibrium points is potentially unbounded, and that the better equilibrium point coincides with the socially-optimal operating point. In Section 5 we present a distributed algorithm which converges to the better equilibrium. Finally, we provide in Section 6 a simple lower bound on the channel throughput that can possibly be obtained with selfish users.

2 Model Description

We consider an ALOHA-like network, shared by a finite set of users $\mathcal{I} = \{1, \ldots, n\}$ who transmit data over a shared collision channel (e.g., wireless stations who transmit to a common base station). Time is slotted, in the sense that all transmitted packets have the same length and require the same time interval (a slot) for transmission. Moreover, all transmissions start at the beginning of a slot and end before the next slot. We assume that a transmission is successful only if no other user attempts transmission simultaneously.

Each user i is characterized by a throughput ρ_i (in packets per slot) which it wishes to deliver over the network. We assume that a user always has packets to send, yet it may postpone transmission, due to the following reasons. First, a user need not transmit in every slot when its average throughput is already met, since it unnecessarily wastes additional resources, such as transmission power. Second, assume a user transmits at every slot; then other users would raise their transmission rates as well, and as a consequence packets will endlessly collide. Hence, each user i chooses a transmission probability p_i, which could be regarded as the transmission rate.

The underlying assumption of our user model is that users are selfish and do not cooperate in any manner in order to obtain their required throughput demands. Define r_i as user i's average throughput. Then

$$r_i = p_i \prod_{j \neq i} (1 - p_j). \tag{1}$$

Note that the transmission probability of each user affects the throughput of all other users through the collision channel. This situation establishes a non-cooperative game [7] between the users. We are interested in the Nash equilibrium point of that game. In our context, a Nash equilibrium point is a vector of user probabilities $\mathbf{p} = (p_1, \ldots, p_n)$, such that

$$r_i = p_i \prod_{j \neq i} (1 - p_j) = \rho_i, \quad i \in \mathcal{I}. \tag{2}$$

We shall refer to the above set of equations (2) as the *equilibrium equations.*

3 Equilibrium Analysis

In this section we analyze the Nash equilibrium point (2) of the network. We start our analysis by considering the number of equilibria.

3.1 Two Equilibria or None

Obviously, if the users' throughput requirements are too high there would not be an equilibrium point, since the network naturally has limited capacity. In case that an equilibrium point does exist, we establish that, generically, there are exactly *two equilibria* (which can be computed efficiently). In addition, we assert that the existence of an equilibrium point could be verified through a computationally efficient procedure. The main result of this section is presented below.

Theorem 1. *Consider the non-cooperative game whose Nash equilibrium point is defined in (2). There are either one, two Nash equilibrium points or none for that game. The case of a single equilibrium point is non-generic (i.e., occurs only for a set of rate vectors ρ of measure zero).*

Proof. See Appendix.

We summarize certain computability properties in the next proposition (proof is omitted due to lack of space).

Proposition 1. *The existence of an equilibrium point can be verified in polynomial time (in the number of users). Additionally, in case an equilibrium exists, both equilibria can be computed in polynomial time (in the number of users).*

3.2 Efficiency and Fairness

Besides the existence and the number of equilibrium points, we wish to characterize the equilibrium points. In particular, we are interested in the following questions:

1. How do the two equilibrium points compare: is one "better" than the other?
2. Is an equilibrium point fair in some sense?

The next theorem addresses the first question raised above. It shows that one equilibrium point is better for all users in the sense that all users transmit at lower rates.

Theorem 2. *Assume there exist two equilibria for the non-cooperative game, whose Nash equilibrium is defined in (2). Let* $\mathbf{p}$ *and* $\tilde{\mathbf{p}}$ *be these two equilibrium points. If* $p_i < \tilde{p}_i$ *for some user* i*, then* $p_j < \tilde{p}_j$ *for every* $j \in \mathcal{I}$*.*

Proof. Define $a_{ik} \overset{\triangle}{=} \frac{\rho_i}{\rho_k}$. For every user $k \neq i$ divide the ith equation in the set (2) by the kth one. We obtain

$$a_{ik} = \frac{p_i(1-p_k)}{p_k(1-p_i)} < \frac{\tilde{p}_i(1-p_k)}{p_k(1-\tilde{p}_i)}. \tag{3}$$

Now since

$$\frac{\tilde{p}_i(1-\tilde{p}_k)}{\tilde{p}_k(1-\tilde{p}_i)} = a_{ik}, \tag{4}$$

it follows that

$$\frac{\tilde{p}_i(1-\tilde{p}_k)}{\tilde{p}_k(1-\tilde{p}_i)} < \frac{\tilde{p}_i(1-p_k)}{p_k(1-\tilde{p}_i)}. \tag{5}$$

We conclude from the last inequality that $p_k < \tilde{p}_k$. □

The last result is significant from the network point of view. Assuming that each transmission is costly (e.g., each transmission consumes a fixed power), we are interested in a network mechanism which will exclude the inferior equilibrium point. This would be our main concern in Section 5. Henceforth, we identify the better equilibrium point as the *Energy Efficient Equilibrium (EEE).*

We now compare the user effort in a given equilibrium point. Our next result suggests that at every equilibrium, the transmission probabilities are ordered in the same order as the throughput demands ρ_i, i.e., users with a larger demand transmit more aggressively.

Theorem 3. *Let* $\mathbf{p}$ *be an equilibrium point of (2). Then if* $\rho_i \geq \rho_j$ *it follows that* $p_i \geq p_j$*.*

Proof. Follows easily from eq. (4). Details are omitted.

The above result indicates that despite user selfishness, some notion of *fairness* is maintained at equilibrium: The higher the throughput requirement, the higher the transmission rate (and consequently, the higher is the power consumption).

4 Efficiency Loss

We now turn to examine the extent to which selfish behavior affects system performance. That it, we are interested in comparing the quality of the obtained equilibrium points to the theoretical case where a central authority can set the users' transmit policies. Recently, there has been much work in quantifying the "efficiency loss" incurred by the selfishness of users in networked systems (see [8] for a comprehensive review). The two concepts which are most commonly used in this context are the *"price of anarchy"*, which is the performance ratio (of a relevant social performance measure) between the global optimum and the *worst* Nash equilibrium, and *"price of stability"*, which is the performance ratio between the global optimum and the *best* Nash equilibrium.

We focus in this section on a wireless system, where W_i represents a (fixed) energy which user i utilizes per transmission. The average power of user i is then given by $J_i(p_i) \stackrel{\triangle}{=} p_i W_i$. A natural performance criterion for evaluating the quality of an equilibrium is given by $\sum_i J_i(p_i)$, which represents the total power consumption in the network. We next show that the price of anarchy with respect to this criterion is unbounded, while the price of stability is always one.

Theorem 4. *Consider the non-cooperative game whose Nash equilibrium point is defined in (2). Define $\sum_i J_i(p_i)$ as the social performance criterion. Then (i) the price of stability is always one, i.e., the better equilibrium point coincides with the social optimum, and (ii) the price of anarchy is generally unbounded.*

Proof. (i) immediate, as a social optimum obeys the equilibrium equations (2). (ii) we establish that the price of anarchy is unbounded by means of an example. Consider a network with two identical users with a throughput requirement of $\rho_i = \epsilon \to 0$ $(i = 1, 2)$, and an average power function given by $J(p) = Wp$, where p is the user's transmission probability (user indexes are omitted in this example, as users are identical). By symmetry, we obtain a single equilibrium equation, namely $p(1-p) = \epsilon$. As ϵ goes to zero, the two equilibria are $p_a \to 1$ and $p_b \to 0$. Obviously, the latter point is also a social optimum; it is readily seen that the price of anarchy equals at the limit to $\frac{2J(1)}{2J(0)} = \infty$.

5 A Distributed Algorithm

5.1 The Algorithm

We have shown so far that when an equilibrium point exists, there are two equilibria, where one obtains lower transmission rates for all users. Moreover, the performance gap between the equilibria could be significantly large. This leads us to find a mechanism which will converge to the better equilibrium. We next suggest a simple distributed algorithm for that purpose.

Let $x_i = \prod_{j \neq i}(1 - p_j)$ denote the current idle probability of all users but the ith one. Each user i iteratively updates its transmission probability through the following rule:

$$p_i := p_i + \epsilon_i \left(\frac{\rho_i}{x_i} - p_i \right), \tag{6}$$

where $0 < \epsilon_i \leq 1$ is the update gain of user i. The motivation for using the above rule follows directly from the equilibrium equations (2).

A synchronized version of the above algorithm (where all users iteratively update their transmission probabilities at the same slots) is considered in [9] pp. 347–349. It was shown there that when $\epsilon_i = 1$ for every $i \in \mathcal{I}$, the algorithm asymptotically converges to the better equilibrium point.

5.2 Practical Considerations and Stability

We pause here to address certain practical implementation issues which are related to the presented algorithm. The quantity x_i required for the probability updates can be obtained by each user through dividing the overall idle probability by its own idle probability. In practice, the channel's idle probability should be estimated by each user by measuring the percentage of idle slots, where the more slots sensed, the better is the estimate. If the transmission probabilities are to be updated on a slow time-scale, users would be able to obtain good estimates of the idle probability. For the sake of our analysis, we assume that the estimation is perfect. Indeed, the frequency of the updates is an important issue. There is obviously a tradeoff between the estimation accuracy of x_i and the wish to enforce the required equilibrium as fast as possible. We leave the quantification of this tradeoff to future research.

We now consider some stability properties of the algorithm. In particular, we wish to verify whether the better equilibrium point is locally stable. Indeed, even when the network operates near a stationary working point, users continuously adjust their probabilities according to (6) (e.g., due to perturbations in their idle estimation). We then have the following stability result.

Theorem 5. *Assume users update their transmission probabilities according to (6). When each ϵ_i is small enough, the EEE is locally stable.*

Proof. (outline) Let $\epsilon_i = \lambda_i \epsilon$, with ϵ-small. The continuous-time limit of (6) as $\epsilon \to 0$ is

$$\dot{p_i} = \lambda_i \left(\frac{\rho_i}{x_i} - p_i \right). \tag{7}$$

We next apply Lyapunov's indirect method (linearization) to study the stability of (7). Consider first the case where $\lambda_i = \lambda_j$ for every $i, j \in \mathcal{I}$. Let Z be the corresponding Jacobian matrix of (7) for this symmetric case. It can be shown that the elements of Z are given by

$$z_{ij} = \begin{cases} -1 & i = j \\ \frac{p_i}{1-p_j} & i \neq j \end{cases}. \tag{8}$$

It can be further shown that the matrix Z is strictly diagonally dominant (see, e.g., [10]) for the better equilibrium point. As such, all eigenvalues of Z have a negative real part ([10], pp. 349). Consider now a general asymmetric case. Let $\lambda = diag(\lambda_1, \ldots, \lambda_n)$. Then the corresponding Jacobian matrix of (7) is given by λZ. It is known that if all eigenvalues of Z have a negative real part, then the same property holds for RZ, where R is any positive diagonal matrix (see [11], pp. 112–121). Consequently, we may conclude that the equilibrium point is asymptotically stable. The local stability of the discrete-time model follows now by continuity argument. $\square$

6 Achievable Throughput

The aim of this section is to provide a lower bound for the maximal throughput which can be obtained in the network.

The theorem below establishes the conditions for the existence of an equilibrium point in the symmetric case.

Theorem 6 (Symmetric users). *Let $\rho_i = \rho$ for every $1 \leq i \leq n$. Then an equilibrium exists if and only if*

$$\sum_{i=1}^{n} \rho_i = n\rho \leq (1 - \frac{1}{n})^{n-1}. \tag{9}$$

Proof. In every equilibrium of the symmetric case $p_i = p_j = p$, for every i, j (immediate by Theorem 3). Thus, the equilibrium equations (2) diminish into a single (scalar) equation:

$$h(p) \overset{\triangle}{=} p(1-p)^{n-1} = \rho. \tag{10}$$

We next investigate the function $h(p)$. The derivative of $h(p)$ is given below.

$$h'(p) = (1-p)^{n-2}\big(1 - p - (n-1)p\big) = (1-p)^{n-2}(1 - np). \tag{11}$$

It can be seen that the maximum value of the function $h(p)$ is obtained at $p = 1/n$. An equilibrium exists if and only if the maximizer of the function obtains a value which is greater than ρ. We assign the maximizer $p = 1/n$ of $h(p)$ in (10) and the result immediately follows. $\square$

The next corollary provides a sufficient condition for the existence of an equilibrium for any number of symmetric users.

Corollary 1. *Let $\rho_i = \rho$ for every $1 \leq i \leq n$. Then an equilibrium exists if $\sum_{i=1}^{n} \rho_i \leq e^{-1}$.*

Proof. Observe that the left hand side of (9) is the total throughput demand. It may be easily verified that the right hand side of (9) decreases with n. Since $\lim_{n\to\infty}(1 - \frac{1}{n})^{n-1} = e^{-1}$, a total throughput demand which is less or equal than this quantity guarantees the existence of an equilibrium. $\square$

It can be shown that the simple bound obtained above holds for non-symmetric users as well, implying that the symmetric case is worst in terms of system utilization. Our result is summarized below. Due to lack of space the (somewhat lengthy) proof is omitted.

Theorem 7 (Asymmetric users). *For any set of n users, an equilibrium point exists if*

$$\sum_{i=1}^{n} \rho_i \leq (1 - \frac{1}{n})^{n-1}. \tag{12}$$

The quantity e^{-1} is also the well-known maximal throughput of a slotted Aloha system with Poisson arrivals and an infinite set of nodes [9]. In our context, if the throughput requirements do not exceed e^{-1}, an equilibrium point is guaranteed to exist. Thus, in a sense, we may conclude that user heterogeneity does not deteriorate the capacity (i.e., the maximal throughput) of the collision channel.

7 Conclusions and Model Extensions

We have investigated in this paper the interaction between heterogenous users, who adjust their transmission rates in order to obtain their individual throughput demands. We established that the network possesses either two Nash equilibria or none. In case that two equilibria exist, in one of the equilibria *all* users transmit at lower rates compared to the transmission rates at the other equilibrium. Translating this fact to power-related terms (which are relevant in wireless systems), we further demonstrated that the performance gap between the two equilibria (in terms of power consumption) could be arbitrarily large. Consequently, network users should be willing to accept a mechanism which ensures convergence to the better equilibrium. We have suggested such a mechanism, and studied some stability properties thereof.

In an ongoing work, we consider a more general scenario of a block fading channel [1], where the channel state of each user varies over time, and affects the data rate. Users, who measure their channel state, base the transmission decision at each time slot on the current measurement of their channel state. Our analysis so far indicates that many of the results of the present paper carry over to this general setup.

Several additional directions remain for future research. Among which are: (i) Further analyzing asynchronous versions of the distributed algorithm. (ii) Incorporating the transmit power as an additional decision variable (as in [5,6]). In some models of practical interest, the transmission power not only affects the data rate, but also the reception chances of the packet. In this context, we plan to include capture models (which sometimes better represent WLAN systems) and multi-packet reception models [12] (as in CDMA systems) in our future work.

References

1. E. Biglieri, J. Proakis, and S. Shamai. Fading channels: Information-theoretic and communications aspects. *IEEE Transactions on Information Theory*, 44(6): 2619–2692, 1998.
2. Y. Jin and G. Kesidis. Equilibria of a noncooperative game for heterogeneous users of an ALOHA network. *IEEE Comm. Letters*, 6(7):282–284, 2002.

3. B. MacKenzie and S. B. Wicker. Stability of slotted aloha with multipacket reception and selfish users. In *Proceedings of INFOCOM*, 2003.
4. E. Altman, R. El-Azouzi, and T. Jimenez. Slotted aloha as a game with partial information. *Computer Networks*, 45(6):701–713, 2004.
5. E. Altman, N. Bonneau, M. Debbah, and G. Caire. An evolutionary game perspective to aloha with power control. In *19th International Teletraffic Congress*, 2005.
6. J. Sun and E. Modiano. Opportunistic power allocation for fading channels with non-cooperative users and random access. In *IEEE BroadNets Wireless Networking Symposium*, 2005.
7. D. Fudenberg and J. Tirole. *Game Theory.* MIT Press, 1991.
8. T. Roughgarden. *Selfish Routing and the Price of Anarchy.* MIT Press, 2005.
9. D. Bertsekas and R. Gallager. *Data Networks.* Prentice-Hall, 1992.
10. R. Horn and C. R. Johnson. *Matrix Analysis.* Cambridge University Press, 1985.
11. R. Horn and C. R. Johnson. *Topics in Matrix Analysis.* Cambridge University Press, 1991.
12. S. Ghez, S. Verdu, and S. C. Schwartz. Stability properties of slotted aloha with multipacket reception capability. *IEEE Transactions on Automatic Control*, 33(7):640–649, 1988.

Appendix – Proof of Theorem 1

We start with the following lemma, which relates the user probabilities in equilibrium.

Lemma 1. *In every equilibrium point the following relation holds for every $i, j \in \mathcal{I}$.*

$$p_j = \frac{a_{ji} p_i}{1 - p_i + a_{ji} p_i}, \tag{13}$$

where $a_{ji} \overset{\triangle}{=} \rho_j / \rho_i$.

Proof. Immediate by dividing the equilibrium equation of the ith user by the equation of the jth one. □

The idea behind the proof of the theorem is to represent the equilibrium conditions through a single scalar equation. The result then follows by showing concavity of this equation.

The equilibrium point $\mathbf{p}$ (if exists) is by (2) such that

$$p_i = \rho_i / \prod_{j \neq i} (1 - p_j), \quad 1 \leq i \leq n. \tag{14}$$

Fixing the ith user and substituting (13) into (14) for every $j \neq i$ we obtain the following equation in p_i only.

$$p_i \prod_{j \neq i} \left(1 - \frac{a_{ji} p_i}{1 - p_i + a_{ji} p_i} \right) = p_i \prod_{j \neq i} \left(\frac{1 - p_i}{1 - p_i + a_{ji} p_i} \right) = \rho_i. \tag{15}$$

Taking log from both sides we have

$$\log p_i + \sum_{j\neq i} \log\left(\frac{1-p_i}{1-p_i+a_{ji}p_i}\right) = \log \rho_i. \tag{16}$$

Let

$$g(p_i) \triangleq \log p_i + \sum_{j\neq i} \log\left(\frac{1-p_i}{1-p_i+a_{ji}p_i}\right). \tag{17}$$

We next investigate the properties of the function $g(p_i)$. Specifically, we concentrate on the user with the maximal demand. Without loss of generality, let user i be the one with the maximal demand, i.e., $\rho_i \geq \rho_j$ for every $j \neq i$. Thus, $a_{ji} \leq 1$. The derivative of $g(p_i)$ is calculated below.

$$\begin{aligned} g'(p_i) &= \frac{1}{p_i} + \sum_{j\neq i} \frac{1-p_i+a_{ji}p_i}{1-p_i} \cdot \frac{-(1-p_i+a_{ji}p_i)+(1-a_{ji})(1-p_i)}{(1-p_i+a_{ji}p_i)^2} \\ &= \frac{1}{p_i} - \sum_{j\neq i} \frac{a_{ji}}{(1-p_i)(1-p_i+a_{ji}p_i)}. \end{aligned} \tag{18}$$

We focus our analysis on $p_i \in [0,1]$. Observe that $g(0) = g(1) = -\infty$. Additionally, $g'(0) = \infty$, $g'(1) = -\infty$. Hence, if the derivative $g'(p_i)$ is monotonously decreasing in $p_i \in [0,1]$, then there are either two roots (which may coincide) or none for the equation (16). Indeed, deriving $g'(p_i)$ yields

$$g''(p_i) = -\frac{1}{p_i^2} + \sum_{j\neq i} \frac{a_{ji}\big[(1-p_i)(-1+a_{ji}) + (-1+p_i-a_{ji}p_i)\big]}{\big((1-p_i)(1-p_i+a_{ji}p_i)\big)^2}. \tag{19}$$

Noting that $a_{ji} \leq 1$, it may be easily verified that $(1-p_i)(-1+a_{ji}) < 0$ for $p_i \in [0,1)$; additionally $(-1+p_i-a_{ji}p_i) = -1+p_i(1-a_{ji}) < 0$ for $p_i \in [0,1)$, so overall $g''(p_i) < 0$ for $p_i \in [0,1]$. We conclude that there are either two roots or none for the equation (16). Hence, there are either two equilibrium points or none for the game. There is a single equilibrium point in the non-generic case where the maximum of $g(p_i)$ equals $\log \rho_i$. $\square$

A Survey of Uniqueness Results for Selfish Routing

Nahum Shimkin

Department of Electrical Engineering
Technion, Israel Institute of Technology
Haifa 32000, Israel
shimkin@ee.technion.ac.il

Abstract. We consider the problem of selfish or competitive routing over a network with flow-dependent costs which is shared by a finite number of users, each wishing to minimize the total cost of its own flow. The Nash Equilibrium is well known to exist for this problem under mild convexity assumptions on the cost function of each user. However, uniqueness requires further conditions, either on the user cost functions or on the network topology. We briefly survey here existing results that pertain to the uniqueness issue. We further consider the mixed Nash-Wardrop problem and propose a common framework that allows a unified treatment of this problem.

1 Introduction

The selfish routing problem involves a number of non-cooperative users, or players, each wishing to ship a certain amount of flow over a shared network, where link costs are flow dependent. A user can choose which route (or routes) to use in order to minimize the total cost of its own flow. This gives rise to a non-cooperative game, with the associated Nash equilibrium as the central solution concept.

Selfish routing was first considered by Wardrop [28] in the context of transportation networks. This paper introduced the notion of shortest-path equilibrium, or Wardrop equilibrium, where only minimal-cost pathes are used between each origin-destination pair. This may be view as the Nash equilibrium of a game between a continuum of infinitesimal users. Recent overviews of the extensive literature that concerns the Wardrop equilibrium and its variants may be found in [3,20,22,25].

The finite-user version of the selfish routing model was introduced in the literature more recently, motivated in part by the non-centralized view of communication networks. The paper [11] shows convergence of the Nash equilibrium (for symmetric users) to the Wardrop equilibrium as the number of users increases to infinity. Existence, uniqueness and some basic properties of the Nash equilibrium are studied in [2,4,21,23]. The notion of a mixed Nash-Wardrop equilibrium, which combines infinitesimal users with positively-sized ones, is considered in [7,10]. Efficient network design and management are considered in

[9,13,14,15,16], while [26] bounds the performance degradation relative to centralized routing (along with similar results for the Wardrop equilibrium). The convergence of some dynamic schemes to the Nash equilibrium is considered in [12], while [17] considers a repeated game version of the routing problem, and [5] considers the addition of side-constraints on link flows.

Our focus here in on the question of uniqueness of the Nash equilibrium in selfish routing. Besides its theoretical interest, uniqueness is of obvious importance for predicting network behavior in equilibrium. From the computational aspect, efficient procedures that find *all* Nash equilibria are virtually non-existent when the equilibrium is non-unique. Uniqueness is also of particular importance for network management, where regulating the user behavior in a single equilibrium (using pricing, for example) is usually much easier than for several equilibria simultaneously.

Uniqueness is well-known to hold for the basic (single-class) Wardrop equilibrium, assuming only that the link costs are strictly increasing in the link flow. In that case the Wardrop equilibrium has been shown in [6] to be equivalent to convex optimization problem, and hence is unique. However, this simple cost monotonicity requirement no longer suffices for the the finite-user Nash equilibrium, as shown through simple counter-examples, nor for the multi-class Wardrop equilibrium problem (where link costs depend on the user class). Therefore, additional conditions are required to guarantee uniqueness in these cases. Existing conditions may be roughly divided into two types: conditions on link cost functions on the one hand, and conditions on the network topology on the other. In this paper we provide a brief survey of these uniqueness results, focusing on the finite-user model. We will also show that the multi-class Wardrop equilibrium may be embedded within the finite-user problem, and outline a general framework that handles the joint Nash-Wardrop problem in a unified manner.

2 The Game Model

Consider a network which is defined by a directed graph $\mathcal{G} = \mathcal{G}(V, L)$, where V is a finite set of vertices (or nodes) and $L \subset V \times V$ is a set of edges or links. This network is shared by a finite set $I = \{1, 2, \ldots, n\}$ of users, where each user i needs to deliver a given positive amount d^i of flow from its source node O_i to its destination node D_i, and may divide its flow between the set of paths π_i that connect these nodes. Denote by f_l^i the flow of user i on link l, and let $f_l = \sum_{i \in I} f_l^i$ denote the total flow on link l. Furthermore, $\mathbf{f}_l = (f_l^i)_{i \in I}$ is the flow vector over link l, $\mathbf{f}^i = (f_l^i)_{l \in L}$ is user i's flow profile, and $\mathbf{f} = (\mathbf{f}^i)_{i \in I}$ denotes the system flow profile.

The flow profile of each user is subject to the standard positivity and conservation requirements. That is, $f_l^i \geq 0$, and the sum of flows at each node (including external incoming or leaving flow) is null. We denote the set of feasible flow profiles $\mathbf{f}^i$ for user i by F^i, which is clearly a closed, convex polyhedron, and by F the set of feasible system profiles.

Let $J^i(\mathbf{f})$ denote the cost function for user i. We consider additive costs of the form

$$J^i(\mathbf{f}) = \sum_{l \in L} J^i_l(\mathbf{f}_l)\,. \tag{1}$$

Thus, the cost for each user is the sum of its link costs, and the cost of any given link depends only on the flow vector on that link. We further impose here the following assumptions:

Assumption A1: $J^i_l(\mathbf{f}_l) = f^i_l T^i_l(f_l)$.

Assumption A2: The function T^i_l takes values in $[0, \infty]$, and is continuously differentiable, strictly increasing (where finite), and convex.

$T^i_l(f_l)$ is the cost per unit flow for user i on link l. Note that the per-unit costs may differ between users; this may arise, for example, due to user-dependent pricing. A simple consequence of these assumptions is that the link cost function $J^i_l(\mathbf{f}^l)$ is strictly convex in f^i_l, hence the user cost $J^i(\mathbf{f})$ is convex in $\mathbf{f}_l$.

We note that more general cost functions of the form $J^i_l(\mathbf{f}_l) = J^i_l(f^i_l, f_l)$ have been considered in [21] and subsequent literature. However, in this review we will focus on the above-mentioned case, which is of most practical interest.

A cost function that is often used in the context of communication networks is the M/M/1 delay functions, namely $T^i_l(f_l) = \frac{1}{C_l - f_l}$ for $f_l < C_l$, and $T^i_l = \infty$ for $f_l \geq C_l$, where C_l is the link capacity. Note that the above assumptions allows the per-unit costs to assume infinite values, as long as the increase to infinity is continuous.

A flow profile $\hat{\mathbf{f}}$ is a Nash equilibrium point (NEP) if each user's flow profile is a best-response against the combined flows of the others. That is, for each $i \in I$,

$$J^i(\hat{\mathbf{f}}) = \min_{\mathbf{f}^i \in F^i} J^i(\hat{\mathbf{f}}^1, \ldots, \hat{\mathbf{f}}^{i-1}, \mathbf{f}^i, \hat{\mathbf{f}}^{i+1}, \ldots, \hat{\mathbf{f}}^I). \tag{2}$$

A simple consequence of Assumptions A1-A2 above is that the link cost function $J^i_l(\mathbf{f}^l)$ is strictly convex in f^i_l, hence the user cost $J^i(\mathbf{f})$ is strictly convex in $\mathbf{f}_l$. If follows that the above model is a convex game, and existence of the NEP essentially follows from classical results [8,24]. As the best-response minimization problem faced by each user is a convex program, its solution is unique (whenever finite). However, as is well known, uniqueness of the best response does not guarantee uniqueness of the equilibrium point.

When cost functions take infinite values, some care is needed in distinguishing finite-cost equilibria from infinite-cost ones, where at least one user does not have a finite-cost response to the flow of the others. To exclude existence of infinite-cost equilibria some additional assumptions are required. An fairly straightforward one is the following:

Assumption A3: For any flow configuration $\mathbf{f}$ which involves infinite costs, at least one user whose cost is infinite can modify its flow configuration to obtain a finite cost.

Irrespectively of Assumption A3, our discussion will henceforth focuse on finite-cost equilibria and their uniqueness.

Nonuniqueness: A first counterexample to the uniqueness of the NEP under reasonable convexity assumptions was given in [21], using a two-user four-node network. The user cost functions were not however given in the form of Assumption A1. Counterexamples with cost functions that do comply with A1-A2 are given in [23] for the networks shown the Figure 2 (we return to these networks in Section 5). In all these examples non-uniqueness is essential, in the sense that the user costs are different in the two equilibria.

Elastic demand: The model considered in this paper assumes that flow demands are fixed. Elastic demand can be incorporated into this model by eliminating the demand constraint and subtracting a flow utility term $U^i(d^i)$ from the cost function (1). The utility function is usually assumed to be convex increasing in the flow, which maintains the convexity of the overall cost. One approach to treat the elastic-demand case is to reduce it to the fixed demand model by adding a dedicated link for each user that absorbs its excess flow, with cost that represents the flow utility. A direct proof of uniqueness for the parallel link network may found in [1] and [18].

3 Cost Function Conditions

A general tool for establishing uniqueness of the NEP in convex games is the notion of Diagonal Strict Convexity (DSC) introduced in [24]. This condition may be applied to the network routing problem to obtain per-link sufficient conditions. It then remains to determine what classes of link cost functions satisfy this property.

Let $g^i(\mathbf{f}) = \frac{\partial J^i(\mathbf{f})}{\partial \mathbf{f}^i}$ denote gradient of user i's cost with respect to its flow vector, and for a fixed vector $\rho \in \mathbb{R}^n$ let $g(\mathbf{f}, \rho) = (\rho^i g^i(\mathbf{f}^i))_{i=1}^n$ (arranged as a row vector). Then the cost functions $\{J^i\}$ satisfy the DSC property if $g(\mathbf{f}, \rho)$ is strictly increasing in $\mathbf{f}$ for some positive vector ρ. That is $\rho^i > 0$, and

$$(g(\hat{\mathbf{f}}, \rho) - g(\mathbf{f}, \rho)) \cdot (\hat{\mathbf{f}} - \mathbf{f}) > 0 \quad \text{for all nonequal } \mathbf{f}, \hat{\mathbf{f}} \in F\,. \tag{3}$$

As established in [24], the DSC property implies uniqueness of the equilibrium in the routing game.

The DSC property (3) may be written in scalar notation as

$$\sum_{l\in L} \sum_{i\in I} \rho_i (g_l^i(\hat{\mathbf{f}}_l) - g_l^i(\mathbf{f}_l))(\hat{f}_l^i - f_l^i) > 0 \tag{4}$$

It is now clear that a *sufficient* condition for the DSC property to hold for the overall game is that a DSC-like property holds for each link separately, but with a common weight vector ρ. We summarize this as follows.

Theorem 1. *Suppose there exist numbers $\rho_i > 0$ so that, for each link l,*

$$\sum_{i\in I} \rho_i (g_l^i(\hat{\mathbf{f}}_l) - g_l^i(\mathbf{f}_l))(\hat{f}_l^i - f_l^i) > 0 \tag{5}$$

for any pair of feasible link flows $\hat{\mathbf{f}}_l \neq \mathbf{f}_l$. Then the NEP is unique.

A second-order sufficient condition given in [24] for the DSC property (3) is that the Jacobian matrix G of $g(\mathbf{f}, \rho)$ with respect to $\mathbf{f}$ be positive definite ($G + G^T > 0$) for every feasible $\mathbf{f}$. Applying this condition on to the last result leads to the following result.

Corollary 1 ([21]). *Suppose that matrix $G_l(\mathbf{f}_l, \rho)$ is positive definite for each link l, where*

$$G_l(\mathbf{f}_l, \rho) = \left\{ \rho_i \frac{\partial^2 J_l^i(\mathbf{f}_l)}{\partial f_l^i \partial f_l^j} \right\}_{i,j \in I} .$$

Then the condition of the last theorem holds, and the NEP is unique.

A couple of simple examples from [21] will be useful for illustrating the nature of these conditions, and in particular the effect of the system load and cost function steepness.

Example 1. Assume two users, $I = a, b$, and consider a link l with capacity C_l and M/M/1 costs: $T_l^i = 1/(C_l - f_l)$, where $f_l = f_l^a + f_l^b$. Then for $f_l < C_l$,

$$G_l(\mathbf{f}_l, \rho) = \frac{1}{(C_l - f_l)^3} \begin{pmatrix} 2\rho_a(C_l - f_l^b) & \rho_l(C_l + f_l^a - f_l^b) \\ \rho_b(C_l + f_l^b - f_l^a) & 2\rho_b(C_l - f_l^a) \end{pmatrix}$$

Assume that the total flows are in a rectangle which is bounded away from the link capacity, namely $f_l^a \le r^a$, $f_l^b \le r^b$ where $r^a + r^b < C_l$. It may be easily verified that the DSC condition on the matrix G_l holds with $\rho^a = r^b$ and $\rho^b = r^a$. However, this is not the case under the alternative constraint $f_l^a + f_l^b < C_l$. Indeed, for any fixed vector ρ the matrix $G_l(\mathbf{f}_l, \rho)$ is not positive definite if f_l^a or f_l^b is close enough to C_l. Evidently, the condition in the Corollary is satisfied in lightly loaded networks (flow requirements $d^a + d^b < C_l$ for each link), but not when the feasible total flow on some link exceeds the capacity.

Example 2. Let $T_l^i = P(f_l)$, $i = a, b$, where P is a monic polynomial with degree $m \ge 1$. Then it may be verified that, with $\rho = (1, 1)$, $G_l(\mathbf{f}_l)$ is positive definite over the entire positive quadrant if $m \le 7$, but not if $m \ge 8$. Thus, DSC is implied here if the cost function is "not too steep".

The next result was established for polynomial-like cost functions of the form

$$T_l^i(f) = a_l f^{p_l} + b_l \,. \tag{6}$$

Such costs find application in the context of road traffic. Let $p^* = \frac{3n-1}{n-1}$, where n is the number of users. Note that $p^* > 3$ for any n.

Theorem 2 ([2]). *Assume the per-link costs (6), with $a_l > 0$ and $0 < p_l < p^*$ for all l. Then the NEP is unique.*

The proof proceeds by demonstrating positive definiteness of the (n by n) matrix $G_l(\mathbf{f}_l, \rho)$, with $\rho_i \equiv 1$.

4 Symmetric Users

A general uniqueness result holds for the case of symmetric users, namely when all users have identical origin-destination pairs, flow demands and link cost functions. In that case the NEP is unique, and indeed turns out to be symmetric (namely with identical link flows) [21]. The proof is by direct analysis, which uses the first-order optimality conditions to show that non-symmetric flows lead to a contradiction.

5 Topological Conditions

Given that uniqueness does not always hold under Assumptions A1-A2 in networks of general topology, the question arises as to whether there exist restricted network topologies for which this general uniqueness property holds. This question was answered in the affirmative in [21] for parallel-link networks. In a recent work, Milchtaich [19] characterized all two-terminal network topologies for which this property holds for the multi-class Wardrop equilibrium. This result was extended in [23] to the finite user model, as described below. We start by defining the following basic property.

Definition 1. *A network $\mathcal{G}$ has the* topological uniqueness property *if the NEP is unique for any routing game over $\mathcal{G}$ that satisfies Assuptions A1-A2.*

The discussion in this section will be focused on *two-terminal networks*, where the source and the destination of all users are the same. The simplest network topology of interest is that of a parallel network: In this case the network has only two nodes, with one serving as the origin node for all users and the other as the destination node. As mentioned, it was shown in [21] that a parallel-link network has the topological uniqueness property.

We proceed to define *nearly parallel networks*, following [19]. As shown there, undirected two-terminal network topologies can be classified into one of two classes. The class of nearly parallel networks essentially contains the networks shown in Figure 1, as well as serial connections of those networks. The complementary class contains all networks in which one of the basic networks shown in Figure 2 is embedded, in the following sense.

Definition 2. *A network $\mathcal{G}'$ is said to be* embedded in the wide sense *in network $\mathcal{G}''$ if $\mathcal{G}''$ can be obtained from $\mathcal{G}'$ by some sequence of the following three operations:*

1. Edge subdivision: *An edge is replaced by two edges with a single common end vertex.*
2. Edge addition: *The addition of a new edge joining two existing vertices.*
3. Terminal vertex subdivision: *The addition of a new edge, joining the terminal vertex O or D with a new vertex v, such that a nonempty subset of the edges originally incident with the terminal vertex are incident with v instead.*

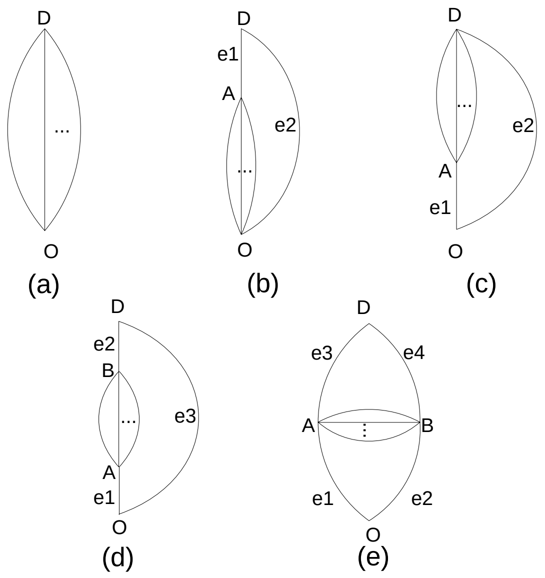


Fig. 1. Basic networks that define the class of nearly-parallel networks

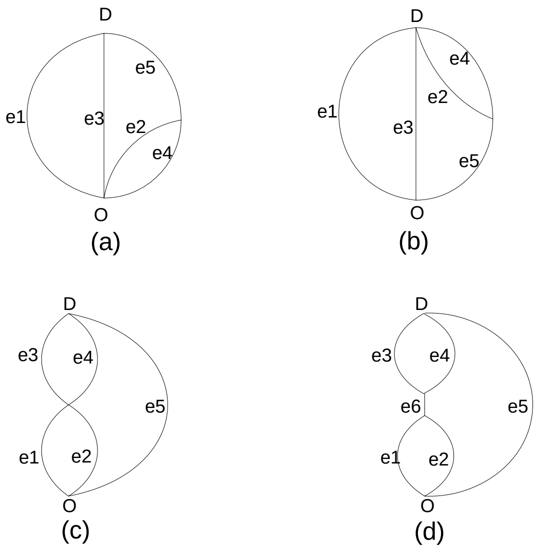


Fig. 2. Basic networks that are not nearly-parallel

Definition 3. *A two-terminal network $\mathcal{G}$ is called* nearly parallel *if it is one of the networks in Figure 1, or can be constructed from one of the networks in Figure 1 by a series of edge subdivisions.*

Theorem 3 ([19]). *For every two-terminal network $\mathcal{G}$, one, and only one, of the following conditions holds: (i) $\mathcal{G}$ is nearly parallel, or is a serial connection of two or more nearly parallel networks. (ii) One (or more) of the networks in Figure 2 is embedded in the wide sense in $\mathcal{G}$.*

The actual (directional) network model is obtained from the non-directional one by replacing each edge with two directional links, one in each direction. Of the five networks in Figure 1, only network (e) supports meaningful bidirectional traffic between some pair of nodes (namely, on the parallel-link network between nodes A and B) given the indicated origin and destination nodes. Indeed, network (e) is the most general of the five, as the other four may be considered a special case of this network for routing purposes. Still, the formal definition of nearly parallel networks does require all these basic networks.

The following result states that topological uniqueness extends to the class of nearly parallel networks, and *only* to that class.

Theorem 4 ([23]). *A two-terminal network $\mathcal{G}$ has the topological uniqueness property if, and only if, $\mathcal{G}$ is a nearly parallel network or is a series connection of such networks.*

The proof of sufficiency uses specific arguments related to monotonicity properties of the marginal link costs. The proof of necessity proceeds by providing a (three-user) counterexample to uniqueness with cost functions that satisfy A1–A2 for each of the networks shown in Figure 2, and then showing that these basic examples can be extended to any network that is not nearly parallel by using the embedding property in Theorem 3(ii).

6 Mixed Nash-Wardrop Routing

Recall that the Wardrop equilibrium may be considered as the limit of the Nash routing problem, where the user size is infinitesimal. A natural extension to the model is to consider jointly both large (atomic) users and a continuum of infinitesimal users that share the same network, to which we refer as the mixed Nash-Wardrop model [7,10]. As in the multi-class Wardrop model, we assume that infinitesimal users belong to a (finite) number of user classes, distinguished by their cost functions.

While the equilibrium conditions for atomic and infinitesimal user classes are defined from different perspectives, they actually share common properties and a unified treatment of these two types of users is desirable. In [23] two different approaches for unified treatment are presented, and used in particular to obtain proper extensions of the above topological uniqueness properties to the mixed model. Due to space limitations we do not provide details here. In broad terms, the two proposed approaches are:

1. Reduction to a finite user Nash model: Here each service class is transformed to a single atomic user with an appropriate cost function. This may

be considered a multi-class extension of the well known representation of the single-class Wardrop equilibrium as a (convex) optimization problem.

2. A continuum-game model: Here the framework of non-atomic games [27] is used to model small users. Thus, each user (large or small) is explicitly modelled as a rational decision maker with an individual cost function. This is in contrast to the usual definition of the Wardrop equilibrium, which specifies the behavior of small-user classes via an aggregate flow condition. As opposed to the previous approach, the model allows for a *continuum* of infinitesimal-user classes alongside the discrete population of large users.

In either case, the cost functions obtained for the infinitesimal users or infinitesimal user classes satisfy somewhat weaker conditions than Assumptions A1–A2 and their natural extensions. Still, these conditions do allow to obtain unified uniqueness results for this model, which recover the known topological uniqueness results for both the Nash and Wardrop equilibrium.

7 Conclusion

While new grounds have been gained recently in the analysis of the uniqueness issue in selfish routing, it appears that much remains to be done. Without further conditions on the cost functions, uniqueness results are limited to a fairly restricted class of network topologies. On the other hand, the sufficient conditions that have been explored so far based on diagonal convexity are link-based and do not bring the network topology into play at all. One may hope to find a middle ground that combines cost function properties with other network and user characteristics. This remains a challenging direction for further research.

Acknowledgment. This research was supported by Grant No. 2002112 from the United States–Israel Binational Science Foundation (BSF).

References

1. E. Altman, R. El Azouzi, T. Basar, and R. Srikant, *Combined competitive flow control and routing games*, Workshop on Networking Games and Resource Allocation, Petrozavodsk, July 2002.
2. E. Altman, T. Başar, T. Jiménez, and N. Shimkin, *Competitive routing in networks with polynomial cost*, IEEE Trans. Automat. Contr. **47** (2002), 92–96.
3. E. Altman, T. Boulogne, R. El Azouzi, T. Jiménez, and L. Wynter, *A survey on networking games in telecommunications*, Comput. Oper. Res. **33** (2005), no. 2, 286–311.
4. E. Altman and H. Kameda, *Equilibria for multiclass routing in multi-agent networks*, 2001, In Proc. 40th IEEE Conf. on Decision and Control, Orlando, Dec. 2001, pp. 604–609.
5. R. Azouzi and E. Altman, *Constrained traffic equilibrium in routing*, IEEE Trans. Automat. Contr. **48** (2003), no. 9, 1656–1660.
6. M. Beckmann, C. B. McGuire, and C. B. Winsten, *Studies in the economics of transportation*, Yale University Press, New Haven, 1956.

7. T. Boulogne, E. Altman, H. Kameda, and O. Pourtallier, *Mixed equilibrium (ME) for multiclass routing games*, IEEE Trans. Automat. Contr. **47** (2002), no. 6, 903–916.
8. G. Debreu, *A social equilibrium existence theorem*, Proc. Natl. Acad. Sci. **38** (1952), 886–893.
9. R. El Azouzi E. Altman and O. Pourtallier, *Avoiding paradoxes in multi-agent competitive routing*, Computer Networks **43** (2003), 133–146.
10. P. Harker, *Multiple equilibrium behaviors on networks*, Transportation Sci. **22** (1988), 39–46.
11. A. Haurie and P. Marcotte, *On the relationship between Nash-Cournot and Wardrop equilibria*, Networks **15** (1985), 295–308.
12. T. Jiménez, E. Altman, T. Başar, and N. Shimkin, *Routing into two parallel links: game-theoretic distributed algorithms*, J. Parallel Distrib. Comput. (Special Issue on Routing in Computer and Communication Systems) **61** (2001), 1367–1381.
13. Y. Korilis, A. Lazar, and A. Orda, *Architecting noncooperative networks*, IEEE J. Selected Areas in Communication **13** (1995), no. 7, 1241–1251.
14. ———, *Capacity allocation under noncooperative routing*, IEEE Trans. Automat. Contr. **42** (1997), no. 3, 309–325.
15. ———, *Avoiding the braess paradox in non-cooperative networks*, J. Appl. Probab. **36** (1999), 211–222.
16. Y. A. Korilis, T. A. Varvarigou, and S. R. Ahuja, *Incentive-compatible pricing strategies in noncooperative networks*, Proc. INFOCOM'98 (San-Francisco, CA), April 1998, pp. 439–446.
17. R. J. La and V. Anantharam, *Optimal routing control: Repeated game approach*, IEEE Trans. Automat. Contr. **47** (2002), 437–450.
18. I Menache and N. Shimkin, *Capacity management for and equilibrium for proportional QoS*, Technical Report CCIT No. 575, Department of Electrical Engineering, Technion, Israel. March 2006. Submitted to IEEE Trans. Networking.
19. I. Milchtaich, *Topological conditions for uniqueness of equilibrium in networks*, Math. Oper. Res. **30** (2005), 225–244.
20. A. Nagurney, *Network economics: A variational inequality approach*, 2nd ed., Kluwer Academic, Dordrecht, The Netherlands, 1999.
21. A. Orda, R. Rom, and N. Shimkin, *Competitive routing in multi-user communication networks*, IEEE/ACM Trans. Networking (1993), 510–521.
22. M. Patriksson, *The traffic assignment problem: Models and methods*, VSP, Utrecht, The Netherlands, 1994.
23. O. Richman and N. Shimkin, *Topological uniqueness of the nash equilibrium for selfish routing with atomic users*, Math. Oper. Res. (2007), in press.
24. J. B. Rosen, *Existence and uniqueness of equilibrium points for concave n-person games*, Econometrica **33** (1965), no. 3, 520–534.
25. T. Roughgarden, *Selfish routing and the price of anarchy*, MIT Press, Cambridge, MA, 2005.
26. T. Roughgarden and E. Tardos, *How bad is selfish routing*, J. ACM **49** (2002), 236–259.
27. D. Schmeidler, *Equilibrium points of nonatomic games*, Journal of Statistical Physics **7** (1973), no. 4, 295–300.
28. J. G. Wardrop, *Some theoretical aspects of road traffic research*, Proc. Inst. Civ. Eng. **2** (1952), 325–378.

Beyond CHOKe: Stateless Fair Queueing

Rade Stanojević and Robert Shorten

Hamilton Institute, NUIM, Ireland

Abstract. Making drop decisions to enforce the max-min fair resource allocation in a network of standard TCP flows without any explicit state information is a challenging problem. Here we propose a solution to this problem by developing a suite of stateless queue management schemes that we refer to as Multi-Level Comparison with index l (MLC(l)). We show analytically, using a Markov chain model, that for an arbitrary network topology of standard TCP flows and queues employing MLC(l), the resource allocation converges to max-min fair as l increases. The analytical findings are verified experimentally using packet level *ns*2 simulations.

1 Introduction

Resource allocation in communication networks has been a topic of interest for some time. In the current Internet most traffic uses TCP as the transport protocol, and most Internet routers do not differentiate packets from different flows. In order to adjust the resource allocation amongst competing users, one can do the following: (1) design the new end-to-end protocol(s) and leave the network infrastructure (routers) unchanged [28,12,8]; (2) design the new end-to-end protocol(s) and network support that will allow cooperation between end-users and network (routers) [11,3,27]; (3) leave the end-to-end protocol(s) unchanged but design the network based scheme that determines desired resource allocation[25,5,22].

Most current proposals have as their performance objective a resource allocation that is max-min fair. In this paper we propose a scheme that belongs to the third group listed above, and whose performance goal is enforcing a max-min fair resource allocation. Its main features are the following.

1. No changes to end-to-end transport protocols are required.
2. The decision to drop (mark) a packet is made locally by each router;
3. No multiple queues or per flow counters are used.

Thus, our goal is to design a stateless active queue management scheme that can enforce max-min fairness in the network of TCP users. While there exists a large amount of work related to analysis and design of distributed algorithms/architecture that enforce max-min fairness, to the best of our knowledge, our algorithm is the first that attempts a stateless active queue management scheme to enforces max-min fairness in the network of TCP users.

1.1 Paper Contributions

Why is reaching the goal stated above so hard? First, recall that in the max-min fair regime, a TCP flow f experiences drops at *one and only one* link l_f at its path (we say that f is bottlenecked at l_f), and therefore must be protected at other links (that can be congested) by receiving lossless service. Second, if two or more flows are bottlenecked at the same link they must receive nonuniform loss rates that are function of their aggressiveness (round-trip times, queuing delays, delayed acks option, etc). Assuming that the router has access to the individual flow rates, or the existence of multiple queues that are appropriately scheduled, a number of solutions to these two problems exist, and are described in previous works [25,11,20,5,22]. However, in our case (routers with no explicit state information) it is highly nontrivial to make the drop (mark) decision without any explicit information. The main contributions of this paper are:

- A stateless queue management scheme, Multi Level Comparisons with index l (MLC(l)), which makes the drop decision based on the structure of packets that are already in the queue (using simple comparisons only).
- A Markov chain analysis of the randomized algorithm MLC(l) that shows that the resource allocation of MLC(l) converge to the max-min fair, for arbitrary network topology and arbitrary set of TCP users, as the index l grows.
- Packet level simulations are presented that support the analytical findings.

2 Power-Drop AQM Schemes

It has been noticed in many studies that both drop tail and RED routers have large bias against large-RTT flows. For example, the authors of [14] have made the empirical observation that for a drop-tail router and two flows with round trip times RTT_1 and RTT_2, the ratio of asymptotic throughput of the first and the second flow is in the ratio $(RTT_2/RTT_1)^a$ for some $a \in (1,2)$. Similarly, it has also been noticed in a number of studies, that oblivious (ones that do not differentiate packets from different flows) AQM schemes (RED [10], BLUE [7], etc.) which attempt to estimate the loss probability for a given traffic pattern and to drop packets according to this estimation, share bandwidth among competing users with round trip times RTT_1 and RTT_2 in the ratio RTT_2/RTT_1, [9,2]. In this section we will investigate RTT unfairness characteristics for more general AQM schemes we call power-drop AQM schemes.

Definition 1. *An AQM is power-drop if it drops a packet from a flow with current throughput*[1] *U with probability $\rho_0 U^{l-1}$, where ρ_0 is variable controlled by router and l positive integer called index of the given power-drop AQM.*

Comment. With $l = 1$ this corresponds to a router which drops packets with loss probability ρ_0. The case when $l = 2$ is similar to CHOKe[21] in the limit when the average queue size does not go the below minimum threshold, and in

[1] Throughput is measured in packets per unit of time.

addition, when there is neither a RED nor an overflow drop. Indeed, comparing a packet at the entrance of queue with a packet from the queue and making drop-decision based on this comparison is actually dropping a packet with probability which is proportional to current throughput of the flow.

In this section we will describe a class of power-drop AQM's called Multi Level Comparison (MLC) AQM's. In particular, we will describe and analyze the fairness characteristics of this queueing discipline for TCP flows competing for bandwidth. We will see that the MLC scheme with index l achieves $1/RTT^{1/(l+1)}$-fairness[2] under the assumption of low loss probability (Theorem 1). More generally, Theorem 2 shows that increasing l leads resource allocation among TCP users arbitrarily close to max-min fairness in general network topologies.

2.1 Description of MLC

The basic strategy in MLC(l) is to extend the core idea from CHOKe of comparing of a packet arriving at the queue with packets which are already in queue; these stored packets are measure of the proportion of bandwidth used by certain flow. MLC(l) maintains a variable h_M which is used to control the probability of dropping an arriving packet: at every packet arrival h_M dropping trials are executed.

> *Dropping trial:* Pick randomly $l - 1$ packets from the queue: if *all* l packets belong to same flow, then drop the arriving packet (if $l = 1$ the arriving packet is dropped by default).

If the arriving packet is not dropped after the execution of h_M dropping trials then it is enqueued. If h_M is not an integer, the number of dropping trials is given as follows. For $h_M < 1$ we execute 1 dropping trial with probability h_M and 0 dropping trial with probability $1 - h_M$. Similarly, $h_M > 1$ we execute $\lfloor h_M \rfloor + 1$ dropping trials with probability $\{h_M\} = h_M - \lfloor h_M \rfloor$ and $\lfloor h_M \rfloor$ dropping trials with probability $1 - \{h_M\}$.

Proposition 1. *For a given h_M, MLC(l) is power-drop scheme with index l.*

Proof. Let U_f be the throughput of a flow f, and U_0 the aggregate throughout on the link. A packet is dropped at one dropping trial with probability

$$q_1 = \left(\frac{U_f}{U_0}\right)^{l-1}.$$

The probability that a packet is dropped after h_M trials is 1-$Prob$[packet is not dropped at any of h_M trials] which is given by

$$q = 1 - (1 - q_1)^{h_M} \approx q_1 \cdot h_M = U_f^{l-1} \frac{h_M}{U_0^{l-1}}.$$

Taking $\rho_0 = \frac{h_M}{U_0^{l-1}}$ we conclude that MLC(l) satisfies Definition 1.

[2] Two flows with round trip times RTT_1 and RTT_2 which have a single bottleneck operating with MLC, obtain bandwidth in ratio: $(RTT_1/RTT_2)^{-1/(l+1)}$.

The higher h_M the more frequent the losses are. Consequently, if the link is under-utilized h_M should be decreased in order to decrease probability of dropping packets. On the other hand if the aggregate traffic on the link is greater than the link capacity then h_M should be increased to reduce traffic load.

Controlling the variable h_M: MLC uses a parameter Δ_0 to affect changes in the variable h_M (in our simulations Δ_0 is set to $100ms$). h_M is adjusted once per Δ_0 using a MIMD (Multiplicative Increase - Multiplicative Decrease) scheme. The performance goal is to keep the utilization at a certain level u_0. Namely, if within the previous Δ_0 the link utilization was less than desired u_0, h_M is set to h_M/γ for some $\gamma > 1$, otherwise h_M is adjusted as $h_M := h_M\gamma$.

At this point it is important to emphasize a few differences between MLC and CHOKe. First, note that CHOKe makes a comparison only when the average queue size becomes greater than min_{th} (RED minimum threshold), and therefore its performance (in terms of resource allocation between TCP users) depends mainly on the number of users: a small number of TCP flows will affect the synchronization of losses, while for large number of users, the number of CHOKe-drops will be much less than number of RED-drops and therefore the effect of CHOKe on TCP fairness would be negligible. Second, the design of CHOKe basically neglects the TCP fairness as performance objective and concentrates on reducing throughput of unresponsive flow(s) [26], while MLC is designed to improve TCP fairness and neglects effects of unresponsive flows.

Our experiments indicate that the parameter Δ_0 should be in the range of round trip times of the connections using the link (in order to allow users to react to changes in h_M). The parameter γ controls the speed of adaptation to changes in network traffic and should be set such that, for a given Δ_0, h_M can be doubled/halved within a few seconds.

2.2 Model and Analysis of Power-Drop AQM

In this section we present a model of power-drop AQM's servicing multiple TCP users. We present results that characterize this situation for both a single bottleneck and for general network topologies.

Single bottleneck case. We consider N TCP-flows with heterogenous round trip times RTT_i, $i = 1, \ldots, N$, traversing a single bottleneck link that employs power-drop AQM with an index l. If we assume that ρ_0 does not fluctuate much (so that we can model it as constant) and that the drop probability for a packet is small, then our analysis shows that the asymptotic rates achieved by TCP users are proportional to $\frac{1}{RTT_i^{2/(l+1)}}$. This is the main result of this section and is given in Theorem 1.

Model. At the flow level, let Δ be the length of sampling interval over which we evaluate changes in throughput. If a flow with a round trip time RTT does not see a drop within interval of length Δ, then its throughput will be increased for Δ/RTT^2 (see [1]). If a flow registers a drop within this sampling interval then its throughput will be halved.[3] The probability that the first event will happen is equal to the probability that each of ΔU packets from the flow are not dropped. This probability is given by:

$$\eta_1 = (1 - \rho_0 U^{l-1})^{\Delta U} \approx e^{-\Delta \rho_0 U^l}.$$

Clearly, the probability that a flow with current throughput U will see a drop within a sampling interval of length Δ is equal to

$$\eta_2 = 1 - (1 - \rho_0 U^{l-1})^{\Delta U} \approx 1 - e^{-\Delta \rho_0 U^l}.$$

The previous approximations are valid under the assumption of a small probability that a packet will be dropped : $\rho_0 U^{l-1} \ll 1$. This assumption seems reasonable, since if this probability is not small, a flow would suffer too many losses an therefore would not get chance to enter the (AIMD) congestion avoidance phase.

Let $U_k^{(\rho)}$ be a stochastic process which describes the evolution of throughput of a TCP flow with round-trip time RTT traversing over link with a power-drop AQM scheme with index l. Here $\rho = \Delta \rho_0$. Since Δ is fixed we can assume that Δ is equal to one unit of time.

We model $U_k^{(\rho)}$ as a Markov chain on $[0, \infty)$ defined by $U_0^{(\rho)} = 0$ and:

$$U_{k+1}^{(\rho)} = U_k^{(\rho)} + \frac{1}{RTT^2} \quad \text{with probability} \quad e^{-\rho (U_k^{(\rho)})^l}$$

$$U_{k+1}^{(\rho)} = \frac{1}{2} U_k^{(\rho)} \quad \text{with probability} \quad 1 - e^{-\rho (U_k^{(\rho)})^l}.$$

The following theorem characterizes the time averaged throughput of a TCP flow with round trip time given by RTT, running over power-drop queue management scheme with index l and its proof can be found in [24].

Theorem 1. *The time averaged throughput of the i'th flow: $\frac{1}{M}\sum_{i=1}^{M} U_i^{(\rho)}$ converges almost surely to:*

$$\lim_{M \to \infty} \frac{1}{M} \sum_{i=1}^{M} U_i^{(\rho)} =: \overline{U}^{(\rho)} = \frac{1}{RTT^{\frac{2}{l+1}} \rho^{\frac{1}{l+1}}} D_{MLC}(l) + \frac{1}{\rho^{\frac{1}{l+1}}} S^{(\rho)}$$

where $D_{MLC}(l)$ is a constant that does not depend on ρ neither RTT and $S^{(\rho)}$ converges to 0 as ρ goes to 0.

Remark. The previous theorem is a generalization of the well known square root formula. Indeed, for $l = 1$, power-drop scheme is an oblivious AQM that drops packets with probability $\rho = h_M$ and Theorem 1 says that time averaged throughput converge to $\frac{1}{RTT\sqrt{\rho}} D_{MLC}(1) + o(\frac{1}{\sqrt{\rho}})$.

[3] Throughout this paper, variations in round trip times are neglected.

To conclude this section we prove that for a given network with routers employing a power-drop AQM with index l, and assuming that the steady state throughput is given by the previous theorem, we can find large enough l such that bandwidth allocation is arbitrary close to max-min fairness. The following characterization of max-min fairness can be found in [23].

Lemma 1. *A set of rates x_r is max-min fair if and only if for every flow r there exists a link on its path, such that the rates of all flows which traverse through that link are less or equal than x_r.*

With this characterization of max-min fair allocation in mind, we shall prove that increasing the index of the MLC will result in allocation of bandwidth in such fashion that each flow will have link on its path such that its asymptotic rate is "almost" the largest among all flows using that link.

Theorem 2. *For any given network topology, and given $\varepsilon > 0$, there exists l such that if all queues employ MLC with index l and loss probabilities are small then for every flow r there exist a link on its path, such that the rates of all flows which traverse through that link are less than $(1+\varepsilon)x_r$ (here x_r is steady state rate of flow r).*

Proof. Let L be the number of links in the network and N the number of flows. We label flows by $i = 1, 2, \ldots, N$ and links by $s = 1, 2, \ldots, L$. By R we denote the routing matrix: $R_{is} = 1$ if flow i uses link s otherwise $R_{is} = 0$. On each link s, a router drops a packet from the flow with current throughput U with probability $\rho(s)U^{l-1}$. Let M be the length (in number of links) of the path of the flow with most links on its route and ν the ratio of the largest and the smallest round trip time in the network. Choose l such that

$$\nu^{\frac{2}{l+1}} M^{\frac{1}{l+1}} < 1 + \varepsilon.$$

For each flow r, let $s_1^{(r)}, \ldots, s_w^{(r)}$ be links used by it and let $s_{max}^{(r)}$ the most congested link on its route in the following sense:

$$\rho(s_{max}^{(r)}) = \max\{\rho(s_j^{(r)}) \mid j = 1, \ldots, w\}. \tag{1}$$

If the current rate of flow r is U, a packet from that flow will be dropped with probability $\lambda_r U^{l-1}$, where $\lambda_r = \sum_{j=1}^{w} \rho(s_j^{(r)})$, and therefore the steady state throughput for flow r is given by

$$x_r = \frac{1}{RTT_r^{\frac{2}{l+1}} \lambda_r^{\frac{1}{l+1}}} C_0.$$

For any other flow t which uses the link $s_{max}^{(r)}$ with $\lambda_t = \sum_{j:R_{jt}=1} \rho(l_j) \geq \rho(s_{max}^{(r)})$ the steady state throughput is given by:

$$x_t = \frac{1}{RTT_t^{\frac{2}{l+1}} \lambda_t^{\frac{1}{l+1}}} C_0.$$

Recall that we have defined the link $s_{max}^{(r)}$ as the most congested link on route of flow r in the sense of (1). This implies that $\lambda_r \leq M\rho(s_{max}^{(r)})$. Now

$$\frac{x_t}{x_r} = \left(\frac{RTT_r}{RTT_t}\right)^{\frac{2}{l+1}} \left(\frac{\lambda_r}{\lambda_t}\right) \leq \left(\frac{RTT_r}{RTT_t}\right)^{\frac{2}{l+1}} \left(\frac{M\rho(s_{max}^{(r)})}{\rho(s_{max}^{(r)})}\right)^{\frac{1}{l+1}} \leq$$

$$\leq \nu^{\frac{2}{l+1}} M^{\frac{1}{l+1}} < 1 + \varepsilon.$$

Remark. Note that for a single bottleneck topology, the resource allocation given by $\frac{C}{RTT_i^{2/(l+1)}}$ is $((RTT_i^2), l+1)$ proportionally fair[18,23]. Indeed, for any resource allocation (x_i), utility $U(x) = \sum_{i=1}^{N} \frac{RTT_i^2}{x_i^l}$, and link capacity c_0 we have (using Holder's inequality):

$$(U(x))^{\frac{1}{l+1}} \cdot c_0^{\frac{l}{l+1}} = \left(\sum_{i=1}^{N} \frac{RTT_i^2}{x_i^l}\right) \cdot (\sum_{i=1}^{N} x_i) =$$

$$\left(\sum_{i=1}^{N} \left(\frac{RTT_i^{2/(l+1)}}{x_i^{\frac{l}{l+1}}}\right)^{l+1}\right)^{\frac{1}{l+1}} \cdot \left(\sum_{i=1}^{N} \left(x_i^{\frac{l}{l+1}}\right)^{\frac{l+1}{l}}\right)^{\frac{l}{l+1}} \leq$$

$$\leq \sum_{i=1}^{N} \frac{RTT_i^{2/(l+1)}}{x_i^{\frac{l}{l+1}}} \cdot x_i^{\frac{l}{l+1}} = \sum_{i=1}^{N} RTT_i^{2/(l+1)}$$

$U(x)$ is maximized if equality holds in the inequality above, which is equivalent to $x_i = \frac{C}{RTT_i^{2/(l+1)}}$ for some constant C. Thus, while spectrum of delay-based end-to-end protocols[18] assume no cooperation from routers to converge to max-min fairness, MLC(l) does not require changes in end-to-end protocol to enforce max-min fairness.

Similar interesting feature is shared between shuffling parameter $\gamma > 0$ of XCP and index l of MLC, but is not discussed here because of space limitations; see [24,16].

3 Experimental Results

In this section we briefly describe some $ns2$ simulations that demonstrate the behavior of proposed AQM schemes.

3.1 Single Bottleneck

The first set of simulations are designed to demonstrate the fairness properties of the MLC in single bottleneck scenario. Specifically, we present results for a single link with service rate of $80Mbps$ that services 100 long-lived TCP users with round trip times uniformly distributed in range $40 - 440ms$. To provide

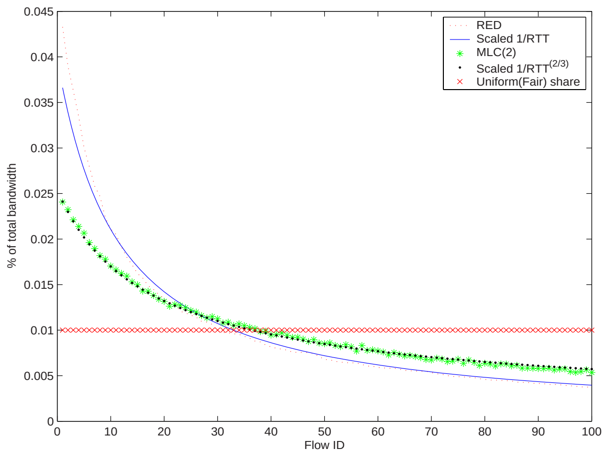


Fig. 1. Scaled throughput for 100 flows over congested link employing RED, MLC(2)

baseline results, we include the performance of RED for the same scenario. Share of total throughput taken by each of 100 flows assuming the bottleneck queue is managed by MLC(2) is depicted in Figure 1.

It can be seen from Figure 1 that the fairness of RED is approximately proportional to the inverse of RTT. This is in accordance with observations made in [2,9]. It can also be observed that the fairness of MLC with index 2 is proportional to $1/RTT^{2/3}$ as predicted by Theorem 1. The MLC parameters used in the simulation are: $l = 2$, $\varDelta_0 = 100ms$, $\gamma = 1.01$, $u_0 = 0.98$.

3.2 Multiple Bottleneck Topologies

Our second set of simulations demonstrate Theorem 2. The network topology that we considered is given in Figure 2. Here, we consider a network of 24 nodes: $n1 - n5, m1 - m5, p1 - p5, q1 - q5$, and $c1$, $c2$, $c3$, $c4$ and 30 flows traversing the network as follows: $n(i) \to p(i); n(i) \to q(i), m(i) \to p(i); m(i) \to q(i); n(i) \to m(i); p(i) \to q(i)$ where $i = 1, 2, 3, 4, 5$.The delays on each of the links in ms are defined as follows:

$$\begin{aligned} ni \to c1 &: 40 \cdot i + 1; & pi \to c3 &: 40 \cdot i + 1 \\ mi \to c2 &: 40 \cdot i + 1; & qi \to c4 &: 40 \cdot i + 1 \end{aligned}$$

and the delays $c1 - c2$, $c2 - c3$, $c3 - c4$ are $10ms$. The capacities of all links are $10Mbps$. With this topology, the max-min fair shares are $0.5Mbps$ for 20 flows that uses link $c2 - c3$, and $1Mbps$ for other 10 flows ($n(i) \to m(i)$ and $p(i) \to q(i)$).

Each flow uses the standard TCP-SACK algorithm, with a packet size 1000B. The aggressiveness of each flow is mainly determined by its RTT. The behavior of the network is evaluated with each link $c1-c2$, $c2-c3$ and $c3-c4$ using: DropTail, RED, MLC(2), MLC(3), MLC(5), MLC(9), and MLC(17) with a queue size of 100 packets. MLC(a) parameters are: $l = a$, $\Delta_0 = 100ms$, $\gamma = 1.01$, $u_0 = 0.98$.

Normalized Jain's fairness index for vector that represents resource allocations $U = (U_1, \ldots, U_N)$ in the network with max-min fair resource allocation given by vector $U_{mm} = (U_{1,mm}, \ldots, U_{N,mm})$ is given by

$$j(U) = \frac{\left(\sum_{i=1}^{N} \frac{U_i}{U_{i,mm}}\right)^2}{N \sum_{i=1}^{N} \left(\frac{U_i}{U_{i,mm}}\right)^2}.$$

Its values for 7 schemes of interest are following:

$$j(U_{DTail})=0.345,\ j(U_{RED})=0.731,\ j(U_{MLC(2)})=0.846,\ j(U_{MLC(3)})=0.884,$$

$$j(U_{MLC(5)}) = 0.956,\ j(U_{MLC(9)}) = 0.989,\ j(U_{MLC(17)}) = 0.997.$$

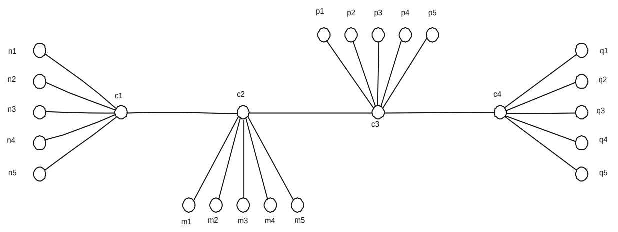


Fig. 2. Network topology

We can see significant unfairness in oblivious schemes: DropTail, RED. As we increase index of MLC scheme, we obtain share of bandwidth very close to max-min share as expected by Theorem 2.

4 Summary

In this paper we developed an AQM scheme for enforcing max-min fairness in TCP networks called MLC(l). MLC(l) is a stateless scheme and belongs to class of queue management schemes that we call power-drop. We showed analytically that by increasing index l, the resource allocation among TCP users using network of MLC(l) queues converge to max-min fair. The presented analytical findings are confirmed by packet level $ns2$ simulations.

References

1. E. Altman, K. Avrachenkov, C. Barakat. "A stochastic model of TCP/IP with stationary random losses". IEEE/ACM Transactions on Networking (TON), vol 13(2), 2005
2. A. Abouzeid, S. Roy. "Analytic understanding of RED gateways with. multiple competing TCP flows". Proc. of IEEE GLOBECOM, 2000.

3. S. Athuraliya, D. Lapsley, S. Low. “Enhanced Random Early Marking algorithm for Internet flow control”. Proc. of IEEE INFOCOM, Tel Aviv, Israel, 2000.
4. A. Das, D. Dutta, A. Goel, A. Helmy, J. Heidemann. “Low State Fairness: Lower Bounds and Practical Enforcement”. Proc. of the IEEE INFOCOM, Miami, FL, USA, March 2005.
5. A. Demers, S. Keshav, S. Shenker. “Analysis and simulation of a fair queueing algorithm”. Proc. of ACM SIGCOMM, Austin, TX, 1989.
6. V. Dumas, F. Guillemin, P. Robert. “A Markovian analysis of additive-increase multiplicative-decrease algorithms”. Adv. in Appl. Probab. 34 (2002), no. 1, 85-111.
7. W. Feng, K.G. Shin, D.D. Kandlur, D. Saha. “The BLUE active queue management algorithms”. *IEEE/ACM Transactions on Networking*, vol. 10, no. 4, 513-528, August 2002.
8. S. Floyd. “HighSpeed TCP for Large Congestion Windows”. RFC 3649, Experimental, December 2003.
9. S. Floyd. “Connections with Multiple Congested Gateways in Packet-Switched Networks Part 1: Oneway Traffic”. ACM Computer Communication Review, 30-47, Vol. 21 , Issue 5, October 1991.
10. S. Floyd, V. Jacobson. “Random early detection gateways for congestion avoidance”. IEEE/ACM Transactions on Networking, vol. 1, pp. 397-413, Aug. 1993.
11. D. Katabi, M. Handley, C. Rohr. “Internet Congestion Control for Future High Bandwidth-Delay Product Environments”. Proc. of ACM SIGCOMM, Pittsburgh, PA, USA, 2002.
12. T. Kelly. “Scalable TCP: Improving Performance in Highspeed Wide Area Networks”. ACM SIGCOMM Computer Communication Review Volume 33, Issue 2, April 2003.
13. M. Kodialam, T. V. Lakshman, Shantidev Mohanty. “Runs based Traffic Estimator (RATE): A Simple, Memory Efficient Scheme for Per-Flow Rate Estimation”. Proc. of IEEE INFOCOM, Hong Kong, March 2004.
14. T. V. Lakshman, U. Madhow. “The performance of TCP/IP for networks with high bandwidth-delay products and random loss”. *IEEE/ACM Transactions on Networking*, vol. 5, no. 3, 336–350, June 1997.
15. D. Lin, R. Morris. “Dynamics of random early detection”. Proc. of ACM SIGCOMM, Cannes, France, 1997.
16. S. Low, L. Andrew, B. Wydrowski. “Understanding XCP: Equilibrium and Fairness”. Proc. of IEEE Infocom, Miami, FL, March 2005.
17. R. Mahajan, S. Floyd, D. Wetherall. “Controlling high-bandwidth flows at the congested router”. Proc. of IEEE ICNP, Riverside, CA, USA, 2001.
18. J. Mo, J. Walrand. “Fair end-to-end window-based congestion control”. IEEE/ACM Transactions on Networking, Vol. 8, No. 5 October, 2000.
19. T. J. Ott, T. V. Lakshman, L. H. Wong. “SRED: Stabilized RED”. Proc. IEEE INFOCOM, New York, March 1999.
20. R. Pan, L. Breslau, B. Prabhakar, S. Shenker. “Approximate Fairness through Differential Dropping”. ACM SIGCOMM Computer Communication Review Volume 33 , Issue 2, April 2003
21. R. Pan, B. Prabhakar, K. Psounis. “CHOKe: A stateless AQM scheme for approximating fair bandwidth allocation”. Proc. of IEEE INFOCOM, Tel Aviv, Israel, 2000.
22. M. Shreedhar, G. Varghese. “Efficient fair queueing using deficit round-robin”. *IEEE/ACM Transactions on Networking*, vol. 4, no. 3, June 1996.

23. R. Srikant. “The Mathematics of Internet Congestion Control”. Birkhauser, 2004.
24. R. Stanojevic, R. Shorten. “AQM’s for achieving fairness between competing TCP flows”. Technical report, available online: www.hamilton.ie/person/rade/trm.pdf.
25. I. Stoica, S. Shenker, H. Zhang. “Core-Stateless Fair Queueing: A Scalable Architecture to Approximate Fair Bandwidth Allocations in High Speed Networks”. *IEEE/ACM Transactions on Networking*, vol. 11, no. 1, 33-46, February 2003.
26. A. Tang, J. Wang, S. H. Low. “Understanding CHOKe: throughput and spatial characteristics”. IEEE/ACM Transactions on Networking. Vol 12 (4),2004.
27. B. Wydrowski, M. Zukerman. “MaxNet: A congestion control architecture for maxmin fairness”. IEEE Communications Letters, vol. 6, no. 11, Nov. 2002, pp. 512-514.
28. Y. Yang, S. Lam. “General AIMD Congestion Control”. Proc. ICNP 2000, Osaka, Japan, 2000.

How Expensive Is Link Utilization?

Rade Stanojević and Robert Shorten

Hamilton Institute, NUIM, Ireland

Abstract. Understanding the relationship between queueing delays and link utilization for general traffic conditions is an important open problem in networking research. Difficulties in understanding this relationship stem from the fact that it depends on the complex nature of arriving traffic and the problems associated with modelling such traffic. Existing AQM schemes achieve a “low delay” and “high utilization” by responding early to congestion without considering the exact relationship between delay and utilization. However, in the context of exploiting the delay/utilization tradeoff, the optimal choice of a queueing scheme’s control parameter depends on the cost associated with the relative importance of queueing delay and utilization. The optimal choice of control parameter is the one that maximizes a benefit that can be defined as the difference between utilization and cost associated with queuing delay. We present a generic algorithm Optimal Delay-Utilization control of t (ODU-t) that is designed with a performance goal of maximizing this benefit. Its novelty lies in fact that it maximizes the benefit in an online manner, without requiring knowledge of the traffic conditions, specific delay-utilization models, nor does it require complex parameter estimation. Moreover, other performance metrics like loss rate or jitter can be directly incorporated into the optimization framework as well. Packet level ns2 simulations are given to demonstrate the behavior of the proposed algorithm.

1 Introduction

Current router buffers are generally sized by the rule-of-thumb given in [25]: router buffers require approximately space for $B = \overline{RTT} \times C$ packets, where $\overline{RTT}$ is the “average” round trip time for connections that use the link and C is capacity of the link. Following this rule, most router buffers are designed in such a fashion that they result in up to $100ms$ to $250ms$ of queueing[1,4]. This, together with TCP’s mechanism of congestion avoidance, serves to ensure a high link utilization.

In the last few years a number of results related to buffer sizing for congested links have appeared [1,2,4,5] that suggest significantly smaller buffers. Although the bounds from these papers yield important theoretical insights into the relation between link utilization and the required buffering they are not immediately applicable to buffers in the real Internet routers for a number of reasons. Firstly, these bounds are functions of various parameters such as the number of active users that are bottlenecked at the link; RTT distribution of TCP users, TCP parameters ($maxcwnd\_$), etc. These quantities vary, and are also usually very hard to estimate [6,14,13,24,3]. Secondly, the mathematical assumptions used in deriving of these bounds are quite restrictive and do not take into account the various and variable traffic mixes possible, the level of loss synchronization, the existence of non-TCP traffic, etc. Most importantly, while it is useful to know that delay and utilization are related in some manner, it is not immediately clear how to utilize this relationship in a meaningful manner.

In this paper we build an optimization framework for the design of queue management schemes in which (low) queueing delays are considered as a scarce resource together with link utilization. The relative importance between queueing delays and utilization is a user[1] specified parameter. Therefore the optimal choice of queueing scheme parameter t is one that maximizes overall benefit $B(t)$ that takes into account the relative importance of queueing delays and utilization. Queueing scheme parameter t can be: (1) available DropTail queue space, or (2) per packet drop probability, or (3) virtual queue service rate or any other parameter that can control utilization and queueing delays.

In Section 3 we propose an online algorithm for control of generic parameter t: Optimal Delay-Utilization control of t (ODU-t). It does not require intricate measurement techniques neither specific assumptions related to the nature of traffic mix. In Section 4 we present a brief simulation study of ODU-t for t denoting available DropTail space.

2 Optimization Framework

Let us consider a synthetic example in which average queueing delay ($aQD(t)$) and utilization $u(t)$ depend on choice of queueing parameter t given in Table 1. For simplicity, assume for the moment that the parameter t is the available buffer space on the congested FIFO Drop-Tail queue; for buffer size equal to t_1 the average queue delay is $100ms$ and the utilization is 100%, for buffer size equal to t_2 the average queue delay is $20ms$ and the utilization is 98%, and so on. Which choice of t is optimal (among 4 possible in this example), depends on the "importance" of low queueing delays. To formalize this, one can identify the "importance" by the relative price between utilization and queueing delays. Let $P : [0, \infty) \mapsto [0, \infty)$ be a function that specifies relative price between utilization and delays. In other words, a queueing delay of d seconds has same value as utilization of $P(d)$. Formally, a price function is any function that satisfies the following definition.

Definition 1. *The function $P : [0, \infty) \mapsto [0, \infty)$ is a price function if it is twice differentiable, increasing and convex. In other words if:*
(a) $\forall d \in [0, \infty)\ \exists P''(d)$
(b) $\forall d \in [0, \infty)\ P'(d) \geq 0$
(c) $\forall d \in [0, \infty)\ P''(d) \geq 0$

[1] In this context user is an ISP or link owner.

Table 1. Synthetic example of $aQd(t)$ and $u(t)$ for 4 different possible choices of parameter t

t	t_1	t_2	t_3	t_4
$aQd(t)(sec)$	0.1	0.02	0.005	0.001
$u(t)$	1.00	0.98	0.90	0.60

Having defined a price function, the overall benefit (in the case given by the parameter t) can be written in the form:

$$B(t) = u(t) - P(aQd(t)). \tag{1}$$

Comment. Notion, similar to the benefit $B(t)$ is introduced in [2] for t representing the available buffer size.

The definition of benefit allows us to define a notion of optimal choice, as the value of t that maximizes the benefit. Formally:

Definition 2. *For a given price function P and set $\mathcal{T}$ of possible choices of t, an optimal Delay-Utilization(D-U) choice is any t_0 such that*

$$B(t_0) = \max\{B(t) \mid t \in \mathcal{T}\}, \tag{2}$$

if the maximum on the right hand side exists.

In the example given in Table 1, if we completely ignore the importance of low queueing delays, by setting $P(d) \equiv 0$ for all d, then the optimal D-U choice is given by t_1, as this maximizes the benefit $B(t) = u(t) - P(aQd(t)) = u(t)$ on the set $\mathcal{T} = \{t_1, t_2, t_3, t_4\}$. For the price function $P(d) = 5 \cdot d$, the optimal D-U choice is t_2, and for the price function $P(d) = 100 \cdot d$, the optimal D-U choice is t_4.

Throughout this paper we assume:

Assumption 1. *Under static traffic conditions the overall benefit given by (1) is a concave function of t.*

Assumption 1 is very hard to analytically check. In a theoretical framework, this would require accurate models of various traffic mixes, and as we already noted, modelling such complex environments is highly nontrivial. Some results related to the convex relationship between utilization and buffer size in non-elastic traffic environments are developed in [17,18]. However, our empirical observations suggest that for the traffic mix that is consisted from the static number of TCP and UDP flows, Assumption 1 holds when t is (1) the available DropTail space or (2) per-packet drop probability or (3) Virtual queue service rate. Technical report [22] discuss this in more detail, and contains the packet level ns2 simulations results that validate Assumption 1 under mentioned circumstances.

3 Optimal Delay-Utilization Control of *t*

Convex optimization has been widely employed in the networking community see [15,23]. In our case, we need an efficient algorithm for solving (2). A standard control strategy for solving (2) is given by

$$t(k+1) = t(k)\left(1 + g(k)\frac{B(t(k)) - B(t(k-1))}{t(k) - t(k-1)}\right),\ g(k) \geq \epsilon > 0, \tag{3}$$

The problem with employing this strategy in the present case is twofold. First, as we do not have explicit relationship between t and $B(t)$, we can not instantly compute the derivative $B'(t(k))$. Second, the noise to signal[2] ratio in measuring of both queueing delays and utilization can be very large (see [22]) especially in the neighborhood of the solution of (2). This would potentially imply low confidence in the estimation of $B'(t)$ in the neighborhood of the solution of (2).

We emphasize again that t is any parameter such that by controlling t, one can control both utilization and queueing delays. Thus, if the performance goal is given by keeping the average utilization at a certain level λ, one can design a strategy for achieving that goal by controlling t. Similarly, if the performance objective is keeping the average queueing delay (at the times of congestion) at a prescribed level d_0, another control strategy can be designed for solving that problem. At this point we should note that by controlling t one can (usually) control not only utilization and queueing delays, but also other (important) performance metrics such as jitter and loss rate. Embedding them into an optimization framework could be done in straightforward manner, but is out of scope of the present paper.

Following the delay-utilization optimization framework developed in the previous section, the performance goal of interest will be the maximization of the benefit $B(t)$. We proceed by presenting an ODU-t algorithm, a strategy with that performance goal.

The ODU-t algorithm controls the variable t such that the value t is updated once per sample time period (Δ) in the following manner:

$$t(k+1) = t(k) \cdot m(k), \tag{4}$$

where $m(k)$ is defined by:

$$m(k) = \alpha, \quad \text{if} \quad \frac{\hat{B}(l(k)) - \hat{B}(l(k-1))}{t(k) - t(k-1)} \geq 0, \tag{5}$$

$$m(k) = \frac{1}{\alpha}, \quad \text{if} \quad \frac{\hat{B}(l(k)) - \hat{B}(l(k-1))}{t(k) - t(k-1)} < 0. \tag{6}$$

Here, $\alpha > 1$ is a constant parameter, close to 1. The choice of α determines the responsiveness of the algorithm. Since t is either multiplied with α or divided by α, in each step k, $t(k) = t(0) \cdot \alpha^{l(k)}$, for some integer $l(k)$. By $\hat{B}(l(k))$ we denote the estimated value of $B(x)$ at the point $x = t(k) = t(0) \cdot \alpha^{l(k)}$. Algorithms of this type can be seen as a version of (3) that do not allow arbitrarily small

[2] By definition $B(t)$ is function of average utilization $u(t)$ and average queueing delay $aQd(t)$. Instantaneous utilization (queueing delay) can be seen as random variable that is sum of $u(t)$ ($aQd(t)$) and appropriate zero mean random variable, that we refer to as noise.

steps. We again emphasize that strategies of the form of (3) are inappropriate in our problem since any algorithm of type (3) that allows very small changes in the parameter t would suffer from a high noise to signal ratio around global maximum of $B(t)$, and would require a long time for accurate estimation of B in the neighborhood of the global maximum. Moreover, it has been proved in [21], using information-theoretical techniques, that any algorithm for finding an optimum using noisy observations of a benefit function has slow expected convergence. Namely, $O(\epsilon^{-4})$ queries have to be made before one can ensure ϵ-accuracy in the estimation of the optimum x^*. Under dynamic, Internet-like traffic conditions, frequent (small) changes of the traffic patterns might not allow such (exact) algorithms to converge, and can potentially cause undesirable large oscillations.

Algorithms of the form of (4) that do not converge to the certain value, but rather continuously search for the optimal value have been extensively used in the networking literature. Examples of such algorithms are AIMD[3] *cwnd_* control in TCP [11], AIAD algorithm for controlling the drop probability in BLUE[8] as well as MIMD algorithm for the adaptation of RED parameters in Self-Configuring RED [7].

The parameters of ODU-t are: $P(d)$ - price function, Δ - length of sampling period and α - MIMD parameter. While in general $P(d)$ can be an arbitrary function that satisfies Definition 1, throughout this paper we will mainly use functions that are linear in d:

$$P_\gamma(d) = \gamma d, \ \gamma > 0. \tag{7}$$

If we restrict ourselves to price functions of this form then the parameter $P_\gamma(d)$ can be specified by a single scalar γ. A higher value of γ assigns more importance to low delays and vice versa. The sampling period time Δ should be chosen to cover several "typical" round trip times, in order to allow traffic to respond to change of t. Choosing Δ in range $[1sec, 5sec]$ usually satisfies this condition. The parameter α determines the responsiveness of ODU-t, and should be selected such that it allows doubling/halving of t within several seconds (up to one minute).

At this point we discuss the notion of variability in the traffic conditions. Measurements from [20] show that on typical 150Mbps+ links, basic IP parameters such as the number of active connections, the proportion of TCP traffic, the aggregate IP traffic, etc., do not change dramatically. Although we do not exclude the possibility that there can be drastic changes in the traffic mixes, our basic presumption in the design of ODU-t is that such events are rare enough to be considered as exception rather than rule. Thus, ODU-t is designed to search for an optimal solution in the "regular" intervals, during which traffic conditions vary slowly. In the cases of dynamic traffic conditions, one can perform self tuning of the parameters Δ and α depending on the level of changes in the traffic conditions.

[3] Additive Increase Multiplicative Decrease.

The following theorem shows that, assuming that estimator $\hat{B}$ preserves order of B on the grid $\mathcal{T}_\alpha = \{t(0) \cdot \alpha^n,\ n \in Z\}$ the controller (4) runs system to the state that is close to global optima. The proof is given in [22].

Theorem 1. *Let t^* be the point where global maximum of B is attained. Suppose that estimator $\hat{B}$ preserves the order on the grid $\mathcal{T}_\alpha$, ie. for all $m_1, m_2 \in Z$:*

$$\hat{B}(m_1) \geq \hat{B}(m_2) \Leftrightarrow B(t(0)\alpha^{m_1}) \geq B(t(0)\alpha^{m_2}).$$

Then there exist m_0 such that for all positive integers r:

$$t(m_0 + 2r) = t(m_0 + 2r + 2) = \bar{t}$$

$$t(m_0 + 4r + 1) = \bar{t}\alpha, \quad \textit{and} \quad t(m_0 + 4r - 1) = \frac{\bar{t}}{\alpha},$$

and the relative error between $\bar{t}$ and t^* satisfies:

$$\frac{\bar{t} - t^*}{t^*} \leq \alpha - 1. \qquad \blacksquare$$

4 Case Study: *t* Is the Available DropTail Space

In this section we evaluate the behavior of ODU-t for t denoting the available DropTail space.

Simulation[4] *1.* Our first set of simulations illustrate the dynamics of t under static conditions of 50 TCP flows with RTT's uniformly distributed in range $[20, 220]msec$ and with packet sizes of 1000 bytes. We run ODU-t with parameters $\Delta = 2sec$, $\alpha = 1.05$. The price function used is $P_{10}(d) = 10 \cdot d$. Initially: $t(0) = 100Kbytes$. The off-line (see Simulation 2) optimal value are approximately $t^* \approx 130Kbytes$. Figure 1 depicts the queue occupancy, evolution of t and utilization for both cases.

Simulation 2. The second set of simulations shows how close the average queueing delays and average utilization are to the optimal values, in static conditions with a constant number TCP flows. We ran the set of 50 TCP flows, with RTT's uniformly distributed in range $[20, 220]ms$ and packet sizes of 1000 bytes, over a bottleneck link with capacity 10MBps. By running a sequence of simulations with buffer of *constant* size we can empirically find $aQd(t)$ and $u(t)$ and thus the optimal values t^* corresponding to different price functions. We refer to these (empirically obtained) optimal values as offline-optimal. Rows 3,5 and 7 in Table 2 contains aQd, u and B_γ for the offline-optimal value of parameter that maximize B_γ in three cases: $\gamma = 2, 10, 20$. For same value of γ's we run ODU using price function $P_\gamma(d) = \gamma \cdot d$ as the parameter. Online averages (5 minutes per simulation) of aQd, u and B_γ are presented in Table 2 in each of these three cases.

[4] Scripts used in all simulations from this paper can be found at http://www.hamilton.ie/person/rade/Optimal/

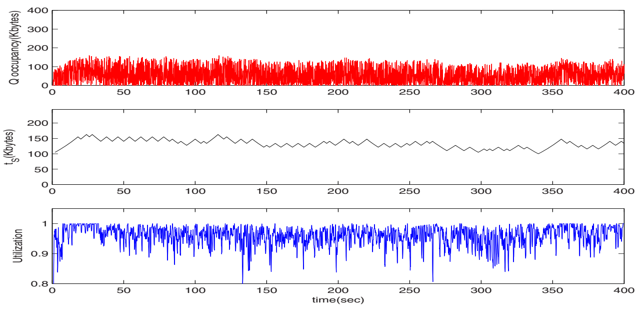


Fig. 1. Simulation 1. Queue occupancy, available buffer space(t), and utilization for ODU-t queue servicing 50 TCP flows

Table 2. Numerical results: off-line optima and online ODU-t. The last column represents $B_\gamma(t) = u(t) - \gamma \cdot aQd(t)$.

Scheme, γ	$aQd(sec)$	u	B_γ
ODU, $\gamma = 2$, online	0.01075	0.9890	0.9675
DT, $\gamma = 2$, off-line	0.01058	0.9876	0.9664
ODU, $\gamma = 10$, online	0.00471	0.9544	0.9074
DT, $\gamma = 10$, off-line	0.00521	0.9589	0.9067
ODU, $\gamma = 20$, online	0.00283	0.9222	0.8655
DT, $\gamma = 20$, off-line	0.00293	0.9269	0.8683

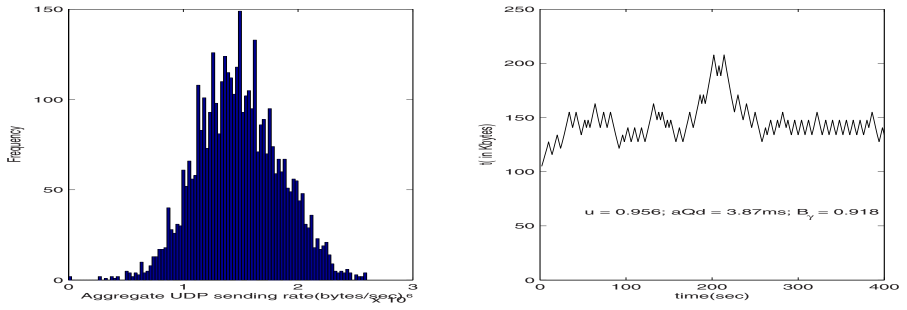


Fig. 2. Simulation 3. Left: histogram of aggregate UDP sending rate (sampling intervals $100ms$). Right: available buffer space as function of time. Queue service 50 TCP flows and 50 on-off UDP flows.

Simulation 3. This simulation shows stable behavior of ODU in the case of mixtures of TCP and (variable) UDP traffic. In this simulation, the same set of 50 TCP flows that were defined previously compete for a bandwidth on $10Mbyte/sec$ link, with 50 UDP flows that have exponentially distributed on and off periods. The on-periods have a mean of $1000ms$, and the off-periods have mean of $3000ms$. The sending rate in on-periods is $1000Kbit/sec$. The aggregate UDP arrival rate has a mean of $1.4867Mbyte/sec$ which is approximately

14.9% of the link's service rate. A histogram, given in Figure 2(left), shows the distribution of the aggregate UDP sending rate sampled on $100ms$ intervals.

The ODU parameters are the same as in previous simulations: $\Delta = 2sec$, $\alpha = 1.05$. The price function used in both cases is $P_{10}(d) = 10 \cdot d$. Initially: $t(0) = 100Kbytes$. Figure 2(right) depicts evolution of t together with obtained values of average utilization, average queueing delay and benefit .

We refer the reader to [22] to explore the behavior of ODU-t some other scenarios which: empirically show stable behavior in cases of sudden changes of traffic pattern; compare ODU that controls DropTail queue size with ODU that controls per-packet drop probability; etc.

Simulation 4. Here we demonstrate how other performance metrics are impacted by changes in available DropTail buffer space. We concentrate on fairness and loss rate. We use Jain's Fairness Index (JFI) [12] as a fairness indicator and is defined as follows. For set of users $u_1, \ldots, u_k$ let $r = (r_1, \ldots, r_k)$ be vector of their achieved average rates during the measurement interval. Then

$$JFI(r) = \frac{\left(\sum_{i=1}^{N} r_i\right)^2}{N \sum_{i=1}^{N} r_i^2}. \tag{8}$$

The simulation setup is same as in Simulation 1 and consists of 50 TCP flows serviced by the 10MBps bottleneck link with RTT's uniformly distributed in $[20, 200]ms$. The bottleneck link has a DropTail queue with size of S kilobytes. We varied S in range 10 to 300. A basic observation is that the performance of TCP-like congestion control algorithms, whose dynamics depend on round-trip time, is significantly affected by queueing delays. By increasing the queueing delay, the aggressiveness of TCP senders is decreased, implying lower loss rates. From a fairness perspective, larger queueing delays decrease bias against long-RTT connections. Indeed, for two TCP connections, with round trip times $RTT_1, RTT_2,\ RTT_1 < RTT_2$, bottlenecked at a single link with queueing delay d_0, the ratio of their expected rates[5] is $\frac{RTT_1+d_0}{RTT_2+d_0}$. Increasing d_0 leads this ratio to a value closer to one. Figure 3 presents the dependance between available space in FIFO Drop-Tail queue and loss rate and JFI. We note that for very small queue sizes ($<$ 50 kilobytes), loss rates are large and TCP dynamics is dominated by timeouts. In this regime the square root formula is not valid and fairness is impacted mainly by timeout mechanism.

5 Summary

In this paper we have addressed the problem of utilizing the tradeoff between queueing delays and link utilization. By specifying the relative importance of queueing delays and utilization, an optimal choice of a queue management parameter is the one that maximizes the overall benefit defined by (1). There could be two possible approaches for solving this problem. First, suppose that one can,

[5] This follows from the square root formula.

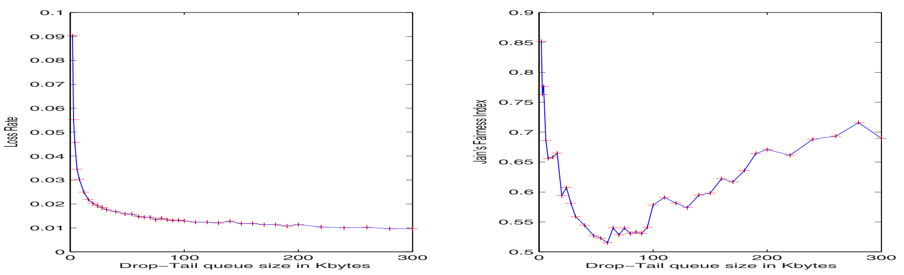


Fig. 3. Simulation F. Loss rates (top) and JFI (bottom) for 50 TCP flows serviced by Drop-Tail queues of different sizes

by accurate modelling and effective estimation, predict the delay/utilization dependance from the control parameter. Then, by an off-line solving of the underlying optimization problem we can set the parameter that controls queue scheme to the optimal value; see [2] for one strategy of this type. And second, where one can adapt the control parameter such that on average the overall benefit is maximized. We argue, that the first approach is not feasible in the current Internet because of both nonexistence of accurate and tractable enough models for the delay/utilization dependance, and the highly nontrivial estimation techniques that such an approach would require. We thus follow the second approach and design an online algorithm Optimal Delay-Utilization control of t which aim to solve the underlying optimization problem by online adaptation of generic parameter t.

The optimization problem (2) assumes a linear dependance between utilization and benefit, and neglected other important performance metrics such as jitter, loss probability, and fairness. In fact, one can define the general overall benefit of the queueing scheme controlled by parameter t as:

$$B_G(t) = V(u(t), aQd(t), j(t), L(t), f(t)), \tag{9}$$

where $j(t)$ is jitter, $L(t)$ is the loss rate, $f(t)$ is a fairness indicator, V is the utility function. We again emphasize the importance of fairness in TCP environments where long-RTT connections could heavily suffer from low queueing delays at the congested links. The embedding of jitter and loss rate into current framework can be done in straightforward manner. However, including fairness into the optimization framework, would be much more challenging as we are not aware of any, computationally light, estimation technique that would faithfully indicate level of fairness. One possible approach to estimate level of the fairness could be by counting runs[6] as suggested in [16].

From the theoretical point of view, an important open issue is convexity (concavity) of the average utilization/Q-delays/ loss-rates as function of control parameter t (available buffer space, random drop probability, virtual queue

[6] Run is event where arriving packet belongs to the same flow as some, previously arrived packet.

capacity, etc). While some results exist for the nonelastic traffic [17,18], in the case of elastic traffic, the arrival process depends on the control parameter, which makes modelling of the corresponding tradeoff curve more difficult.

Other AQM schemes could be seen in the optimization framework as well. For example the AVQ algorithm developed in [19] or PI[10] have strict performance goals in terms of utilization(AVQ) or queueing delay(PI). Such schemes can be easily incorporated into our framework, taking appropriate utility functions (see [22]).

The MIMD nature of ODU-t algorithm introduced here is just one possible approach for solving the optimization problem (2). In Section 3 we discussed the rationale for choosing MIMD algorithm that continuously searches for optimal value instead of an algorithm that will search for an exact optimal value under noisy measurements. It will be interesting to investigate other control strategies as part of future work.

References

1. G. Appenzeller, I. Keslassy, N. McKeown. “Sizing router buffers”. ACM SIGCOMM, USA, 2004.
2. K. Avrachenkov, U. Ayesta, A. Piunovskiy. “Optimal choice of the buffer size in the Internet routers”. Proc. of the IEEE CDC, Spain, 2005.
3. A. Dhamdhere, C. Dovrolis. “Open Issues in Router Buffer Sizing” ACM CCR, Jan. 2006.
4. A. Dhamdhere, H. Jiang, C. Dovrolis. “Buffer sizing for congested internet links”. Proc. of the IEEE INFOCOM, Miami, FL, USA, 2005.
5. M. Enachescu, Y. Ganjali, A. Goel, N. McKeown,T. Roughgarden. “Part III: routers with very small buffers”. ACM Computer Communication Review 35(3): 83-90 (2005).
6. C. Estan, G. Varghese, M. Fisk. “Bitmap algorithms for counting active flows on high speed links”. Proc. of 3rd ACM SIGCOMM conference on Internet measurement, 2003.
7. W. Feng, D. D. Kandlur, D. Saha, K. G. Shin. “A Self-Configuring RED Gateway”. Proc. of INFOCOM, New York, NY, USA, 1999.
8. W. Feng, K.G. Shin, D.D. Kandlur, D. Saha. “The BLUE active queue management algorithms”. *IEEE/ACM Transactions on Networking*, vol. 10, no. 4, 513-528, August 2002.
9. R. J. Gibbens, F. P. Kelly. “Distributed Connection Acceptance Control for a Connectionless Network”, 16th International Teletraffic Conference, Edimburgh, June 1999, pp. 397-413.
10. C. Hollot, V. Misra, D. Towsley, W.B. Gong. “Analysis and design of controllers for AQM routers supporting TCP flows” *IEEE Transactions on Automatic Control*, pp. 945-959 June, 2002.
11. V. Jacobson. “Congestion avoidance and control”. Proc. of the ACM SIGCOMM, 1988.
12. R. Jain. “The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling”. John Wiley and Sons, INC., 1991.

13. S. Jaiswal, G. Iannaccone, C. Diot, D. F. Towsley. "Inferring TCP connection characteristics through passive measurements". IEEE INFOCOM, March 2004.
14. H Jiang , C. Dovrolis. "Passive estimation of TCP round-trip times". ACM SIGCOMM Computer Communication Review, v.32(3), July 2002.
15. F.P. Kelly, A.K. Maulloo, D.K.H. Tan. "Rate control for communication networks: shadow. prices, proportional fairness and stability". J. Oper. Res. Soc., Vol. 49 (3), March 1998.
16. M. Kodialam, T. V. Lakshman, S. Mohanty. "Runs bAsed Traffic Estimator (RATE): A Simple, Memory Efficient Scheme for Per-Flow Rate Estimation". Proc. of the IEEE INFOCOM, 2004.
17. K. Kumaran, M. Mandjes. "The buffer-bandwidth trade-off curve is convex". Queueing Systems, 38 (2001), no. 4, 471–483.
18. K. Kumaran, M. Mandjes, A. Stolyar. "Convexity properties of loss and overflow functions". Operations Research Letters, 31(2), 2003.
19. S. Kunniyur, R. Srikant. "Analysis and Design of an Adaptive Virtual Queue (AVQ) Algorithm for Active Queue Management". *IEEE/ACM Transactions on Networking*, 12(2), 2004.
20. Online: http://pma.nlanr.net/Special/.
21. B. Pearlmutter. "Bounds on query convergence". Preprint 2005. Online: http://arxiv.org/abs/cs.LG/0511088.
22. R. Stanojevic, R. Shorten. "How expensive is link utilization". Technical report, available online: www.hamilton.ie/person/rade/QP.pdf.
23. R. Srikant, *The Mathematics of Internet Congestion Control*. Birkh̀auser, 2004.
24. B. Veal, K. Li, D. Lowenthal. "New Methods for Passive Estimation of TCP Round-Trip Times". Proc. of PAM, Boston, MA, USA, 2005.
25. C. Villamizar, C. Song. "High Performance TCP in ANSNET". ACM Computer Communication Review, 24(5), 1994

Two Different Models of FAST TCP and Their Stable and Efficient Modification

Kyungmo Koo[1], Joon-Young Choi[2], and Jin S. Lee[1]

[1] Department of Electronic and Electrical Engineering, Pohang University of Science and Technology, Korea
{pumpkins,jsoo}@postech.ac.kr
[2] Department of Electronic Engineering, Pusan National University, Korea
jyc@pusan.ac.kr

Abstract. In this paper, we introduce two different models of FAST TCP. One is from the original FAST TCP model, and the other is from the *ns-2* implementation of FAST TCP. Interestingly, these two models show significantly different dynamic performance and stability characteristics. That is, while the latter model is always globally exponentially stable, the former model is faster than the latter model and possible to go unstable. Motivated from these two models, we suggest a modified congestion control algorithm. By tuning the gain of the terms caused by the difference of the two models, the modified algorithm can be made to become globally asymptotically stable and more responsive than the *ns-2* implementation model of FAST TCP. The stability condition of the modified algorithm is decoupled from the network parameters and does not change the equilibrium state.

Keywords: Internet congestion control, FAST TCP, Global stability.

1 Introduction

While the commonly implemented transmission control protocols (TCP) such as TCP Reno and its variants rely on packet loss to measure congestion, there has been recent interest in using queuing delay at the links as a new congestion measure. Some of the TCP examples implementing this idea in their congestion control mechanisms are TCP Vegas [1] and more recently FAST TCP [9]. When the queuing delays are used for congestion measure, the users at the sources are able to anticipate the onset of congestion in time to adjust their congestion window size, thereby keeping a reasonable number of packets buffered in the links and then preventing the loss of packets assuming enough buffering spaces at the links. The congestion control mechanism developed using queuing delay as a congestion measure is more responsive to network changes and well suited to high speed networks. Being sensitive to network congestion, it becomes all the more important to ensure not only stability but also responsiveness of the algorithm with this mechanism.

FAST TCP is a new TCP congestion control algorithm for high-speed long-latency networks and uses the queuing delays as congestion measure [9]. Even though extensive experiments of FAST TCP have been conducted, and the results are promising [9], the stability property of FAST TCP has not been sufficiently studied yet. Local stability of FAST TCP without network feedback delay was proved for the case of a single link in [9]. A sufficient condition for local asymptotic stability of FAST TCP was achieved for the general network with network feedback delay in [8]. The global asymptotic stability of FAST TCP is analyzed for the single-link network in [2,3]. Recent study [3] reveals that FAST TCP can be always globally exponentially stable in a single-link multi-source network with a slightly different source model based on *ns-2* implementation [11] of FAST TCP.

In this paper, we introduce two different available models of FAST TCP. One is from the original FAST TCP model which is used in almost all of the literature analyzing FAST TCP, e.g. [2,7,8,9], and the other is from the *ns-2* implementation of FAST TCP which is adopted in [3]. We investigate that these two models show significantly different dynamic performance and stability characteristics. While the latter model is always globally exponentially stable, the former model is possible to go unstable even though the former model is faster than the latter model. Motivated from these two models, we suggest a modified version of FAST TCP congestion control algorithm. By tuning the gain of the terms caused by the difference of two FAST TCP models, the modified algorithm can be always globally asymptotically stable compared to the former model and still faster than the latter model. We establish a sufficient condition for global asymptotic stability of the modified algorithm in a single-link single-source network by adopting the Lyapunov-Razumikhin theorem [4]. Since the condition is decoupled from the network parameters, it does not change the equilibrium state.

This paper is organized as follows. Section 2 presents a continuous-time model for the single-link single-source network with a FAST TCP source and a link with FIFO queue. Section 3 introduces two different models of FAST TCP and compares their dynamic performance and stability characteristics. In section 4, a modified version of FAST TCP congestion control algorithm is proposed, and its global asymptotic stability is proved in a single-link single-source network. Section 5 provides the simulation results to illustrate the performance and the stability of the modified algorithm. And, section 6 makes conclusions.

2 Network Model

In this section, we develop a network model to describe the behavior of FAST TCP and its modified versions. Analyzing a general multi-link multi-source network with TCP sources considering global stability causes mathematical difficulty and complexity, and hence we focus on the case of a single-link single-source network in which a pair of sender and receiver node is connected through a single bottleneck link. The link has a finite transmission capacity c and is assumed to have infinite buffering storage. Associated with the link is the queuing delay $p(t)$ and with the source is the congestion window $w(t)$. We assume at time t that the source observes as a feedback signal the measured queuing delay

$$q(t) \triangleq p(t - \tau^b) \ ,$$

where τ^b denotes the backward delay in the feedback path from link to source, and the link observes incoming TCP packets

$$y(t) \triangleq w(t - \tau^f) \ ,$$

where τ^f denotes the forward delay from source to link. The round trip time (RTT) $T(t)$ is defined for the source as $T(t) \triangleq d + q(t)$, where d is the constant round trip propagation time, and the round trip feedback delay is assumed to be $T(t) = \tau^f + \tau^b$.

FAST TCP periodically updates its congestion window as follows [9]

$$\mathtt{w(t + \Delta)} = \frac{1}{2}\left(\mathtt{w(t)} + \frac{\mathtt{baseRTT}}{\mathtt{RTT}}\mathtt{w(t)} + \alpha\right) ,$$

where $\mathtt{w(t)}$ is the congestion window, $\mathtt{baseRTT}$ is the minimum RTT observed, α is the tuning parameter, and $\mathtt{\Delta}$ is the update unit time. Using the Euler's method $\dot{w}(t) \approx \frac{w(t+\Delta)-w(t)}{\Delta}$, and defining $\gamma \triangleq \frac{1}{2\Delta}$, FAST TCP can be expressed as a differential equation

$$\dot{w}(t) = \gamma\left(-w(t) + \frac{d}{d + q(t)}w(t) + \alpha\right) \ .$$

Based on the self-clocking property of TCP [5] and ignoring the fast dynamics at the link, a static approximation model for the link is given as [2]

$$\frac{w(t - \tau^f)}{d + p(t)} \begin{cases} = c & \text{if } p(t) > 0 \\ \leqslant c & \text{if } p(t) = 0 \end{cases} .$$

This static model can be interpreted as the queuing delay at the link is algebraically determined by the delayed congestion window of the sources. From $q(t) \triangleq p(t - \tau^b)$ and $T = \tau^f + \tau^b$, the link model can be expressed at the source side as [2]

$$\frac{w(t - T)}{d + q(t)} = c \ . \tag{1}$$

Note that this static link model adequately describes the RTT period of delay between the congestion window $w(t)$ and measured queuing delay $q(t)$, where $q(t)$ is the delayed feedback information of $w(t)$.[1]

3 Two Different Models of FAST TCP

In this section, we introduce two different available models of FAST TCP and show that their dynamic performance and stability characteristics are significantly different.

[1] The original idea of the static link model was presented in [8] in discrete-time. However, in continuous-time, the model, as it is, does not capture the delay between $w(t)$ and $q(t)$.

As derived in Section 2, the original FAST TCP congestion control algorithm developed in [9] can be modelled as a differential equation (2). Comparing the numerical results of FAST TCP network to those of *ns-2* simulation, however, we find that numerical results do not clearly match with those of the *ns-2* simulation. Especially, in some scenarios, while the numerical simulation of the original FAST TCP model shows unstable results, the *ns-2* simulation results are still stable with the same network parameters. Taking a close look at the FAST TCP implementation of *ns-2* simulator [11], we notice that their models are slightly different as follows:

- Original FAST TCP model :

$$\dot{w}(t) = \gamma\left(-w(t) + \frac{d}{d+q(t)}w(t) + \alpha\right) \tag{2}$$

- *ns-2* implementation model :

$$\dot{w}(t) = \gamma\left(-w(t) + \frac{d}{d+q(t)}w(t-T) + \alpha\right) \tag{3}$$

To compare the stability and the dynamic performance of these two models, we conduct the simulation for a single bottleneck link utilized by a single source. The source has the propagation delay $d = 100$ ms and the tuning parameter $\alpha = 50$. And, the link capacity c is changed from 10 pkts/ms to 20 pkts/ms at 10 second. For this model, we show in Fig. 1 the numerical results as well as the *ns-2* simulation result.

As shown in Fig. 1, the original model (2) of FAST TCP is unstable even in a single-link single-source network when the tuning parameter α is not sufficiently large. In fact, FAST TCP network is guaranteed to be globally asymptotically stable when the parameter α is larger than the bandwidth-delay product cd of the network [2]. However, this condition is hard to meet in a real network because, while α implies the expected number of backlogs on the link, the buffer

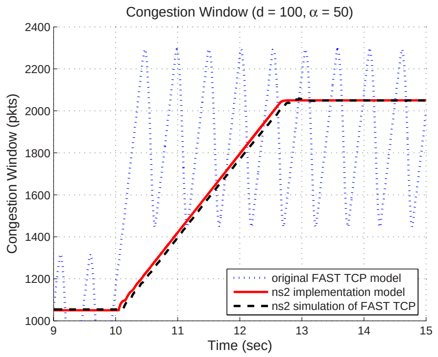


Fig. 1. Comparison of two different models of FAST TCP ($d = 100$ ms, $\alpha = 50$, and c is changed from 10 pkts/ms to 20 pkts/ms.)

size of the routers is usually not larger than bandwidth-delay product cd of the link. Moreover, the equilibrium queuing delay $\frac{\alpha}{c}$ must be larger than propagation delay d to meet this condition.

On the other hand, even though the link model (1) was not be intended to implement FAST TCP module in the *ns-2* simulator, the model (3) always shows the globally exponentially stable response when the link model (1) is adopted. Since the $w(t-T)$ term of (3) is automatically clocked to the link capacity c, the closed-loop system of the single-link single-source network simply becomes

$$\dot{w}(t) = \gamma(-w(t) + cd + \alpha) ,$$

and its solution is

$$w(t) = (w(0) - cd - \alpha)e^{-\gamma t} + cd + \alpha .$$

That is, the single-link single-source network with (3) is always globally exponentially stable. Rigorous global stability analysis of (3) in a single-link multi-source network can be found in [3].

The original model (2) of FAST TCP can be decomposed into

$$\dot{w}(t) = \gamma \left(-w(t) + \frac{d}{d+q(t)} w(t-T) + \alpha \right) + \gamma \frac{d}{d+q(t)} \left(w(t) - w(t-T) \right) ,$$

which is the *ns-2* implementation model (3) plus an additional term

$$\gamma \frac{d}{d+q(t)} \left(w(t) - w(t-T) \right) . \tag{4}$$

Using the Euler's method, we have

$$\gamma \frac{d}{d+q(t)} \left(w(t) - w(t-T) \right) \approx \gamma d \dot{w}(t-T) ,$$

where the additional term can be considered as a differential term of the PD-type controller. Hence, we can interpret that this additional differential term makes the original model (2) not only respond faster than (3) but also possible to go unstable.

However, even though the two different models (2) and (3) show different dynamic performance, they have the same equilibrium point which is proportionally fair [6,9].

4 Modified Congestion Control Algorithm

In this section, we suggest a modified FAST TCP congestion control algorithm and show that it is globally asymptotically stable by appropriately tuning the involved parameter. The main objective of the modified algorithm is to reduce the additional term (4) of (2) as small as possible while guaranteeing the stability of the algorithm. Hence, the modified algorithm is slower than the original model (2), but global asymptotic stability is guaranteed and still faster than (3).

By replacing $\frac{d}{d+q(t)}$ with a tuning parameter κ in (4), the additional term becomes

$$\gamma\kappa\left(w(t) - w(t-T)\right) \ ,$$

where $\kappa \in [0,1]$ because $q(t) \geqslant 0$, and the modified congestion control algorithm becomes as follows.

- Modified FAST TCP algorithm:

$$\dot{w}(t) = \gamma\left(-(1-\kappa)w(t) + \left(\frac{d}{d+q(t)} - \kappa\right)w(t-T) + \alpha\right) \ , \qquad (5)$$

where $\kappa \in [0,1]$.

The tuning parameter κ has a trade-off between the convergence speed and the stability. As described in Section 5, as κ grows, the convergence speed of the algorithm increases, but the overshoot also increases. Particularly when $\kappa = 0$, the modified algorithm reduces to (3). And, the modified algorithm has the same equilibrium properties as the original FAST TCP algorithm. They include the existence and uniqueness of the equilibrium, and the proportional fairness.

In order to make the network stable with the modified algorithm, we need to find the appropriate condition on the parameter κ. Adopting the static link model (1), the closed-loop system of the modified FAST TCP network becomes

$$\dot{w}(t) = \gamma\left(-(1-\kappa)w(t) - \kappa w(t-T) + cd + \alpha\right) \ , \qquad (6)$$

and by rewriting (6) in terms of the congestion window error $\tilde{w}(t) \triangleq w(t) - cd - \alpha$, we have

$$\dot{\tilde{w}}(t) = \gamma\left(-(1-\kappa)\tilde{w}(t) - \kappa\tilde{w}(t-T)\right) \ .$$

Theorem 1 shows that, whenever $\kappa \in [0, 0.5)$, the network with the modified algorithm is globally asymptotically stable independently with any delay.

Theorem 1. *If* $\kappa \in [0, 0.5)$*, then the modified FAST TCP network described by (6) is globally asymptotically stable.*

Proof. If $V(\tilde{w}(t)) = \tilde{w}^2(t)/2$, then

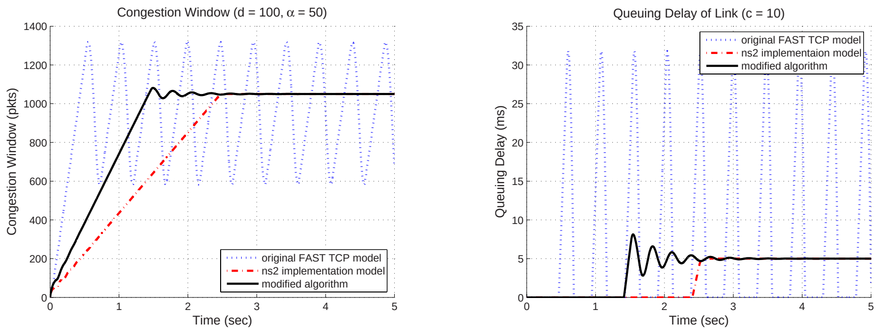
$$\begin{aligned}\dot{V}(\tilde{w}(t)) &= -\gamma(1-\kappa)\tilde{w}^2(t) - \gamma\kappa\tilde{w}(t)\tilde{w}(t-T) \\ &\leqslant -\gamma(1-\kappa)\tilde{w}^2(t) + \gamma\kappa|\tilde{w}(t)||\tilde{w}(t-T)| \\ &\leqslant -\gamma(1-2\kappa)\tilde{w}^2(t)\end{aligned}$$

whenever $|\tilde{w}(t)| \geqslant |\tilde{w}(t-T)|$. Therefore, if $\kappa < 0.5$, $\dot{V}(\tilde{w}(t)) \leqslant 0$ and we conclude from the Lyapunov-Razumikhin Theorem [4] that the modified FAST TCP network (6) is globally asymptotically stable as long as $\kappa \in [0, 0.5)$.

In contrast to the stability condition, $\alpha > cd$, of the original FAST TCP model (2), the stability condition of the modified algorithm is decoupled from the network parameters such as c, d, and α. Hence, the condition does not change the equilibrium state and only controls the convergence speed and the stability of the algorithm.

5 Simulation

In this section, we present a set of *MATLAB* and *ns-2* simulation results to illustrate the performance and the stability of the modified algorithm. The network used in the first and the second simulation test consists of a single bottleneck link utilized by a single source. We conduct the simulation test for the network with capacity $c = 10$ pkts/ms and round trip latency $d = 100$ ms.



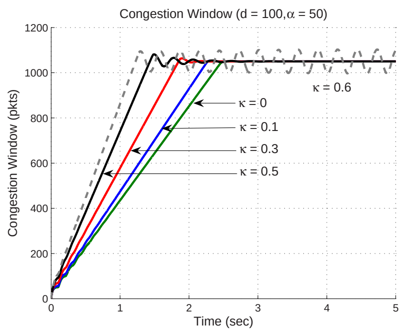
(a) Congestion window (b) Queuing delay

Fig. 2. *MATLAB* simulation results of FAST TCP networks ($d = 100$ ms, $c = 10$ pkts/ms, $\alpha = 50$, and $\kappa = 0.499$)

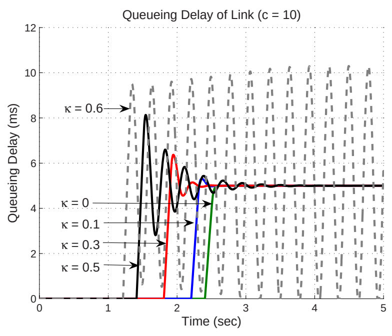
The first simulation test is conducted to show that the modified algorithm is stable in contrast to the original model (2) and still faster than the *ns-2* implementation model (3). The tuning parameter α is set to 50 for all three kinds of TCP sources to violate the condition $\alpha > cd$, and κ is set to 0.499 for the modified algorithm to satisfy the stability condition of the Theorem 1. Fig. 2 illustrates that the response of the system implementing (2) is oscillating because α does not satisfy $\alpha > cd$, but the system employing (3) or the proposed algorithm converge to their equilibrium. Moreover, we notice that the modified algorithm is faster than (3).

The second simulation test is conducted not only to validate the stability condition of Theorem 1, but also to show the trade-off of the tuning parameter κ between the convergence speed and the stability. The tuning parameter α is set to 50, and κ is increased from 0 to 0.6. Fig. 3 shows that, as κ grows, the convergence speed of the modified algorithm increases, but the overshoot also increases. And, in the case of $\kappa = 0.6$ where the stability condition $\kappa < 0.5$ of Theorem 1 is not satisfied, the modified algorithm becomes unstable.

The third simulation is conducted to illustrate that the modified algorithm can be successfully implemented in a packet-level TCP congestion control algorithm. The proposed algorithm is implemented as a module of *ns-2* simulator [10] and simulated in a network of a single bottleneck link utilized by three TCP sources as depicted in Fig. 4. Three sources have identical round trip latency



(a) Congestion window



(b) Queuing delay

Fig. 3. *MATLAB* simulation results of the modified algorithm ($d = 100$ ms, $c = 10$ pkts/ms, $\alpha = 50$ and $\kappa = 0, 0.1$, 0.3, 0.5, and 0.6)

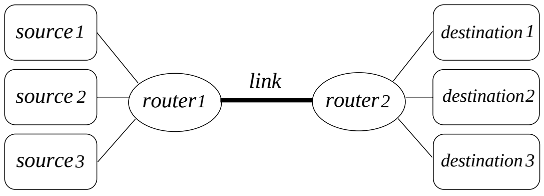
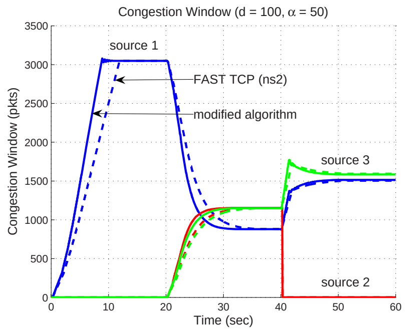
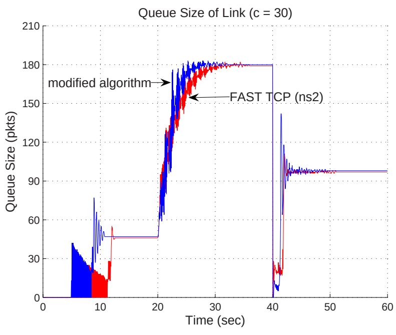


Fig. 4. Network topology for *ns-2* simulation



(a) Congestion window



(b) Queue size of link

Fig. 5. *ns-2* simulation results of the modified algorithm ($d = 100$ ms, $\alpha = 50$, and $\kappa = 0.5$ for all sources, and $c = 30$ pkts/ms)

$d = 100$ ms and the tuning parameter $\alpha = 50$. And, the bottleneck link capacity is set to 30 pkts/ms. The source 1 is active from the beginning of the simulation test, and the source 2 and 3 are activated after 20 seconds. After 40 seconds, the source 2 becomes inactive. The simulation results in Fig. 5 show that the modified algorithm is asymptotically stable and more responsive than the *ns-2* implementation version of FAST TCP.

6 Conclusions

In this paper, we introduce two different models of FAST TCP congestion control algorithm. One is from the original FAST TCP model which is used in almost all of the literature analyzing FAST TCP, and the other is from the *ns-2* implementation of FAST TCP. Interestingly, these two models have the same equilibrium properties, but they show significantly different dynamic performance and stability characteristics. That is, while the latter model is always globally exponentially stable, the former model is faster than the latter model and is possible to go unstable.

Motivated from these two models, we propose a modified FAST TCP congestion control algorithm. By tuning the gain in the algorithm, it can be made to be always globally asymptotically stable and still faster than the *ns-2* implementation version. In addition, since the stability condition of the tuning parameter is decoupled from the network parameters, the condition does not change the equilibrium state. The algorithm is proved to be globally asymptotically stable in a single-link single-source network, and the simulation results illustrate that it shows stability and good convergence performance.

References

1. Brakmo, L.S., Peterson, L.L.: TCP Vegas: end-to-end congestion avoidance on a global Internet, IEEE Journal on Selected Areas in Communications **13** (1995) 1465–1480
2. Choi, J.-Y., Koo, K., Lee, J.S., Low, S.H.: Global Stability of FAST TCP in Single-Link Single-Source Network, IEEE Conference on Decision and Control (2005)
3. Choi, J.-Y., Koo, K., Wei, D.X., Lee, J.S., Low, S.H.: Global Exponential Stability of FAST TCP, IEEE Conference on Decision and Control (2006)
4. Hale, J.K., Lunel, S.M.V.: Introduction to Functional Differential Equation, Springer-Verlag (1993)
5. Jacobson, V.: Congestion Avoidance and Control, ACM SIGCOMM (1988)
6. Kelly, F.: Charging and rate control for elastic traffic, European Transactions on Telecommunications **8** (1997) 33–37
7. Tang, A., Jacobsson, K., Andrew, L.L.H., Low, S.H.: An Accurate Link Model and Its Application to Stability Analysis of FAST TCP, IEEE Infocom (2007)
8. Wang, J., Wei, D.X., Low, S.H.: Modelling and Stability of FAST TCP, IEEE Infocom (2005)
9. Wei, D.X., Jin, C., Low, S.H., Hegde, S.: FAST TCP: motivation, architecture, algorithms, performance, IEEE/ACM Transaction on Networking **14** (2006) 1246–1259
10. The Network Simulator – *ns-2*
    `http://www.isi.edu/nsnam/ns`
11. FAST TCP Simulator Module for *ns-2* version 1.1
    `http://www.cubinlab.ee.mu.oz.au/ns2fasttcp`

Revisiting Adaptive RED: Beyond AIMD Algorithms

Richard Marquez*, Isbel González, Niliana Carrero, and Yuri Sulbarán

Departamento de Sistemas de Control, Escuela de Ingeniería de Sistemas, Facultad de Ingeniería, Universidad de Los Andes, Mérida 5101, Venezuela
marquez@ula.ve

Abstract. We propose a new MIMD Adaptive RED and revisit two well-known adaptive algorithms for ARED, (Feng et al., A Self Configuring RED Gateway, *Infocom*, 1999) and (Floyd et al., Adaptive RED). We use the steady-state relation between the maximum marking (dropping) probability, $\max_p$, and the average queue length, $\bar{q}$, to argue that different adaptive schemes, AIMD, MIMD, etc., can be proposed when $\max_p$ varies slowly. We model MIMD ARED and study stability. It is shown through *ns-2* simulations that the performance and robustness properties of MIMD ARED are in fact similar to those of Floyd's ARED.

1 Introduction

RED parameters are difficult to tune [1] due to changing and *unknown*[1] network conditions. This is also true for most active queue management (AQM) algorithms, such as the (fixed-gain) PI-AQM [2]. To overcome this problem, adaptive schemes for online parameter tuning have been proposed, see e.g. [3,4,5,6,7] and their references therein. In particular, we study in this paper two ARED (Adaptive RED) approaches: Feng et al.'s Self-Configuring RED [8] and Floyd et al.'s Adaptive RED [9].

Floyd et al. [9] proposed an additive-increase multiplicative-decrease (AIMD) policy to adapt max_p instead of the multiplicative-increase multiplicative-decrease (MIMD) approach of [8] because they considered that an AIMD algorithm is, in some sense, more robust. It is believed, see for example [10], that MIMD algorithms could "oscilate wildly" and deliver poor performance. In our opinion, this is a misconception; it hardly depends on the particular application and parameter tuning. We show that an AIMD policy is just one of many possible alternatives to define an adaptive algorithm to RED. To validate our approach, we propose a MIMD-based Adaptive RED algorithm, similar to Feng's [8], which has a performance comparable to that of Floyd's ARED [9].

* Current address: CINVESTAV-IPN, Área de Mecatrónica, Departamento de Ingeniería Eléctrica, Av. IPN No. 2509, Col. San Pedro Zacatenco, C.P. 07360, México D.F., México.

[1] Although routers are at the convergence of flows, Drop-Tail and actual AQM algorithms at routers do not dispose of enough information to estimate (compute) precise and timely network conditions.

This paper is organized as follows. First, in section 2, we model TCP flows and RED router connections. Section 3 is devoted to review Adaptive RED approaches in terms of their stability properties. We propose and model a MIMD based adaptive scheme for RED algorithm in section 4. We provide precise tuning rules for this algorithm. We illustrate the performance of the proposed MIMD algorithm through *ns-2* simulations. We finish with some conclusions and recommendations for future research.

2 TCP/RED Model

We propose a steady-state analysis of TCP-RED dynamics. Steady state refers here to make (state) derivatives equal to zero. The intuitive (and obvious) relation between the maximum marking probability, $\max_p$, and the average queue length, $\bar{q}$ is then obtain.

Fluid model. The following simplified equations represent TCP flows from N averaged (identical) sources, a queue controlled by RED, and a round-trip time RTT, see for instance [2] (TCP model is based on NewReno version of [11]):

a) System: TCP source and queue

$$\begin{aligned} \frac{dw}{dt} &= \frac{a - \left(a + \frac{2b}{2-b} w\right) wp}{\text{RTT}} \\ \frac{dq}{dt} &= \frac{Nw}{\text{RTT}} - C \\ \text{RTT} &= \frac{q}{C} + T_p \end{aligned} \tag{1}$$

b) Controlling mechanism: RED

$$\begin{aligned} \frac{d\bar{q}}{dt} &= \omega_q (q - \bar{q}) \\ p &= \max_p \frac{\bar{q} - \min_{th}}{\Delta_{th}} \end{aligned} \tag{2}$$

where $\Delta_{th} = \max_{th} - \min_{th}$.

Here w represents the congestion window size, q is the queue length, $\bar{q}$ is the filtered (averaged) queue signal, p is the packet-marking (dropping) probability, $0 \le p \le 1$; a and b are, respectively, increase and decrease TCP parameters, N is a load factor (number of TCP sources), C is the link capacity, T_p is the propagation delay, ω_q is the pole of low-pass filter in (2), $\max_{th}$ and $\min_{th}$ are, resp., maximum and minimum thresholds for $\bar{q}$. Finally, $\max_p$ is the (maximum) value of p at $\bar{q} = \max_{th}$, $0 \le \max_p \le 1$. For the sake of simplicity, delays are neglected in equation (1); these can added to further complete our analysis.

We only consider values of p on the linear region given by RED algorithm, refer to [12], i.e. $\min_{th} \le \bar{q} \le \max_{th}$, because adaptive RED mechanisms try to adapt $\max_p$ in order to maintain the system (1)-(2) in this region. See section 3.

Equations (1)-(2) represent a closed-loop system where all parameters are fixed. Let us assume that $\max_p$ is not a constant but an external control signal susceptible to be manipulated.

Analyzing the "open-loop" system. Parameter tuning. Since $\max_p$ is introduced in our context as a sort of controlled (manipulated) signal, previous system may be regarded as an "open-loop" system, so that additional controller design strategies are to be investigated.

Equilibrium state $(w^*, q^*, \bar{q}^*, p^*, \max_p^*)$ of the system (1)-(2) is then obtained in terms of the filtered queue size $\bar{q}^*$ as follows:

$$\begin{aligned} w^* &= \frac{\bar{q}^* + T_pC}{N} \\ q^* &= \bar{q}^* \\ p^* &= \frac{(2-b)aN^2}{(\bar{q}^* + T_pC)(2b(\bar{q}^* + T_pC) + (2-b)aN)} \end{aligned} \tag{3}$$

and

$$\max_p^* = \frac{\Delta_{th}(2-b)aN^2}{(\bar{q}^* + T_pC)(\bar{q}^* - \min_{th})(2b(\bar{q}^* + T_pC) + (2-b)aN)} \tag{4}$$

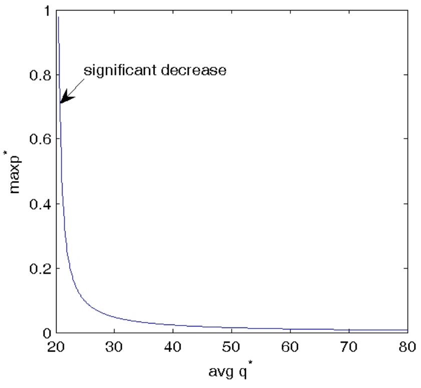
Several interesting conclusions can be deduced from the last two equations. First, in equilibrium the dropping probability p^* must be always positive. From (3), we have $p^* < 1$:

$$p^* = \frac{(2-b)aN \times N}{(\bar{q}^* + T_pC)(2b(\bar{q}^* + T_pC) + (2-b)aN)} < \frac{N}{\bar{q}^* + T_pC} < 1$$

provided that $N/C < \text{RTT}^*$. This condition is also equivalent to $w^* > 1$.

Equation (4) shows the steady-state relation between the (*desired*) average queue size $\bar{q}^*$ and the maximum marking probability $\max_p^*$. The case $\bar{q}^* < \min_{th}$ is not taken into account (it yields $\max_p^* < 0$). The value of $\max_p^*$ is undefined for $\bar{q}^* = \min_{th}$ which means that as far as $\bar{q}^*$ approaches $\min_{th}$, a varying $\max_p^*$ would grow unbounded. For $\bar{q}^* > \min_{th}$, the behavior of $\max_p^*$ is inversely proportional to $\bar{q}^*$ (monotonically decreasing): an increase of $\bar{q}^*$ results on a decrease in $\max_p^*$ which is significant when $\bar{q}^*$ is near $\min_{th}$. Notice $\lim_{\bar{q}^* \to \infty} \max_p^* = 0$. This behavior agrees with discussions and experimental results of [8].

Figure 1a shows a plot of $\max_p^*$ in terms of $\bar{q}^*$ for the parameter values shown in the appendix; in particular, $N = 60$ TCP sources. As $\bar{q}^*$ is increased, there is a significant decrease of $\max_p^*$ near $\min_{th} = 20$; $\max_p^*$ remains between 0 and 1 on the interval $\min_{th} \leq \bar{q} \leq \max_{th} = 80$. When changing the value of N to $N = 800$ sources, i.e. $N/C = \text{RTT}^*$ packets/s, $\max_p^*$ values changes dramatically. See Figure 1b. There is a narrow interval, $0.77 < \max_p^* < 1$, by means of which $\bar{q}^*$ remains less than $\max_{th}$ ($67.5 < \bar{q}^* < \max_{th}$). This agrees with the previous results related to p^*: a larger N yields larger values of p^* and $\max_p^*$, see (4). As we will see later, Floyd et al. [9] proposes to constrain the value of $\max_p$ to $0.01 < \max_p < 0.5$; these constraints make not possible to get an average queue size $\bar{q}^* = 50$ packets, below $\max_{th}$ (at $\max_p^* = 0.5$, $\bar{q}^* = 87.5 > \max_{th}$).

a) 

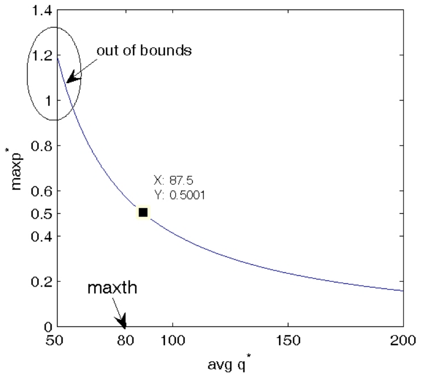
b) 

Fig. 1. $\max_p^*$ vs $\bar{q}^*$: a) $N = 60$ sources, b) $N = 800$ sources

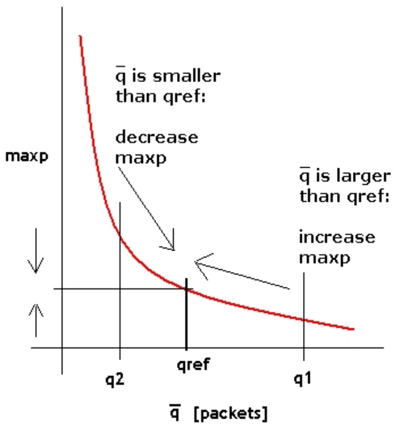


Fig. 2. $\max_p$ parameter tuning

The shape of the curve relating $\max_p^*$ and $\bar{q}^*$ may lead to a simple mechanism to adapt $\max_p$ depending only on the present value of $\bar{q}$. See Figure 2. Let q_{ref} be a desired queue size (the queue reference)[2]. Consider the case $\bar{q} = q_1 > q_{\text{ref}}$. We propose to increase $\max_p$ in order to obtain a decrease on $\bar{q}$. For the case $\bar{q} < q_{\text{ref}}$, $\max_p$ is decreased to obtain a higher value of $\bar{q}$. This results in the generic adaptation algorithm given in Listing 1.1. This algorithm is reminiscent of sliding mode control strategies, see e.g. [13]. In the following sections we discuss other ARED cases and specify precisely what 'increase/decrease $\max_p$' means.

[2] Maintaining $\bar{q} = q_{\text{ref}}$ is related to link utilization: 100% occupation of the link is obtained when the queue size q remains all the time in the interval $0 < q < q_{\max}$, which is indeed the case when the dynamics of $\bar{q}$, not too slow, attains q_{ref}.

Listing 1.1. A generic adaptation algorithm

```
if q_ave > qref
  increase maxp
else if q_ave < qref
  decrease maxp
else
  (* do nothing *)
end
```

Listing 1.2. Simplified Self-Configuring RED

```
                                     (* generic ARED *)
if q_ave > maxth                  (* if q_ave > qref       *)
  maxp = maxp * beta              (*      increase maxp    *)
else if q_ave < minth             (* else if q_ave < qref  *)
  maxp = maxp / alpha             (*      decrease maxp    *)
else
  (* do nothing *)
end
```

3 Two Adaptive RED Algorithms

We give a brief overview of two well known ARED algorithms, and compare them to our generic ARED.

3.1 Feng's ARED

The aim of Self-Configuring RED [8] is to maintain $\min_{th} < \bar{q} < \max_{th}$, see Listing 1.2. This MIMD algorithm has been simplified[3] for comparison purposes. It has two parameters: $\alpha > 1$ and $\beta > 1$ (Feng et al. [8] propose $\alpha = 3$ and $\beta = 2$ without further explanation).

Although there is no explicit target q_{ref}, Feng's algorithm behaves similarly to the previous proposed algorithm: the higher the value of $\bar{q}$, the smaller is $\max_p$ (see Figure 2). In particular, assuming $\min_{th} < q_{\text{ref}} < \max_{th}$, both 'if conditions' of Feng's ARED, i.e. $\bar{q} > \max_{th} (> q_{\text{ref}})$ and $\bar{q} < \min_{th} (< q_{\text{ref}})$, are related to those employed in the generic algorithm.

As shown, instead of considering tight bounds on the desired value of $\bar{q}$, Feng et al. [8] use a pretty large interval, which results in: first, variation of $\max_p$ depends on large values of α and β, and, second, a 'status' variable must be

[3] A variable called 'status', not shown in Listing 1.2, has been also considered in [8]: the value of $\max_p$ is increased (decreased) only when $\bar{q}$ varies around $\max_{th}$ (resp. $\min_{th}$), if $\bar{q} > \max_{th}$ (resp. $\bar{q} < \max_{th}$) in two consecutive updates nothing has to be done.

Listing 1.3. Simplified Floyd's algorithm

```
                                     (* generic ARED *)
if q_ave > target                  (* if q_ave > qref     *)
  maxp = maxp + alpha_floyd        (*      increase maxp  *)
else if q_ave < target             (* else if q_ave < qref *)
  maxp = maxp * beta_floyd         (*      decrease maxp  *)
else
  (* do nothing *)
end
```

introduced into this algorithm to prevent excessive variations on the value of $\max_p$ (see footnote 3).

3.2 Floyd's ARED

Floyd et al. [9] have modified the original proposal [8]. First, a narrow target interval is introduced $[\min_{th} + 0.4(\max_{th} - \min_{th}), \min_{th} + 0.6(\max_{th} - \min_{th})]$; this is equivalent in the generic algorithm to selecting a target value q_{ref}. Instead of using a MIMD algorithm, an AIMD algorithm is proposed,[4] where α_{floyd} is the additive increase parameter ($\alpha_{\mathrm{floyd}} = 0.01$) and β_{floyd} is the decrease factor, $\beta_{\mathrm{floyd}} = 0.9$, see Listing 1.3. In this case, compared to Feng's algorithm, we have $\alpha = 1/\beta_{\mathrm{floyd}} > 1$ (observe the little change in notation), which is greater than 1 as before. Moreover, $\max_p$ is constrained to remain within the interval $0.01 < \max_p < 0.5$. As we already mentioned, this could be too restrictive.

Whereas Feng's algorithm updates $\max_p$ on each packet arrival, Floyd's ARED proposes to update the value of $\max_p$ every Δt seconds (sampling period $\Delta t =$ 0.5s). That is, $\max_p$ is updated at a slower rate than RED average queue size $\bar{q}$. This is also the case of parameters α_{floyd} and β_{floyd} which are so selected as to allow a slowly adaptation of $\max_p$, compared to RED dynamics itself.

The work [9] also describes a set of tuning rules for parameters α_{floyd}, β_{floyd} (related to $\max_p$ dynamics) and ω_q (average queue dynamics). We will see in the next section that $\max_p$ and low-pass filter dynamics determine stability and oscillations of queue dynamics, at least on an average sense.

Remark that the analysis and conclusions of [9] are based on intuition and current network practices, supported via *ns-2* simulations; in fact, a large number of simulations illustrating the behavior of RED and the performance of ARED under a variety of scenarios is presented. As in [8], there is no explicit mention to mathematical equations (discrete or continuous) describing the behavior of such a kind of dynamical system. In the following section, we model a simple MIMD Adaptive RED and study the dynamical properties of the controlled system (1)–(2).

[4] Remark that Floyd's ARED also behaves as a MIMD algorithm under particular circumstances: increments of $\max_p$ would be given by maxp = (1+1/4)*maxp = 1.25*maxp for $\max_p < 0.04$, see [9].

Listing 1.4. MIMD ARED

```
if q_ave > qref
  maxp = maxp * kappa      (*   increase maxp   *)
else if q_ave < qref
  maxp = maxp / kappa      (*   decrease maxp   *)
else
  (* do nothing *)
end
```

4 MIMD ARED: Model, Stability, *ns-2* Implementation

Listing 1.4 shows a proposed MIMD Adaptive RED based on previous discussions. This algorithm can be applied on each packet arrival or every Δt seconds. To analyze this algorithm, we will employ a similar approach to that of [11] for TCP modelling: to translate this discrete algorithm into a differential equation by assuming a continuous dynamics (i.e. a very short sampling period).

Parameter κ = kappa can be expressed as $\kappa = 1 + \delta$ and $1/\kappa \approx 1 - \delta$, for $0 < \delta \ll 1$. If $\bar{q} > q_{\text{ref}}$ we have $\texttt{maxp}[n+1] = \texttt{maxp}[n] * (1+\delta)$. Then, the *rate of change* of maxp is given by

$$\frac{d}{dt}\texttt{maxp} \approx \frac{\texttt{maxp}[n+1] - \texttt{maxp}[n]}{\Delta t} = \frac{\delta}{\Delta t}\texttt{maxp}[n]$$

i.e.

$$\frac{d}{dt}\max{}_p = \gamma \max{}_p$$

where $\gamma = \delta/\Delta t$. Consequently, $\max_p$ dynamics can be written as follows:

$$\frac{d}{dt}\max{}_p = \gamma\,\text{sign}(\bar{q} - q_{\text{ref}})\max{}_p = \begin{cases} \gamma \max_p & \text{if} \quad \bar{q} > q_{\text{ref}} \\ -\gamma \max_p & \text{if} \quad \bar{q} < q_{\text{ref}} \end{cases} \tag{5}$$

This is a *discontinuous* differential equation which can be analyzed by the methods given in [14]. To simplify our analysis, we replace the discontinuous function sign(x) by a continuous one, given by $\frac{2}{\pi}\arctan(\lambda x)$, $\lambda \gg 1$.

Therefore, the (closed-loop) nonlinear fluid model (1)–(2)–(5) results

$$\begin{aligned}
\frac{dw}{dt} &= \frac{a - \left(a + \dfrac{2bw}{2-b}\right) w \dfrac{\max_p(\bar{q} - \min_{th})}{\Delta_{th}}}{\frac{q}{C} + T_p} \\
\frac{dq}{dt} &= \frac{Nw}{\frac{q}{C} + T_p} - C \\
\frac{d\bar{q}}{dt} &= \omega_q(q - \bar{q}) \\
\frac{d\max_p}{dt} &= \gamma \frac{2}{\pi}\arctan(\lambda(\bar{q} - q_{\text{ref}}))\max{}_p
\end{aligned} \tag{6}$$

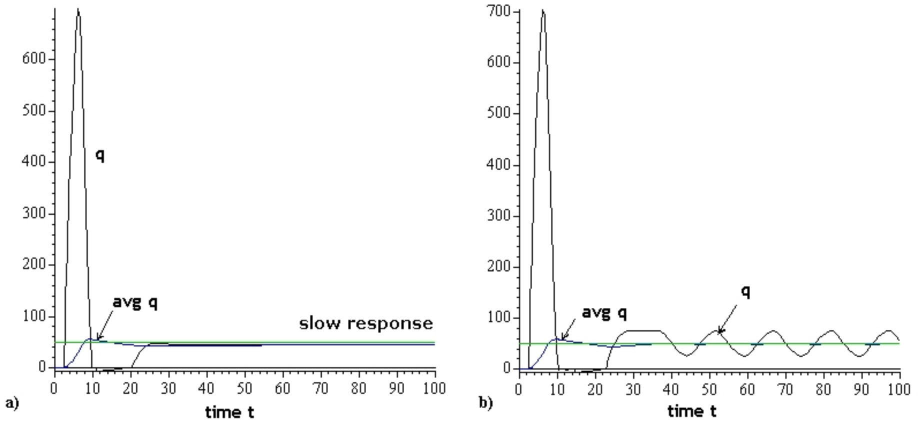


Fig. 3. Queue size $\bar{q}$ response (simulation of fluid model): a) $\gamma = 0.001$, b) $\gamma = 0.02$. Queue size is restricted to remain positive, $q \geq 0$.

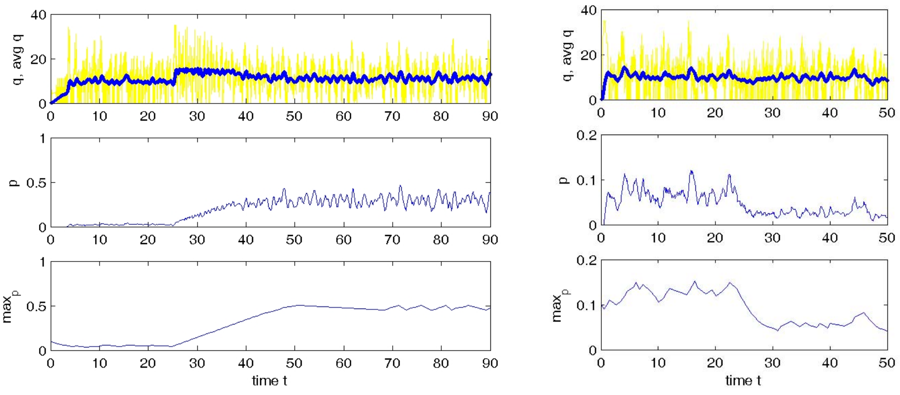


Fig. 4. Behavior of MIMD ARED (*ns-2*): q, $\bar{q}$, p, and $\max_p$ responses: (left) an increase and (right) a decrease in congestion

This model has several interesting properties from a dynamical systems point of view. Consider the parameter values in the appendix with $q_{\text{ref}} = 50$ packets. Linearization of (6) around $(w^*, q^*, \bar{q}^*, p^*, \max_p^*) = (13.3, 50, 50, 0.00758, 0.01516)$ results in the following characteristic polynomial:

$$s^4 + 5.375073738s^3 + .9414912881s + 3.236353367s^2 + 167.8587290\gamma \qquad (7)$$

This polynomial, which depends on parameter γ, permits to evaluate the local stability of (6). It is Hurwitz on the interval $0 < \gamma < 0.003194$, this means that fluid model is (local) asymptotically stable. For $\gamma > 0.003194$, the linearization becomes unstable. At $\gamma = 0.003194$, (7) exhibits two purely imaginary roots. This is indication of the presence of Hopf bifurcation. As a result, an asymptotically periodic orbit appears. Figure 3 shows a numerical simulation for two values of γ. An even higher value of γ may result in an unstable nonlinear system. A small γ leads to a slow response. Hence, we have another explanation to the parameter choices[5] proposed in [9], overcoming the tuning problems of [8].

MIMD ARED (Listing 1.4) has been implemented on *ns-2*. We illustrate the behavior of MIMD ARED by using *ns-2* tests for Adaptive RED (cf. Figures 7 and 9 of [9]). A low value[6] of δ was chosen, $\delta = 0.0008$. Then, we set `kappa_ = 1.0008` and `q_weight_ = 0.0001`. Results are shown in Figure 4. As expected, our results are quite similar to those of [9].

5 Conclusions

We obtain a MIMD-based ARED which has similar performance properties as those of Floyd's AIMD ARED. We propose continuous and discontinuous models which can be used to analyze and understand the parameter tuning and behavior of ARED. Through this model we obtain precise stability results. A similar approach can be used to model Floyd's ARED and to propose new adaptive algorithms. In forthcoming publications, we further investigate the stability properties and limit cycles of the closed-loop system (6) given a discontinuous $\max_p$ dynamics; we also evaluate the impact on *ns-2* of a nonlinear adaptation of $\max_p$, given by $\text{maxp} = \text{maxp} + \kappa \times 2/\pi \arctan(\lambda(\bar{q} - q_{\text{ref}}))\text{maxp}$, where κ is a gain factor. In the setting presented, it will be rather easy to propose classical, nonlinear or sliding mode control strategies to define $\max_p$ dynamics according to appropriate stability/performance criteria.

Acknowledgments

Authors would like to thank David Ros and Eitan Altman for their elucidating remarks and encouragement. This work has been supported by Fonacit (Venezuela) under grant S1-2002000472, French-Venezuelan collaboration ECOS-Nord under Grant V04M01 (PI-2003000207), as well as by INRIA grant ARC TCP and CONACYT (Mexico) under Grant 2007-61372.

References

1. Floyd, S.: RED: Discussions of setting parameters. Available at http://www.aciri.org/floyd/REDparameters.txt (1997)
2. Hollot, C., Misra, V., Towsley, D., Gong, W.: Analysis and design of controllers for AQM routers supporting TCP flows. IEEE Trans. Automat. Contr. **47** (2002)

[5] Moreover, the amplitude and frequency of the oscillation can be modified when increasing the value of ω_q: a higher value results in smaller amplitude but a higher frequency. This agrees with current practice. This study is not included here.

[6] As stated before, higher values of δ yields a faster response but more oscillatory.

3. Kunniyur, S., Srikant, R.: An adaptive virtual queue (AVQ) algorithm for active queue management. IEEE/ACM Transactions on Networking **12** (2004) 286–299
4. Wu, W., Ren, Y., Shan, X.: A self-configuring PI controller for active queue management. In: 7th Asia-Pacific Conference on Communications (APCC), Session T53, Tokyo, Japan (2001)
5. Zhang, H., Hollot, C., Towsley, D., Misra, V.: A self-tuning structure for adaptation in tcp/aqm networks. In: Proc. of Globecom. (2003) 3641–3646
6. Deng, X., Yi, S., Kesidis, G., Das, C.: A control theoretic approach for designing adaptive AQM schemes. In: Proc. of Globecom. (2003) 2947–2951
7. Chang, X., Muppala, J.: A stable queue-based adaptive controller for improving AQM performance. Computer Networks **50** (2006) 2204–2224
8. Feng, W., Kandlur, D., Saha, D., Shin, K.: A self-configuring RED gateway. In: Proc. of IEEE/Infocom. (1999) 1320–1328
9. Floyd, S., Gummadi, R., Shenker, S.: Adaptive RED: an algorithm for increasing the robustness of RED's active queue management. Technical report, AT&T Center for Internet Research at ICSI (2001) under submission.
10. Jacobson, V.: Congestion avoidance and control. ACM Computer Communication Review **18** (1988) 314–329
11. Marquez, R., Altman, E., Solé, S.: Time-averaging of high-speed data transfer protocols. IEEE Trans. Automat. Contr. **50** (2005) 2065–2069
12. Floyd, S., Jacobson, V.: Random early detection gateways for congestion avoidance. IEEE/ACM Transactions on Networking **1** (1993) 397–413
13. Utkin, V.: Sliding Modes in Control and Optimization. Springer-Verlag, Berlin (1992)
14. Filippov, A.: Differential Equations with Discontinuous Righthand Sides. Kluwer Academic Publishers, Dordrecht (1988)

A Parameter Values

Here are some parameter values used along the paper: $C = 3750$ packets/s $(= 15\text{Mb}/(8 \text{ bit/byte})/(500 \text{ bytes/packet}))$, $N = 60$ sources, $a = 1.0$, $b = 0.5$, $\min_{th} = 20$, $\max_{th} = 80$, $\Delta_{th} = 60$. Here we are considering similar values to those employed in [9]. Additionally, $T_p = 0.2$ s, $\omega_q = 0.02$. For $\max_p$ continuous dynamics, $\lambda = 10$.

The Practical Performance of Subgradient Computational Techniques for Mesh Network Utility Optimization

Peng Wang and Stephan Bohacek

Department of Electrical and Computer Engineering
University of Delaware, Newark DE 19716, USA
Phone.: 302-831-4274, Fax: 302-831-4316
{pwang,bohacek}@udel.edu

Abstract. In the networking research literature, the problem of network utility optimization is often converted to the dual problem which, due to nondifferentiability, is solved with a particular subgradient technique. This technique is not an ascent scheme, hence each iteration does not necessarily improve the value of the dual function. This paper examines the performance of this computational technique in realistic mesh network settings. The traditional subgradient technique is compared to a subgradient technique that is an ascent algorithm. It is found that the traditional subgradient techniques suffer from poor performance. Specifically, for large networks, the convergence is slow. While increasing the step size improves convergence speed, due to stability problems, the step size cannot be set arbitrarily high, and suitable step sizes result in slow convergence. The traditional subgradient technique also suffers from difficulties when used online. The ascent scheme performs well in all respects, however, it is not a distributed technique.

Keywords: Network capacity optimization, subgradient techniques.

1 Introduction

There has been extensive effort focused on finding time division multiplexing schedules that maximize the capacity of wireless networks [1]- [9]. A common approach is to maximize the sum of flow utilities subject to constraints related to interference. Specifically, we consider

$$\min -\sum_{\phi\in\Phi} U_\phi(f_\phi) \tag{1}$$

$$\text{subject to: } \sum_{\{\phi|l\in P(\phi)\}} f_\phi \le \sum_{v\in V} \alpha_v R(v,l) \text{ for all } l \text{ and } \sum_{v\in V}\alpha_v = 1,\ \alpha_v \ge 0.$$

where f_ϕ is the data rate of flow ϕ, $U_\phi(f_\phi)$ is the utility of flow ϕ when the flow rate is f_ϕ, $\mathcal{P}(\phi)$ is the set of links that flow ϕ traverses (i.e., $\mathcal{P}(\phi)$ is the path of flow ϕ), $R(v,l)$ is the data rate over link l when assignment v is used, and

α_v is the duration that assignment v is used. We define an *assignment* to be a specification of which links transmit and the transmit powers. Thus, a schedule is a weighted combination of assignments where the weights are α_v. We let V denote the set of considered assignments. If power control is not used, then there are 2^L distinct assignments, where L is the number of links in the network, and if power control is used, the space of assignments is $[0,1]^L$. In [1], a technique is presented that generates a small set V that results in nearly optimal utility. Hence, currently, utility optimization is tractable for networks with hundreds of links. Most efforts to solve (1) use dual or primal-dual techniques. Specifically, after some manipulation, the dual function is written as

$$q(\mu) = \sum_{\phi \in \Phi} \inf_{f_\phi \geq 0} \left(-U_\phi(f_\phi) + f_\phi \sum_{l \in \mathcal{P}(\phi)} \mu_l \right) - \max_{v \in V} \sum_{l=1}^{L} R(v,l)\mu_l, \tag{2}$$

where μ_l is the Lagrange multiplier associated with link l. The dual problem is

$$\max_{\mu \geq 0} q(\mu). \tag{3}$$

Due to the term $\max_{v\in V} \sum_{l=1}^{L} R(v,l)\mu_l$, the dual function, q, is not differentiable for all μ. Hence, computational methods based on the gradient are not available. To circumvent this difficulty, supergradient[1] techniques can be employed. In the networking literature [2]-[9], the most popular supergradient technique is to iterate

$$\mu_l(k+1) = \left(\mu_l(k) + \gamma_k \left(\sum_{\{\phi | l \in \mathcal{P}(\phi)\}} f_\phi^*(\mu(k)) - R(v(k), l) \right) \right)^+ \tag{4}$$

where

$$v(k) \in \arg\max_{v \in V} \sum R(v,l)\mu_l(k), \tag{5}$$

γ_k is a step size, and $f_\phi^*(\mu(k))$ is the optimal flow given $\mu(k)$, i.e., f_ϕ^* is the solution to the infimum in (2). Since this scheme is widely used, it will be referred to as the *traditional supergradient scheme.*

This paper examines the practical performance of (4) through extensive computational experiments. The conclusions are that the traditional supergradient scheme suffers from poor performance. Specifically, for large networks, the convergence is slow. While increasing the step size improves convergence speed, due to stability problems, the step size cannot be set arbitrarily high, and suitable step sizes result in slow convergence. On the other hand, this method does not find the exact solution, but merely oscillates around the optimal solution. However, the oscillations are small, hence in terms of error, (4) works well. Often the traditional supergradient techniques are used for online and distributed

[1] Subgradient is a more common term. However, subgradient and supergradient techniques are the same, the only difference is that the former refers to minimization while the later refers to maximization, which is the focus here.

computation. However, this approach suffers from several problems. Finally, an alternative ascent algorithm is also investigated. While this approach does not appear to lend itself to distribution, it does perform well in all other aspects.

The remainder of the paper proceeds as follows. In the next section, a few theoretical aspects of supergradient based optimization are presented. In Section 3, some details of the computational experiments are provided. The rest of the paper is focused on the performance of the traditional supergradient scheme, specifically, Section 4 examines the convergence rate, Section 5 examines the error, Section 6 examines stability, and Section 7 examines the performance when the traditional supergradient scheme is used as an online and distributed computational method. Finally, Section 8 provides some concluding remarks.

2 Theoretical Results on Supergradient Optimization

The performance of (4) has been extensively investigated (e.g., see [10]). In [8], the following is proved.

Theorem 1. *Let γ_k be a constant γ and let $G = \max_\mu \|\partial q(\mu)\|$, where $\|\partial q(\mu)\|$ is the norm of the largest element in the superdifferential $\partial q(\mu)$ and let $u(k)$ be given by (4). Then*

$$\lim_{K\to\infty} \sup \frac{1}{K} \sum_{k=1}^{K} |q(\mu(k)) - q(\mu^*)| < \gamma G^2/2.$$

Thus, one can expect that if a fixed step size is used, then $\mu(k)$ will enter a ball around μ^* and remain in this ball, where μ^* is the solution to (3). Hence, using the terminology of [8], we can consider that the $\mu(k)$ has *stochastically converged* when it enters this ball. The ball can be made smaller by using a smaller step size. In fact, by slowly decreasing the step size, this scheme will converge. However, in order to guarantee convergence, the step size must converge slowly. Specifically, in general, we must have $\lim_{k\to\infty} \gamma_k g_k = 0$ and $\sum_{k=1}^{\infty} 1/(\gamma_k g_k)^2 = \infty$, where $g_k = \left(\sum_l \left(\sum_{\{\phi | l\in\mathcal{P}(\phi)\}} f_\phi^*(\mu(k\Delta t)) - R(v(k), l)\right)^2\right)^{1/2}$ [10].

When $q(\mu)$ is not differentiable, the superdifferential, $\partial q(u)$, is a set of vectors. The algorithm (4) arbitrarily selects one element from the superdifferential and uses it as if it was a direction of ascent. As just mentioned, if the step size is selected correctly, then this scheme will converge. However, it is possible to more carefully select the direction so that it is a direction of ascent[2].

Theorem 2 (Thm 1.11 in [10]). *Let $\partial q(\mu)$ be the superdifferential of q at μ. Suppose $0 \notin \partial q(\mu)$ and let η be the element of $\partial q(\mu)$ that is nearest to the origin, i.e.,*

$$\eta = \arg\min \|g\|^2 \qquad (6)$$
$$\text{subject to: } g \in \partial q(\mu).$$

Then η is a direction of steepest descent at μ.

[2] Note due to convexity, there must be a direction of ascent, unless $\mu(k) = \mu^*$.

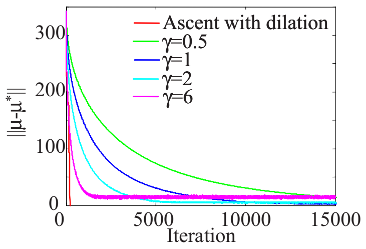


Fig. 1. Example of the convergence of (4) for a 60 link network

Therefore, steepest ascent is an alternative computational scheme to the traditional supergradient scheme (4). However, as is well known, steepest ascent can lead to oscillations that result in slow convergence. The steepest ascent algorithm can be further improved by using space dilation (see page 69 in [10]). We refer to this approach as the *ascent algorithm*. More details on the ascent algorithm can be found in [1]. Section 4 compares the convergence rate of this ascent algorithm to the traditional supergradient algorithm (4). In the other sections, this ascent algorithm is used to find μ^*, the optimal solution to (3) as well as optimal flow and link rates.

3 Experiment Set Up

The performance of (4) and the ascent algorithm were examined in realistic mesh network scenarios that were based on downtown Chicago. Specifically, random mesh networks were generated by placing one infrastructure node randomly on each block in a region of downtown Chicago. A centrally located infrastructure node was designated as the base station. All other nodes were set to be wireless relays. These wireless relays were also set as destinations. Hence, for each relay, there was one flow from the base station to the relay. Shortest path routing was used, where the channel loss along each hop was required to be no more than 55 dB. The propagation was determined from the UDelModels ray-tracing tool [11]. If some relays were disconnected from the network, then the relay was excluded from the topology. Finally, by adjusting the size of the region of Chicago where the mesh network was constructed, the number of links could be approximately controlled. Topologies were grouped together based on the number of links. Twenty topologies were generated for each number of links, where the number of links ranged from 15 to 75 links in steps of five links.

As mentioned in the Introduction, when there are L links, there are 2^L possible assignments. Hence, for large topologies it is intractable to consider all possible assignments. Instead, the scheme described in [1] was used to construct a good set of assignments. In [1], it is shown that this technique results in network utility that is within 0.05% of optimal. Thus, the set V in the Introduction was set to be this set of good assignments.

Finally, the utility function used was $U(f) = \log(f)$ and data rates were given by Shannon's Theorem, i.e., $\log_2(1 + SNIR)$ bps/Hz.

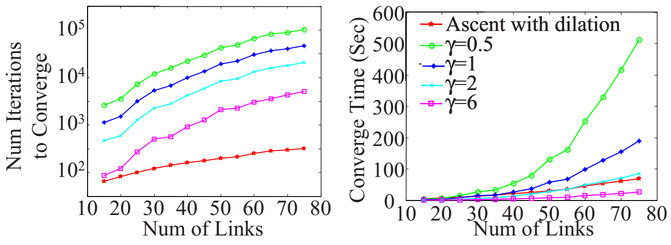


Fig. 2. Left: Number of Iterations until convergence. Right: Computation time on an 2.8GHz 64 bit PC until convergence.

4 Convergence Rate

There are few theoretical results on the convergence rate of (4). However, it is intuitive that a smaller step size results in a slower convergence. Figure 1 shows examples of $\|\mu(k) - \mu^*\|$ for (4) with several step sizes and for the ascent algorithm. Note that the ascent algorithm will eventually converge, hence the curve representing the ascent algorithm is only shown for the iterations before convergence.

We will say that the traditional supergradient scheme has converged when

$$\|\mu(k) - \mu^*\| \leq \lim_{k\to\infty} E(\|\mu(k) - \mu^*\|).$$

Once this condition has been met, we can assume that $\mu(k)$ remains in a ball around μ^* and the flow and link rates will be approximately correct.

Figure 2 shows the number of iterations until convergence and the computation time until convergence. Here we assume the computation is performed centrally. Thus, the computation time for one iteration is the time to update μ_l for each link.

The left-hand frame of Figure 2 shows that the convergence time does not grow exponentially with the number of links. However, the right-hand frame shows a superlinear growth in the convergence time with the number of links. On the other hand, the ascent algorithm shows a slower growth than the traditional supergradient method. To see this, note that for $\gamma = 2$, the traditional supergradient method converges in less time than ascent algorithm when the number of links is small, but requires more time when the numbers of links is large. Similarly, while $\gamma = 6$ takes less time than the ascent algorithm when there are 75 or fewer links, it takes more time for large networks. For example, we found for a set of networks with 280 links, the traditional supergradient method with $\gamma = 6$ takes approximately 1834 seconds, whereas the ascent algorithm takes approximately 816 seconds.

5 Error

The relationship between the number of iterations to reach convergence and the step size indicates that if γ is selected very large, then convergence will be very

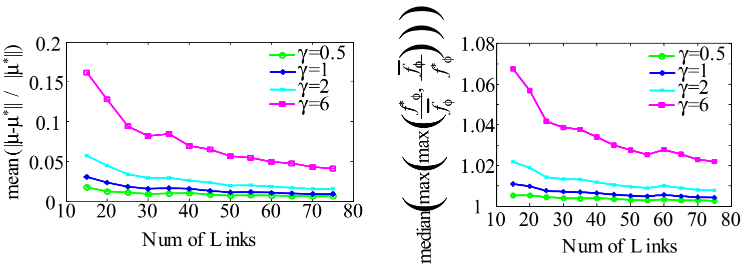


Fig. 3. Left: The relative error of μ. The mean is over all topologies with L links. Right: The ratio of average flow rates to optimal flow rate.

fast. On the other hand, Theorem 1 indicates that the error $\|\mu(k) - \mu^*\|$ grows with γ.

To investigate this, we consider the average relative error after convergence (i.e., the average value of $\|\mu(k) - \mu^*\| / \|\mu^*\|$ for very large k). The left-hand frame of Figure 3 shows that the relative error is quite small and decreases with the number of links.

The right-hand frame of Figure 3 provides another view of the error. To understand this plot, recall that given $\mu(k)$, the flow rates, $f_\phi(k)$, can be determined. If $\mu(k)$ differs from μ^*, then $f_\phi(k)$ will differ from f_ϕ^*. To examine the size of this difference, we compare f_ϕ^* and $\bar{f}_\phi$, the average value of $f_\phi(k)$ for k very large, i.e., after convergence. Specifically, we examine

$$\underset{\text{over all topologies}}{\text{median}} \; \max_{\phi} \max\left(\frac{f_\phi^*}{\bar{f}_\phi}, \frac{\bar{f}_\phi}{f_\phi^*}\right).$$

Note that the inner maximization forces the ratio to always be greater than one. The outer maximization is the maximization over all flows, i.e., the worst case flow. The median is taken over all topologies.

In both views of the error, we see that the error decreases with the number of links. Further investigation is required to understand why this is the case. Nonetheless, in both cases the error is quite small.

6 Stability

Section 4 showed that the time to convergence decreases when the step size, γ, is increased. Furthermore, the previous section showed that the error is quite small even for $\gamma = 6$. Moreover, the error decreases with the number of links. Hence, increasing the step size may improve convergence while maintaining acceptable error. However, we find that large step sizes can lead to instability and divergence.

For a particular topology, we define $\bar{\gamma}$ to be the maximum value of γ such that (4) is stable. Figure 4 shows minimum value of $\bar{\gamma}$ where the minimum is over all topologies with L links. Figure 4 also shows the median value of $\bar{\gamma}$ over all topologies with L links. While Figure 4 indicates that in some cases large

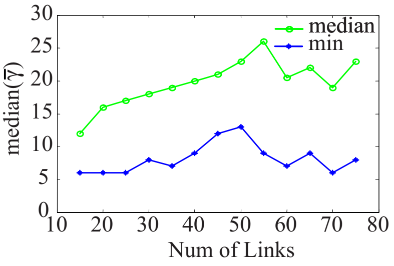


Fig. 4. The median and minimum value of $\overline{\gamma}$, the maximum allowable value of γ such that the traditional subgradient method is stable, i.e., does not diverge. The median and minimum are over all topologies with L links.

values of γ might not cause instability, there are other topologies such that γ must be rather small. Indeed, from Figure 4, we conclude that it is not possible to reliably set γ larger than 6.

7 Online and Distributed Supergradient Optimization

In this section the possibility of distributing the supergradient optimization in such a way that it supports online computation of assignments. By online we mean that at each iteration, the assignment $v(k)$ is used, i.e., the link bit-rates are $R(v(k), l)$. This assignment is used for Δt seconds before a new assignment, $v(k+1)$, is generated. Note that Δt is not necessarily the same as γ. However, here we assume that $\gamma = \Delta t$.

We assume that the computation of a new assignment requires communication with neighboring nodes. Recall that we assume that the $v(k) \in \arg\max_{v \in V} \sum R(v, l) \mu_l(k)$. Thus, in order to compute $v(k)$, each link must be aware of $\mu_l(k)$ for all other links. Consequently, each iteration can be expensive in terms of bandwidth, the resource that is being optimized. In order to preserve bandwidth, one can set Δt large. In this section, the values of Δt studied range from 500 msec. to 6 sec. Refer to Figure 2 for the number of iterations required for convergence. For example, with 75 links and $\Delta t = 500$ msec, it will take 50000 seconds until convergence.

Besides slow convergence, there are two performance problems with the online approach, namely, the actual link utilization of congested links may be small and queues occupancies can be very large. These problems are discussed next.

7.1 Link Utilization

When using TDM, a link is not able to transmit at all times. However, for some time-slots, the link is able to transmit and it is expected that the link will transmit continuously during that time-slot. If the link is unable to transmit data throughout the entire time-slot, then it might be possible to either increase the flow rates or use different assignment so that other links can transmit. If either of these options is possible, then the network utility can be increased. On the other hand, it is possible that at optimality a link will have more bandwidth allocated to it than is required to transmit the data passing over it. However, from complementary sensitivity, for link l where this occurs, we must have $\mu_l^* = 0$. Thus, if $\mu_l^* > 0$, then we expect that link l will always send data when it is allocated bandwidth, that is, the link will be fully utilized.

Since a radio cannot simultaneously transmit and receive on the same bandwidth, when a node is transmitting, it must transmit data that is stored in its queue. Thus, letting $Q_l(k)$ denote the queue occupancy of link l at the beginning of the kth time-slot, a link is underutilized if $Q_l(k) < \Delta t R(v(k), l)$, i.e., more data can be sent than is available in the queue. Thus, we define the utilization of a link to be

$$\rho_l := \frac{\sum\limits_{\{k:R(v(k),l)>0\}} \min\left(Q_l(k), \Delta t R(v(k), l)\right)}{\sum\limits_{\{k:R(v(k),l)>0\}} \Delta t R(v(k), l)},$$

where $\{k : R(v(k), l) > 0\}$ is the set of time-slots for which link l is transmitting. In the analysis that follows, the utilization is computed once the algorithm has stochastically converged.

We approximate the queue occupancy with the following

$$Q_l(k+1) = \min\left(Q_{\max}, \left(Q_l(k) + \mu_l(k+1) - \mu_l(k)\right)^+\right) \tag{7}$$

where $Q_{\max}$ is the size of the queue. Note that if $Q_{\max} = \infty$ and $\mu(0) = Q(0)$, then $\mu_l(k) = Q_l(k)$ for all k. Also, note that (7) is only an approximation of the queue occupancy since it assumes that the arrival flow rate for link l is $\sum_{\{\phi:l\in\mathcal{P}(\phi)\}} f_\phi^*(\mu(k))$. However, upstream queue overflows could result in arrival rates less than $\sum_{\{\phi:l\in\mathcal{P}(\phi)\}} f_\phi^*(\mu(k))$. Nonetheless, the analysis that follows uses (7).

There are two ways in which the traditional supergradient method results in congested links having utilization that is less than one. First, as shown in Figure 3, the larger that γ is, the larger the variations experienced by $\mu(k)$, and hence the larger the variations experienced by $Q(k)$. Consequently, when μ_l^* is small for some link l, variations in $\mu_l(k)$ around μ_l^* will occasionally result in $\mu_l(k) = 0$. Similarly, occasionally $Q(k) = 0$, and hence $\rho_l < 1$.

While $Q(k) = 0$ will result in $\rho_l < 1$, this problem is most significant for links with small μ_l^*. Considering sensitivity analysis[3], these links with small μ_l^* are not as critical as links with larger μ_l^*. Hence, if these less critical links do not reach full utilization, it will not have a significant impact on the network utility unless Δt is quite large (in which case, occasionally we will have $Q_l(k) = 0$ for links with large μ_l^*). However, as discussed in Section 6, due to instability, it is difficult to have Δt large.

[3] By sensitivity, the Lagrange multiplier μ_l is related to the change in the network utility due to a change in link resources. Hence, if μ_l is small, then decreasing the data rate across link l will only have a small impact on the network utility. Thus, if μ_l^* is small, then link l is less critical to the network utility.

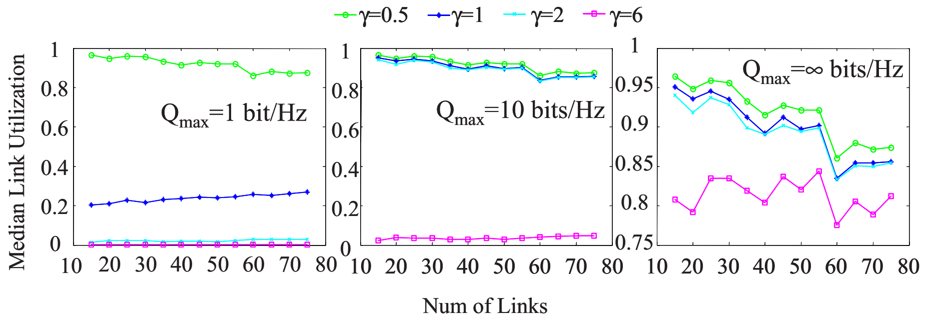


Fig. 5. Traditional subgradient methods can result in link utilization of less than one for congested links (i.e., links with $\mu_l > 0$). The above shows the median link utilization where the median is over all links and all sampled topologies with L links. As the $Q_{\max} \to \infty$, the median link utilization converges.

Finite queue sizes is a second cause of reduced link utilizations. For example, since a node cannot send and transmit at the same time, if the maximum queue size is zero (i.e., there is no queue), then $\rho = 0$. In general, finite queue size is not a problem if $Q_{\max} \geq \max_l \max_{v \in V} \Delta t R(v, l)$. This condition can be conservative since links with very high data rates might not be critical links (i.e., $\mu_l^* = 0$), and not all assignments v are used.

Figure 5 shows the median link utilization for different maximum queue sizes, $Q_{\max}$ (only links with $\mu_l^* > 0$ are considered). As expected, for very small sizes of $Q_{\max}$, the utilization is quite low. This is due to $Q_{\max} < \Delta t R(v, l)$ for some l and some v that is used by the schedule. When $Q_{\max} = \infty$, then the utilization is less than one due to $Q_l(k) = 0$ for some k and l. As can be seen, the median link utilization is far from one in all cases.

7.2 Queue Size and Delay

An important drawback of the online implementation of the supergradient method is that the link cost, μ_l, is tightly associated with the queue occupancy, Q_l. In the typical approach, $\mu_l = Q_l$. This is problematic since if the link cost is high, then the queue occupancy will be large, resulting in long delays and consuming large amounts of memory resources. For example, in our experiments, it was not uncommon to have $\mu_l^* > 100$ bits/Hz. If the bandwidth is 20 MHz, as is the case in 802.11b/g, this would result in nominal queue occupancies of 2Gb. As discussed above, limiting the queue to smaller values decreases link utilization.

Another possible option is to somehow force $Q_l(k) = \mu_l(k) - \mu_l^*$. In this case, the queue is nominally empty and only grows when $\mu_l > \mu_l^*$. In this case, delay is only caused by positive variations in μ_l. However, as shown in Figure 6, even in this ideal situation, we find that the queue must be large. Indeed, when $\Delta t = 500$ msec and the bandwidth is 20 MHz, we have some queue occupancies that exceed 10 Mb.

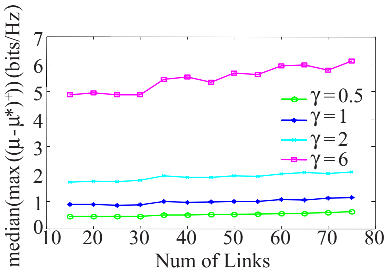


Fig. 6. Median of the Maximum Positive Deviation of $\mu - \mu^*$. The maximum is over all links in the topology and over all time, and the median is over all topologies with L links.

8 Conclusions

It is common to use a particular supergradient technique to maximize network utility. This paper examines the performance of the traditional supergradient technique and finds that in practice, it performs poorly. Specifically, convergence is slow, and instability results if the step size is increased in an attempt to improve convergence speed. An alternative ascent algorithm is found to converge much faster. Another problem with the traditional supergradient approach is that if it is distributed, then queue occupancies can become very large and link utilization of critical links is below one. On the other hand, while the supergradient methods do not provide the exact solution (they oscillate around it), the error is small.

References

1. S. Bohacek and P. Wang, "Toward tractable computation of the capacity multihop wireless networks," in *Infocom*, 2007, available at: http://udelmodels.eecis.udel.edu/publications1.php.
2. M. Chiang, "Balancing transport and physical layers in wireless multihop networks: Jointly optimal congestion control and power control," *IEEE Journal on Selected Areas in Communications*, vol. 23, pp. 104–116, 2005.
3. R. Cruz and A. Santhanam, "Optimal routing, link scheduling and power control in multi-hop wireless networks," in *IEEE INFOCOM*, March 2003.
4. J. Yuan, Z. Li, W. Yu, and B. Li, "A cross-layer optimization framework for multihop multicast in wireless mesh networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 11, pp. 2092–2103, Nov 2006.
5. N. Jin, G. Venkitachalam, and S. Jordan, "Dynamic congestion-based pricing of bandwidth and buffer," *IEEE/ACM Transactions on Networking*, vol. 13, no. 6, pp. 1233–1246, Dec 2005.
6. D. S. Lun, N. Ratnakar, M. Médard, R. Koetter, D. R. Karger, T. Ho, E. Ahmed, and F. Zhao, "Minimum-cost multicast over coded packet networks," *IEEE/ACM Transactions on Networking (TON)*, vol. 14, pp. 2608 – 2623, June 2006.
7. X. Wang and K. Kar, "Cross-layer rate control for end-to-end proportional fairness in wireless networks with random access," in *MobiHoc*, May 2005.

8. L. Chen, S. H. Low, M. Chiang, and J. C. Doyle, "Cross-layer congestion control, routing and scheduling design in ad hoc wireless networks," in *Infocom*, 2006.
9. X. Lin and N. B. Shroff, "The impact of imperfect scheduling on cross-layer congestion control in wireless networks," *IEEE/ACM Transactions on Networking*, vol. 14, no. 2, pp. 302–315, April 2006.
10. N. Z. Shor, *Minimization Methods for Non-Differentiable Functions.* Berlin: Springer-Verlag, 1985, p69.
11. S. Bohacek, V. Sridhara, and J. Kim, "UDel Models," available at: http://udelmodels.eecis.udel.edu/.

Channel Dependent Interference and Decentralized Colouring

P. Clifford and D.J. Leith

Hamilton Institute, NUI Maynooth, Ireland

Abstract. We consider channel allocation to mitigate interference between wireless LANs. The channel allocation task is often formulated in the literature as finding a proper colouring of a single graph. We show that the interference between WLANs can be channel dependent in which case a different conflict graph is associated with each channel. Channel allocation then corresponds to a multi-graph colouring problem. This potentially has profound implications as the behaviour of many proposed colouring-based algorithms for channel allocation is unclear in a multi-graph context. We prove that a recently proposed decentralized colouring algorithm performs correctly in the multi-graph setting. We also present a new, extended version of this algorithm suited to a wide range of multi-radio architectures.

1 Introduction

We consider how a group of access-points/base-stations[1] can configure their channel choice so as to minimise interference between one another. This problem has recently been the subject of an upsurge of interest in the WLAN literature, e.g. see [2,3,4,5,6,7,8,9,10,11,12,13]. The channel allocation task is often formulated in the literature as finding a proper colouring of a single graph. That is, a conflict graph is constructed by associating a graph vertex with each WLAN and inserting edges between WLANs that interfere. A non-interfering channel allocation then corresponds to a proper colouring of this conflict graph. We demonstrate that this formulation may be unrealistic. Specifically, we show that the interference between WLANs can be channel dependent in which case a different conflict graph is associated with each channel. Channel allocation then corresponds to a *multi-graph* colouring problem. This potentially has profound implications as the behaviour of many proposed colouring-based algorithms for channel allocation is unclear in a multi-graph context.

Our second main contribution is to establish that a recently proposed decentralized colouring algorithm does indeed generalise to the multi-graph setting. We also present a new, extended version of this algorithm suited to a wide range of multi-radio architectures.

[1] We use the term access point or AP to denote the co-ordinating station in a WLAN that is responsible for channel selection. Consideration is not restricted to a specific WLAN technology. Each AP has associated wireless client stations and we refer to the collection of clients plus AP as a WLAN.

2 Channel Allocation and Graph Colouring

The channel allocation task is usually formulated as a standard graph colouring problem. For example, Figure 1 shows four interfering WLANs. Transmissions within the AP1 and AP2 WLANs can interfere, with the interference range of each WLAN indicated by the dashed circles in Figure 1. The level of interference between any particular pair of transmissions depends on the physical locations of the communicating stations. This can easily lead to complex hidden/exposed terminal problems. For example, if AP2 transmits data to client 1 at the right-hand edge of the figure at the same time as the client 2 station located at the left-hand edge of the figure sends data to AP1, then reception by AP1 may be blocked by AP2's transmission while AP2's transmission is successfully received at the right-hand station as this is beyond the interference range of AP1. This is, of course, an example of hidden terminal behaviour, known to have the potential to induce gross unfairness and reduced network utilisation. Similarly AP3 and AP4's transmissions can interfere creating further potential for four-way hidden/exposed terminal behaviour.

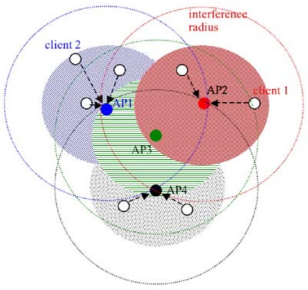


Fig. 1. Example of interfering 802.11 WLANs. Dashed circles indicate interference radius, shaded circles indicate communication radius.

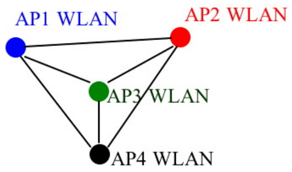


Fig. 2. Interference graph of Figure 1

The underlying channel selection problem in this example is equivalent to graph colouring. To see this, define the interference graph by associating a node with each WLAN (e.g. with each BSS in an 802.11 network) and inserting an edge between nodes that interfere. For example, Figure 2 shows the interference graph corresponding to the wireless network in Figure 1. A colouring of the graph assigns colours to each node, and a proper colouring is an assignment of colours to each node such that no adjacent nodes share the same colour. A non-interfering channel allocation is then equivalent to a proper colouring of the interference graph associated with a wireless network. Similar considerations also apply in multi-hop multi-radio situations. For example, AP3 might be a multiradio intermediate relay station with AP4 the only access point with a wired backhaul link.

3 Channel Dependent Interference

It is important to stress here that the use of circles to denote interference regions in Figure 1 is an idealisation. Importantly, we note that since channel characteristics are dependent on the frequency used, we can expect that the shape of the interference regions will be *channel dependent*.

To investigate this question, we took measurements on an experimental testbed. The testbed consists of 10 PC-based embedded Linux boxes based on the Soekris net4801, 5 boxes configured as APs in infrastructure mode and 5 as client stations. We also use 5 PCs acting as monitoring stations to collect measurements – this is to ensure that there is ample disk space, RAM and CPU resources available so that collection of statistics does not impact on the transmission of packets. These machines are setup as five WLANs (denoted WLAN A - WLAN E) located in a university office space. All systems are equipped with an Atheros 802.11a/b/g mini-PCI card with an external antenna. All nodes use a Linux 2.6.16.20 kernel and the MADWiFi wireless driver. All of the systems are also equipped with a 100Mbps wired Ethernet port, which is used for control of the testbed from a PC. Specific vendor features on the wireless card, such as turbo mode and channel scanning, are disabled.

The testbed hardware supports operation both in the 802.11a 5GHz band and in the 802.11b 2.4GHz band. While spectrum analyzer measurements revealed little external interference in the 5GHz band (a noise floor of around -80dB being typical), significant external interference was observed in the 2.4GHz band which is attributed to bluetooth devices . Focussing on the 5GHz band, our measurements indicated that the level of interference between WLANs can be strongly channel dependent. For example, the measured interference level between WLANs B and C as the channel number is varied (with WLANs B and C always sharing the same channel), varied from 0 on channel 36, to 27% on channel 56, and back to 1% on channel 64. This behaviour is perhaps unsurprising as we can expect path propagation characteristics to be frequency dependent. Nevertheless, it has profound implications for channel allocation algorithms. In particular, it is in general not sufficient to confine consideration to a single conflict graph as shown for example in Figure 2, but rather a different conflict graph may be associated with each available frequency channel. An immediate consequence is that the channel allocation problem is not necessarily equivalent to the standard colouring task on a single graph, but rather may involve a more general multi-graph colouring task.

4 Implications for Channel Allocation Algorithms

The chromatic number (minimum number of colours for a proper colouring) of the multi-graph problem is only weakly related to the constituent individual graphs. We illustrate this by example.

Figure 3 shows the conflict graphs associated with channels 1,2 and 3 in a network of 6 interfering WLANs and also shows a successful channel allocation

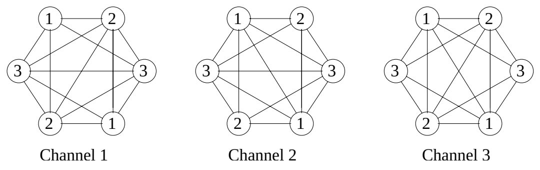


Fig. 3. Multi-graph example 1. Individual channel conflict graphs shown with proper colouring requiring only 3 colours.

using 3 channels. Although for each channel there is only one pair of nodes which do not interfere, the arrangement is such that only three channels are necessary to avoid interference, rather than the six which would be required if every node interfered with every other on every channel. This example demonstrates that the problem of multi-graph colouring is dramatically different to normal graph colouring. To our knowledge, no analytic results are available on the performance of colouring algorithms on multi-graphs. Existing convergence proofs for distributed algorithms such as those in [8,10,13] relate to colouring of a single graph. Centralised channel allocation algorithms based on single graph colouring may exhibit unexpected behaviour in a multi-graph context.

Channel dependent interference also has direct implications for frequency hopping approaches to channel allocation such as that in [5] and elsewhere. The performance of heuristic algorithms is unclear.

5 Main Result

We refer to [13] for the decentralized channel allocation algorithm (Section 6 in this paper contains a generalisation). Let $G(i) = (V, E(i))$ denote the interference graph associated with use of channel i in a wireless network. That is, the vertices V of $G(i)$ are the network WLANs and the edge set $E(i)$ contains an edge between vertices (u, v) when WLAN u and v interfere on channel i. The interference environment is then characterised by the family of graphs $\{G(i), i \in [1, 2, .., c]\}$. A non-interfering channel allocation is one where each WLAN uses a channel i that is different from all of its neighbours in $G(i)$. Note that in the special case where $G(i) = G \forall i$ then the interference graph is the same on every channel and we recover a standard single graph colouring problem.

Theorem 1. *Suppose each vertex in V operates the CFL algorithm. Assume that the channel allocation problem is feasible (i.e. a non-interfering channel allocation does indeed exist). Then the CFL algorithm converges, with probability one, to a non-interfering channel allocation.*

Our proof also provides a partial answer to a further question, namely how quickly the algorithm converges to a non-interfering allocation. The stopping time is the time taken for the algorithm to converge. We have the following property.

Corollary 1. *Let τ denote the stopping time of the CFL algorithm. Then $prob[\tau > k] < \alpha e^{-\gamma k}$, for positive constants α, γ.*

Our argument does not yield a tight estimate of the exponent γ, which determines the precise convergence rate of the algorithm, but given that the underlying colouring problem is NP-hard this is unsurprising. Extensive simulations not presented here demonstrate that the convergence is rapid on average, similarly to the simulations presented in [13] for the single graph case.

We will show that in a determined finite amount of steps the system has some minimum positive probability of convergence. We show that starting from any configuration the system can reach some standard state after two steps. From this standard state we show that the system can then potentially reach a state where every node experiences a failure simultaneously, allowing convergence without issues of dependence between nodes. Hence the network always has positive probability of global success and so will almost surely converge.

In the sequel we refer to two nodes choosing the same channel as a "collision". We say that the state $\mathbb{S}$ consists of the set of all possible configurations where (i) the channel selections of at least two nodes interfere and (ii) at all colliding nodes the selection probability for every channel is bounded away from zero (in fact, we will consider the case where they are strictly greater than $\frac{b(1-b)}{c-1}$). We define the master graph: an edge is in the master graph if it is in any of the individual channel graphs $G(i), i \in [1, 2, .., c]$. Denote the maximum node degree of the master graph by md and the diameter of the master graph (length of the longest shortest path between two nodes) by D.

Consider a colliding node. Observe from [14] that a node colliding on one colour and then on a different colour ensures that its selection probabilities for all channels are strictly greater than $b(1-b)/(c-1)$. Similarly if a node succeeds and then collides. However, it can be seen that repeated collisions on the same channel can result in the channel selection probability becoming arbitrarily small. Thus, the system may avoid state $\mathbb{S}$ by some node undergoing repeated same channel collisions. We show in Lemma 1 that if the system has reached a configuration with some channel selection probabilities lower than $b(1-b)/(c-1)$ at one or more colliding nodes, then there is a positive lower-bounded probability that it will return in two steps to our standard state $\mathbb{S}$. The following Lemma is proved in [14].

Lemma 1. From any configuration of the system, if after two steps the system has not converged, it is in state $\mathbb{S}$ with some probability $pr_5 > 0$. ■

We proceed by defining the directed graph DG which is dependent on the current channel selection by the network nodes. There is an edge in DG from node u to node v if an edge exists between u and v in the graph $G(i_v)$ where i_v is the channel currently chosen by node v. We say v is a DG-neighbour of u. The edges directed *into* a node v are determined by the channel selection of that node together with the conflict graphs G, but are unaffected by the channel selections of other nodes. Existence of a directed path in graph DG from node u to node v indicates that the node u can potentially force a collision at node v (by first generating a collision with its immediate neighbour, which in turn can generate a collision with its neighbour, and so on until node v is reached).

DG-graphs associated with an example network are illustrated in Figure 5. Consider the lower left node. Edges involving this node only exist on channel R. Hence, this node can potentially create collisions with its neighbours by selecting channel R. However, by selecting channel B, the lower left node can always avoid interference from any of the other nodes regardless of their channel selection. This asymmetric nature of the relationship between the lower left node and its neighbours is indicated by the directional arrows on the DG-graph links.

Note also that once it chooses channel B, the lower left node in Figure 5 is unreachable from the other nodes. Since it is unreachable, no collisions can occur, choice of channel B will yield a "success" in the CFL algorithm and the node will remain on channel B thereafter i.e. the node will be converged and permanently unreachable. That is, the CFL algorithm therefore ensures that unreachable nodes remain permanently unreachable. A second example illustrating this behaviour is also given in the right-hand graphs in Figure 5. These examples illustrate the general point that as the CFL algorithm proceeds connectivity can change and, in particular, certain nodes may become permanently unreachable and we need to take account of this when analyzing convergence.

We define the set of nodes CN to be all nodes which are unreachable from any node which just collided. We note that any node $w \in CN$ must have just been successful. In addition, no matter what colour choices other nodes make in the future, w will never subsequently undergo a collision (since w is unreachable). Hence any nodes in CN are converged and can be ignored for the remainder of the proof. Note that the graph DG changes as the algorithm proceeds, and nodes can join CN but will never leave. In Figure 5 we see two stages of the algorithm, the corresponding DG, and the set CN illustrated by nodes in bold.

Lemma 2. Suppose that the system is in state $\mathbb{S}$. There exists a specific evolution $\mathbb{E}$ of the system which results in all nodes not in CN colliding.

Proof of Lemma 2. Consider one of the collisions. Two nodes k_1 and k_2, say, have just experienced a collision. By way of notational convenience we say these

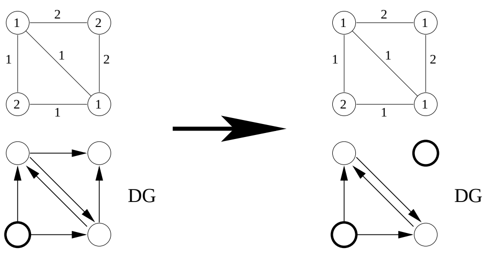


Fig. 4. Illustrating definition of DG-graphs. Upper graphs show node channel selections and the channel-dependence of edges is indicated by labels. Lower graphs show corresponding DG-graphs. The set CN indicated by nodes in bold.

two nodes were *visited* at step 2. Suppose now that k_1 collides with its first non visited DG-neighbour k_3 (if any) at step 3. Suppose also that k_2 collides with its first non visited DG-neighbour (if any, potentially k_3 also) at step 3 also. We say that such nodes are *visited* at step 3. Inductively suppose now that a node once visited collides with all its nonvisited DG-neighbours in consecutive steps. This is possible because a visited node having just collided can potentially choose any channel. Note that a node being visited simultaneously (along two different equal length paths from k_1 and k_2 say) is also possible.

Suppose that once a node has collided with all its nonvisited neighbours it then repeatedly chooses channel 1 until step $T_1 = T_0 + 3 + md \times D$. We note that as a node k_4 is colliding with its nonvisited DG-neighbours some of them may become visited from other nodes before they collide with k_4; we suppose then that k_4 does not visit such nodes.

Concurrently with this visiting procedure starting at the nodes k_1 and k_2, we can suppose that the same visiting procedure starts at all nodes in JC, and traverses the graph as before. Again as a node k_5 is colliding with its nonvisited DG-neighbours some of them may become visited from other nodes, and we again suppose that k_5 does not visit such nodes.

When all the visited nodes have visited all their neighbours, every node not in CN has been visited and is choosing channel 1. Some nodes which are now choosing channel 1 may of course have entered the set CN and are ignored. Hence every node not in CN is colliding. At the next time step we suppose that every node chooses a colour so that no collisions occur. ■

Lemma 3. Suppose that there exists a choice of channels that yields a non-interfering allocation. There is a strictly positive lower bound pr_8 on the probability of the evolution $\mathbb{E}$ occurring from any configuration in state $\mathbb{S}$.

Proof of Lemma 3. Given the initial colour selection probabilities and the set JC, the evolution $\mathbb{E}$ is well defined. The duration of $\mathbb{E}$ is at most $md \times D$ timesteps. Hence $\mathbb{E}$ has some positive (computable) probability pr_6 of occurring since the system is finite.

By assumption the system begins in state $\mathbb{S}$ and so the initial colour selection probabilites of just collided nodes are lower bounded; therefore there is some probability $pr_7 > 0$ such that $pr_6 > pr_7$ irrespective of the initial colour selection probabilities.

The set JC is one of finitely many possibilities and so again there is some probability $pr_8 > 0$ such that $pr_7 > pr_8$ irrespective of the initial choice of JC. ■

Proof of Theorem 1. Defining $pr_9 = pr_8 pr_5$ gives the probability that the system is in state $\mathbb{S}$ after the first two steps and then follows evolution $\mathbb{E}$. Hence every $2 + md \times D$ steps the system will converge with probability at least pr_9. Hence after $j(2 + md \times D)$ steps we have converged with probability at least $1 - (1 - pr_9)^j$ which converges to 1 as $j \to \infty$. ■

6 Multiple Radios

The use of wireless access points equipped with multiple radios has been the subject of much recent interest. The CFL algorithm can be applied without change to multi-radio access points by running a separate copy of the CFL algorithm for each radio. This will yield a non-interfering channel allocation for every radio. In this section we illustrate that the CFL algorithm can be further generalised to take explicit account of bit rate requirements in a multi-radio setting.

Specifically, we consider the following task. Suppose we have a set of interfering WLANs (possibly with channel-dependent interference) and a set $\mathcal{C}$ of available channels. Let b_i denote the bit rate associated with channel i. At access point j we require to select a non-interfering set of channels $C \subseteq \mathcal{C}$ such that $\sum_{i \in C} b_i \geq B$ and with cardinality $|C| \leq r$, where r is the number of radios at the access point. Note that the channel bit rate b_i, the target bit rate B, number of radios r and set of available channels $\mathcal{C}$ may be different for each access point.

The change here over our previous discussion is the inclusion of the bit rate constraint $\sum_{i \in C} b_i \geq B$. Such a bit rate requirement arises, for example, when striping data across multiple radios. One advantage over simply allocating a channel to every available radio is that it may be that fewer radios are sufficient to provide the required bandwidth, thereby reducing the load on the spectrum in dense WLAN deployments. This formulation also allows us to take explicit account of the different quality of each channel – this can be important in multi-radio settings where radios are heterogeneous e.g. some radios might be 802.11 based and others 802.16 based. We note that the bit-rate constrained channel allocation problem is also relevant to dynamic spectrum management in wired DSL lines (where cross-talk across wiring bundles is a significant source of interference) [15]. We introduce the following generalised version of the CFL algorithm to solve the multiple radio bit-rate constrained channel allocation problem.

Let c denote the number of available channels at an access point and the access point maintain a c element state vector p with element p_i corresponding to the probability of transmitting on the ith channel. Since we allow use of multiple radios, note that we do not require the p_i's to sum to one. Consider the following decentralized algorithm for updating p.

Generalised CFL Algorithm

1. Initialise $p = [r/c, r/c, \dots, r/c]$
2. Pick a random ordering of the channels. In that order, toss coins to activate channel i with probability p_i. Stop immediately once the AP's target bit rate is met. This results in a set C of active channels.
3. If $\sum_{i \in C} b_i < B$, multiply every probability by $1 + b$, set $C = \emptyset$, and repeat[2] step 2. We note that the random selection process at step 2 above together with the redistribution of probability in step 6 below, ensures that there is a positive lower bound on the probability of any feasible allocation after a collision.

[2] The precise procedure here is not important. The feasible active set may be found in any reasonable fashion provided any channel with nonzero p_i might be active and that channels with larger p_i are more likely.

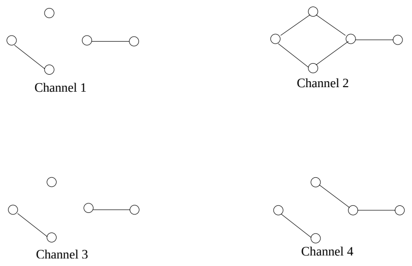


Fig. 5. Five AP example of multiple radios. Each graph gives the interference for a particular channel.

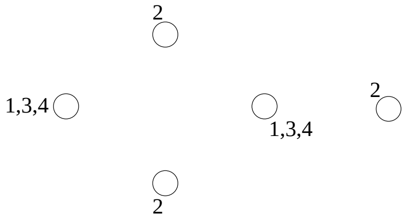


Fig. 6. Possible final result of multiple radio algorithm. The figure shows a successful channel allocation.

4. Sense the quality of the channels in set C. We obtain "success" on channel $i \in C$ if this does not interfere with any neighbouring WLAN and otherwise have a "failure".
5. If we have success on all active channels, update p as

$$p_i = 1 \ \forall i \in C, p_j = 0 \ \forall j \notin C \tag{1}$$

   i.e. on a successful choice we use the same set of channels for the next round. This ensures that any channel allocation that satisfies the target bit rate and also removes interference between all WLANs is an absorbing state.
6. Otherwise let S denote the set of channels which were successful, F the set of failed channels and I the set of inactive channels. Update p as

$$\begin{aligned} p_i &= 1 \ \forall i \in S, \\ p_j &= (1-b)p_j + b \ \forall j \in I, \\ p_k &= (1-b)p_k \ \forall k \in F. \end{aligned}$$

   The lower bound on these probabilities after the node fails is much more important than the exact choice of parameters.
7. Return to 2.

This algorithm maintains the three key properties of the original CFL algorithm, namely (i) that if every WLAN is successful the system remains in this successful configuration henceforth; (ii) after a collision any feasible channel allocation is possible; and (iii) if one WLAN is failing the failure can propagate to neighbouring WLANs and force them (with some probability) to change their channel allocation. Hence by a similar proof to that for Theorem 1 the generalized CFL algorithm will converge with probability 1 to a non-interfering channel allocation satisfying the specified bit rate requirements, provided one exists.

In Figure 5 we present an example of the multiple radio problem. Suppose that every AP has bit rate demand 3 units; suppose that channel 2 has bit rate 3 units and that all other channels have bit rate 1 unit. Figure 6 illustrates a feasible channel allocation which is a result of the algorithm.

7 Conclusions

We show that the interference between WLANs can be channel dependent in which case a different conflict graph is associated with each channel. This potentially has profound implications as the behaviour of proposed colouring-based algorithms for channel allocation is unclear in a multi-graph context. We are, however, able to show that a recently proposed decentralized colouring algorithm does generalise to the multi-graph setting. We also present a new, extended version of this algorithm suited to a wide range of multi-radio architectures. This work was supported by Science Foundation Ireland grant IN3/03/I346.

References

1. L. Tassiulas, A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks", *IEEE Trans Automatic Control*, 37 (12), 1992.
2. A. Akella, G. Judd, P. Steenkiste, and S. Seshan. "Self management in chaotic wireless deployments". In *MobiCom*, 2005
3. H. Luo, P. Medvedev, J. Cheng, S. Lu, "A self coordinating approach to distributed fair queuing in ad hoc wireless networks", *Proc. of IEEE INFOCOM '01*, 2001.
4. A. Raniwala, T. Chiueh. "Architecture and algorithms for an IEEE 802.11-based multi-channel wireless mesh network". In *Proc IEEE International Conference on Computer Communications*, 2005.
5. A.Mishra, V.Shrivastava, D.Agarwal, S.Banerjee, S.Ganguly, Distributed Channel Management in Uncoordinated Wireless Environments. In *MobiCom*, 2006.
6. A.Mishra, V.Brik, S.Banerjee, A.Srinivasan, W.Arbaugh, "A Client-Driven Approach for Channel Management". *Proc. IEEE INFOCOM*, 2006
7. B.J. Leung, K.K. Kim, "Frequency assignment for IEEE 802.11 wireless networks". *Proc. 58th IEEE Vehicular Technology Conference*, 2003.
8. B. Kauffmann, F. Baccelli, A. Chaintreau, K. Papagiannaki, C. Diot, "Self Organization of Interfering 802.11 Wireless Access Networks,", *INRIA Technical Report*, August 2005.
9. A. Subramanian, H. Gupta and S. R. Das, "Minimum interference channel assignment in multi-radio wireless mesh networks" *Proc. Mobicom*, 2006.
10. B.J.Ko, V.Mishra, J.Padhye, D.Rubenstein, "Distributed channel assignment in multi-radio 802.11 mesh networks". `http://www1.cs.columbia.edu/~danr/publish/2006/jun-tr06.pdf`
11. A. K. Das, S. Roy and R. Vijaykumar, "Static Channel Assignment in Multi-radio Multi-channel 802.11 Wireless Mesh Networks: Issues, Metrics and Algorithms". *Proc. IEEE Globecom*, 2006
12. K. Ramachandran, E. Belding, K. Almeroth, M. Buddhikot, "Interference-Aware Channel Assignment in Multi-Radio Wireless Mesh Networks". *Proc. IEEE INFOCOM*, 2006.
13. D.J. Leith, P. Clifford, "A Self-Managed Distributed Channel Selection Algorithm for WLANs". *Proc. ACM/IEEE RAWNET*, Boston, 2006.
14. D.J. Leith, P. Clifford, "Convergence of Distributed Learning Algorithms for Optimal Wireless Channel Allocation". *Proc. IEEE CDC*, San Diego, 2006.
15. T. Starr, J.M. Cioffi, P. Silverman, "Understanding Digital Subscriber Lines", Prentice Hall: Upper Saddle River, NJ, 1999.

Optimal Call Admission Control for an IEEE 802.16 Wireless Metropolitan Area Network

Sondes Khemiri[1], Khaled Boussetta[2], Nadjib Achir[2], and Guy Pujolle[1]

[1] Université de Paris 6, LIP6, Paris, France
{Sondes.Khemiri,Guy.Pujolle}@rp.lip6.fr
[2] Université de Paris 13, L2TI, Villetaneuse, France
{Khaled.Boussetta,Nadjib.Achir}@l2ti.univ-paris13.fr

Abstract. This paper focus on optimal Call Admission Control (CAC) policy for an IEEE 802.16 Wireless MAN. This policy has two objectives: (i) statistically guarantee the QoS of UGS, rtPS and nrtPS connections, (ii) maximize the average revenue of the wireless link. To find such optimal policy, we model our CAC agent as a Constrained Semi-Markov Decision Process (CSMDP). To the best of our knowledge, the only algorithm able to compute the optimal control policy of a CSMDP is based on the Linear Programming (L.P.) approach. Unfortunately, a realistic CAC problem with a large states space is intractable with the L.P. algorithm. Our work presents two contributions. First, the proposition of an optimal CAC for Triple-Play services support over a 802.16 WMAN. Second, the presentation of an alternative iterative algorithm that can be used to overcome the difficulties faced by the L.P. approach.

1 Introduction

The IEEE 802.16 standard [1] has been developed keeping in view the stringent QoS requirements of Triple-Play applications. Indeed, IEEE 802.16 defines four QoS scheduling services that should be treated appropriately by the MAC layer, namely; Unsolicited Grant Service (UGS), Real-Time Polling Service (rtPS), Non-Real-Time Polling Service (nrtPS) and Best Effort service (BE). UGS is designed to support real-time services that generate CBR or CBR-like flows, such as Voice over IP. When using UGS the Base Station (BS) provides fixed size data grants on a periodic basis, without any explicit Subscriber Stations (SS) requests. rtPS is designed to support real-time services that generate delay sensitive VBR flows, such as MPEG video or VoIP (with silence suppression). Each service flow is polled using periodic unicast request opportunities, in order to meet the flow's real-time needs and allow the SS to specify the size of the desired grants. nrtPS is designed to support delay-tolerant data delivery with variable size packets, such as high bandwidth FTP. The nrtPS connections use more periodic spaced request opportunities than rtPS. And finally, the BE service is proposed to be used for best effort traffic where no throughput or delay guarantees are provided.

The Base Station manages the radio resources. Consequently, it incorporates a Call Admission Control (CAC) agent, which guarantees that the QoS requirements are satisfied. Precisely, before any connection establishment, each SS inform the BS about their QoS requirements. The CAC agent decides whether a connection request could be accepted or should be rejected.

Although there are substantial literatures on Call Admission Control (CAC), little of them concentrated on the IEEE 802.16 Standard. In both [2] and [3] the authors propose a CAC strategy based on thresholds. Unfortunately, none of them attempted to compare the performance of their proposal to an optimal CAC policy. In this paper, we investigate the proposition of an optimal CAC policy for an IEEE 802.16 Wireless MAN. This policy has two objectives: (i) statistically guarantee the QoS of the various scheduling services defined in the IEEE 802.16 standard, (ii) maximize the average gain of the wireless link. To find such optimal policy, we model our CAC agent as a Constrained Semi-Markov Decision Processes (CSMDP).

This paper is organized as follows. In the next section we will describe the background and formulate the CAC problem. We also detail the resulting CSMDP model. Our resolution algorithm will be described in section 3. Results will be discussed in section 4. Finally, we conclude this paper in section 5.

2 Problem Statement

2.1 System Modeling

We are interested in an IEEE 802.16 2004 cell, which serves fixed residential Triple play subscribers. Those can have access to various types of applications like telephony, Video on Demand and file downloading. The standard 802.16 2004 defines two modes of QoS grant; either by SS (aggregated traffic) or by connection. In this work, we address the second case. Thus the base station has to provide QoS guarantees for 3 service classes. Namely, UGS, rtPS and nrtPS, which we denote as class 1, 2 and 3, respectively. In this study, we consider that the wireless link is the main bottleneck and that the bandwidth is a crucial resource that has to be shared.

The available bandwidth in an 802.16 cell depends on the physical layer characteristics. Especially, on the modulation technique that is used in an area. Generally, the modulation technique is chosen according to the distance separating the SS to the BS (More robust techniques are used for distant SSs). We assume that the cell's bandwidth is totally partitioned, so that each partition is adapted to a specific modulation scheme. The system S that we are modelling in this paper, focus on one link direction (e.g. downlink) and address one modulation technique's clients area. Note that this modelling assumption relies on the fact that we are considering residential subscribers.

We assume that the offered bandwidth capacity of the system S is a fixed number of bandwidth units, that we note C. During its duration, each UGS call needs a constant bit rate guarantee. That is, in S, a class 1 call requires the allocation of a given number of bandwidth units. For convenient presentation reason, we indifferently denote this number as, s_1, $\underline{s}_1$or $\overline{s}_1$.

rtPS calls generate variable bit rate traffic and require strong guarantees on QoS delay parameters. We consider that these constraints could be satisfied if s_2, the bandwidth units number assigned to a class 2 call in S, is comprised between two limit numbers, which we note as $\underline{s}_2$ and $\overline{s}_2$ for lower and upper limits, respectively. Similarly, nrtPS calls, generate variable bit rate traffic but are less constringent on QoS delay parameters than rtPS ones. Therefore, a class 3 call needs the allocation of bandwidth units number, which we note s_3. We also suppose that nrtPS QoS guarantees could be meet if $s_3 \in [\underline{s}_3, \overline{s}_3]$.

In order to obtain a tractable analytical modelling of the CAC problem (with a limited states space dimension), we suppose that all calls of a given class i are assigned the same $s_i \in [\underline{s}_i, \overline{s}_i]$ bandwidth units. This assumption may not leads to an optimal radio resource efficiency, but is motivated by the fact that the resulting states space description of our system will be given by only the number of ongoing traffic i calls.

Since an IEEE 802.16 2004 cell's coverage can reach a diameter of several kilometers, we could reasonably assume an infinite population traffic model. Precisely, we suppose that calls of class $i \in \{1, 2, 3\}$ arrive according to a Poisson process with rate λ_i. A call of class $i \in \{1, 2, 3\}$ may be rejected if its acceptance violates the QoS guarantee of ongoing calls or if the system does not have enough resources. Otherwise, it is accepted and will generate a revenue at a rate R_i during the call holding time, which is exponentially distributed with mean μ_i^{-1}. For instance, the revenue rate could correspond to the amount of money per second earned by the system for carrying this call. Note that, although s_2 and s_3 may vary according to system states (see section 2.2), we assume that the holding times of class $i \in \{1, 2, 3\}$ calls are independent of the allocated bandwidth.

Following this system modelling, we consider that the packet-level's QoS parameters (loss rates, delays, jitters etc.) of a class i call are statistically guaranteed in the wireless link part of the WiMAX network as long as the allocated bandwidth s_i remains in the $[\underline{s}_i, \overline{s}_i]$. We can then focus on the connection-level's QoS parameters, namely the call blocking probability. Formally, we would like to find the optimal CAC policy solution, π^*, to the following optimization problem:

1. maximize the average revenue $R(\pi^*)$ of S
2. $\forall i \in \{1, 2, 3\}$, guarantee that the call blocking probability $P_i(\pi^*)$ remains under a threshold β_i,

2.2 Priority-Based Bandwidth Sharing Scheme

For an UGS traffic the required amount of bandwidth units is fixed. So, only acceptance or rejection of a class 1 call has to be decided by the CAC policy. However, for $i \in \{2, 3\}$, s_i only needs to remain in $[\underline{s}_i, \overline{s}_i]$. We propose to exploit the flexibility of rtPS and nrtPS bandwidth requirements, so that the possible number of simultaneous calls carried by the system (and consequently the revenue) is maximized. For that purpose, we consider a classical priority-based bandwidth sharing strategy. Our choice is not only motivated by its implementation simplicity, but also because it led us to a tractable analytical modelling of the CAC problem.

Priority levels of our bandwidth allocation strategy are decremental from class 1 down to class 3. Precisely, let Ω be the states space descriptor of the system and $S_i(x)$ is the aggregated bandwidth allocated to class i calls when the system is in state x. Our bandwidth sharing strategy relies on these rules:

– When a class 1 call request arrives and the remaining available bandwidth units are less than s_1, then the BS resource manager check if it's possible to iteratively decrease $S_3(x)$ until that s_1 free bandwidth units are made available. Degradation attempt of the aggregated nrtPS bandwidth is stopped if $s_3 = \underline{s}_3$. If this threshold is reached, while the freed bandwidth is not enough to handle the arriving UGS call, then degradation of aggregated rtPS traffics is checked in the same meaner as for nrtPS. This process is stopped as soon as s_1 bandwidth units are made available, or if $\forall i \in \{2,3\}$, $s_i = \underline{s}_i$. Note that the degradation is operationally performed only if (1) it leads to liberate s_1 bandwidth units and (2) the CAC policy decides to accept the new class 1 call.
– When an rtPS call request arrives and the remaining available bandwidth units are less than s_2, then the degradation attempt of aggregated nrtPS bandwidth, eventually of rtPS, is performed following the rules described in previous paragraph.
– When an nrtPS call request arrives while the remaining available bandwidth units are smaller than s_3, then only degradation attempt of aggregated nrtPS bandwidth is authorized.
– When a call leaves the system and $\exists i \in \{2,3\}$ such that $s_i < \overline{s}_i$, then the freed bandwidth is shared following an upgrading priority-based strategy. First, the available bandwidth is fairly shared between rtPS calls. Then, once $s_2 = \overline{s}_2$ the remaining bandwidth is equitably allocated to nrtPS calls with the aim to make them reach the upper limit $\overline{s}_3$.

Here, it's important to notice that as $\forall i \in \{2,3\}$ $\underline{s}_i > 0$ then our degradation strategy never leads to the preemption of any ongoing call. We also recall the reader, that all calls of a given class i are assigned the same bandwidth capacity s_i. Therefore, the bandwidth upgrading (resp. degrading) of a given class is performed if all calls of that class are identically augmented (resp. reduced) by the same amount of bandwidth units. Based on our system description, the optimal CAC policy could be computed using the CSMDP framework.

2.3 The Constrained Semi-markov Decision Process

Markov Decision Processes (MDP) are often used to obtain optimal control policies. A complete survey of successful use of MDP in telecommunications problems can be found in [4]. An CSMDP can be described by a tuple $\{\Omega, A, R, P\}$, where each element is detailed in the following paragraphs:

The states space descriptor Ω. The states space descriptor Ω is defined as the set of allowed states of S. In our case study, each state of S can be described by a state vector $x = (x_1, x_2, x_3)$ where, $\forall i \in \{1,2,3\}$, x_i denotes the number of ongoing traffic i calls. Taking into account the priority-based bandwidth sharing scheme, we could easily see that:

$$\Omega = \left\{ x \,\middle|\, \sum_{i=1}^{3} x_i \underline{s}_i \leq C \right\} \tag{1}$$

The actions space of the control process A. For each request, the CAC agent's action could be an acceptance or a rejection. The agent that we model in this section has the same behaviour as the one described in [15]: our CAC agent "computes" its decision to accept or to block a new call in advance, i.e. before the arriving of such request. Thus, when S is in state $x \in \Omega$, in prevision of the arrival of a new call request, the CAC agent choose an action vector $a^{\pi}(x) = (a_1^{\pi}(x), a_2^{\pi}(x), a_3^{\pi}(x)) \in A$ following a CAC policy π. The value of each element $a_i^{\pi}(x)$, $\forall i \in \{1,2,3\}$, indicates to the system the appropriate action to be executed if the next arriving event is call request of class i. More precisely, each element takes the value 1 whenever the CAC agent recommends accepting the call, and 0 otherwise. Therefore, the action space can be expressed as follow

$$A = \{a = (a_1, a_2, a_3) \mid a_i \in \{0,1\} \, \forall i \in \{1,2,3\}\} \tag{2}$$

Note that in some states of Ω, the remaining free resources in S are not sufficient to accept a new call. Therefore, for some call requests, the only allowed action in these states is the rejection. Thus, each vector action $a^{\pi}(x)$ belongs to the subset $A(x) \subseteq A$ of the allowed actions when S is in state x. Obviously,

$$A(x) = \{a \in A \mid a_i = 0 \,\text{if}\, x + e_i \notin \Omega, \, \forall i \in \{1,2,3\}\} \tag{3}$$

Here, the event e_i refers to the arrival in S of a class i call request, and $x + e_i$ denotes the new transition state if the CAC agent decides to accept the call.

The set of rewards R perceived by the system. By definition, $R = \{R(x,a) \mid \forall x \in \Omega, \, \forall a \in A(x)\}$, where $R(x,a)$ is the reward perceived by the system if the CAC agent chooses the vector action $a \in A(x)$ while S is in the state $x \in \Omega$. Let $\tau(x,a)$ be the expected sojourn time of S in the state x when the agent chooses the action vector $a \in A(x)$, and note $R(x)$ as the reward rate perceived by the system when S is in the state $x \in \Omega$. Then, $R(x,a) = \tau(x,a) R(x)$. According to our model description, we can easily get that:

$$\tau(x,a) = \left[\sum_{i=1}^{3} \lambda_i a_i + x_i \mu_i \right]^{-1} \quad \text{and} \quad R(x) = \sum_{i=1}^{3} R_i x_i \tag{4}$$

The set of transition Probabilities $P = \left\{ P_{xay} \,|\, \forall (x,y) \in \Omega^2, \forall a \in A(x) \right\}$. where P_{xay} denotes the transition probability from the state x to the state y if the vector action a is chosen by the agent while X_t was in state x. Obviously:

$$P_{xay} = \begin{cases} \lambda_i a_i \tau(x,a) \,\text{if}\, y = x + e_i \; \forall i \in \{1,2,3\} \\ x_i \mu_i \tau(x,a) \,\text{if}\, y = x - e_i \; \forall i \in \{1,2,3\} \\ 0 \qquad\qquad\quad \text{otherwise} \end{cases} \tag{5}$$

2.4 Limitations of the Classical CSMDP Resolution Method

The general form of an optimal control policy of a CSMDP belongs to a particular subset, called the set of *randomized* stationary policies [5]. A policy π is said randomized stationary policy if the action a is chosen when the system S is in state x according to a probability $p^{\pi}(x, a)$. To the best of our knowledge, the only method that can determine the optimal randomized control policy of a CSMDP is based on the Linear Programming (L.P.) approach [5]. In our case study, the execution of this algorithm will requires the storage of a data matrix who's size is proportional to $|\Omega|*8$ (8 is the cardinality of A). Unfortunately, the size of $|\Omega|$ will increase exponentially with the number of resources units C. For example, if $C = 100$ and $\forall i \in \{1, 2, 3\}$ $\underline{s}_i = 1$ then $|\Omega|$=176851. Clearly, finding the probabilities $p^{\pi^*}(x, a)$ $\forall x \in \Omega$ and $\forall a \in A(x)$ will necessitate important storage capacity and processing time by the computing machine. Several papers have pointed out the computational limitations relating to Markov Decision Process framework [6,7]. Nevertheless, in the best of our knowledge, all of the proposed solutions to overcome this problem have been made targeting the basic (unconstrained) MDP context. Thus, they are unfortunately not directly applicable to our Constrained Semi-Markov Decision Process.

3 Our Proposed Computational Method

3.1 Basic Idea

The solution that we propose in this paper uses some well known results issued from the optimization under constraints field: a very popular approach to resolve a constrained optimization problem is to use a relaxation technique. The basic idea is to relax the constraints of the problem by integrating them into the objective function. The objective function will be penalized each time a solution did not satisfy a constraint. Consequently, using a relaxation technique, a primal constrained problem is turned into two dual sub-problems: one master subproblem focusing on how to relax the constraints of the primal problem, and a slave submaster, which resolves the resulting unconstrained optimization problem. In the literature, there are many relaxations methods (see [8,9] for a complete review). They differ on the way how the objective function is penalized each time a solution did not satisfy a constraint. A well adapted method for our CAC problem is the Lagrange multiplier method.

Applying the Lagrange multiplier method to our case study, we introduce the Lagrange function:

$$L\left(\pi^*(\omega), \omega\right) = R\left(\pi^*(\omega)\right) - \sum_{i=1}^{3} \omega_i \left(P_i\left(\pi^*(\omega)\right) - \beta_i\right) \tag{6}$$

where, $\omega = (\omega_1, \omega_2, \omega_3)$ is the Lagrange Multiplier vector and $\pi^*(\omega)$ is the optimal CAC policy obtained by resolving the slave sub-problem. Since the salve subproblem resolution is executed after the master subproblem, then $\pi^*(\omega)$ is dependent on ω. Our primal problem is now transformed into a dual problem. The latter one is decomposed into two subproblems, which have to be successively resolved:

1. A master subproblem, which seeks to obtain the optimal vector
$$\omega^* = \arg \min_{\omega \in I\!R^3_+} L\left(\pi^*\left(\omega\right), \omega\right) \tag{7}$$
2. A slave subproblem, which, depending on a given ω, computes the optimal CAC policy
$$\pi^*\left(\omega\right) = \arg\max_{\pi} L\left(\pi\left(\omega\right), \omega\right) \tag{8}$$

3.2 Resolving the Slave Subproblem

The idea of using the Lagrange multiplier method in the context of CMDP was first discussed in [10]. Then, the investigations were extended in [11] to the case of CSMDP. Latter, [12] generalized these studies to CMDP with counted states space. In all these papers, the authors consider only one constraint. In addition, none of the mentioned works have provided details on how to compute the optimal Lagrange multiplier value.

Multi-Constrained SMDP have been addressed by Altman (see [13] for a complete review). Compared to these works, our main contribution relies on the resolution of the master and slave subproblems. The main idea of our method is based on the results obtained by Altman in [14]. Applying this work to our case study, we could show that resolving the slave subproblems leads to obtain the optimal CAC policy of a particular unconstrained SMDP(ω). The tuple characterizing this SMDP is dependent on the value of the Lagrange multipliers vector ω. Precisely, if we note $R\left(\omega, x, a\right) = \sum_{i=1}^{3} R_i(x) + \omega_i a_i$, then the SMDP($\omega$) is described by the tuple $\{\Omega,\ A,\ R\left(\omega\right),\ P\}$, where
$$R\left(\omega\right) = \left\{R\left(\omega, x, a\right) \mid \forall x \in \Omega,\ \forall a \in A\left(x\right)\right\} \tag{9}$$

In his papers, Altman suggest the use of an L.P. algorithm to resolve the slave subproblem and a dual L.P. algorithm for the master subproblem. Due to the computational states space difficulties discussed previously, our CAC problem is intractable using the Altman's approach.

Since solving the slave subproblem leads to finding the optimal control policy of the SMDP(ω), we suggest the use of the Value Iteration Algorithm with dynamic relaxation factor [5]. This method is more robust and less resources demanding. In particular, as in [15] we could easily show that only 3 tests are needs in each iteration to determine the optimal action vector of a state. That is, the algorithm complexity is on O($|\Omega|$3) (instead of O($|\Omega|$8) for the L.P. approach). In addition, methods proposed to address (unconstrained) MDP states space reduction, could now be considered when resolving the SMDP(ω).

Note that the CAC policy obtained by our method is not necessarily the optimal solution of our CAC optimization problem. In fact, the optimal control policy of an unconstrained SMDP is not a randomized stationary policy. It rather belongs to a more restrictive set: the subset of *deterministic stationary* policies. A policy π is a deterministic stationary policy if $\forall x \in \Omega$ and $\forall a \in A$, $p^{\pi}(x, a) = 1$ or 0. In reality, our CAC policy is quasi-optimal. Indeed, in [15] the authors show that the optimal control policy of a CSMDP with m constraints will have at most m actions chosen according to a probability value, while all the remaining actions will be chosen in a deterministic way (I.e. with probability equal to 1 or 0).

Applying the Value Iteration method to the SMDP(ω) will provides the optimal action vectors $a^{\pi^*}(x) \in A$ for all states $x \in \Omega$. We could then compute the call blocking probability $\forall i \in \{1, 2, 3\}$ as,

$$P_i(\pi) = \sum_{x \in \Omega^{\pi}} (1 - a^{\pi}(x)) P^{\pi}(x) \tag{10}$$

where Ω^{π} is the states space accessible by the system, and $P^{\pi}(x)$ is the resulting steady state x probability. These steady state probabilities could be obtained using a numerical method (e.g. Gauss Seidel) [16]. Finally, let $\rho_i = \frac{\lambda_i}{\mu_i}$, $\forall i \in \{1, 2, 3\}$. We could easily derive the average revenue rate obtained by the system, when the CAC agent apply the policy π, as

$$R(\pi) = \sum_{i=1}^{3} R_i \rho_i (1 - P_i(\pi)) \tag{11}$$

3.3 Resolving the Master Subproblem

Following the Lagrange Multiplier method, the master subproblem resolution aims to determine the solution to equation 7. For that purpose, we use the Gradient Incremental Method, which is based on iterative executions. During its n*th* iteration a vector ω^n is computed following the direction given by the gradient $\nabla\omega L\left(\pi^*\left(\omega^{n-1}\right), \omega\right)$. Here, $\pi^*\left(\omega^{n-1}\right)$ is the optimal CAC policy of the SMDP$\left(\omega^{n-1}\right)$ and $s^n \in I\!R^+$ is a chosen step. Formally,

$$\omega_i^n = \left[\omega_i^{n-1} - s^n \frac{\partial L\left(\pi^*\left(\omega^{n-1}\right), \omega\right)}{\partial \omega_i}\right]^+, \forall i \in \{1, 2, 3\} \tag{12}$$

Clearly, $\forall i \in \{1, 2, 3\}$,

$$\frac{\partial L\left(\pi^*\left(\omega^{n-1}\right), \omega\right)}{\partial \omega_i} = \beta_i - P_i\left(\pi^*\left(\omega^{n-1}\right)\right) \tag{13}$$

Note that the step s^n have to be judiciously adjusted during the n*th* iteration, so that convergence to the optimal solution of equation 7 is (1) guaranteed and (2) speeder. In our case study, we have used the Bertsekas's method [8] .

4 Results Analysis

In this section we present some results. We have run several test scenarios. In all our evaluations, we have considered that; $\underline{s}_1 = \overline{s}_1 = \underline{s}_3 = 1$, $\underline{s}_2 = 2$, $\overline{s}_2 = \overline{s}_3 = 4$, $R_1 = 5$, $R_2 = 3$, $R_3 = 1$, $\beta_1 = 0.001$, $\beta_2 = 0.01$ and $\beta_3 = 0.01$.

In our five test scenarios, we have considered that $\forall i \in \{1, 2, 3\}$, $\rho_i = \rho$, where $\rho \in \{1, \ldots, 5\}$. For each ρ value we have computed the optimal CAC policy π^*. Note that, we have not fixed C. In fact, for each test scenario, C is chosen as the smallest integer value for which an optimal CAC policy exists (i.e. the call blocking probabilities are satisfied). In other words, we provide here the answer to the question on how to dimension the WiMAX cell capacity (C), so that the QoS of the carried services are statistically satisfied. Results of our experimentations are summarized in table bellow:

ρ	C	$e(\pi^*)$	$R(\pi^*)$	$P_1(\pi^*)$	$P_2(\pi^*)$	$P_3(\pi^*)$	$E[s_2]$	$E[s_3]$
1	13	0.6094	8.970	0.000666	0.007572	0.003884	3.887	3.456
2	19	0.7773	17.917	0.000400	0.009904	0.009500	3.839	3.113
3	25	0.8488	26.908	0.000616	0.007985	0.003634	3.798	2.848
4	31	0.8844	35.904	0.000411	0.006413	0.002677	3.767	2.629
5	36	0.9079	44.865	0.000359	0.006382	0.006112	3.701	2.357

The table shows that the link efficiency, denoted $e(\pi^*)$, and the average revenue rate, $R(\pi^*)$, of the optimal CAC policy π^*, increase with the load ρ. We also clearly see that, $\forall i \in \{1, 2, 3\}$, $P_i(\pi^*) \leq \beta_i$. The table shows the average bandwidth allocated to each rtPS or nrtPS call[1], denoted $E[s_2]$ and $E[s_3]$ respectively. Obviously, $\forall i \in \{2, 3\}$, $\underline{s}_i \leq E[s_i] \leq \overline{s}_i$. One could also observe that as ρ increases, the degradation of $E[s_3]$ is more important than $E[s_2]$. This is a direct consequence of our Priority-based bandwidth sharing scheme. Note that, assuming our bandwidth sharing scheme, we have also evaluated the Complete Sharing (CS) as a basic CAC policy. This simple policy is known for not being able to satisfy calls blocking probability constraint. Using the same test scenarios' parameters described in this section, and assuming the values of C given in the result table we were able to see that, under a CS policy, UGS calls blocking probability is always higher than β_1.

5 Conclusion

This work presents two contributions. First, we formalize the optimal CAC policy problem for a tripe play WiMAX services access. Second, in order to determine the nearly optimal CAC policy we propose an alternative algorithm from the classical Linear Programming approach. The basic idea is to use the Lagrange multiplier method in order to overcome the numerical problems faced by the L.P. algorithm. The main advantage of our algorithm is its iterative nature.

[1] Recall that for UGS; $s_1 = \underline{s}_1 = \overline{s}_1$.

Although, our method is more robust than the L.P. approach to states space dimension problem, numerical problems could not totally being avoided for large state spaces. Precisely, when considering realistic WiMAX deployment scenario. Our ongoing works focus on designing simple WiMAX CAC policies. We do not necessarily seek for optimal CAC. We rather look for simple CAC policies, whose mathematical modelling will lead to a tractable analytical resolution, more suitable for realistic WiMAX dimensioning studies. The CAC policy presented in this paper will serve as the optimal reference for our new policies.

References

1. IEEE Std 802.16-2004 (Revision of lEEE Std 802.16-2001). IEEE Standard for Local and Metropolitan Area Networks Part 16: Air Interface for Fixed Broadband Wireless Access Systems.
2. H. Wang, W. Li and Dharma P. Agrawal. Dynamic admission control and QoS for 802.16 wireless MAN. In *Wireless Telecommunications Symposium*, April 2005.
3. H. Hou, W. P. H. Ho and X. Shen. Performance Analysis of a Reservation-based Connection Admission Scheme in 802.16 Networks. In *IEEE Global Telecommunications Conference*, November 2006.
4. Altman, E.: Applications of Markov Decision Processes in Communication Networks. Handbook of Markov Decision Processes (2002) 489–536
5. Tijms, H.: Stochastic models - an algorithmic approach. J. Wiley & Sons (1994)
6. Ross, K.W., Tsang, D.H.K.: Optimal circuit access policies in an ISDN environment: A Markov decision approach. IEEE Transactions on Communications **37**(9) (1989)
7. Kwon, T., Choi, Y., Naghshineh, M.: Optimal Distributed Call Admission Control for Multimedia Services in Mobile Cellular Networks. In: Mobile Multimedia Conference (MoMuC '98). (1998)
8. Bertsekas, D.P., Nedic, A., Ozdaglar, A.E.: Convex Analysis and Optimization. Athena Scientific (2003)
9. Ben-Tal, A., Zibulevsky, M.: Penalty/Barrier Multiplier Methods for Convex Programming Problems. SIAM Journal of Optimization **7**(2) (1997) 347–366
10. Beutler, F.J., Ross, K.W.: Optimal policies for controlled Markov chains with a constraint. Journal of Mathematical Analysis and Applications **112** (1985) 236–252
11. Beutler, F.J., Ross, K.W.: Time-average optimal constrained semi-Markov decision processes. Advances of Applied Probability **18** (1986) 341–359
12. Sennott, L.I.: Constrained discounted Markovdecision chains. Probability in the Engineering and Informational Sciences **5** (1991) 463–475
13. Altman, E.: Constrained Markov Decision Processes. Chapman & Hall/CRC (1999)
14. Altman, E., Spieksma, F.: The Linear Program approach in Markov Decision Problems revisited. ZOR - Methods and Models in Operations Research **42**(2) (1995) 169–188
15. Ross, K.W.: Randomized and past dependent policies for Markov decision processes with multiple constraints. Operations Research **37**(3) (1989) 474477
16. Stewart, W.: Numerical Methods for Computing Stationary Distributions of Finite Irreducible Markov Chains. In: Chapter 3 of Advances in Computational Probability, Kluwer Academic Publishers (1999)

A Survey of Throughput Versus Complexity Tradeoffs in Wireless Networks

Saswati Sarkar*

Department of Electrical and Systems Engineering, University of Pennsylvania
Sophia, Antipolis
swathi@seas.upenn.edu

Abstract. Attaining stability guarantees using distributed scheduling policies is an important design goal in multi-hop wireless networks. Several recent results have recently characterized tradeoffs tradeoffs between computation times and stability guarantees in multi-hop wireless networks. We summarize existing results that have substantially advanced the state of the art in this context, and discuss problems that remain open.

1 Introduction

Attaining the maximum stability region, or a guaranteed fraction thereof, through dynamic link scheduling, is a key design goal in multihop wireless networks. The scheduling problem involves determination of which links transmit packets at any given time. Appropriate scheduling of links is key towards attaining stability guarantees as the success of transmission in any given link depends on which other links transmit packets simultaneously. The transmission schedules can not be pre-computed, and needs to be determined at every transmission epoch, as the congestion levels in the nodes and the transmission conditions in the wireless medium vary with time, and the statistics of these temporal variations are oftentimes not known a-priori. Thus, the time required to determine which links would transmit at any transmission epoch is a key performance metric for any dynamic scheduling policy.

Owing to the lack of a central controller, at every transmission epoch each link needs to determine whether it would transmit based on its own state and the information it acquires about the states of other nodes. The stability guarantees usually improve with increase in the information each link (or rather a node which is the source of the link) acquires about the states of other links. The time required for each link to decide whether to transmit at any given time depends on the time required (a) to exchange messages with other links to learn their states and (b) to perform the computations required to arrive at an appropriate decision based on the information acquired. We refer to the total time required in both parts as the schedule computation time, or rather the computation time. The time required in each part increases with increase in the amount of information

* Currently visiting INRIA, Sophia, Antipolis.

a link acquires about the states of other links. Thus, an important question is how much information a link should acquire about the states of other links.

The scheduling policies that have been widely investigated can be classified in two broad classes based on the above qualifier: the policies that require each link to know some attribute that depends on the states of (a) all links in the network [3,9,10] and (b) only the links that interfere with it [1,5,6,8,11]. We refer to the two classes as INFORMATION(N) and INFORMATION(1) policies respectively, where N is the number of links in the network. By this nomenclature, INFORMATION(k) is the class of policies that require each link to learn the states of their k-hop interferers.

The contribution of this paper is to survey the recent results that characterize the tradeoffs between stability guarantees, computation times and the amount of state information required for scheduling policies for different classes of wireless networks. We subsequently summarize several open problems that are theoretically intriguing and also important from a practical perspective. The article is organized as follows. We describe the existing work in Section 2, and describe the open problems in Section 3.

2 Existing Work

A seminal result in this context has been obtained by Tassiulas *et al.* who have characterized the maximum stability region and provided a policy that attains this stability region in an arbitrary wireless network [10]. This policy schedules the maximum weighted independent set of links in each slot, and hence requires $\Omega(e^N)$ computation time unless $P = NP$. Later, Tassiulas [9] provided randomized scheduling schemes that attain the maximum achievable stability region, which can be implemented in fully distributed manner using gossip based algorithms [4]. In each slot, this policy randomly selects an independent set of links, compares its weight with the weight of the set of links scheduled in the previous slot and schedules the set that has the larger weight. This policy requires $\Theta(N)$ computation time. Recently, Dimakis *et al.* [3] have shown that a greedy maximal weight scheduling, which requires $\Theta(N)$ computation time, attains the maximum stability region in several different networks. All these policies are in the INFORMATION(N) class.

Recently, provable stability guarantees have been obtained with some policies in INFORMATION(1) class. Dai *et al.* [2], Lin *et al* [6] and Wu *et al.* [11] proved that a simple greedy scheduling scheme, maximal matching, attains half the maximum stability region for the primary interference model; the computation time for maximal matching is $\Theta(\Delta_G \log N)$, where Δ_G is the maximum degree and N is the number of nodes in the network. Chaporkar *et al.* [1] proved that maximal matching can be generalized to attain guaranteed fraction of the maximum stability region for arbitrary interference models, while retaining the logarithmic computation time. Sarkar *et al.* [8] proved that for primary interference model and tree graphs, a queue length dependent maximal matching attains 2/3 of the stability region while using $\Theta\left(\Delta_G (\log(N))^2\right)$ computation time. Lin *et al.* [5]

proved that a random access scheme, where links access the medium with a probability that depends on their and their interferers' queue lengths, attains 1/3 and $1/\Delta_G$ the stability region for arbitrary networks under primary and secondary interference models respectively, while requiring a $O(\Delta_G)$ computation time.

Sarkar *et al.* [7] introduced the class of INFORMATION(k) policies and proved that for appropriate choices of k, policies can be designed in the INFORMATION(k) class so as to obtain arbitrary tradeoffs between the best stability guarantees and the computation times. They first consider the primary interference model which mandates that any set of links can be simultaneously scheduled provided they do not have any common node. Under this interference model, when the network topology is a tree, given any positive constant ϵ, they obtained a distributed scheduling policy in INFORMATION(1) class that (a) approximates the stability region within a factor of $1 - \epsilon$ and (b) requires a computation time of $O(\Delta_G/\epsilon)$. The policy is simple to implement as the computation of each schedule needs each link to communicate only 1 bit of control message to each of the links that interferes with it. Next, they present a general framework for designing INFORMATION(k) policies for approximating the stability region arbitrarily closely for arbitrary networks and interference models. They subsequently use this framework for obtaining arbitrary tradeoffs between stability guarantees and computation times for large classes of networks, e.g., graphs with limited cyclicity and primary interference models, geometric and quasi-geometric graphs under both primary and secondary interference models. For example, when nodes are embedded in a plane and there exists an edge between two nodes if and only if the distance between them is less than a given constant D (i.e., for geometric graphs), given any positive constant ϵ, they obtain a distributed scheduling policy in INFORMATION$\big(O(\Delta_G/\epsilon^2)\big)$ class that (a) approximates the stability region within a factor of $1 - \epsilon$ and (b) requires a computation time of $O(\Delta_G^2/\epsilon^2)$. The stability and computation time guarantees extend to networks where sessions traverse multiple links.

We now briefly describe the design of the above policies, and provide the intuition behind the performance guarantees. The proposed policies partition the network in a collection of components - the size of the components depend only on Δ_G and ϵ. The links that originate in a component but interfere with those in another component are "shut down" i.e., not scheduled. Thus, the links scheduled in each component will not interfere with those scheduled in other components irrespective of the scheduling policy in each component. Hence, the scheduling in different components can now be determined in parallel. Thus, the time required to compute the overall schedule now depends only on the size of each component and can be determined only by Δ_G and ϵ for appropriate partitioning schemes. The links are scheduled in each component so as to maximize the stability region of the component. Thus, the reduction in the overall stability region may only happen because links are occasionally shut down. But, the resulting reduction in stability is small because each link is shut down only a small fraction of time since different partitioning schemes are used at different times and the size of the components in each partition is large enough. The proofs for the stability guarantees rely on a combination of graph-partitioning techniques and lyapunov arguments.

3 Open Problems

The existing results focus on obtain arbitrary guarantees between computation times and stability guarantees. An intriguing question is to obtain guarantees for end-to-end delays. A basic question, which remains open, is the characterization of a policy that attains any feasible delay vector in an arbitrary constrained queueing network. The characterization is likely to be significantly more challenging than characterization of the maximum stability guarantees simply because the class of policies that minimize end-to-end delays is more limited than the class of policies that maximize the stability region. Furthermore, the delay-optimal policies are likely to have exponential computation times. Hence, the next natural question, will be to obtain arbitrary tradeoffs between computation times and delay guarantees. It is worthwhile to investigate whether the techniques that lead to tradeoffs between computation times and throughput guarantees lead to similar tradeoffs for delay guarantees as well.

Next, the existing results assume that the transmitters do not control power levels. When this assumption is relaxed, the nodes need to jointly optimize power levels and transmission policies such that the stability region is maximized subject to limitations on available power. The problem of obtaining arbitrary tradeoffs between computation times and stability guarantees remains open in this context.

References

1. P. Chaporkar, K. Kar, and S. Sarkar. Throughput guarantees through maximal scheduling in multihop wireless networks. In *Proceedings of 43d Annual Allerton Conference on Communication, Control and Computing*, Allerton, Monticello, Illinois, September 28-30 2005.
2. J. Dai and B. Prabhakar. The throughput of data switches with and without speedup. In *Proceedings of INFOCOM*, pages 556–564, Tel Aviv, Israel, Mar 2000.
3. A. Dimakis and J. Walrand. Sufficient conditions for stability of longest queue first scheduling: second order properties using fluid limits. *Advances of applied Probability*, 38(2):505–521, June 2006.
4. D. Shah E. Modiano and G. Zussman. Maximizing throughput in wireless networks via gossiping. In *Proc. ACM SIGMETRICS / IFIP Performance'06*, June 2006.
5. X. Lin and S. Rasool. Constant-time distributed scheduling policies for ad hoc wireless networks. In *Proceedings of IEEE CDC-ECC'05*, San Diego, CA, Dec 2006.
6. X. Lin and N. Shroff. The impact of imperfect scheduling on cross-layer rate control in multihop wireless networks. In *Proceedings of INFOCOM*, Miami, FL, Mar 2005.
7. S. Ray and S. Sarkar. Arbitrary throughput versus complexity tradeoffs in wireless networks using graph partitioning. In *Proceedings of the Second Workshop on Information Theory and Applications*, San Diego, CA, January 29-February 2 2007.

8. S. Sarkar and K. Kar. Achieving 2/3 throughput approximation with sequential maximal scheduling under primary interference constraints. In *Proceedings of 44th Annual Allerton Conference on Communication, Control and Computing*, Allerton, Monticello, Illinois, September 27-29 2006.
9. L. Tassiulas. Linear complexity algorithms for maximum throughput in radio networks and input queued switches. In *Proceedings of INFOCOM*, pages 533–539, 1998.
10. L. Tassiulas and A. Ephremidis. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37(12):1936–1948, Dec 1992.
11. X. Wu and R. Srikant. Regulated maximal matching: a distributed scheduling algorithm for multihop wireless networks with node-exclusive spectrum sharing. In *Proceedings of IEEE CDC-ECC'05*, Seville, Spain, Dec 2005.

Finite Horizon Control Problems Under Partial Information

Jens Winter

Universität Ulm
Institute of Optimization and Operations Research
Helmholtzstr. 18, 89069 Ulm, Germany
jens.winter@uni-ulm.de

Abstract. A control model over a finite horizon is considered, where the state process is not observable and has to be estimated with an observation process, where each state of the observation process represents a group of states of the unobservable process. We show how the model with partial information can be transformed in one with complete information with the help of the filter technique and conditional probabilities. As a main result we prove the connection between the original and the reduced model and we show an explicite representation of the conditional probability.

Keywords: Markovian jump process, partial information, conditional probability, stochastic control theory.

1 Introduction

Queueing systems, call centers or telecommunication networks are modeled mostly as a continuous time and discrete state space Markov chain. What they have also in common, that the state process is very often not observable in a direct way. We may think for a admission-control-model, where the gatekeeper is only informed if an accepted customer finds an idle server or not, but he is not informed, when service is completed (see [7]). In [1] an inventory model is considered, where the only information about the inventory level is, whether the level is empty or not. As a third example one consider the modeling of a congested path system in the framework of TCP-internet-system done in [4]. In all of this examples there exists an observation process, which is correlated to the unobserved state process and which gives some information about the current state or changes in the system.

In this paper such a system with incomplete information is considered in a more general framework. Our underlying state process is a Markovian jump process, which is estimated again by a Markovian jump process. Each state of the observation process is a representative of an unobserved state and so jumps in the unobserved process may lead to changes in the observation state. Based on this state process a control problem over a fixed finite horizon is considered, which is not solvable in a direct way, since of the incomplete information structure.

We show how this problem can be transformed into a model with complete information with the help of conditional probabilities and show the relation between the original and the reduced model. Also an explicite formula for the conditional probability is given.

The paper is organized as follows: in section 2 we introduce our state process and the corresponding control problem over a finite horizon T. The reduction and the transformation into a problem with complete information structure is pointed out in section 3.

2 The Model

On a measurable space $(\Omega, \mathcal{F})$ consider a process (X_t) taking values in a finite set, which is identified for mathematical reason by $S_X := \{e_1, \ldots, e_n\}, n \in \mathbb{N}$, where e_i ist the i-th unit vector of $\mathbb{R}^n$. $X_t = (X_t^1, \ldots, X_t^n)$ is a pure jump process and can be characterized by the number of jumps into and out of state e_i, i.e.

$$X_t^i = \sum_{j=1}^{n} N_t^X(j,i) - \sum_{j=1}^{n} N_t^X(i,j)\,,$$

where $N_t^X(i,j)$ counts the jumps of X_t from state e_i to e_j with $i \neq j$ (see [5]). Such a jump occurs with intensity $q_{ij}^X(u)X_{t-}^i$, where u is the control parameter, specified later on. For completeness we set $N_t^X(i,i) = 0$.

The process X_t is not observable in a complete way, whereas an observation process exists, which gives us some information about the unobserved current state of X_t.

Definition 1. *Let $m \in \mathbb{N}$. We call $(I(k), k = 1, \ldots, m)$ an information structure, if*

(i) $\emptyset \neq I(k) \subset \{1, \ldots, n\}$ for all k and
(ii) $I(k) \cap I(l) = \emptyset$ for all $k \neq l$.

The set $I(k)$ is identified with f_k and f_k is called a representative of e_i, if $i \in I(k)$, where f_k is the k-th unit vector of $\mathbb{R}^m$. By the definiton of the information structure, each e_i has at most one representative, and $m \leq n$ is finite.

Example 1

1. one-to-one-representation: $m = n$ and $I(k) = \{k\}$ for all $k = 1, \ldots, m$.
2. group-representation: $m < n$ and there exists at least one k with $|I(k)| \geq 2$.
3. no-information: $m = 1, I(1) = \{1, \ldots, n\}$.

The observation process is modeled as a pure jump process on the same measurable space $(\Omega, \mathcal{F})$ and is denoted by Y_t. Y_t, taking values in the finite set $S_Y := \{f_1, \ldots, f_m\}$, is characterized by the processes $N_t^Y(k,l)$ $(k \neq l)$, where $N_t^Y(k,l)$ counts the jumps of Y_t from f_k to f_l, i.e.

$$Y_t^k = \sum_{l=1}^{m} N_t^Y(l,k) - \sum_{l=1}^{m} N_t^Y(k,l) \qquad (1)$$

and $N_t^Y(k,l)$ is defined for $k \neq l$ by

$$N_t^Y(k,l) = \sum_{i \in I(k)} \sum_{j \in I(l)} N_t^X(i,j) Y_{t-}^k . \tag{2}$$

So the intensity of $N_t^Y(k,l)$ is given by

$$q_{kl}^Y(u, X_{t-}) Y_{t-}^k := \Big(\sum_{i \in I(k)} \sum_{j \in I(l)} q_{ij}^X(u) X_{t-}^i \Big) Y_{t-}^k .$$

For completeness define $N_t^Y(k,k) = 0$.

Denote by $(\mathcal{F}_t^{X,Y})$ the filtration generated by (X_t, Y_t) and $(\mathcal{F}_t^Y)$ the filtration generated by (Y_t) and assume that both filtrations satisfy the usual conditions. By the construction of Y_t of course it is true, that $\mathcal{F}_t^{X,Y} = \mathcal{F}_t^X$.

Let $U \subset \mathbb{R}$ be a compact set, such that $\forall u \in U$ holds: $q_{ij}^X(u) \geq 0 \; \forall i \neq j$. We assume that the control process (u_t) satisfies the following assumption:

$$(A) \begin{cases} (u_t) \text{ is a càdlàg process} \\ u_t \text{ is } \mathcal{F}_t^Y\text{-predictable for all } t \\ u_t \in U \text{ for all } t . \end{cases}$$

So we can give as a summary the following martingale representation of our process (X_t, Y_t) :

$$dX_t = Q^X(u_t) X_t dt + dM_t^X \tag{3}$$
$$dY_t = Q^Y(u_t, X_t) Y_t dt + dM_t^Y , \tag{4}$$

where

- $(Q^X(u))^\top = (q_{ij}^X(u))$ with $q_{ii}^X(u) := -\sum_{j \neq i} q_{ij}^X(u)$ is the intensity matrix of X_t
- $(Q^Y(u,x))^\top = (q_{kl}^Y(u,x))$ with $q_{kk}^Y(u,x) := -\sum_{l \neq k} q_{kl}^Y(u,x)$ is the intensity matrix of Y_t
- M_t^X and M_t^Y are n- and m-dimensional martingales with expectation 0 defined by

$$M_t^X(i) = \sum_{j=1}^{n} \Big(N_t^X(j,i) - N_t^X(i,j) - \int_0^t q_{ji}^X(u_s) X_s^j ds \Big)$$
$$M_t^Y(k) = \sum_{l=1}^{m} \Big(N_t^Y(l,k) - N_t^Y(k,l) - \int_0^t q_{lk}^Y(u_s, X_s) Y_s^l ds \Big) .$$

By the construction of X_t and Y_t we have immediately, that Y_t jumps only if X_t jumps (see (2)). We also get the quadratic variation of this both processes as:

$$[X_\cdot, Y_\cdot]_t = [M_\cdot^X, M_\cdot^Y]_t = \sum_{0<s\leq t} \sum_{k=1}^{m} \sum_{l=1}^{m} \sum_{i \in I(k)} \sum_{j \in I(l)} (e_j - e_i)(f_l - f_k)^\top X_{s-}^i Y_{s-}^k .$$

Note that X_t, Y_t, M_t^X, M_t^Y depend also on the control u, so formally we have to write X_t^u, etc. For simplicity we drop this dependency in the notation.

Let $T > 0$ be a finite time horizon. Since only Y_t is observable, control at time t is allowed to depend on the observed past $\mathcal{F}_t^Y$ only, i.e. we call a control $u = (u_s)$ admissible in $[t, T]$ if

$$u \in \mathcal{U}[t,T] := \{u = (u_s) \mid (u_s)_{s\in[t,T]} \text{ satisfies } (A)\} .$$

The set $\mathcal{U}[t,T]$ is called the set of admissible controls for the time interval $[t,T]$.

Lemma 1. *There exists a probability measure $\mathcal{P}_u$ such that on $(\Omega, \mathcal{F}_T^{X,Y}, \mathcal{P}_u)$ there exists a process satisfying the dynamic equations (3) and (4).*

Proof. This proof is an immediate consequence of Kolmogorov's Theorem. □

X_t is called Markovian jump process with generator $(Q^X(u))^\top$. Note, that if the controls are Markovian, (X_t) is a continuous time Markov chain and (3) is just the martingale representation. In general this is not the case, but the intensity $q_{ij}^X(u)$ for a jump from e_i to e_j depends only on the current control and the current state, so the notion Markovian is justifed. A similiar interpretation holds for Y_t.

After defining our state process and the set of admissible controls, we finally introduce our control model over the fixed finite horizon T :

$$(P)\begin{cases} E_u\Big[\int_0^T g(s,X_s,Y_s,u_s)ds + h(X_T,Y_T)\Big] \to \min \\ dX_t = Q^X(u_t)X_t dt + dM_t^X \\ dY_t = Q^Y(u_t,X_t)Y_t dt + dM_t^Y \\ (X_0,Y_0) = (x_0,y_0) \\ u \in \mathcal{U}[0,T] \end{cases}$$

where the expectation E_u is taken with respect to $\mathcal{P}_u$.

The initial state can also be stochastic with some given initial distribution. For simplicity however we choose the initial state deterministic.

The optimal value of (P) is denoted by

$$v(P) := \inf_{u\in\mathcal{U}[0,T]} E_u\Big[\int_0^T g(s,X_s,Y_s,u_s)ds + h(X_T,Y_T)\Big]$$

and an admissible control u^* is called optimal if the induced cost are equal $v(P)$ (note once more, that (X_t,Y_t) also depends on the control).

Problem (P) can not be solved in a direct way, since the process (X_t) is not observable and has to be estimated with the help of the observation process (Y_t), which gives us indeed some information, since each state of Y_t stands for a group of states of X_t. Note, that we have in general less information, since

$$\mathcal{F}_t^Y \subset \mathcal{F}_t^{X,Y} .$$

3 The Reduction

We show now, how the model with partial information can be transformed in a model with complete information, which is then solvable in a direct way. This reduction will be done with the help of the well-known filter-technique. We prove how a stochastic differential equation for the conditional probabilities

$$\widehat{X}_t^i := \mathcal{P}_u(X_t = e_i \mid \mathcal{F}_t^Y)$$

can be derived in an explicite way. Based on this result we transform problem (P) into a problem (P_{red}), which depends only on $\mathcal{F}_t^Y$-measurable processes. We show also some equivalence structure between the two problems.

Because of (1) and $N_t^Y(k,l) = \int_0^t Y_{s-}^k dY_s^l$ it holds:

$$\mathcal{F}_t^Y = \sigma\left(N_s^Y(k,l), \text{ for } k \neq l, s \leq t\right) .$$

So the filter technique can be applied to each $N_t^Y(k,l)$ instead of Y_t and the conditional probability $\widehat{X}_t$ can be expressed in terms of the Poisson-processes $N_t^Y(k,l)$.

It is well known by [6], that the predictable $\mathcal{F}_t^Y$-intensity of $N_t^Y(k,l)$ is given by

$$E_u[q_{kl}^Y(u_t, X_{t-})Y_{t-}^k \mid \mathcal{F}_{t-}^Y] = \Big(\sum_{i\in I(k)} \sum_{j \in I(l)} q_{ij}^X(u_t)\widehat{X}_{t-}^i\Big)Y_{t-}^k =: \widehat{q}_{kl}^Y(u_t, \widehat{X}_{t-})Y_{t-}^k .$$

With (4) in mind we get the $\mathcal{F}_t^Y$-representation of Y_t as follows:

$$dY_t = \widehat{Q}^Y(u_t, \widehat{X}_t)Y_t dt + d\widehat{M}_t^Y , \tag{5}$$

where $\widehat{M}_t^Y$ is a $\mathcal{F}_t^Y$-martingale with expectation 0, which consists of the compensated $\mathcal{F}_t^Y$-Poisson-process $N_t^Y(k,l)$ having $\mathcal{F}_t^Y$-intensity $\widehat{q}_{kl}^Y(u_t, \widehat{X}_{t-})Y_{t-}^k$ (similar to section 2).

The following theorem offers an explicite (unique) representation of the conditional probability $\widehat{X}_t^i$. This formula is an extension of the results given in [2], [6] and for the unnormalized case considered in [3].

Theorem 1. *It holds:*

$$d\widehat{X}_t = Q^X(u_t)\widehat{X}_t dt + \sum_{k=1}^m \sum_{l=1}^m \Psi_{(k,l)}(u_t, \widehat{X}_{t-})\Big(dN_t^Y(k,l) - \widehat{q}_{kl}^Y(u_t, \widehat{X}_t)Y_t^k dt\Big) , \tag{6}$$

with $\Psi_{(\nu,\nu)}(u,x) = 0$ for $\nu \in \{1,\dots,m\}$ and for $k \neq l$

$$\Psi_{(k,l)}(u,x) = \frac{1}{\widehat{q}_{kl}^Y(u,x)} \begin{pmatrix} \sum_{i\in I(k)} \sum_{1\in I(l)} q_{i1}^X(u)x_i \\ \vdots \\ \sum_{i\in I(k)} \sum_{n\in I(l)} q_{in}^X(u)x_i \end{pmatrix} - x .$$

Proof. From (3) follows $d\widehat{X}_t = Q^X(u_t)\widehat{X}_t dt + d\widehat{M}_t$, where $\widehat{M}_t = E_u[M_t^X \mid \mathcal{F}_t^Y]$. Since $\widehat{M}_t$ is an $\mathcal{F}_t^Y$-martingale, it admits the representation (see e.g. [2]): $\widehat{M}_t = \int_0^t \phi_s d\widehat{M}_s^Y$, where ϕ_t is a square integrable, predictable process. As mentioned above $\widehat{M}_t^Y$ can be written as sums of the compensated $\mathcal{F}_t^Y$-Poisson-process $N_t^Y(k,l)$, so we get:

$$d\widehat{X}_t = Q^X(u_t)\widehat{X}_t dt + \sum_{k=1}^m \sum_{l=1}^m \phi_{(k,l)}(t)(dN_t^Y(k,l) - \widehat{q}_{kl}^Y(u_t, \widehat{X}_t)Y_t^k dt)\,.$$

Using Itô's formula for calculating $X_t N_t^Y(k,l)$ and $\widehat{X}_t N_t^Y(k,l)$ and using a Fubini argument by [8] we get the assertion by comparing the expectation of these two expressions. □

Equation (6) offers an explicite representation of the conditional probability $\widehat{X}_t$, but the equation is nonlinear in x which makes things very difficult for a closed formula of $\widehat{X}_t$. Since between two jumps of Y_t, $\widehat{X}_t$ follows the deterministic differential equation

$$\dot{x} = f(u,x,y)\,,$$

with $f(u,x,y) := Q^X(u)x - \sum_{k=1}^m \sum_{l=1}^m \Psi_{(k,l)}(u,x)\widehat{q}_{kl}^Y(u,x)y_k$ and if a jump of $N_t^Y(k,l)$ occurs, the new state of $\widehat{X}_t$ is $\widehat{X}_t = \widehat{X}_{t-} + \Psi_{(k,l)}(u_t, \widehat{X}_{t-})$ with probability one, $\widehat{X}_t$ is a piecewise-deterministic jump process.

If Y_t jumps from f_k into f_l (with $k \neq l$ under the assumption $m \geq 2$) we have $\widehat{X}_{t-}^i = 0$ for all $i \in I(l)$ and $\widehat{X}_t^i \geq 0$ (where at least one is > 0) for all $i \in I(l)$ and $\widehat{X}_t^i = 0$ for all $i \notin I(l)$. So each jump of Y_t leads to changes in $\widehat{X}_t$. So we have $\Psi_{(k,l)}(u,x) \neq 0$ for $k \neq l$ and therefore a one-to-one relation between Y_t and $\widehat{X}_t$. Summarizing we have

$$\mathcal{F}_t^Y = \mathcal{F}_t^{\widehat{X}}\,. \tag{7}$$

Equation (6) can be extended to the case of noised observation, where the noise is modeled as external induced jumps of Y_t, i.e. the counting process $N_t^Y(k,l)$ is of the following form:

$$N_t^Y(k,l) = \sum_{i \in I(k)} \sum_{j \in I(l)} N_t^X(i,j)Y_t^k + N_t^E(k,l)\,,$$

where $N_t^E(k,l)$ is a Poisson-process, counting some external events.

Example 2

1. one-to-one-representation: then $N_t^Y(k,l) = N_t^X(k,l)Y_t^k$ and so if $Y_t = X_t$ then $\widehat{X}_s = Y_s = X_s$ for all $s \geq t$ (i.e. $Y_0 = X_0$ then $\widehat{X}_t \equiv Y_t \equiv X_t$). For completeness: $\Psi_{(k,l)}(u,x) = f_l - x$.

2. two-group-representation: wlog we assume $I(1) = \{1, \dots, v\}$ and $I(2) = \{v+1, \dots, n\}$. Then

$$\Psi^i_{(1,2)}(u,x) + x = \begin{cases} 0 & i = 1, \dots, v \\ \frac{1}{\widehat{q}^Y_{12}(u,x)} \sum\limits_{j=1}^{v} q^X_{ji}(u) x_j & i = v+1, \dots, n \end{cases}$$

where $\widehat{q}^Y_{12}(u,x) = \sum\limits_{j=1}^{v} \sum\limits_{i=v+1}^{n} q^X_{ji}(u) x_i$. Since after a jump of Y_t from f_k to f_l, the new state of $\widehat{X}_t$ is $\widehat{X}_{t-} + \Psi_{(k,l)}(u_t, \widehat{X}_{t-})$ we see here: the probability to be then in a state, which is represented by $I(1)$ is equal to 0 and the probability for the states of representation group $I(2)$ is proportional to the jump intensities weighted with the current conditional state probability $\widehat{X}_{t-}$. A analogous result holds of course for $\Psi_{(2,1)}(u,x)$.
3. no-information: $m = 1$ and $I(1) = \{1, \dots, n\}$: then $\Psi_{(k,l)}(u,x) \equiv 0$ and $\widehat{X}_t$ follows the deterministic differential equation $\dot{x} = Q^X(u)x$, which is Kolmogorov's backward differential equation.

With equation (5) and theorem 1 in mind we can transform our problem (P) with the help of Fubini into the following reduced model.

$$(P_{\text{red}}) \begin{cases} E_u\Big[\int_0^T \bar{g}(s, \widehat{X}_s, Y_s, u_s)ds + \bar{h}(\widehat{X}_T, Y_T)\Big] \to \min \\ d\widehat{X}_t = Q^X(u_t)\widehat{X}_t dt \\ \qquad\qquad + \sum\limits_{k=1}^{m} \sum\limits_{l=1}^{m} \Psi_{(k,l)}(u_t, \widehat{X}_{t-})\Big(dN^Y_t(k,l) - \widehat{q}^Y_{kl}(u_t, \widehat{X}_t) Y^k_t dt\Big) \\ dY_t = Q^Y(u_t, \widehat{X}_t) Y_t dt + d\widehat{M}^Y_t \\ (\widehat{X}_0, Y_0) = (x_0, y_0) \\ u \in \mathcal{U}[0,T] \end{cases}$$

where

$$\bar{g}(s,x,y,u) := \sum_{i=1}^{n} g(s, e_i, y, u_s) x_i$$

$$\bar{h}(x,y) := \sum_{i=1}^{n} h(e_i, y) x_i \,.$$

Note, that the goal function is linear in the state variable $\widehat{X}_t$, but this is not the case for the stochastic differential equation of $\widehat{X}_t$, as mentioned earlier.

Problem (P_{red}) is now a problem with complete information structure and with state space $\triangle^n \times S_Y$, where $\triangle^n$ is the n-dimensional probability simplex. So it is solvable in a direct way. It is equivalent in the sense of the following reduction theorem:

Theorem 2 (Reduction). *It holds for all $t \in [0,T]$:*

a) $E_u\Big[\int_t^T g(s,X_s,Y_s,u_s)ds + h(X_T,Y_T) \mid \mathcal{F}_t^Y\Big]$
$$= E_u\Big[\int_t^T \bar{g}(s,\widehat{X}_s,Y_s,u_s)ds + \bar{h}(\widehat{X}_T,Y_T) \mid \mathcal{F}_t^Y\Big]$$

b) $\sup\limits_{u\in\mathcal{U}[t,T]} E_u\Big[\int_t^T g(s,X_s,Y_s,u_s)ds + h(X_T,Y_T) \mid \mathcal{F}_t^Y\Big]$
$$= \sup_{u\in\mathcal{U}[t,T]} E_u\Big[\int_t^T \bar{g}(s,\widehat{X}_s,Y_s,u_s)ds + \bar{h}(\widehat{X}_T,Y_T) \mid \mathcal{F}_t^Y\Big].$$

Proof. Part a) follows with the theorem of Fubini. Part b) is a direct consequence of part a). □

Corollary 1. *The following assertions are immediate consequences of theorem 2:*

a) $u = (u_t)$ *is optimal for* $(P_{red}) \Longleftrightarrow u = (u_t)$ *is optimal for* (P)
b) The optimal values of (P_{red}) *and* (P) *are the same.*

Theorem 2 simplifies with (7) in mind in the case of an (optimal) Markovian control to the following:

Theorem 3. *If the (optimal) control* (u_t^*) *is Markovian and* $(\widehat{X}_t^*,Y_t^*)$ *is the corresponding state process, then:* $E_{u^*}\Big[\int_t^T \bar{g}(s,\widehat{X}_s^*,Y_s^*,u_s^*)+\bar{h}(\widehat{X}_T^*,Y_T^*) \mid \mathcal{F}_t^{Y^*}\Big] = E_{u^*}\Big[\int_t^T \bar{g}(s,\widehat{X}_s^*,Y_s^*,u_s^*)+\bar{h}(\widehat{X}_T^*,Y_T^*) \mid \widehat{X}_t^*,Y_t^*\Big]$.

One way for solving (P_{red}) is with the help of the Hamilton-Jacobi-Bellman-equation and the verification technique, described in the next theorem.

Theorem 4 (Verification). *Let* $C^{1,1_p}$ *the set of functions, who are* C^1 *in the first and piecewise* C^1 *in the second variable. Let* $\tilde{V} : [0,T]\times\triangle^n\times S_Y \to \mathbb{R}$ *and let* $\tilde{V}(t,x,y) \in C^{1,1_p}$ *be a solution of the following HJB-equation:*

$$\begin{cases} V_t(t,x,y) + \inf\limits_{u\in U}\Big\{\bar{g}(t,x,y,u) + V_x(t,x,y)f(u,x,y) \\ \qquad + \sum\limits_{k=1}^{m}\sum\limits_{l=1}^{m}\big(V(t,x+\Psi_{(k,l)}(u,x),f_l) - V(t,x,f_k)\big)\widehat{q}_{kl}^Y(u,x)y_k\Big\} = 0 \\ V(T,x,y) = \bar{h}(x,y)\,. \end{cases}$$

If there exists $\widehat{u}(t,x,y) \in U$ *such that* $\forall (t,x,y) \in [0,T]\times\triangle^n\times S_Y$ *the* inf *is attained for* $u_0 := \widehat{u}(t,x,y)$*, then*

a) $\big(u_t^* := \widehat{u}(t,\widehat{X}_t^*,Y_t^*)\big)_{t\in[0,T]}$ *is an optimal control, where* $(\widehat{X}_t^*,Y_t^*)$ *is the to* (u_t^*) *corresponding state process*
b) $\tilde{V}(t,x,y) = E_u\Big[\int_t^T \bar{g}(s,\widehat{X}_s^*,Y_s^*,u_s^*)+\bar{h}(\widehat{X}_T^*,Y_T^*) \mid \widehat{X}_t^* = x, Y_t^* = y\Big]$ *and i.e.* $v(P_{red}) = \tilde{V}(0,x_0,y_0)$.

Proof. Apply the formula of Itô on $\tilde{V}(t,x,y)$ and calculate the expectation the assertion in a) is proven. Part b) follows from a), with theorem 3 in mind, since (u_t^*) is Markovian. □

4 Conclusions

In this paper we considered a general framework for a control model, where only groups of the unobserved state process are observable. We pointed out one possible reduction technique for solving the incomplete information model with the help of the filter technique and we showed the equivalence between the original and the reduced model. The here given construction and formula for the conditional probabilities are very general and can also be applied in many applications, like queueing models.

Acknowledgments. The author thanks an anonymous reviewer for valuable comments.

References

[1] Bensoussan, A., Cakanyildirim, M., Sethi, S.: Partially Observed Inventory Systems. Proceedings of the 44th IEEE Conference on Decision and Control, p. 1023-1028, (2005)
[2] Brémaud, P.: Point Processes and Queues: Martingale Dynamics. Springer-Verlag, (1981)
[3] Elliott, R.; Aggoun, L.; Moore, J.: Hidden Markov Models: Estimation and Control. Springer-Verlag, (1995)
[4] Miller, B., Avrachenkov, K., Stepanyan, K., Miller, G.: Flow Control as a Stochastic Optimal Control Problem with Incomplete Information. Problems of Information Transmission, 41 (2), p. 150-170, (2005)
[5] Rogers, L., Williams, D.: Diffusions, Markov Processes and Martingales. Cambridge University Press, (2003)
[6] Last, G., Brandt, A.: Marked Point Processes on the Real Line: The Dynamic Approach. Springer, (1995)
[7] Lin, K., Ross, S.: Admission Control with Incomplete Information of a Queueing System. Operations Research, 51 (4), p. 645-654, (2003)
[8] Wong, E., Hajek, B.: Stochastic Processes in Engineering Systems. Springer-Verlag, (1985)

A Tandem Queueing Network with Feedback Admission Control

Lasse Leskelä[1] and Jacques Resing[2]

[1] Helsinki University of Technology
P.O. Box 1100, FI-02015 TKK, Finland
lasse.leskela@iki.fi
[2] Eindhoven University of Technology
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
j.a.c.resing@tue.nl

Abstract. Admission control in queueing networks is often based on partial information on the network state. This paper studies how the lack of state information affects performance by considering a simple model for admission control. The model is analyzed by studying a related censored process that has a matrix-geometric steady-state distribution. Numerical results show how partial information may cause some performance characteristics in queueing networks to be nonmonotone with respect to service rates.

1 Introduction

Admisson control can be used to avoid congestion in heavily loaded queueing networks. In many queueing networks of practical interest, the admission controller may not be able to fully monitor the state of all queues in the network. This might happen for example in models of Internet-based services that rely on the infrastructure of several network operators. As a consequence, the admission decisions must be based on partial information on the network state. The goal of this paper is to study how the lack of state information affects the performance in this type of queueing models.

To deal with the question mathematically, we will restrict our attention to simple two-station tandem queues with unlimited buffers. Probably the simplest admission control mechanism based on partial information is the one where the admission decisions are based on the number of customers at the first server only, so that arriving customers are rejected whenever the length of the first queue exceeds a certain threshold. The steady-state decay rate of the number of customers in the second queue in this model was recently studied by Kroese, Scheinhardt, and Taylor [6]. Another well-studied class of tandem queues where the control is based on partial state information are the models where the first server stops processing when the second queue becomes too long [3,4,5,7].

In this paper we look at a different type of system where the control is based on the state of the second queue, but in such a way that only the admissions to the network, not the operation of the first server, can be controlled. More precisely, we consider a two-station tandem queueing network, where the interarrival times at station 1 and the service times in stations 1 and 2 are independent and exponentially distributed with parameters λ, μ_1, and μ_2, respectively. We denote the number of customers in station i by X_i, and assume that the admission controller accepts new customers to the system if and only if $X_2 \le K$, see Figure 1. The stability of this system was recently studied by Leskelä [8], who showed that the queue length process $X = (X_1, X_2)$ is positive recurrent if and only if the triple (λ, μ_1, μ_2) satisfies the relation

$$\lambda \left(1 - (\mu_1/\mu_2)^{K+1}\right) < \mu_1. \tag{1}$$

Here, we will focus on determining the steady-state distribution of X. Observe that, in contrast with the other aforementioned control models, in this system both queues can grow arbitrarily big.

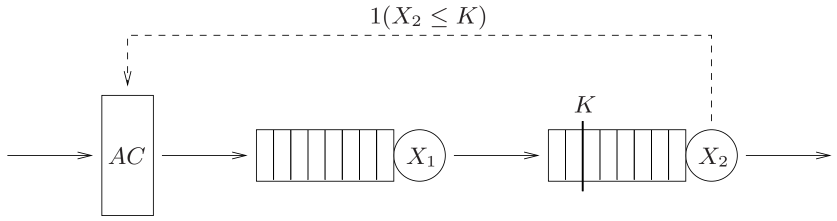


Fig. 1. The admission control mechanism

The rest of the paper is organized as follows. In Section 2, the problem of finding the steady-state distribution of the system is reduced using censoring to a simpler problem with a matrix-geometric structure. Section 3 presents an efficient numerical approach for performance evaluation of the model, with examples that illustrate how partial information affects the system. Finally, Section 4 concludes the paper.

2 Queue Length Analysis

We will assume from now on that (1) holds, so that the continuous-time Markov process $X = (X_1, X_2)$ is positive recurrent. In the sequel we will first study a modification of X restricted to periods of time during which $X_2 \le K$. This censored process has a generator with a special block structure of the so-called $G/M/1$ type [9], which will be exploited to evaluate its steady-state distribution. Afterwards, we will find the steady-state probabilities of the process X itself.

2.1 Censoring

The behavior of the process $X = (X_1, X_2)$ during periods of time when $X_2 \le K$ can be studied by censoring the parts of the sample path where X does not belong to the set $S^- = \{(n,k) \in \mathbb{Z}_+^2 : k \leq K\}$. The censored process $Y = (Y_1, Y_2)$ is defined by

$$Y_i(t) = X_i(\gamma(t)), \quad t \geq 0,$$

where

$$\gamma(t) = \inf\{\tau \geq 0 : \int_0^\tau 1_{\{X_2(s) \leq K\}}\, ds > t\}.$$

It follows from the strong Markov property [10, Section III.21] that Y is a Markov process on S^-.

To conveniently describe the infinitesimal generator of Y, we will employ the following notation for $(K+1)$-dimensional square matrices. Denote by I the identity matrix, while T_L and T_R will stand for the left and right shift matrices given by $(T_L)_{i,j} = \delta_{i-1,j}$ and $(T_R)_{i,j} = \delta_{i+1,j}$ for $0 \leq i,j \leq K$ where $\delta_{i,j}$ denotes the Kronecker delta. Further, denote the projection matrices onto 0-th and K-th coordinate by U_0 and U_K, that is, $(U_0)_{i,j} = \delta_{i,0}\delta_{j,0}$ and $(U_K)_{i,j} = \delta_{i,K}\delta_{j,K}$. Ordering the states in S^- lexicographically, the generator of Y can be written in the form

$$Q = \begin{pmatrix} B_0 & A_0 & 0 & 0 & \cdots \\ B_1 & A_1 & A_0 & 0 & \cdots \\ B_2 & A_2 & A_1 & A_0 & \cdots \\ B_3 & A_3 & A_2 & A_1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \end{pmatrix},$$

where the matrices A_n and B_n are given by

$$\begin{aligned} A_0 &= \lambda I, \\ A_1 &= \mu_2 T_L - (\lambda + \mu_1 + \mu_2) I + \mu_2 U_0, \\ A_2 &= \mu_1 (T_R + q_1 U_K), \\ A_{n+1} &= \mu_1 q_n U_K, \quad n \geq 2, \end{aligned}$$

and

$$\begin{aligned} B_0 &= \mu_2 T_L - (\lambda + \mu_2) I + \mu_2 U_0, \\ B_1 &= \mu_1 (T_R + U_K), \\ B_{n+1} &= \mu_1 (1 - q_1 - \cdots - q_n) U_K, \quad n \geq 1. \end{aligned}$$

The numbers q_n represent the probabilities that, if the process X leaves the set S^- in some state $(m+n, K)$, it enters S^- again in state (m, K), where $m \geq 1$. It is not hard to check that q_n is equal to the probability that a random walk on the integers starting at state 0 and with probabilities $\mu_1/(\mu_1+\mu_2)$ and $\mu_2/(\mu_1+\mu_2)$ of going to the right and left, respectively, reaches state -1 for the first time in exactly $2n-1$ steps. It is well-known [2] that this quantity equals

$$q_n = C_{n-1} \left(\frac{\mu_1}{\mu_1 + \mu_2} \right)^{n-1} \left(\frac{\mu_2}{\mu_1 + \mu_2} \right)^n,$$

where $C_n = \frac{1}{n+1}\binom{2n}{n}$ are the Catalan numbers.

For $k = 0, \dots, K$, denote by e_k the k-th basis vector of the $(K+1)$-dimensional euclidean space, and let $e = \sum_{k=0}^{K} e_k$. By convention, all vectors are treated as row vectors. One can check that Neuts' mean drift condition [9, Formula (1.7.11)] for the stability of Y is equivalent to (1). Furthermore, the steady-state probabilities of the censored process are given [9] in the matrix-geometric form

$$\mathrm{P}(Y = (n,k)) = x_0 R^n e_k^T, \quad (n,k) \in S^-, \tag{2}$$

where the matrix R is the unique minimal non-negative solution of

$$\sum_{n=0}^{\infty} R^n A_n = 0, \tag{3}$$

and x_0 is the unique positive row vector satisfying

$$x_0 \sum_{n=0}^{\infty} R^n B_n = 0, \quad \text{and} \quad x_0 (I - R)^{-1} e^T = 1. \tag{4}$$

2.2 Steady-State Queue Lengths

Once the steady-state distribution of the censored process Y is found, we will next show how this distribution can be used to obtain the steady-state probabilities for the queue length process X. First, note that

$$\mathrm{P}(X = (n,k)) = \mathrm{P}(X_2 \le K)\, \mathrm{P}(Y = (n,k)), \quad (n,k) \in S^-. \tag{5}$$

Second, because the steady-state rate at which customers are accepted into the system must be equal to the rate of customers coming out of the system, it follows that

$$\lambda\, \mathrm{P}(X_2 \le K) = \mu_2 (1 - \mathrm{P}(X_2 = 0)). \tag{6}$$

Note that (5) implies $\mathrm{P}(X_2 = 0) = \mathrm{P}(X_2 \le K)\, \mathrm{P}(Y_2 = 0)$. Substituting this into (6) we get

$$\mathrm{P}(X_2 \le K) = \frac{\mu_2}{\lambda + \mu_2\, \mathrm{P}(Y_2 = 0)}. \tag{7}$$

Thus, (5) and (7) yield the steady-state probabilities of X for all states $(n,k) \in S^-$, and what remains is to find the corresponding quantities on $S^+ = \mathbb{Z}_+^2 \setminus S^-$.

For the states in S^+, we will first find out the probabilities $\mathrm{P}(X = (n, K+k))$ for $n, k > 0$ by inspecting the excursions X makes in S^+. Note that if X visits the state $(n, K+k)$, the time it spends there has mean $1/(\mu_1 + \mu_2)$. Furthermore, note that the steady-state rate of transitions from $(n+m, K)$ to S^+ equals $\mu_1\, \mathrm{P}(X = (n+m, K))$. Now we see by conditioning on the state in S^- from where X enters S^+ that for all $n, k > 0$,

$$\mathrm{P}(X = (n, K+k)) = \frac{\mu_1}{\mu_1 + \mu_2} \sum_{m=k}^{\infty} q_{k,m}\, \mathrm{P}(X = (n+m, K)), \tag{8}$$

where $q_{k,m}$ is the probability that X will visit state $(n, K+k)$ during an excursion in S^+ which was initiated from state $(n+m, K)$. It is not hard to see that $q_{k,m}$ does not depend on the values of n and K, and is equal to the probability that a random walk on the integers starting from state 1 at time 0 and with probabilities $\mu_1/(\mu_1+\mu_2)$ and $\mu_2/(\mu_1+\mu_2)$ of going to the right and to the left, respectively, will visit state k at time $2m-k-1$ without visiting state 0 in any time inbetween. Using the ballot theorem (see Takács [11]) one can verify that this quantity equals

$$q_{k,m} = \frac{k}{m}\binom{2m-k-1}{m-1}\left(\frac{\mu_1}{\mu_1+\mu_2}\right)^{m-1}\left(\frac{\mu_2}{\mu_1+\mu_2}\right)^{m-k}. \tag{9}$$

Finally, the probabilities $\mathrm{P}(X=(0,k))$ with $k>K$ can be found by observing that the steady-state rate of transitions out of the set $\{(n,k') : n=0, k' \geq k\}$ equals the corresponding rate into that set, so that

$$\mu_2\,\mathrm{P}(X=(0,k)) = \mu_1 \sum_{m=k-1}^{\infty} \mathrm{P}(X_1=1, X_2=m), \quad k>K. \tag{10}$$

Alternatively, the probabilities on the left-hand side of equations (8) and (10) can be recursively determined from the balance equations, starting from the ones for $X_2=K$. In this way one avoids the infinite sums appearing on the right-hand side of equations (8) and (10).

Remark 1. In the special case where $K=0$, (3) degenerates into a scalar equation. In this case one can explicitly solve the balance equations to conclude that the steady-state distribution of the system equals

$$\mathrm{P}(X_1=n, X_2=k) = \begin{cases} \dfrac{\lambda}{\lambda+\mu_2}(1-R)\left(1-\dfrac{\mu_1}{\lambda}R\right)^{k-1}, & n=0,\ k\geq 1,\\[2ex] \dfrac{\mu_2}{\lambda+\mu_2}(1-R)\,R^n\left(1-\dfrac{\mu_1}{\lambda}R\right)^{k}, & \text{otherwise,}\end{cases}$$

where

$$R = \frac{\lambda}{2\mu_1\mu_2}\left(\sqrt{(\lambda+\mu_1-\mu_2)^2+4\mu_1\mu_2} - (\lambda+\mu_1-\mu_2)\right).$$

This result has also been independently derived by Adan and Weiss [1], who studied a two-machine 3-step re-entrant line with an infinite supply of work.

3 Performance Analysis

3.1 Throughput and Sojourn Time

We will analyse the steady-state performance of the system in terms of the throughput θ, measured as the number of customers served per unit time, and the mean sojourn time $\mathrm{E}(D)$ of accepted customers. To evaluate θ and $\mathrm{E}(D)$, one could use formulas (5) – (10) derived in Section 2.2. However, this approach is computationally not very appealing, because it involves multiple infinite summations over the state space. As an alternative, the following theorem shows how θ and $\mathrm{E}(D)$ can be calculated directly in terms of the steady-state distribution of the censored process Y.

Theorem 1. *Assume that (1) holds. Then the steady-state throughput θ and mean sojourn time $\mathrm{E}(D)$ are given in terms of the steady-state distribution of the censored process Y by*

$$\theta = \left(\frac{1}{\lambda} \mathrm{P}(Y_2 = 0) + \frac{1}{\mu_2} \right)^{-1}, \tag{11}$$

and

$$\mathrm{E}(D) = \frac{1}{\lambda} \mathrm{E}(Y_1 1_{\{Y_2=0\}}) + \frac{1}{\mu_2} \mathrm{E}(Y_1 + Y_2 + 1). \tag{12}$$

Proof. Because $\theta = \lambda \mathrm{P}(X_2 \le K)$, the validity of (11) follows immediately from (7). To prove the second claim, let us consider the level transitions for the total amount of customers $X_1 + X_2$. Under stability, the rate of events where the value of $X_1 + X_2$ changes from n to $n+1$ is given by $\lambda \mathrm{P}(X_1 + X_2 = n, X_2 \le K)$, while the corresponding rate backwards from $n+1$ to n equals $\mu_2 \mathrm{P}(X_1 + X_2 = n+1, X_2 > 0)$. Thus,

$$\lambda \mathrm{P}(X_1 + X_2 = n, X_2 \le K) = \mu_2 \mathrm{P}(X_1 + X_2 = n+1) - \mu_2 \mathrm{P}(X_1 = n+1, X_2 = 0) \tag{13}$$

for all $n \ge 0$. Multiplying both sides of (13) by $n+1$ and then summing over n we see that

$$\lambda \mathrm{E}(\, (X_1 + X_2 + 1) 1_{\{X_2 \le K\}}\,) = \mu_2 \mathrm{E}(X_1 + X_2) - \mu_2 \mathrm{E}(X_1 1_{\{X_2 = 0\}}). \tag{14}$$

Because $\mathrm{E}(D) = \theta^{-1} \mathrm{E}(X_1 + X_2)$ by Little's law, the validity of (12) now follows from solving (14) for $\mathrm{E}(X_1 + X_2)$ and using $\theta = \lambda \mathrm{P}(X_2 \le K)$.

3.2 Long-Term Behavior of the Unstable System

To better understand how the choice of parameters affects the performance of the system, it is also interesting to see what happens in the unstable parameter region. Recall from (1) that instability of the system implies $\mu_1 < \mu_2$, so that the rate at which work is fed into the second server is strictly less than its service capacity. Thus, intuition suggests that only the first queue will grow to infinity. The next theorem verifies the validity of these heuristics.

Theorem 2. *Assume $\lambda(1 - (\mu_1/\mu_2)^{K+1}) > \mu_1$. Then the process X started from an arbitrary initial state satisfies as $t \to \infty$,*

$$\begin{aligned} X_1(t) &\to \infty \quad \textit{almost surely}, \\ X_2(t) &\to Z \quad \textit{in distribution}, \end{aligned}$$

where Z is a geometrically distributed random variable with parameter μ_1/μ_2.

Proof. Let $N_\lambda, N_{\mu_1}, N_{\mu_2}$ be independent Poisson processes with rates λ, μ_1 and μ_2, respectively. Then X can be represented as the unique solution of

$$\begin{aligned} X_1(t) &= X_1(0) + \int_{(0,t]} 1_{\{X_2(s) \le K\}} \, N_\lambda(ds) - \int_{(0,t]} 1_{\{X_1(s)>0\}} \, N_{\mu_1}(ds), \\ X_2(t) &= X_2(0) + \int_{(0,t]} 1_{\{X_1(s)>0\}} \, N_{\mu_1}(ds) - \int_{(0,t]} 1_{\{X_2(s)>0\}} \, N_{\mu_2}(ds). \end{aligned} \tag{15}$$

Let $\tilde{X}_2(t)$ be the solution of

$$\tilde{X}_2(t) = X_2(0) + N_{\mu_1}(t) - \int_{(0,t]} 1_{\{\tilde{X}_2(s)>0\}} \, N_{\mu_2}(ds). \tag{16}$$

Then a pathwise comparison of (15) and (16) shows that $\tilde{X}_2(t) \ge X_2(t)$ for all t almost surely. This implies that $X_1(t) \ge U(t)$ for all t a.s., where

$$U(t) = X_1(0) + \int_{(0,t]} 1_{\{\tilde{X}_2(s) \le K\}} \, N_\lambda(ds) - N_{\mu_1}(t).$$

Note that $\tilde{X}_2$ equals the number of customers in a stable $M/M/1$ queue with arrival rate μ_1 and mean service time $1/\mu_2$. Thus, $\tilde{X}_2(t) \to Z$ in distribution, where Z is geometric with parameter μ_1/μ_2. Since N_λ is independent of $\tilde{X}_2$, it is not hard to see that $\lim_{t\to\infty} \frac{1}{t} \int_{(0,t]} 1_{\{\tilde{X}_2(s)\le K\}} \, N_\lambda(ds) = \lambda \, \mathrm{P}(Z \le K)$ a.s. Now using $\lim_{t\to\infty} \frac{1}{t} N_{\mu_1}(t) = \mu_1$ a.s., we see that with probability one,

$$\lim_{t\to\infty} U(t)/t = \lambda(1 - (\mu_1/\mu_2)^{K+1}) - \mu_1.$$

Since the above limit is strictly positive, $U(t) \to \infty$ and thus $X_1(t) \to \infty$ almost surely.

To verify that $X_2(t) \to Z$ in distribution, it is enough to show that X_2 and $\tilde{X}_2$ will couple in finite time. Let $T_0 = \sup\{t : X_1(t) = 0\}$. Since $X_1(t) \to \infty$, T_0 is a.s. finite. Define $T_1 = \inf\{t \ge T_0 : \tilde{X}_2(t) = 0\}$. Since $\tilde{X}_2$ represents the state of a stable $M/M/1$ queue, T_1 is finite a.s. Further, since X_2 is dominated by $\tilde{X}_2$, we see that $X_2(T_1) = \tilde{X}_2(T_1) = 0$. Since the pathwise dynamics of X_2 and $\tilde{X}_2$ coincide for $t \ge T_1$, we conclude that $X_2(t) = \tilde{X}_2(t)$ for all $t \ge T_1$.

3.3 Numerical Results

The steady-state distribution of the censored process $Y = (Y_1, Y_2)$ can be numerically calculated by first solving the matrix R from equation (3) using the method of successive substitutions [9], and then solving the vector x_0 from (4). The steady-state throughput θ and mean sojourn time $\mathrm{E}(D)$ are then found by combining the expressions of Theorem 1 with formula (2).

Figure 2 illustrates numerically computed contours of the throughput θ for varying μ_1 and μ_2, where $\lambda = 1$ and $K = 5$. The thick curve in the middle indicates the boundary of the stability region, so that the queue length process

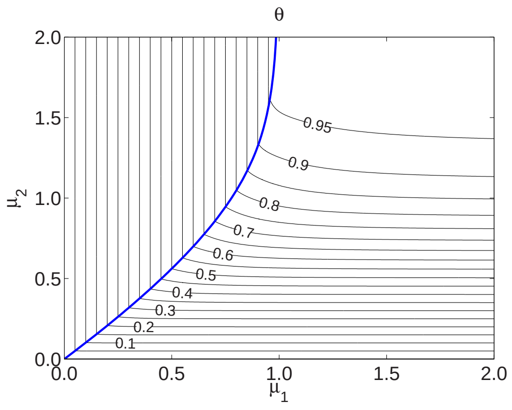


Fig. 2. Contours of θ as a function of μ_1 and μ_2 with $\lambda = 1$ and $K = 5$

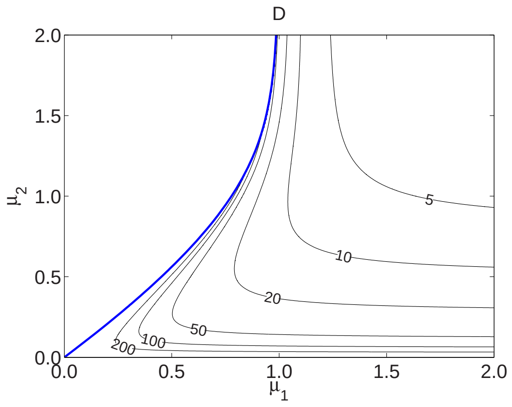


Fig. 3. Contours of $\mathrm{E}(D)$ as a function of μ_1 and μ_2 with $\lambda = 1$ and $K = 5$

$X = (X_1, X_2)$ is not positive recurrent for values of μ_1 and μ_2 located to the left of the thick curve. In the unstable region, the value of throughput does not dependent on μ_2, and is in fact equal to μ_1. This reflects the fact that $X_1(t) \to \infty$ almost surely when the system is unstable (Theorem 2), so in the long run the system serves customers at the bottleneck rate μ_1.

In Figure 3, where the corresponding contours for the mean sojourn time $\mathrm{E}(D)$ are plotted, we see that increasing the service rate for queue 2 may either increase or decrease $\mathrm{E}(D)$. In particular, if $\mu_1 < 1$, then making μ_2 large enough will eventually drive the system unstable and $\mathrm{E}(D)$ becomes infinite.

4 Conclusion

We analyzed a tandem queue with admission control based on partial information on the network state. We showed that, although the two-dimensional state space of the system is infinite in both coordinate directions, the steady-state distribution can still be analyzed using matrix-analytic methods. The approach used in Section 2 extends to a wider class of models. For example, by making suitable modifications to the matrices A_n and B_n in Section 2.1, we can model situations where during congestion the input traffic is gradually thinned by randomly rejecting a certain proportion of the newly arriving customers. The approach of the paper can still be used as long as there exists a certain maximum threshold so that all newly arriving customers are rejected whenever the length of queue 2 exceeds this maximum threshold. A different modification to A_n and B_n allows to replace the second queue in the network by a delay node of the $M/M/\infty$ type. An interesting direction for future research is to extend this approach to networks with more than two queues.

Acknowledgments. The work presented in this article was done while both authors were visiting the Mittag–Leffler Institute.

References

1. Adan, I. J. B. F. and Weiss, G.: Analysis of a simple Markovian re-entrant line with infinite supply of work under the LBFS policy. Queueing Systems **54** (2006) 169–183.
2. Feller, W.: An Introduction to Probability Theory and Its Applications, volume I. Wiley, third edition (1966).
3. van Foreest, N. D., Mandjes, M., van Ommeren, J. C. W., and Scheinhardt, W. R. W.: A tandem queue with server slow-down and blocking. Stochastic Models **21** (2005) 695–724.
4. Grassmann, W. K. and Drekic, S.: An analytical solution for a tandem queue with blocking. Queueing Systems **36** (2000) 221–235.
5. Konheim, A. G. and Reiser, M.: A queueing model with finite waiting room and blocking. Journal of the ACM **23** (1976) 328–341.
6. Kroese, D. P., Scheinhardt, W. R. W., and Taylor, P. G.: Spectral properties of the tandem Jackson network, seen as a quasi-birth-and-death process. Ann. Appl. Probab. **14** (2004) 2057–2089.
7. Latouche, G. and Neuts, M. F.: Efficient algorithmic solutions to exponential tandem queues with blocking. SIAM J. Algebra. Discr. **1** (1980) 93–106.
8. Leskelä, L.: Stabilization of an overloaded queueing network using measurement-based admission control. J. Appl. Probab. **43** (2006) 231–244.
9. Neuts, M. F.: Matrix-Geometric Solutions in Stochastic Models. John Hopkins University Press (1981).
10. Rogers, L. C. G. and Williams, D.: Diffusions, Markov Processes, and Martingales, volume I. Wiley, second edition (1994).
11. Takács, L.: Combinatorial Methods in the Theory of Stochastic Processes. Wiley (1967).

Marginal Productivity Index Policies for Admission Control and Routing to Parallel Multi-server Loss Queues with Reneging

José Niño-Mora[⋆]

Universidad Carlos III de Madrid
Department of Statistics
C/ Madrid 126
E-28903 Getafe (Madrid), Spain
jnimora@alum.mit.edu
http://alum.mit.edu/www/jnimora

Abstract. This paper addresses the problem of designing tractable dynamic admission control and/or routing policies in a Markovian model of parallel multi-server loss queues with reneging, which seek to optimize performance objectives of concern. Such problems are relevant in a variety of systems that provide distributed telecommunication or computing services. The paper shows the direct applicability of the author's results in Niño-Mora (2002) [Dynamic allocation indices for restless projects and queueing admission control: a polyhedral approach. Math. Program. **93** 361–413], where index policies based on restless bandits were developed in a broader setting. A well-grounded and tractable index policy for admission control and/or routing is thus proposed for the model of concern. Results of preliminary computational experiments are reported, showing that the proposed index policy is nearly optimal, and substantially outperforms conventional benchmark policies in the instances investigated.

Keywords: dynamic routing; admission control; index policies; parallel queues; multi-server; loss queues.

1 Introduction

This paper addresses the problem of designing tractable dynamic admission control and/or routing policies in a Markovian model of parallel multi-server loss queues with reneging, which seek to optimize performance objectives of concern. A single customer class arrives to the system as a Poisson stream with rate λ.

[⋆] The author's work has been supported in part by the Spanish Ministry of Education & Science under grants BEC2000-1027, MTM2004-02334 and a Ramón y Cajal Investigator Award, by CAM-UC3M's grant UC3M-MTM-05-075, and by the European Union's Network of Excellence Euro-NGI. He has presented earlier versions of this work at the Schloss Dagstuhl Seminar on Scheduling in Computer and Manufacturing Systems (Wadern, Germany, 2002), and at the Workshop on Topics in Computer Communication and Networks (Cambridge Univ., UK, 2002).

Upon a customer's arrival, the controller decides (i) whether to admit the customer into the system or to reject it; and, if admitted, (ii) to which of K queues in parallel to route the customer for service. Queue $k \in \mathbb{K} \triangleq \{1, \ldots, K\}$, which serves customers in FCFS order, is endowed with a pool of m_k identical parallel exponential servers, each working at rate μ_k, and has a finite buffer of size $n_k \geq m_k$. We will denote by $X_k(t)$ the state of queue k at time t, given by the number of customers it holds waiting or in service, and by $a_k(t) \in \{0, 1\}$ the action indicator taking value 1 when a customer arriving at time t would *not* be routed to queue k.

Once in a queue, customers that have not started service become impatient, being prone to renege from the system. As introduced in Palm (1957), we assume that the reneging-time distribution is exponentially distributed with rate θ_k at queue k, and that interarrival, service and reneging times are mutually independent.

The following types of cost and reward accrue: (1) *holding costs*, at the convex nondecreasing rate $h_k(j_k)$ per unit time that j_k customers are in queue k; (2) *reneging costs*, at rate c_k per customer that reneges from queue k; (3) *completion rewards*, at rate r_k per service completed at queue k; and (4) *loss costs*, at rate ν per rejected customer.

We will find it convenient to write the equivalent total net cost rate per unit time when the joint system state is $\boldsymbol{j} = (j_k)_{k\in\mathbb{K}}$ and joint action $\boldsymbol{a} = (a_k)_{k\in\mathbb{K}}$ prevails as

$$-(K-1)\lambda\nu + \sum_{k\in\mathbb{K}} \left\{C_k(j_k) + \nu Q_k^{a_k}(j_k)\right\}, \tag{1}$$

where

$$C_k(j_k) \triangleq h_j(j_k) + (j_k - m_k)^+ c_k\theta_k - \min(j_k, m_k) r_k\mu_k$$

and

$$Q_k^{a_k} \triangleq \lambda a_k.$$

We will consider two versions of the problem: (i) the case where the admission control capability is enabled; and (ii) the case where it is not, in which rejections occur only when all buffers are full. The system is operated by adopting an admission control and routing policy (for version (i)), or just a routing policy (for version (ii)), denoted by π, which is drawn from the corresponding class Π of history-dependent randomized policies.

The operation of such a system raises the following optimization problems: (i) find a policy minimizing the expected total discounted value of net costs accrued,

$$\min_{\pi\in\Pi} \mathbb{E}_{\boldsymbol{i}}^{\pi}\left[\int_0^\infty e^{-\alpha t} \sum_{k\in\mathbb{K}} \left\{C_k\big(X_k(t)\big) + \nu Q_k^{a_k(t)}\right\} dt\right], \tag{2}$$

where $\alpha > 0$ is the discount rate and $\boldsymbol{i} = (i_k)_{k\in\mathbb{K}}$ is the initial joint state; and (ii) find a policy minimizing the long-run average net cost rate per unit time,

$$\min_{\pi\in\Pi} \limsup_{T\to\infty} \frac{1}{T}\mathbb{E}_{\boldsymbol{i}}^{\pi}\left[\int_0^T \sum_{k\in\mathbb{K}} \left\{C_k\big(X_k(t)\big) + \nu Q_k^{a_k(t)}\right\} dt\right]. \tag{3}$$

Notice that in (2)–(3) we have disregarded the additive constant $-(K-1)\lambda\nu$ in (1).

Such problems are relevant in a variety of applications, most notably in the provision of geografically distributed telecommunication or computing services. Thus, e.g., in a distributed call center, calls may be initially accepted or rejected, depending on current congestion levels. If accepted, they are routed to one of multiple operator pools. The present model assumes that operators within a pool are homogeneous, whereas their skills might differ across pools. It further captures the fact that customers are prone to become impatient, possibly abandoning the system before receiving service. A similar situation arises in the operation of a distributed grid of computing nodes, consisting of workstation clusters, to which jobs are to be dynamically routed. For earlier related work see, e.g., Houck (1987), Bassamboo et al. (2005), and references therein.

While the above problems are readily formulated as discrete-time finite Markov decision processes (MDPs), their solution via the conventional dynamic programming approach is computationally intractable in large-scale models, due to the exponential growth of the state space's size on the number of queues. We will thus focus attention on the goals of designing, computing and testing well-grounded heuristic policies that are readily implementable.

Since in such problems the controller must dynamically assess the relative values of alternative rejection and routing actions, it is both intuitively appealing and practical to do so based on an *index policy.* In the present model, such policies are based on attaching to each queue k an *index* $\nu_k(j_k)$, which can be thought of as a *measure of undesirability for routing a customer to queue* k, given as a function of its current state j_k. Then, in the problem version with admission control capability, an arriving customer would be admitted if $\nu > \nu_k(j_k)$ for at least one queue k with $j_k < n_k$, i.e., if the cost of rejecting the customer exceeds the undesirability of routing it to some nonfull queue; otherwise, the customer would be rejected. If accepted, the customer would then be routed to a queue with *smallest* current index value, among nonfull queues k for which $\nu > \nu_k(j_k)$.

Indeed, for problem version (ii), the classic *Shortest Queue Routing* and *Shortest Expected Delay Routing* rules are examples of such index policies, with the former known to be optimal in special symmetric cases. See, e.g., Winston (1977), Johri (1989), and Hordijk and Koole (1990). We are thus led to address the issue of how to define appropriate indices $\nu_k(j_k)$ for the above problems.

Such an issue was actually resolved by the author in a broader setting in Niño-Mora (2002a). That paper, which drew on earlier work in Whittle (1988) and in Niño-Mora (2001), introduced the idea that problems of dynamic admission control and/or routing to parallel queues can be formulated as *restless bandit problems* (RBPs). It also established the existence of a corresponding *marginal productivity index* (MPI) for the constituent *bandits* (which correspond to a single queue subject to admission control), under rather general birth-death dynamics and nonlinear cost rate functions. Further, it furnished efficient index-computing algorithms, as well as closed index formulae for some special cases. See also the presentation in Niño-Mora (2002b).

However, the direct applicability of such work to models such as those addressed in this paper appears to have been overlooked. This paper thus sets out to clarify how such results bear on the present model. While the results are presented here in abridged form, a full version of this paper with complete analyses and extensive experimental results is currently under preparation. For related work on the theory and applications of restless bandit indexation, see also, e.g., Niño-Mora (2006a, 2006b).

The remainder of the paper is organized as follows. Section 2 discusses the reformulation of the above problems in the framework of the RBP, which allows us to deploy the corresponding MPI policy. Section 3 reviews the indexability analysis for appropriate single-queue admission control subproblems, along with index-computing algorithms, which are exploited to obtain closed-form index formulae in some special cases. Finally, Section 4 reports on the results of some preliminary computational experiments on the performance of the proposed MPI policy. These show that the proposed index policy is nearly optimal, and substantially outperforms conventional benchmark policies in the instances investigated.

2 RBP Reformulation and MPI Policy

The formulations given in (2)–(3) have been chosen to make it apparent that such problems fit into the framework of the RBP introduced by Whittle (1988). The RBP concerns the optimal dynamic allocation of effort to a collection of stochastic *projects*, modelled as binary-action (1/work/active; 0/rest/passive) MDPs. Whittle considered the problem version where a fixed number M of projects are to be engaged at each time. He showed that, for a limited class of restless projects, which he termed *indexable*, there exists a state-dependent index that characterizes their optimal policies, and proposed to use the resulting index policy for the multi-project RBP: engage at each time M projects with currently largest index values.

In the present model, as introduced in Niño-Mora (2002a), we identify a "project" with a single queue fed with a Poisson arrival stream, subject to admission control. It is convenient to imagine that such a control action is exercised by a *gatekeeper* who, when active, blocks access to the queue by shutting an entry gate, and, when passive, allows customers to enter the queue by letting the gate open. See Fig. 1. Notice that, when the queue's buffer is full, both actions yield identical dynamics. We will thus term such a full state *uncontrollable*, and the remaining states, where there is an effective choice of action, *controllable*. Further, for consistency with the interpretation of action a_k used in (1), where $a_k = 1$ means *not* routing to queue k, we will adopt the convention that the active action is taken at uncontrollable states, i.e., $a_k = 1$ if queue k is full.

Now, to view the present model as a multi-project RBP we represent it as a collection of queues as above, *each fed by its own arrival stream with rate* λ and having its own gatekeeper, where the actions of gatekeepers are coordinated in such a way as to yield the required dynamics. Thus, in the problem version with admission control capability, *at least* $M = K - 1$ projects are to be engaged at each time, so that at least $K-1$ queues' entry gates are to be shut: if $K-1$ gates are shut, an arriving customer would be routed to the queue whose gate is open, which must be nonfull; if the K gates are shut, the customer would be rejected. In the problem version without admission control capability, exactly exactly $K-1$ queues' entry gates are shut at any time, and an arriving customer would be routed as in the previous case, provided there is at least one nonfull queue; if all queues are full, the K entry gates must be shut.

Now, to deploy restless bandit indexation we *decouple* problems (2)–(3) into the *single-project subproblems*

$$\min_{\pi_k \in \Pi_k} \mathbb{E}_{i_k}^{\pi_k} \left[\int_0^\infty e^{-\alpha t} \Big\{ C_k\big(X_k(t)\big) + \nu Q_k^{a_k(t)} \Big\} \, dt \right] \tag{4}$$

and

$$\min_{\pi_k \in \Pi_k} \limsup_{T \to \infty} \frac{1}{T} \mathbb{E}_{i_k}^{\pi_k} \left[\int_0^T \Big\{ C_k\big(X_k(t)\big) + \nu Q_k^{a_k(t)} \Big\} \, dt \right], \tag{5}$$

where i_k is the initial state of queue k, and Π_k is the class of admissible admission control policies for operating the queue *in isolation*.

We now use the rejection cost rate ν as a parameter, and consider how the optimal policies for such subproblems vary as ν ranges over $\mathbb{R}$. We will say that any of the above queue k's subproblems is *indexable* if there exists an *index* $\nu_k^*(j_k)$, defined for controllable states $0 \le j_k < n_k$, such that, for any initial state i_k, it is optimal to take the active action, i.e., reject an arrival when the queue occupies state j_k, iff $\nu_k^*(j_k) \ge \nu$. In such case, we term $\nu_k^*(j_k)$ the queue's *marginal productivity index* (MPI). Such a term was introduced in Niño-Mora 2006a, motivated by the interpretation of the index given there as the state-dependent marginal cost vs. work trade-off rate.

Hence, if it were established that such an MPI exists for the above single-queue subproblems, we could use it to implement an index policy as discussed above.

3 Single-Queue Subproblems: Indexability and MPI Computation

In fact, the required indexability analysis was carried out in Niño-Mora (2002a), in a broader setting of admission control problems with birth-death dynamics and nonlinear cost rates, which includes the above subproblems as special cases. We review here such results, which further include an efficient index-computing algorithm, and highlight their application to the present model.

3.1 Review of Relevant Results in Niño-Mora (2002a)

Consider the system portrayed in Fig. 1, which represents a single-server facility catering to an incoming customer stream, which is endowed with a finite buffer capable of holding up to and including n customers, waiting or in service. Customer flow is regulated by a *gatekeeper*, who dynamically opens or shuts an *entry*

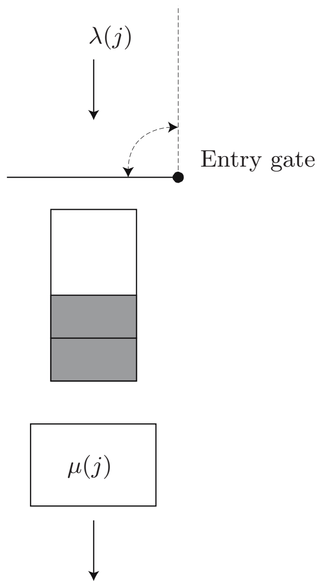


Fig. 1. Control of admission to a single queue

gate which customers must cross to enter the buffer; those finding a shut gate, or a full buffer, on arrival are rejected and lost.

The *state* $X(t)$, recording the number in system at times $t \geq 0$, evolves as a controlled *birth-death process* with state space $N = \{0, \ldots, n\}$. While the system occupies state j, customers arrive at rate $\lambda(j)$ (being then admitted or rejected), and the server works at rate $\mu(j)$.

The system is governed by an *admission control policy* π, prescribing the action $a(t) \in \{0, 1\}$ to take at each time t, where action 1 means shutting the entry gate. As before, we assume that such an action must be taken when the queue is full.

The system continuously accrues holding costs, at rate $C(j)$ while in state j, along with rejection charges, at rate ν per customer rejected. The equivalent rejection cost rate under action a is therefore $\nu Q^a(j)$, where $Q^a(j) \triangleq \lambda(j)a$ is the rejection cost rate.

In such a setting, we consider the following optimization problems: (i) find a discount-optimal admission control policy,

$$\min_{\pi \in \Pi} \mathbb{E}_i^\pi \left[\int_0^\infty e^{-\alpha t} \Big\{ C\big(X(t)\big) + \nu Q^{a(t)}\big(X(t)\big) \Big\} \, dt \right], \tag{6}$$

where $\alpha > 0$ is the discount rate and i is the initial state; and (ii) find an average-optimal policy,

$$\min_{\pi \in \Pi} \limsup_{T \to \infty} \frac{1}{T} \mathbb{E}_i^\pi \left[\int_0^T \Big\{ C\big(X(t)\big) + \nu Q^{a(t)}\big(X(t)\big) \Big\} \, dt \right]. \tag{7}$$

We will assume that the model parameters satisfy the following regularity conditions, where we use the notation $\Delta x(j) \triangleq x(j) - x(j-1)$, and write

$$d(j) = \mu(j) - \lambda(j),$$

and

$$\rho(j) = \frac{\lambda(j)}{\mu(j+1)}.$$

Assumption 1. *The following conditions hold:*

(i) *Concave nondecreasing* $d(j)$*:* $0 \le \Delta d(j+1) \le \Delta d(j), \quad 1 \le j \le n-1$*, with* $\Delta d(1) > 0$.
(ii) *Convex nondecreasing* $C(j)$*:* $\Delta C(j+1) \ge \Delta C(j) \ge 0, \quad 1 \le j \le n-1$.

Now, in Niño-Mora (2002a[Th. 7.2 and Cor. 7.1]) it is proven that, under Ass. 1, such problems are indexable, so that their optimal policies are characterized by an MPI $\nu^*(j)$. Further, the latter is consistent with the intuitive class of *threshold policies*, which prescribe to reject customers if the queue is long enough. Namely, the MPI is monotone nondecreasing in the queue length j:

$$\nu^*(0) \le \nu^*(1) \le \cdots \le \nu^*(n-1).$$

That paper further gives a recursive index algorithm, which we include here for ease of reference:

$$\begin{aligned}
\nu^*(0) &= \frac{\Delta C(1)}{\alpha + \Delta d(1)} \\
\nu^*(j) &= \nu^*(j-1) + \frac{\Delta C(j+1) - \nu^*(j-1)\big(\alpha + \Delta d(j+1)\big)}{\alpha + \Delta d(j+1) + \dfrac{w^{S(j+1)}(j-1)}{\rho(j-1)}}, \quad 1 \le j \le n-1,
\end{aligned} \tag{8}$$

where $S(j+1) \triangleq \{j, \ldots, n-1\}$ is an *active-state set* corresponding to a threshold policy, and quantities $w^{S(j+1)}(j-1)$ are *marginal workloads*, which are recursively computed by

$$\begin{aligned}
w^{S(2)}(0) &= \lambda(0) \frac{\alpha + \Delta d(1)}{\alpha + \lambda(0) + \mu(1)} \\
w^{S(j+1)}(j-1) &= \lambda(j-1) \frac{\alpha + \Delta d(j) + \dfrac{w^{S(j)}(j-2)}{\rho(j-2)}}{a(j)\big(\alpha + \lambda(j-1) + \mu(j)\big)}, \quad 2 \le j \le n-1.
\end{aligned} \tag{9}$$

In (9), quantities $a(j)$ are also recursively computed, by letting $a(1) = 1$, and

$$a(j) = 1 - \frac{\lambda(j-1)\mu(j-1)}{\big(\alpha + \lambda(j-2) + \mu(j-1)\big)\big(\alpha + \lambda(j-1) + \mu(j)\big)} \frac{1}{a(j-1)}, \quad 2 \le j \le n. \tag{10}$$

For the average criterion, one need simply set the value $\alpha = 0$ in the above recursions.

3.2 Application to the Present Model

To apply the above results to the present model, we must identify the corresponding parameters for each single-queue subproblem, and verify that they satisfy Ass. 1. Clearly, the parameters of concern should be defined as follows, where we have dropped the queue label k in the notation:

$$\begin{aligned} &\lambda(j) \triangleq \lambda, \quad 0 \leq j \leq n, \\ &\mu(j) \triangleq (j-m)^{+}\theta + \min(j,m)\mu, \quad 1 \leq j \leq n \\ &C(j) \triangleq h(j) + (j-m)^{+}c\theta - \min(j,m)r\mu, \quad 0 \leq j \leq n. \end{aligned} \tag{11}$$

With such definitions, it is readily verified that Ass. 1 holds, and, therefore, the results reviewed above apply to the model of concern in this paper.

3.3 Some Closed-Form First- and Second-Order MPI Formulae

In certain special cases, it is possible to solve explicitly the above index recursions to obtain closed-form expressions for the MPI.

Thus, in the classic case of multiple $M/M/1$ queues (no reneging) in parallel, with $h(j) = hj$ and $r = 0$, under the average criterion, it is shown in Niño-Mora (2002a) that the MPI is given by

$$\nu^*(j) = \frac{c}{\mu}\sum_{i=1}^{j+1}\left(1+\cdots+\rho^{i-1}\right) = \begin{cases} \dfrac{c}{\mu}\left[\dfrac{\rho^{j+2}-1}{(\rho-1)^2} - \dfrac{j+2}{\rho-1}\right] & \text{if } \rho \neq 1 \\ \dfrac{c}{\mu}\,\dfrac{(j+1)\,(j+2)}{2} & \text{if } \rho = 1, \end{cases} \tag{12}$$

where $\rho = \lambda/\mu$.

Another interesting case is that where the objective of concern is to *maximize the system's throughput.* In the case of multiple $M/M/1$ queues in parallel, with $h(j) \equiv 0$ and $r = 1$, under the average criterion, the index recursion above yields the *constant* MPI $\nu^*(j) \equiv -1$. Since such an index is noninformative, we proceed as in Niño-Mora (2006b) by introducing a tie-breaking, *second-order MPI* $\gamma^*(j)$, based on the MacLaurin expansion on the discounted MPI,

$$\nu^*(j) = -1 + \alpha\gamma^*(j) + O(\alpha), \qquad \text{as } \alpha \searrow 0.$$

Thus, the corresponding index policy routes customers to a queue with smallest second-order MPI.

It is readily verified that such a second-order MPI is computed recursively by

$$\begin{aligned} &\gamma^*(0) = 1/\mu \\ &\gamma^*(j) = \gamma^*(j-1) + \frac{1}{\mu}\left(1+\cdots+\rho^{j}\right), \quad 1 \leq j < n. \end{aligned} \tag{13}$$

Further, the solution to such a recursion is as follows. In the case $\rho = \lambda/\mu \neq 1$,

$$\gamma^*(j) = \frac{j+1-(j+3)\rho+\rho^2+\rho^{j+1}}{\mu(\rho-1)^2},$$

and, in the case $\rho = 1$,

$$\gamma^*(j) = \frac{2 + j + j^2}{2\mu}.$$

4 Some Computational Experiments

This section reports on some preliminary experimental results on the relative performance of the proposed MPI policy. More thorough results, resulting from a large-scale computational study, will be reported in the full version of this paper.

In the following experiments, the performance of the MPI policy is compared against the optimal performance, and against the performance of two benchmark policies: the classic *Shortest Queue* (SQ) routing policy, and the *Smallest Expected Cost* (SEC) routing policy, which is the individually optimal policy. The experiments were performed with MATLAB 2006b, using implementations developed by the author. The optimal performance was computed by solving the linear programming formulation of the corresponding DP equations, using the CPLEX solver interfaced with MATLAB via TOMLAB. The alternative policies considered were evaluated by solving the appropriate linear evaluation equations.

In the first experiment, we investigate how relative performance varies with the arrival rate λ, using the the base instance having $K = 2$ queues in parallel, and the following parameters. Buffer sizes are $(n_1, n_2) = (25, 25)$, the numbers of servers are $(m_1, m_2) = (4, 5)$, the service rates are $(\mu_1, \mu_2) = (0.60, 0.40)$, the reneging rates are $(\theta_1, \theta_2) = (0.2, 0.2)$, the holding cost rates are $(h_1, h_2) = (1, 1)$, the reneging cost rates are $(c_1, c_2) = (0.3, 0.3)$, the completion rewards are $(r_1, r_2) = (1, 1)$, and the cost per rejected customer is $\nu = 0$. We investigate the model under the long-run average criterion, and in the version where only routing decisions are allowed.

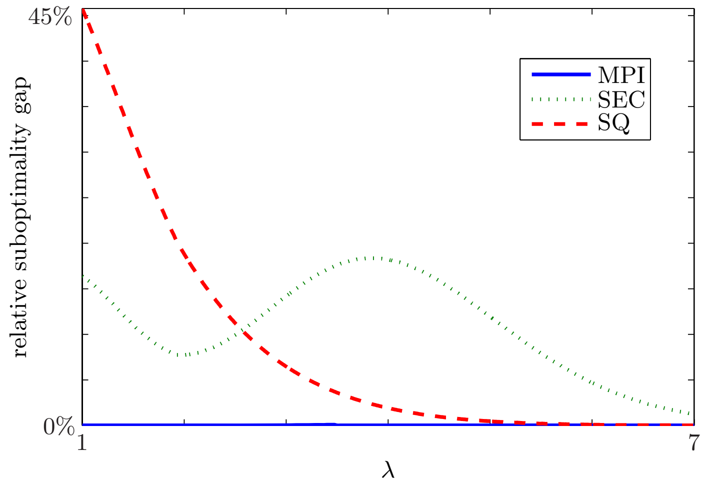


Fig. 2. Exp. 1: Relative suboptimality gaps of MPI, SEH and SQ policies as λ varies

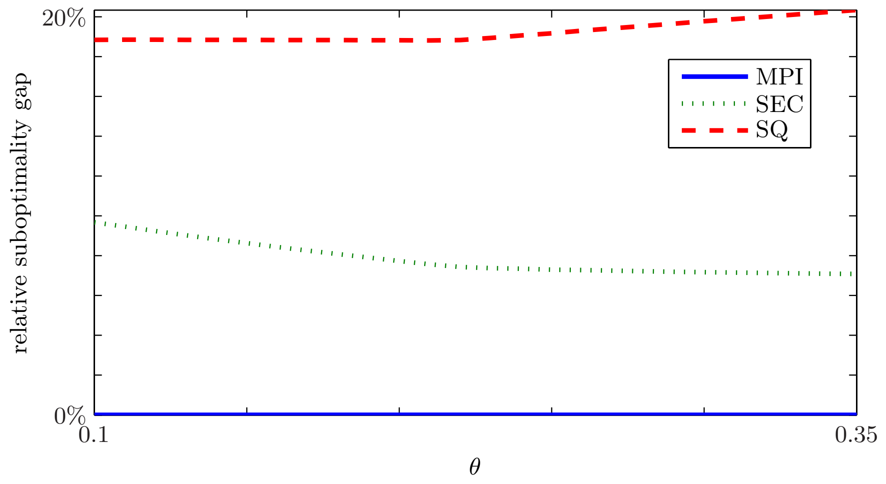


Fig. 3. Exp. 2: Relative suboptimality gaps of MPI, SEH and SQ policies as θ varies

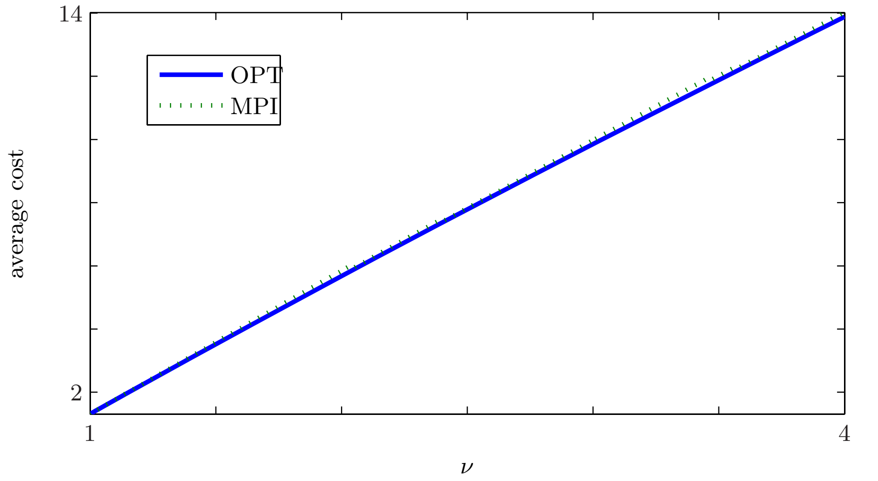


Fig. 4. Exp. 3: Performance of MPI and optimal (OPT) routing policies as ν varies

Fig. 2 plots the relative suboptimality gaps of the MPI, SEC and SQ policies as the arrival rate λ varies over the interval $[1, 7]$. The plot shows that the MPI policy is nearly optimal throughout such a range, consistently outperforming the SEC and SQ policies. The SEC policy is better than the SQ policy in relatively light traffic, whereas the opposite holds in heavier traffic. The plot further shows that the three policies are asymptotically optimal in heavy traffic.

In the second experiment, we investigated how relative performance varies with a constant reneging rate θ, using a base instance as in the previous experiment, fixing the arrival rate to $\lambda = 2$. Fig. 3 plots the relative suboptimality gaps

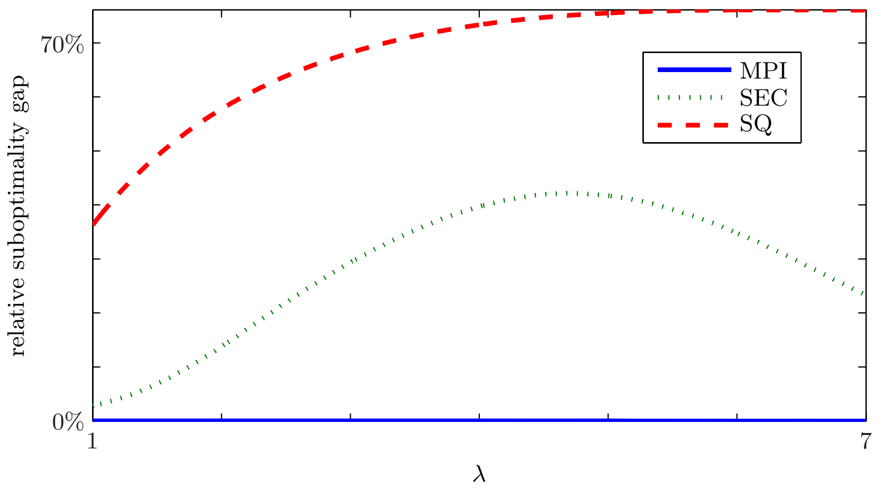


Fig. 5. Exp. 4: Relative suboptimality gaps of MPI, SEH and SQ policies as λ varies

of the MPI, SEC and SQ policies as the reneging rate θ varies over the interval $[0.1, 0.35]$. Again, the plot shows that the MPI policy is nearly optimal throughout such a range, consistently outperforming the SEC and SQ policies. Among the latter, the SEC policy is the better one, with its relative suboptimality gap ranging from between 8% to 10%.

In the third experiment, we investigated how relative performance varies with the rejection cost rate ν, in the problem version with admission control capability, using a two-queue base instance with parameteres $(n_1, n_2) = (25, 25)$, $(m_1, m_2) = (4, 5)$, $(\mu_1, \mu_2) = (0.60, 0.40)$, $(\theta_1, \theta_2) = (0.2, 0.2)$, $(h_1, h_2) = (1, 1)$, $(c_1, c_2) = (0.3, 0.3)$, $(r_1, r_2) = (3, 3)$, and $\lambda = 4$. Fig. 4 plots the average cost incurred by the MPI policy and the optimal average cost, as nu ranges over the interval $[1, 4]$. Again, the plot shows that the MPI policy is nearly optimal throughout such a range.

Finally, in the fourth experiment, we investigated how relative performance varies with the arrival rate λ in a problem where the objective is to maximize expected total discounted throughput, using the the base instance having $K = 2$ queues in parallel, and the following parameters: $(n_1, n_2) = (25, 25)$, $(m_1, m_2) = (4, 5)$, $(\mu_1, \mu_2) = (0.60, 0.40)$, $(\theta_1, \theta_2) = (0.2, 0.2)$, $(h_1, h_2) = (0, 0)$, $(c_1, c_2) = (0, 0)$, $(r_1, r_2) = (1, 1)$, and the cost per rejected customer is $\nu = 0$. We investigate the model under the discounted criterion, with discount rate $\alpha = 0.01$, letting λ range over $[1, 7]$.

The results are shown in Fig. 5, where it is seen that, again, the MPI policy is nearly optimal throughout the parameter range, while the SEC policy exhibits a moderately poor performance, and the SQ policy shows a substantially worse performance.

References

Bassamboo, A., Harrison, J. M., Zeevi, A.: Dynamic routing and admission control in high-volume service systems: asymptotic analysis via multi-scale fluid limits. Queueing Syst. **51** (2005) 249–285

Hordijk, A., Koole, G.: On the optimality of the generalized shortest queue policy. Probab. Eng. Inf. Sci. **4** (1990) 477–487

Houck, D. J.: Comparison of policies for routing customers to parallel queueing sytems. Oper. Res. **35** (1987) 306–310

Johri, P. K.: Optimality of the shortest line discipline with state-dependent service rates. European J. Oper. Res. **41** (1989) 157–161

Niño-Mora, J.: Restless bandits, partial conservation laws and indexability. Adv. in Appl. Probab. **33** (2001) 76–98

Niño-Mora, J.: Dynamic allocation indices for restless projects and queueing admission control: a polyhedral approach. Math. Program. **93** (2002a) 361–413

Niño-Mora, J.: Admission control and routing to parallel queues via Whittle's restless bandit index policy. Presentation at the Workshop on Topics in Computer Communication and Networks, Isaac Newton Institute for Mathematical Sciences, Cambridge Univ., UK (available at http://www.newton.cam.ac.uk/webseminars/pg+ws/2002/CMPW03/1217/nino-mora/) (2002b)

Niño-Mora, J.: Restless bandit marginal productivity indices, diminishing returns and optimal control of make-to-order/make-to-stock $M/G/1$ queues. Math. Oper. Res. **31** (2006a) 50–84

Niño-Mora, J.: Marginal productivity index policies for scheduling a multiclass delay-/loss-sensitive queue. Queueing Syst. **54** (2006b) 281–312

Palm, C.: Contributions to the theory of delay systems. Tele **1** (1957) 37–67

Whittle, P.: Restless bandits: Activity allocation in a changing world. A Celebration of Applied Probability, J. Gani (ed.), J. Appl. Probab. **25A** (1988) 287–298

Winston, W.: Optimality of the shortest line discipline. J. Appl. Probab. **14** (1977) 181–189

Some Examples of Stochastic Approximation in Communications

Vivek S. Borkar

Tata Institute of Fundamental Research,
Mumbai 400005, India
borkar@tifr.res.in

Abstract. A brief overview of some interesting dynamics commonly arising as scaling limits of stochastic approximation type algorithms is given, with sample applications to communications.

Keywords: stochastic approximation, power control, routing, dynamic pricing.

1 Introduction

Stochastic approximation [16] is a tried and tested paradigm for adaptive algorithms, which exploit its incrementality (leading to graceful convergence), low per iterate computational and memory requirements, and its 'averaging' property which makes it ideal for noisy situations. While it retains its place of pride in its traditional application domains such as adaptive control and signal processing, it has also found new applications in communications, such as in adaptive resource allocation problems arising in wireless and wireline networks. This is a brief summary of some key paradigms that fall in this category. An important development in stochastic approximation theory, going back to [7], [15], has been the 'o.d.e.' (for '*ordinary differential equations*') approach which allows us to analyze the algorithm in terms of a limiting o.d.e. In turn, an o.d.e. can be mapped onto an algorithm. Thus we list certain classes of convergent o.d.e.s as proto-algorithms and give examples of actual algorithms where their convergence properties have been exploited.

2 Stochastic Approximation

These are stochastic recursions in $\mathcal{R}^d$ of the form

$$x_{n+1} = x_n + a(n)[h(x_n) + M_{n+1}],\ n \geq 0,$$

where h is Lipschitz, $\{M_n\}$ a martingale difference (i. e., $E[M_{n+1}|x_m, M_m, m \leq n] = 0$) satisfying $E[\|M_{n+1}\|^2|x_m, M_m, m \leq n] \leq K(1 + \|x_n\|^2)$ for some $K > 0$, and $\{a(n)\}$ is a stepsize schedule, i. e., positive scalars, satisfying $\sum_n a(n) = \infty$ and $\sum_n a(n)^2 < \infty$. (This is a typical set of conditions, variations are possible.)

The o.d.e. approach says that under suitable conditions (which in particular should ensure the a. s. boundedness of iterates), the iterates have a. s. the same asymptotic behavior as the o.d.e.

$$\dot{x}(t) = h(x(t)).$$

In case of the o.d.e. having a finite set of isolated attractors, one can refine this (under mild additional assumptions) to say that $\{x_n\}$ a. s. converges to a stable attractor. Some important variations are:

1. *Projected schemes:* When the a. s. boundedness of $\{x_n\}$ cannot be ensured a priori, one may project the iterate to a compact convex set whenever it exits from this set. This leads to a 'projected o.d.e.' limit. One needs to ensure, however, that no spurious boundary equilibria are introduced by the projection.
2. *Markov noise:* Here $h(x_n)$ above gets replaced by $\tilde{h}(x_n, Z_n)$, where $\{Z_n\}$ is an ergodic Markov process with marginal μ. The same o.d.e. applies with $h(x) = \int \tilde{h}(x, z)\mu(dz)$.
3. *Two timescale systems:* These are coupled iterations

$$x_{n+1} = x_n + a(n)[h(x_n, y_n) + M_{n+1}],$$
$$y_{n+1} = y_n + b(n)[g(x_n, y_n) + M'_{n+1}],$$

   with conditions similar to the above on $h, g, \{M_n\}, \{M'_n\}$, and with $\sum_n a(n) = \sum_n b(n) = \infty$, $\sum_n a(n)^2 + \sum_n b(n)^2 < \infty$, $\frac{b(n)}{a(n)} \to 0$. Thus the second iteration moves on a slower timescale and sees the first as quasi-equilibrated, which in turn sees it as quasi-static. To analyze this, consider the o.d.e. $\dot{x}(t) = h(x(t), y)$ and suppose it has a globally asymptotically stable equilibrium $\lambda(y)$ for a 'nice' $\lambda(\cdot)$. Also consider $\dot{y}(t) = g(\lambda(y(t)), y(t))$ and suppose it has a unique globally asymptotically stable equilibrium y^*. Then under reasonable conditions, $(x_n, y_n) \to (\lambda(y^*), y^*)$ a. s.
4. *Asynchronous schemes:* Suppose different components of x_n are updated by different processors, not all of which update at each time (i. e., they have their own 'clocks') and which communicate their outputs to each other with random delays. The $i-$th component is then updated according to

$$x_{n+1}(i) = x_n(i) + a(\nu(i,n))I\{i \in Y_n\}[h_i(x_{n-\tau_{i1}(n)}(n), \\ \cdots, x_{n-\tau_{id}(n)}) + M_{n+1}(i)],$$

   where Y_n is the set of components updated at time n, $\{\tau_{ij}(n)\}$ the delays, and $\nu(i,n) = \sum_{m\le n} I\{i \in Y_m\}$ the local clock. One can show that the iterates then track the nonautonomous o.d.e. $\dot{x}(t) = \Lambda(t)h(x(t))$, where $\Lambda(t)$ is a diagonal matrix with nonnegative entries. The same is true when $a(\nu(i,n))$ is replaced by $a(n)$. This o.d.e. in some cases of interest can be shown to have the same asymptotic behavior as that corresponding to $\Lambda(\cdot) \equiv$ the identity matrix, if $\liminf_{n\to\infty} \nu(i,n)/n > 0$ for all i, i. e., the components are updated comparably often.

5. *Constant stepsize:* One often considers $a(n) \equiv a > 0$, e.g., when tracking a slowly varying environment. Then the standard claims of 'converges a. s. to $\cdots$' for decreasing stepsizes above need to be replaced by the corresponding statement 'concentrates with a high ($\approx 1 - O(a)$) probability near $\cdots$'.

See [3] for details.

3 Stochastic Gradient Schemes

Here $h = \nabla f$ for a suitable f and thus $\{x_n\}$ converge a. s. to a local maximum of f. As an example [1], consider N users sharing an ergodic Markov channel with stationary distribution ν. The aim is to minimize average power subject to a minimum rate constraint. Let A denote the set of unit coordinate vectors in $\mathcal{R}^N$, the $i-$th vector signifying the choice of $i-$th user for the slot. Let $p_2(y|x)$ denote the conditional distribution of the user given channel state and $p_1(q|y,x)$ the conditional distribution of this user's power consumption. The problem becomes

$$\min \int \nu(dx) \sum_{y \in A} \int_0^\infty p_1(dq|y,x) p_2(y|x) q \quad \text{subject to}$$

$$\int \nu(dx) \sum_{y \in A} \int_0^\infty p_1(dq|y,x) \log(1 + q y_i x_i) \geq C_i \ \forall i.$$

The optimal solution is to select user

$$k = \text{argmin}_i \left((\lambda_i - \frac{1}{x_i})^+ - \lambda_i [\log(1 + (\lambda_i - \frac{1}{x_i})^+ x_i) - C_i] \right),$$

who will transmit power

$$q^* = (\lambda_k - \frac{1}{x_k})^+,$$

λ_i being the Lagrange multiplier associated with the $i-$th constraint. The latter can be learnt adaptively by the stochastic gradient scheme

$$\lambda_i(n+1) = \Gamma(\lambda_i(n) - a(n) y_i(n) [\log(1 + (\lambda_i - \frac{1}{x_i})^+ x_i(n)) - C_i]), \ \forall i.$$

Here $y_i(n) = I\{\alpha_i \leq \alpha_j, j \neq i\}$ for

$$\alpha_i = q_i^* - \lambda_i(n) [\log(1 + (\lambda_i - \frac{1}{x_i(n)})^+ x_i(n)) - C_i], \ 1 \leq i \leq N,$$

and Γ is projection to $[0, L]$ for a large L. The idea is to use the Lagrange multiplier formulation in order to cast the constrained optimization problem as an unconstrained min-max (= max-min) problem, do the minimization over both the users and power explicitly as above, and the maximization over Lagrange multipliers by stochastic approximation. The foregoing theory ensures desired asymptotics, which is also verified by simulation experiments [1].

4 Fixed Point Iterations

Here $h(x) = F(x) - x$, where F is a contraction w.r.t. a suitable norm and the aim is to find its unique fixed point x^* given by $x^* = F(x^*)$. This can be shown to be the unique asymptotically stable equilibrium for the o.d.e. limit $\dot{x}(t) = F(x(t)) - x(t)$. This can be extended to nonexpansive maps in some cases.

The standard application of this is to dynamic programming. Consider, e.g., the nonnegative integer valued queue process $\{X_n\}$ given by $X_{n+1} = X_n - u_n + W_{n+1}$, where $\{W_n\}$ is the i.i.d. packet arrival process with law μ and $u_n \in [0, x_n]$ is the number of packets transmitted at time n. Consider a constrained Markov decision process that seeks to minimize

$$\limsup_{n\to\infty} \frac{1}{n} \sum_{m=0}^{n-1} c(X_m, u_m) \text{ s. t. } \limsup_{n\to\infty} \frac{1}{n} \sum_{m=0}^{n-1} c_i(X_m, u_m) \leq C_i, 1 \leq i \leq N.$$

Using a standard Lagrange multiplier formulation [2], this can be reduced to an unconstrained MDP with running cost $c + \sum_i \lambda_i c_i$, with λ_i's the Lagrange multipliers. The corresponding dynamic programming equation is

$$\tilde{V}(x) = \min_u [c(x,u) + \sum_{i=1}^{N} \lambda_i c_i(x,u) - \beta + \sum_w \mu(w) \tilde{V}(x - u + w)].$$

Here β is the optimal cost. One may view the transition $X_n \to X_{n+1}$ as a composition of $X_n \to X_n^+ = X_n - u_n$ (the 'post-state') and $X_n^+ \to X_{n+1} = X_n^+ + W_{n+1}$. In terms of $\{X_n^+\}$, the dynamic programming equation becomes

$$V(x) = \sum_w \mu(w) \min_u [c(x+w,u) + \sum_{i=1}^{N} \lambda_i c_i(x+w,u) - \beta + V(x - u + w)].$$

The difference is that the minimization is now inside the expectation. This allows us to write the stochastic approximation version of the corresponding 'relative value iteration':

$$\begin{aligned} V_{n+1}(i) = V_n(i) + a(\nu(i,n)) I\{X_n^+ = i\} [\min_u [c(X_{n+1}, u) + \\ \sum_{i=1}^{N} \lambda_i(n) c_i(X_{n+1}, u) - V_n(i_0) + V(X_{n+1} - u)]. \end{aligned}$$

Here i_0 is a prescribed state and $\nu(i,n) = \sum_{m \leq n} I\{X_m = i\}$. The Lagrange multipliers are updated on a slower timescale by the stochastic ascent.

$$\lambda_i(n+1) = \lambda_i(n) + b(n)[c_i(X_n, u_n) - C_i] \ \forall i.$$

The convergence can be proved by using the two timescale analysis above, that the slow component performs the correct gradient ascent being a consequence of the generalized 'envelope theorem' from mathematical economics. See [17] for details and simulations.

5 Cooperative o.d.e.

This is the case when $\frac{\partial h_i}{\partial x_j} > 0,\ j \neq i$. (More generally, '$\geq 0$' and the Jacobian matrix of h is irreducible.) These are called *cooperative* o.d.e.s because any increase in the $i-$th component leads to an increase in $j-$th component of the driving vector field for $j \neq i$. If the trajectories remain bounded, then for all initial conditions belonging to an open dense set, $x(\cdot)$ converges to the set of equilibria [12], [19].

As an application, we consider the problem of dynamic pricing in a system of parallel queues [5], [9]. There are K parallel queues and an entry charge $p_i(n)$ is charged for the $i-$th queue. Let $y_i(n)$ denote the queue length in $i-$th queue at time n. There is an 'ideal profile' $y^* = [y_1^*, \cdots, y_K^*]$ of queue lengths which we want to stay close to, and the objective is to manage this by modulating the respective prices dynamically. Let Γ_i denote the projection to $[\epsilon_i, B_i]$ for $0 \leq i \leq K$, where $\epsilon_i > 0$ is a small number and B_i is a convenient a priori upper bound. Let $a > 0$ be a small constant stepsize. The scheme is: for $1 \leq i \leq K$,

$$p_i(n+1) = \Gamma_i\left(p_i(n) + ap_i(n)[y_i(n) - y_i^*]\right),\ n \geq 0.$$

The idea is to increase the price if the current queue length is above the ideal (so as to discourage new entrants) and decrease it if the opposite is true (to encourage more entrants). The scalar ϵ_i is the minimum price which also ensures that the iteration does not get stuck at zero. We assume that if the price vector is frozen at some $p = [p_1, \cdots, p_K]$, the process of queue lengths is ergodic. Ignoring the boundary of the box $\mathcal{B} = \Pi_i[\epsilon_i, B_i]$, the limiting o.d.e. is

$$\dot{p}_i(t) = p_i(t)[f_i(p(t)) - y_i^*],\ 1 \leq i \leq K,$$

where $p(t) = [p_1(t), \cdots, p_K(t)]$ and $f_i(p)$ is the stationary average of the queue length in the $i-$th queue when the price vector is frozen at p. It is reasonable to assume that if the ϵ_i's are sufficiently low and the B_i's are sufficiently high, then $p(t)$ is eventually pushed inwards from the boundary of $\mathcal{B}$, so that we may ignore the boundary effects and the above is indeed the valid o.d.e. limit for the price adjustment mechanism. It is also reasonable to assume that $\frac{\partial f_i}{\partial p_j} > 0$, as increase in the price of one queue keeping all else constant will force its potential customers to other queues. Thus the foregoing applies. One can say more: Letting $f(\cdot) = [f_1(\cdot), \cdots, f_K(\cdot)]^T$, it follows by Sard's theorem that for almost all choices of y^*, the Jacobian matrix of f is nonsingular on the inverse image of y^*. Hence by the inverse function theorem, this set, which is also the set of equilibria for the above o.d.e. in $\mathcal{B}$, is discrete. Thus the o.d.e. converges to a point for almost all initial conditions. Hence the stationary distribution of $\{p(n)\}$ concentrates near the equilibria of the above o.d.e. Note that all equilibria in $\mathcal{B}$ are equivalent as far as our aim of keeping the vector of queue lengths near y^* is concerned. Thus we conclude that the dynamically adjusted prices asymptotically concentrate near one point which achieves the desired queue length profile, giving it the right 'pushes' if it deviates.

6 Replicator Dynamics

This is the o.d.e. from evolutionary biology [13] given by

$$\dot{x}_i(t) = x_i(t)[D_i(x(t)) - \sum_j x_j(t)D_j(x(t))], 1 \leq i \leq N.$$

The original interpretation is that $x_i(t)$ is the fraction of population of the $i-$th species (among N species) and $D_i(x(t))$ its payoff given the current composition of the population. The dynamics, which evolves on the simplex of probability vectors in $\mathcal{R}^N$, then says that the population of the $i-$th species increases, resp. decreases in proportion to the difference between its payoff and the average payoff of the population, depending on the sign of this difference. Two important cases when it converges are when $-D$ is 'monotone', i.e., $\langle x-y, D(x)-D(y)\rangle < 0$, and when $D_i(x) = \frac{\partial f}{\partial x_i}(x)$ for some f, which is called a potential function - see [18] for an application to congestion pricing. More generally, however, it can display nonconvergent behavior.

This has been applied to a network routing problem in [4], where $x(t)$ is in fact of the somewhat more general form $x(t) = [[x_{ij}(t)]]_{1\leq i\leq N, j\in J(i)}$, where N is the number of nodes and $J(i)$ is the set of neighbors of i. One has $x_{ij}(t) \geq 0, \sum_j x_{ij}(t) = 1$ for each i. Thus each vector $x_{i\cdot}(t)$ is *separately* a probability vector, indicating the probabilities with which $i-$th node directs its traffic to its neighbors. This leads to a coupled system of replicator dynamics. Other important features are as follows:

- The payoffs are negative of the mean delays associated with the routing decision. These are separately estimated by an averaging scheme on a faster timescale.
- The actual update omits the last component of each probability vector, setting it equal to one minus the sum of the rest. The updates are then projected to a subprobability simplex. This avoids too many projection operations.

The details may be found in [4].

7 Generalized Cohen-Grossberg Model

The o.d.e.

$$\dot{x}_i(t) = a_i(x(t))[b_i(x_i(t)) - \sum_j c_{ij} f_j(\sum_k c_{jk} g_k(x_k(t)))].$$

generalizes the Cohen-Grossberg model in neural networks [10]. Assume:

- $a_i(x), b_i(y), f_i(y), g_i(y), g_i'(y) > 0$ and are Lipschitz, where $g_i' = \frac{dg_i}{dy}$.
- $c_{ij} = c_{ji}$.

Then

$$V(x) = \sum_i \left(\int_0^{\sum_j c_{ij} g_j(x_j)} f_i(y)dy - \int_0^{x_i} b_i(y) g_i'(y)dy \right)$$

serves as a Liapunov function because $dV(x(t))/dt \leq 0$, as can be easily verified. This subsumes the Kelly-Maulloo-Tan model of dynamic pricing [14]. Another 'network' interpretation is described below: One can have a 'neighbourhood structure' by having $N(i) \overset{\Delta}{=}$ the set of neighbours of i, with the requirement that

$$c_{ij} = c_{ji} > 0 \Longleftrightarrow i \in N(j) \Longleftrightarrow j \in N(i).$$

One can allow $i \in N(i)$. Suppose:

- $j \in N(i)$ if j's transmission can be 'heard' by i.
- $x_j(t) =$ the traffic originating from j at time t.
- $\{c_{ij}\}$ capture the distance effects.
- $\sum_k c_{jk} g_k(x_k(t)) =$ the net traffic 'heard' by j at time t.
- Each node reports the volume of the net traffic heard by it to its neighbours and updates its own traffic so that the higher the net traffic it hears $\Longrightarrow$ it decreases its own flow correspondingly more.

An equilibrium of this o.d.e. will be characterized by

$$b_i(x_i) = \sum_j c_{ij} f_j \left(\sum_k c_{jk} g_k(x_k) \right) \ \forall i.$$

8 Other Models

Other interesting dynamics include:

- the 'fictitious play' from evolutionary game theory wherein each agent plays his best response to the adversary's empirical policy [8],
- the 'double bracket' equation [11],
- various models of flocking behavior [6].

These don't seem to have found applications in communications yet.

Acknowledgements. Work supported in part by a J. C. Bose Fellowship.

References

1. Bhorkar, A., Karandikar, A., Borkar, V. S.: Power Optimal Opportunistic Scheduling. IEEE GLOBECOM 2007, Washington DC (2007).
2. Borkar, V. S.: Convex Analytic Methods in Markov Decision Processes. in 'Markov Decision Processes: Models, Methods, Directions and Open Problems' (E. Feinberg and A. Shwartz, eds.). Kluwer Academic, Norwell, Mass. (2001), 347-375.
3. Borkar, V. S.: Stochastic Approximation: A Dynamic Viewpoint. Book in preparation.

4. Borkar, V. S., Kumar, P. R.: Dynamic Cesaro-Wardrop Equilibration in Networks. IEEE Trans. Automatic Control 48(3), (2003), 382-396.
5. Borkar, V. S., Manjunath, D.: Charge-based Control of Diffserve-like Queues. Automatica 40, (2004), 2040-2057.
6. Cucker, F., Smale, S.: Emergent Behavior in Flocks. Preprint available at $http : //www.tti - c.org/smale_papers/flock.pdf$ (2005).
7. Derevitskii, D. P., Fradkov, A. L.: Two Models for Analyzing the Dynamics of Adaptation Algorithms. Automation and Remote Control 35, (1974), 59-67.
8. Fudenberg, D., Levine, D.: Theory of Learning in Games. MIT Press, Cambridge, MA (1998).
9. Garg, D., Manjunath, D., Borkar, V. S.: Network Pricing for QoS: A 'Regulation' Approach. Advances in Control, Communication Networks, and Transportation Systems (E. H. Abed, ed.). Birkhauser, Boston (2005).
10. Haykin, S.: Neural Networks: A Comprehensive Foundation. 2nd ed. McMillan Publ. Co., New York (1999).
11. Helmke, U., Moore, J. B.: Optimization and Dynamical Systems. Springer Verlag, London (1994).
12. Hirsch, M. W.: Systems of Differential Equations That Are Competitive or Cooperative II: Convergence Almost Everywhere. SIAM Journal on Mathematical Analysis 16 (1985), 423-439.
13. Hofbauer, C., Siegmund, K.: Evolutionary Games and Population Dynamics. Cambridge Uni. Press, Cambridge, UK (1999).
14. Kelly, F. P., Maulloo, A., Tan, D.: Rate Control in Communication Networks: Shadow Prices, Proportional Fairness and Stability. Journal of Operational Research Society 49 (1998), 237-252.
15. Ljung, L.: Analysis of Recursive Stochastic Algorithms. IEEE Trans. on Automatic Control 22 (1977), 551-575.
16. Robbins, H., Monro, S.: A Stochastic Approximation Method. Annals of Mathematical Statistics. 22, (1951) 400-407.
17. Salodkar, N., Bhorkar, A., Karandikar, A., Borkar, V. S.: A Power Optimal Scheduling Algorithm on a Point to Point Wireles Link. Preprint.
18. Sandholm, W.: Evolutionary Implementation and Congestion Pricing. Review of Economic Studies. 69, (2002) 667-689.
19. Smith,H. L.: Monotone Dynamical Systems: An Introduction to the Theory of Competitive and Cooperative Systems. American Math. Society, Providence, RI (1995).

Optimisation-Based Overload Control

Marc Wennink, Phil Williams, Nigel Walker, and Ben Strulo

BT Group Chief Technology Office,
Adastral Park, Martlesham Heath, IP5 3RE, UK
{`marc.wennink,phil.m.williams,nigel.g.walker,ben.strulo`}`@bt.com`

Abstract. We discuss the design of an overload control protocol from an optimisation perspective. The design process starts with the formulation of a convex optimisation problem. We then construct a distributed algorithm for this problem by applying Lagrangian decomposition. The different components make use of models of their environment to ensure convergence of the algorithm. We show how the constructed algorithm is implemented by GOCAP, an overload control protocol currently undergoing standardisation.

Keywords: Overload Control, Convex Optimisation, Lagrangian Decomposition, Distributed Algorithms, Protocol Design.

1 Introduction

GOCAP (Generic Overload Control Application Protocol) is a protocol currently undergoing standardisation within the European Telecommunications Standards Institute (ETSI) [1]. Its purpose is to provide a general mechanism for protecting hosts and servers in Next Generation Networks against processing overload, and its design has been informed by previous experience of overload control in PSTN and Intelligent Network Services [2]. In this paper we motivate the design of GOCAP from the viewpoint of distributed optimisation. We present some of the key requirements as specified by the GOCAP design team and argue that these requirements can be associated with the formulation of a convex optimisation problem. We then apply Lagrangian decomposition techniques to construct a distributed algorithm for this problem. It turns out that GOCAP can be viewed as an implementation of this algorithm, which provides useful verification of its design.

The optimisation-centric approach to protocol design has received significant interest in recent literature [3]. This paper introduces two contributions to this research area. The first contribution is the *component graph*, a graphical representation of the Lagrangian associated with an optimisation problem. This representation is useful for laying out different decompositions and exposing the communication requirements for distributed algorithms based on such decompositions. The second contribution is an approach to designing the algorithm. Most algorithms proposed in the literature use variants of gradient search methods, which can have expensive communication requirements. In our approach, each agent participating in the algorithm is solving a local problem in a single step. The local problem changes as neighbouring agents solve their local problems and this can give rise to oscillations or instability. We show how this can be avoided by allowing the agents to construct a simple internal model of their environment.

2 The Overload Control Problem

2.1 Requirements

The processing capacity of servers in Next Generation Networks will be dimensioned against predictable peak load profiles. Occasionally, due to media stimulated events, disasters, or network failures, request rates may still exceed a server's capacity. In such a scenario, as the demand for processing capacity increases, there will be a point at which the server will start to reject some requests. As the rejection of requests itself requires processing effort, a further increase in the offered load may ultimately lead to a performance collapse, characterised by buffer overflows and unacceptable response times. Typically, the throughput as a function of the offered load is as depicted in Fig. 1.

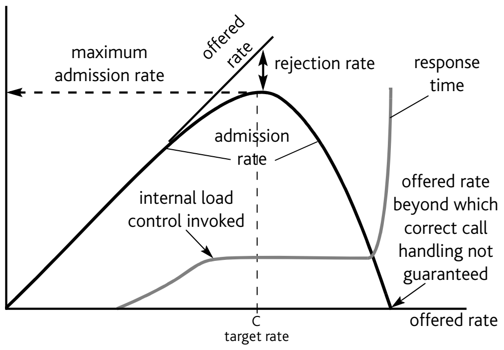


Fig. 1. Throughput and response time as a function of offered load (From [2])

A server will generally receive different types of requests from a variety of sources. The main ingredient of the overload control mechanism is a *rate restrictor* associated with each request type and each source, which is implemented at the source by means of a 'leaky bucket'. For each of the rate restrictors, the mechanism will determine the maximum rate at which requests can be offered to the server. The requirements for the control mechanism are outlined in the ETSI technical report [1], from which we extract the following: different request types can have different importance levels and should be prioritised accordingly; processing resources should be shared fairly between sources offering requests of the same type; throughput should be maximised.

The requirements state explicitly that the system should converge to an acceptable steady state when demands are constant. The first step in our process of designing an overload control protocol is to describe that steady state behaviour as the solution to an optimisation problem. Demand levels will appear as parameters in the problem formulation. In general, demand levels change over time and the optimal solution changes correspondingly. Our protocol will have to implement a distributed algorithm that converges quickly, so that the system can track this time-varying optimal solution sufficiently closely.

2.2 Optimisation Formulation

We use the term *stream* for any pair of location and request type for which a rate restrictor is introduced. Let $\mathcal{I}$ be the collection of streams. For each $i \in \mathcal{I}$, requests for stream i are generated at a demand rate d_i. We will control the rates x_i at which these requests are offered to the server. Since we want to maximise throughput, the aggregate rate $x_a = \sum_{i \in \mathcal{I}} x_i$ should not exceed the target level C (see Fig. 1). In order to differentiate between all possible rate allocations, we introduce the 'utility function'

$$U(\{x_i\}) = \sum_{i \in \mathcal{I}} u_i(x_i) = \sum_{i \in \mathcal{I}} \pi_i \log(x_i) \ . \tag{1}$$

The weights π_i, $i \in \mathcal{I}$, enable us to attach different levels of importance to different streams. If two streams have the same weight and experience the same demand, maximising the sum of the logarithmic utility functions ensures that both streams will receive equal allocations. Alternative allocation strategies, for example implementing guaranteed service rates, can be captured by adopting different utility functions.

For constant demand levels d_i and target rate C, the overload control mechanism should converge to offered rates x_i that solve the following problem:

$$\begin{aligned} \max \ & \sum_{i \in \mathcal{I}} \pi_i \log x_i \\ \text{s.t.} \ & \sum_{i \in \mathcal{I}} x_i = x_a \\ & x_a \leq C \\ & 0 \leq x_i \leq d_i \ , i \in \mathcal{I} \end{aligned} \tag{2}$$

Problem (2) is a relatively simple convex optimisation problem: Stefanov [4] gives an $\mathcal{O}(n^2)$ algorithm for a slightly more general version. The algorithms presented in the literature typically require knowledge of all the parameters in the formulation. In our scenario, however, the server can only measure the aggregate rate x_a and it cannot derive the values of d_i from observations of x_a. In Section 3 we will construct a distributed algorithm in which the sources and the server collectively find the appropriate rates.

2.3 The Lagrangian Component Graph

With problem (2) we can associate the Lagrangian

$$L(x; y) = \sum_{i \in \mathcal{I}} \pi_i \log x_i - y_a(\sum_{i \in \mathcal{I}} x_i - x_a) - y_s(x_a - C) - \sum_{i \in \mathcal{I}} y_i(x_i - d_i) \ , \tag{3}$$

where y_s and $x_i, y_i, i \in \mathcal{I}$ are restricted to be nonnegative. We are interested in finding the saddle point of L, solving

$$\max_x \min_y L(x; y) . \tag{4}$$

The Lagrangian L can be represented as a *component graph* as in Fig. 2, for a scenario in which $\mathcal{I} = \{1, 2, 3\}$. The primal variables x_a and $x_i, i \in \mathcal{I}$, are represented by circles, while the dual variables y_a, y_s, and $y_i, i \in \mathcal{I}$, are represented by squares. Individual components of the Lagrangian (3) are represented by 'blobs' connecting the variables that appear in them. Non-negativity conditions are represented by open blobs connected to the relevant variables.

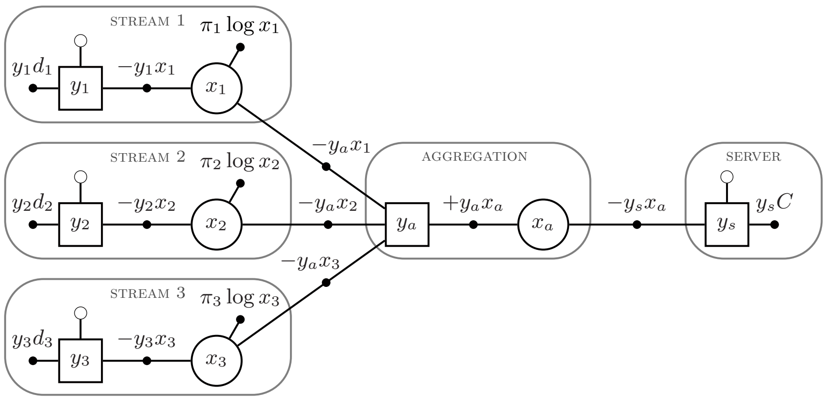


Fig. 2. Component graph representation of the Lagrangian L

L is a concave-convex function, and its saddle point is characterised by the Karush-Kuhn-Tucker (KKT) conditions (see, e.g., [5]). These conditions can be associated with the individual variables: each primal (dual) variable selects its value such as to maximise (minimise) the value of the Lagrangian for given values of all other variables. For each variable, only those components of the Lagrangian that contain that variable need to be considered (and can be read directly from the graph). For example, the condition for y_s requires that $L^{y_s}(y_s) = y_s C - y_s x_a$ is minimised over $y_s \geq 0$. If $x_a > C$, this problem is unbounded. Therefore, the condition translates into

$$x_a \leq C , \qquad y_s(C - x_a) = 0 . \tag{5}$$

3 Distributed Algorithm

3.1 Decomposition

The first step in our design process was to formulate the optimisation problem and the corresponding Lagrangian. In the second step, we create a number of sub-systems, each responsible for a number of variables. The choice of decomposition is often imposed on us by the physical distribution of the elements in a system. Some variables in the overload control problem are naturally 'owned' by the sources and others by the server. The decomposition can also be purely functional, with different functions implemented on the same host.

In Fig. 2, we have identified a partition of the set of variables into five subsets. With each such subset, we can now associate a *partial* Lagrangian. For example, the partial Lagrangian for the 'local' variables y_1 and x_1, i.e., for the sub-system of stream 1, is

$$L_1(x_1; y_1) = \pi_1 \log x_1 - y_1(x_1 - d_1) - y_a x_1 \ , \tag{6}$$

defined for nonnegative y_1.

Each partial Lagrangian is typically defined with respect to some given values of other, 'environmental' variables. For example, y_a appears as an environmental variable in the definition of L_1. In the component graph, environmental variables can be identified quite easily as those variables that are connected to at least one of the local variables.

The KKT condition associated with any variable z for the saddle point of the global Lagrangian L, is now represented by an identical KKT condition for the partial Lagrangian for which z is a local variable. Thus, if all partial Lagrangians are simultaneously in their saddle points, L itself is also in its saddle point.

The saddle point for the partial Lagrangian of stream i is

$$x_i^* = \min\{\frac{\pi_i}{y_a}, d_i\} \ , \quad y_i^* = \max\{0, \frac{\pi_i}{d_i} - y_a\} \ . \tag{7}$$

For the aggregation sub-system, it is

$$x_a^* = \sum_{i \in \mathcal{I}} x_i \ , \quad y_a^* = y_s \ , \tag{8}$$

and for the server sub-system, we find

$$y_s^* \begin{cases} = 0 & \text{if } x_a < C \\ \geq 0 & \text{if } x_a = C \\ = \infty & \text{if } x_a > C \end{cases} \ . \tag{9}$$

3.2 A Naive Algorithm

Given a decomposition, we now have to determine how each sub-system chooses the values of its local variables in response to changes in its environment. It is tempting to try out the following naive algorithm. Whenever a sub-system receives a new value of one of its environmental variables from a neighbouring sub-system, it determines the new saddle point for its local Lagrangian. It then sends the new values of its variables to its neighbours, who will update their local saddle points, and so on. If such an algorithm converges, it converges to the saddle point of the global Lagrangian.

The corresponding required communication channels stand out clearly as links that connect sub-systems in the component graph. In Fig. 2, we can identify four such channels: one between the aggregation component and each of the three streams, and one between the aggregation component and the server. Note that the channels are bi-directional. For example, stream 1 will communicate the value of x_1 to the aggregation component and, reciprocally, the aggregation component will send the value of y_a to (the source of) stream 1.

In [6] we showed how this approach applied to the network flow formulation of the shortest path problem resulted in the Bellman-Ford algorithm, which is at the heart of so called distance vector routing protocols. Unfortunately, in the decomposition of Fig. 2, convergence is not guaranteed. Assume that the server has just determined a new value of y_s. The aggregation sub-system will respond by setting $y_a = y_s$, and the sources will then set the offered rates according to (7): $x_i = \min\{\pi_i/y_a, d_i\} = \min\{\pi_i/y_s, d_i\}$. Thus, for any value of y_s set by the server, the rest of the system will respond by generating a unique total offered rate

$$x_a = x_a(y_s) = \sum_{i\in\mathcal{I}} x_i = \sum_{i\in\mathcal{I}} \min\{\frac{\pi_i}{y_s}, d_i\} \ . \tag{10}$$

But, observing a particular value of x_a, the server's response will be to set $y_s = 0$, if $x_a < C$, and $y_s = \infty$, if $x_a > C$. If $\sum_{i\in\mathcal{I}} d_i > C$, this leads to a constant flipping between $y_s = 0$ and $y_s = \infty$.

One approach that will remedy this problem is to adjust y_s more gradually. For example, we could use something similar to the dual algorithm proposed by Kelly et al. in [7] for the flow control problem. In a setting in which all sources adjust their rates instantaneously to new values of y_s, the dynamic system

$$\dot{y}_s = \mu(x_a(y_s) - C) \tag{11}$$

can be shown to converge to the value y_s^* at which the entire system will be in its saddle point, for any positive value of the update rate μ. If the adjustment process requires a significant amount of time, however, μ would have to be small and convergence would be slow. We will now discuss a scheme in which y_s can make 'big' adjustments while avoiding the type of oscillations identified above.

3.3 Modelling the Environment

One interpretation of the difficulty that leads to the oscillation in the naive algorithm is that the server, in choosing a new value for y_s, fails to adequately anticipate the effect, made explicit in (10), that this new value will have on the returned load, x_a. On the other hand, if the server had available all the parameters of the global optimisation problem, it could find the optimal value y_s^* in a single step, communicate this to the aggregation sub-system, let the sources limit the load they offer, and the problem would be solved in a single iteration. But, while the server knows the weights, π_i, it is considered infeasible for it to learn directly the demands, d_i. As a consequence, it can only construct an *approximate* model, but hopefully one that is good enough to make the algorithm converge. We adopt a model in which the server treats all the traffic as if it belongs to a single stream of the same form as the individual streams in Fig. 2. Since there is no need for explicit aggregation, it imagines it is participating in the Lagrangian

$$\tilde{L}(x_m; y_s, y_m) = \pi_m \log x_m - y_m(x_m - d_m) - y_s(x_m - C) \ , \tag{12}$$

which is a simpler version of (3), where d_m models the total demand, and x_m models the total offered rate. The server has to choose appropriate values for the parameters π_m and d_m. It does this by reconciling the model (12) with the actual aggregate load x_a that it has received in response to its current value of y_s. In the model, given a value for y_s, the value of x_m is given by $x_m = \min(\pi_m/y_s, d_m)$, which, as a model of a single stream, is equivalent to (7). For this to be compatible with the actual load we must have $x_m = x_a$. Then, depending on whether 1) π_m/y_s or 2) d_m is limiting, we obtain two possibilities for the parameters π_m and d_m:

$$1)\ \pi_m = x_a y_s,\ d_m > \frac{\pi_m}{y_s} = x_a \qquad 2)\ \pi_m > d_m y_s = x_a y_s,\ d_m = x_a \ . \tag{13}$$

The inequalities are resolved conservatively. In the first case the server assumes $d_m = \infty$. In the second case the server knows the utilities, $\pi_i \log(x_i)$, of the individual flows and can derive an upper bound, $y_s x_a < \sum_{i\in\mathcal{I}} \pi_i$, from (10), to use as a value for π_m. With these values (13) becomes

$$1)\ \pi_m = y_s x_a,\ d_m = \infty \qquad 2)\ \pi_m = \sum_{i\in\mathcal{I}} \pi_i,\ d_m = x_a \ . \tag{14}$$

Finally the server must decide whether to choose model 1 or model 2. It does this on the basis of an externally configured parameter ϵ. If $y_s > \epsilon$ it assumes the stream is being restricted and it chooses model 1. In the saddle point of (12), we then find $y_s' = y_s x_a/C$. If $y_s \le \epsilon$ the server assumes the stream is demand limited and it adopts model 2, and finds $y_s' = 0$ if $x_a < C$, and $y_s' = \sum_{i\in\mathcal{I}} \pi_i/C$ if $x_a \ge C$. The behaviour of all the sub-systems is summarised in Algorithm 1.

As an example, consider a server with a target rate $C = 100$, and three streams with priorities $\pi_1 = 1, \pi_2 = 5, \pi_3 = 10$, and demands $d_1 = 50, d_2 = 60, d_3 = 10$. The total offered rate x_a as a function of y_s, captured in (10), is depicted as the black curve in Fig. 3. The grey curve represents the combinations of y_s and x_a that satisfy the KKT condition (5) for y_s. Their intersection gives the desired operating point.

Initially, $y_s = 0$, and $x_a = \sum_{i\in\mathcal{I}} d_i = 120$. The server is overloaded and responds by setting y_s to a high value $\sum_{i\in\mathcal{I}} \pi_i/C$. The sources reduce the offered rates in response, ensuring $x_a < C$. In each of the subsequent iterations, y_s is reduced, maintaining $x_a < C$ and converging monotonically towards $x_a = C$ ([1], p34). A number of iterations for y_s have been depicted in Fig. 3. Open squares represent the state of the system that the server is aiming at when it is updating y_s, i.e., the saddle point of $\tilde{L}(x_m; y_s, y_m)$ as given in (12). Open circles represent the state of the system after the sources have responded to a change in the value of y_s.

Algorithm 1.

```
SERVER
    Initialisation: $y_s = 0$
    Update step: given current value $y_s$ and received total rate $x_a$, set
```

$$y'_s = \begin{cases} y_s x_a / C & \text{if } y_s > \epsilon \\ 0 & \text{if } y_s \le \epsilon \text{ and } x_a < C \\ \sum_{i \in \mathcal{I}} \pi_i / C & \text{if } y_s \le \epsilon \text{ and } x_a \ge C \end{cases} . \qquad (15)$$

```
    Communication: send $y'_s$ to AGGREGATION

AGGREGATION
    Initialisation: $y_a = 0, x_a = 0$
    Update step: given values of $x_i$ and $y_s$, set $y'_a = y_s$ and $x'_a = \sum_{i \in \mathcal{I}} x_i$.
    Communication: send $y'_a$ to all STREAMs and $x'_a$ to SERVER

STREAM $i$
    Initialisation: $y_i = 0, x_i = 0$
    Update step: given demand $d_i$ and received $y_a$, set
```

$$y'_i = \max\{\frac{\pi_i}{d_i} - y_a, 0\} , \quad x'_i = \min\{\frac{\pi_i}{y_a}, d_i\} . \qquad (16)$$

```
    Communication: send $x'_i$ to AGGREGATION
```

4 GOCAP

GOCAP has originally been designed to satisfy the requirements that led us to formulating problem (2). It is therefore not entirely unexpected that GOCAP can be shown to solve that problem. Still, it is quite striking to see how closely the protocol matches our distributed algorithm.

In the GOCAP architecture [1], three GOCAP-specific functional components have been specified, labelled M, R, and D. The monitoring and restriction mastering function (M), measures the offered load and uses that information to adapt the restriction level. Restrictors (R) restrict the demand at the source, on the basis of updates of the restriction level, which are distributed by the distribution function (D). The three components can be associated with the server, stream, and aggregation sub-systems in the component graph and our algorithm.

The monitoring and restriction mastering function in GOCAP maintains a control parameter X, which we can identify as being equal to $1/y_s$. It measures the total offered rate x_a and then uses the target rate C to adjust X to $X' = XC/x_a$, which is equivalent to the update rule used in (15) for $y_s > \epsilon$ (taking into account $X = 1/y_s$). GOCAP can be operating in either of two modes, "Closed-Loop Control" or "Monitoring", which can be associated with the case when $y_s > \epsilon$ or $y_s \le \epsilon$, respectively. If it is in the Closed-Loop mode and the offered rate is significantly less than the target rate C for a considerable period of time,

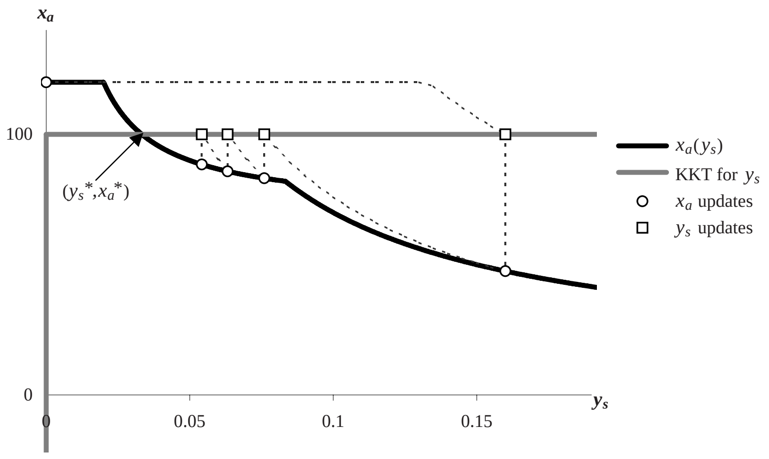


Fig. 3. A number of steps in the algorithm

GOCAP switches to the Monitoring mode. In our algorithm, y_s approaches 0 in the same scenario and the mode switch occurs when $y_s \leq \epsilon$.

The distribution function in GOCAP receives the value of the control parameter X and sends restriction levels $\pi_i X$ to the restrictors. Our algorithm assumes that the sources know the values of the π_i, in which case they only need the value of y_s $(= 1/X)$ to compute the restriction level. The distribution function is then performed by the aggregation sub-system sending the value of y_s to all sources. The effect of the aggregation component on primal variables is to compute $x_a = \sum_{i \in \mathcal{I}} x_i$ and send this to the server. In a deployment of GOCAP this operation is implemented in the data path of the application, and by the server measuring the received rate (or aggregate load).

The restrictor function in GOCAP is implemented by means of a leaky bucket with a variable maximum leak rate r. If traffic arrives at a constant rate d, the leaky bucket will let traffic through at a rate $x = \min\{d, r\}$. In Monitoring mode, the restrictors are switched off. In Closed-Loop mode, the leaky bucket rates are set to values $r_i = \pi_i X = \pi_i / y_a$, thus implementing the primal part of (16).

Since GOCAP's original conception, its functionality has been extended slightly. Streams now may have guaranteed service rates s_i and any spare processing capacity is allocated proportionally using weights π_i. The leak rates are thus set to $r_i = s_i + \pi_i / y_a$.

5 Conclusion

We have presented the design of an overload control protocol from an optimisation perspective, identifying three stages in the design process. In the first stage, a convex optimisation problem or, more generally, a convex-concave Lagrangian, is specified. In the second stage the corresponding saddle point problem is decomposed into a number of smaller problems. Finally, adaptation rules are specified, determining how each subsystem responds to changes in its environment. We have discussed how various design choices can be made at each stage. We highlighted the component graph representation of the Lagrangian as a tool to visualise decomposition options and to expose communication requirements associated with each decomposition.

The three design stages are not completely independent. Convexity of the optimisation problem specified in the first stage makes it possible to establish convergence of the adaptation rules described in the third stage. Convexity is not sufficient though, as the naive algorithm in Section 3 illustrated. To overcome this problem we have exposed a further aspect of the design space, whereby each subsystem can make use of a local model of its environment. This strategy applied to the overload control problem successfully reproduced the main aspects of GOCAP, which would have remained outside the scope of a more conventional optimisation decomposition approach.

We are currently studying generalisations of the overload control problem, where requests are routed through a network of servers. Here, complex interactions arise between the routing mechanism and different instantiations of the overload control protocol. More sophisticated models of the environment are required to obtain fast convergence in this setting. The convergence proof for GOCAP, given in [1], does not apply in the server network scenario, but we are exploring more powerful and more generic convergence proof techniques.

The optimisation problem at the heart of our design process only captures the steady state behaviour of the protocol. We are currently exploring ways in which problem formulations can be extended to also capture requirements and trade-offs for the transient behaviour.

References

1. ETSI TR 182 015: Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN); Next Generation Networks; Architecture for control of processing overload. Technical report, European Telecommunications Standards Institute (2006)
2. Whitehead, M., Williams, P.: Adaptive network overload controls. BT Technology Journal **20**(3) (2002)
3. Chiang, M., Low, S.H., Luo, Z., Shroff, N.B., Yu, W.: Nonlinear optimization of communication systems. IEEE Journal on Selected Areas in Communications **24**(8) (2006)
4. Stefanov, S.M.: An efficient method for minimizing a convex separable logarithmic function subject to a convex inequality constraint or linear equality constraint. Journal of Applied Mathematics and Decision Sciences **2006** (2006) Article ID 89307
5. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press (2004)
6. Walker, N., Wennink, M.: Interactions in transport networks. Electronic Notes in Theoretical Computer Science **141**(5) (2005) 97–114
7. Kelly, F., Maulloo, A., Tan, D.: Rate control in communication networks: shadow prices, proportional fairness and stability. Journal of the Operational Research Society **49** (1998) 237–252

Lyapunov Convergence for Lagrangian Models of Network Control

Ben Strulo, Nigel Walker, and Marc Wennink

BT Group Chief Technology Office,
Adastral Park, Ipswich, UK, IP5 3RE
{ben.strulo,nigel.g.walker,marc.wennink}@bt.com

Abstract. Many network control problems can be formulated and studied using the machinery of optimisation theory and Lagrange duality. The goal of the control process is to find the saddle point of the Lagrangian. We present a stability result for a class of dynamic processes for this problem. Our formulation automatically derives a Lyapunov function from the form of the dynamic equations. We show how several stability results from the literature of distributed flow control in networks fit into this formalism.

Keywords: Optimisation, Lagrange Duality, Flow Control.

1 Introduction

A strong and growing theme in the design and analysis of network control systems is to state the control objectives in terms of optimisation theory, and then to present the network dynamics (or protocol behaviour) as a process that seeks a solution of this optimisation problem [1]. Flow control algorithms, including TCP, have been extensively analysed through various utility maximisation problems, as has power control in mobile radio systems. Routing also yields to a similar treatment though typically expressed as cost minimisation rather than utility maximisation.

The optimisation approach has some very attractive features. It encourages a 'separation of concerns' between the statement of the network *objectives* and the *process* by which it achieves those objectives. It provides a notion of *decomposition*, whereby the system problem is broken down into sub-problems that must be solved by the local subsystems or agents. This in turn leads to distributed algorithms, and allows the design space of these algorithms to be more thoroughly and systematically explored. It also allows for *composition*. For example, it is possible to combine separate problems, such as routing and rate control, into a single joint problem. Optimisation theory also points to the importance of *convexity* as a structural property of problems that can be solved by effective distributed algorithms.

A central concern in this methodology is that the distributed algorithm or protocol should converge to the solution of the optimisation problem. Convergence, or *stability*, is also central from the viewpoint of control theory. Convergence criteria can appear quite different (and complicated) according to whether they assume continuous processes or discrete processes (either over time or over state-space), whether they allow for communication delay or capacity limits, whether they apply to linear or nonlinear systems, and whether they consider local or global stability. In the context of optimisation, convergence has been studied through gradient descent arguments, a passivity framework [2], local linear approximations, use of the frequency domain [3], and also, commonly, by concocting a Lyapunov function for the particular algorithm under discussion. Framing convergence criteria for solving optimisation problems, and rationalising and relating different approaches is an active area of research.

The purpose of this paper is to present a class of continuous (nonlinear) dynamic equations which we prove come equipped automatically with a Lyapunov function which ensures global convergence. The proof brings together the optimisation problem, the algorithm and a corresponding Lyapunov function into one combined package. This not only offers potential for labour saving over an approach in which these are motivated separately, but also offers useful insight and understanding of the convergence conditions. Another attractive feature is that it applies directly to a *Lagrangian* setting where equal status is accorded to primal and dual variables, and we can naturally derive joint primal-dual algorithms. We show how our result captures some of the models of dynamics presented in the literature on flow control in congested networks.

2 The Lagrangian Approach

Although network control problems are often first expressed as a constrained optimisation problem (utility maximisation or cost minimisation), it is common to then combine the objective and constraints through a Lagrangian formulation. The Lagrangian formulation also seems to help in decomposing a global joint optimisation problem into processes running locally in the network. We are seeking to give the Lagrangian function primary role, emphasising the equal status of primal and dual variables. Elsewhere we have developed a graphical syntax and a variety of idioms which exploit the structure of the Lagrangian to allow control processes to be clearly expressed [4]. These techniques then allow these formulations to be manipulated and explored until an efficient decomposition and associated dynamics can be chosen and refined towards implementation.

Our model of the control problem will be a general Lagrangian system, in which the global objective is to find a saddle point, by solving

$$\max_{\mathbf{x}} \min_{\mathbf{y}} L(\mathbf{x}, \mathbf{y})$$

over its convex domain $E \subseteq \mathbb{R}^n \times \mathbb{R}^m$. We assume L is strictly concave in its primal variable $\mathbf{x} \in \mathbb{R}^n$ and convex in its dual variable $\mathbf{y} \in \mathbb{R}^m$.

Typically the primal variables represent the controlled parameters of our system (for example, flow values or rates) while the dual variables arise as Lagrange multipliers associated with constraints, and are interpreted as, for example, prices (in flow control) or distance labels (in routing).

3 Dynamics and Lyapunov Convergence

We now define the trajectories that will take our system through its state space to its equilibrium. These are derived from two functions, $F(\mathbf{x}) \in \mathbb{R}$ and $G(\mathbf{y}) \in \mathbb{R}$ from which we will also derive the Lyapunov function to prove convergence. Our proof will rely fundamentally on the Legendre transform of these functions. Both F and G and their Legendre transforms are analogous to energy in electrical circuits, and as a convenience we will refer to all of these as "energy" functions.

We assume F and G are differentiable throughout E. We require some additional technical conditions to be sure that their Legendre transforms exist. Using definitions from Rockafeller [5] Theorem 26.5, we require them to be of *Legendre type*. We say a function F is of concave Legendre type if (writing $C = \operatorname{int}(\operatorname{dom} F)$ where $\operatorname{int}(S)$ denotes the interior of the set S and $\operatorname{dom}(F)$ the domain of the function F)

1. F is closed and strictly concave in C,
2. F is differentiable throughout C,
3. $\lim_{i\to\infty} |\nabla F(\mathbf{x}_i)| = \infty$ whenever $\mathbf{x}_i$ is a sequence in C converging to a boundary point of C.

These somewhat technical conditions ensure that the Legendre transform is well-defined, differentiable and strictly concave, and are usually straightforward; for example they are trivial for strictly concave/convex functions that are differentiable everywhere. For completeness, we say that G is of convex Legendre type if $-G$ is of concave Legendre type.

We now present our main result

Theorem 1. *Given a strictly concave convex Lagrangian $L : E \mapsto \mathbb{R}$ with its unique saddle point in its convex domain E and primal and dual energy functions F and G of concave and convex Legendre type respectively and with* $\operatorname{int}(\operatorname{dom} F) \times \operatorname{int}(\operatorname{dom} G) \supseteq E$*, then trajectories with*

$$-\frac{d}{dt}\nabla F \in \partial_x L \qquad\qquad -\frac{d}{dt}\nabla G \in \partial_y L \tag{1}$$

are globally asymptotically stable.

Here $\partial_x L$ is the sub-gradient of L with respect to primal variables only, $\partial_y L$ the sub-gradient with respect to dual variables only. Definitions of these sub-gradients are provided later in (4) and (5). Our proof of global asymptotic stability will be by a Lyapunov function method (see e.g. [6]).

Note that these conditions on the trajectories of the dynamical system do not necessarily imply that such trajectories exist or are unique. In fact, we will describe an example in which the trajectories are not uniquely defined by these conditions; instead any of a set of possible trajectories are stable. However, simple sufficient conditions for the trajectories to be uniquely defined can be straightforwardly found. For example, assume first that L is differentiable and

F, G are *twice* differentiable. Then $\nabla^2 F$ and $\nabla^2 G$ are negative or positive definite and so invertible and our trajectories are defined by

$$\dot{\mathbf{x}} = -(\nabla^2 F)^{-1}\nabla_x L \qquad \dot{\mathbf{y}} = -(\nabla^2 G)^{-1}\nabla_y L \tag{2}$$

which have unique solutions as long as $(\nabla^2 F)^{-1}\nabla_x L$ and $(\nabla^2 G)^{-1}\nabla_y L$ are Lipschitz.

3.1 Proof: Equilibrium and Zero Energy at Origin

We first prove convergence for the special case where the saddle point of the Lagrangian is at the origin, $(\mathbf{0},\mathbf{0})$. The saddle point condition states that

$$L(\mathbf{x},\mathbf{0}) \leq L(\mathbf{0},\mathbf{0}) \leq L(\mathbf{0},\mathbf{y}) \tag{3}$$

for all $(\mathbf{x},0)$ and $(0,\mathbf{y})$ in E. The Lagrangian is assumed to be concave-convex, so, from [5] (chapter 23), we can define the *subgradients*, $\partial_x L$ and $\partial_y L$, to be the maximal sets such that

$$\forall \mathbf{r} \in \partial_x L(\mathbf{x_1},\mathbf{y}_1) \quad L(\mathbf{x}_2,\mathbf{y}_1) - L(\mathbf{x}_1,\mathbf{y}_1) \leq (\mathbf{x}_2 - \mathbf{x}_1).\mathbf{r} \tag{4}$$

$$\forall \mathbf{s} \in \partial_y L(\mathbf{x}_1,\mathbf{y}_1) \quad L(\mathbf{x}_1,\mathbf{y}_2) - L(\mathbf{x}_1,\mathbf{y}_1) \geq (\mathbf{y}_2 - \mathbf{y}_1).\mathbf{s} \tag{5}$$

We assume that the energy function $F(\mathbf{x})$ is negative with maximum value 0 at $\mathbf{x} = \mathbf{0}$, and $G(\mathbf{y})$ is positive with minimum value 0 at $\mathbf{y} = \mathbf{0}$, so

$$F(\mathbf{0}) = 0 \qquad G(\mathbf{0}) = 0 \tag{6}$$

$$\nabla F(\mathbf{0}) = \mathbf{0} \qquad \nabla G(\mathbf{0}) = \mathbf{0} \tag{7}$$

$$F(\mathbf{x}) \leq 0 \qquad G(\mathbf{y}) \geq 0 \tag{8}$$

We will remove these restrictions in the next section so the following proof is essentially without loss of generality.

In constructing a Lyapunov function for (1) we will require the Legendre transforms of F and G, for which we follow a definition provided by [5] (Chapter 26). First note that ∇F and ∇G provide one-to-one mappings from $C = \operatorname{int}(\operatorname{dom} F)$ and $D = \operatorname{int}(\operatorname{dom} G)$ to their conjugate spaces $\overline{C} = \nabla F(C)$ and $\overline{D} = \nabla G(D)$. Introduce two auxiliary variables $\mathbf{p} \in \overline{C}$ and $\mathbf{q} \in \overline{D}$ such that

$$\mathbf{p}(\mathbf{x}) = \nabla F(\mathbf{x}) \qquad \mathbf{q}(\mathbf{y}) = \nabla G(\mathbf{y}) \tag{9}$$

$$\mathbf{x}(\mathbf{p}) = (\nabla F)^{-1}(\mathbf{p}) \qquad \mathbf{y}(\mathbf{q}) = (\nabla G)^{-1}(\mathbf{q}) \tag{10}$$

where $(\nabla F)^{-1}$ and $(\nabla G)^{-1}$ are the inverses of ∇F and ∇G respectively. Then the Legendre transforms, $\overline{F}$ and $\overline{G}$, of F and G are defined as

$$\overline{F}(\mathbf{p}) = \mathbf{x}(\mathbf{p}) \cdot \mathbf{p} - F(\mathbf{x}(\mathbf{p})) \qquad \overline{G}(\mathbf{q}) = \mathbf{y}(\mathbf{q}) \cdot \mathbf{q} - G(\mathbf{y}(\mathbf{q})) \tag{11}$$

We know from [5] that $\overline{F}(\mathbf{p})$ and $\overline{G}(\mathbf{q})$ are also of Legendre type. Moreover, from [5] Theorem 26.5

$$(\nabla F)^{-1} = \nabla\overline{F} \qquad (\nabla G)^{-1} = \nabla\overline{G} \tag{12}$$

Also from [5] is Theorem 26.4 which states that $\overline{F}$ is the concave conjugate of F and $\overline{G}$ the convex conjugate of G, and so

$$\overline{F}(\mathbf{p}) = \min_{\mathbf{x}}\big(\mathbf{x}\cdot\mathbf{p} - F(\mathbf{x})\big) \qquad\qquad \overline{G}(\mathbf{q}) = \max_{\mathbf{y}}\big(\mathbf{y}\cdot\mathbf{q} - G(\mathbf{y})\big) \tag{13}$$

The functions $\overline{F}$ and $\overline{G}$ obey

$$\overline{F}(\mathbf{0}) = 0 \qquad\qquad \overline{G}(\mathbf{0}) = 0 \tag{14}$$

$$\nabla\overline{F}(\mathbf{0}) = \mathbf{0} \qquad\qquad \nabla\overline{G}(\mathbf{0}) = \mathbf{0} \tag{15}$$

$$\overline{F}(\mathbf{p}) \le 0 \qquad\qquad \overline{G}(\mathbf{q}) \ge 0 \tag{16}$$

To show (15), invert (7) to give $\mathbf{0} = (\nabla F)^{-1}(\mathbf{0})$, then use (12). To show (14), substitute $\mathbf{p} = \mathbf{0}$ into (11), noting also that $\mathbf{x}(\mathbf{0}) = \nabla\overline{F}(\mathbf{0}) = \mathbf{0}$ from (15). To show (16), we have from (13) that $\overline{F}(\mathbf{p}) \le \mathbf{0}\cdot\mathbf{p} - F(\mathbf{0}) = 0$, and similarly for $\overline{G}$. We will also need, using (12) and (9)

$$\nabla\overline{F}(\mathbf{p}(\mathbf{x})) = \mathbf{x} \qquad\qquad \nabla\overline{G}(\mathbf{q}(\mathbf{y})) = \mathbf{y} \tag{17}$$

We are now ready to demonstrate the existence of a Lyapunov function, ϕ, defined as follows

$$\phi(\mathbf{x},\mathbf{y}) = \overline{G}(\mathbf{q}(\mathbf{y})) - \overline{F}(\mathbf{p}(\mathbf{x}))$$

Firstly $\phi(\mathbf{0},\mathbf{0}) = 0$ from (14). Also from (16), ϕ is positive and if $\phi(\mathbf{x},\mathbf{y}) = 0$ then $\overline{F}(\mathbf{p}) = 0$ hence $\mathbf{x} = 0$ and $\overline{G}(\mathbf{q}) = 0$ hence $\mathbf{y} = 0$. Finally we have

$$\begin{aligned}
\frac{d}{dt}\phi(\mathbf{x},\mathbf{y}) &= \frac{d}{dt}\overline{G}(\mathbf{q}(\mathbf{y})) - \frac{d}{dt}\overline{F}(\mathbf{p}(\mathbf{x})) \\
&= \nabla\overline{G}(\mathbf{q}(\mathbf{y}))\cdot\frac{d}{dt}(\mathbf{q}(\mathbf{y})) - \nabla\overline{F}(\mathbf{p}(\mathbf{x}))\cdot\frac{d}{dt}(\mathbf{p}(\mathbf{x})) && \text{by chain rule} \\
&= \nabla\overline{G}(\mathbf{q}(\mathbf{y}))\cdot\frac{d}{dt}(\nabla G(\mathbf{y})) - \nabla\overline{F}(\mathbf{p}(\mathbf{x}))\cdot\frac{d}{dt}(\nabla F(\mathbf{x})) && \text{by (9)} \\
&= \mathbf{y}\cdot\frac{d}{dt}(\nabla G(\mathbf{y})) - \mathbf{x}\cdot\frac{d}{dt}(\nabla F(\mathbf{x})) && \text{by (17)} \\
&= -\mathbf{y}\cdot\mathbf{s} + \mathbf{x}\cdot\mathbf{r} \\
&\qquad \text{for some } \mathbf{s}\in\partial_y L(\mathbf{x},\mathbf{y}) \text{ and } \mathbf{r}\in\partial_x L(\mathbf{x},\mathbf{y}) && \text{by (1)} \\
&\le \big(L(\mathbf{x},\mathbf{0}) - L(\mathbf{x},\mathbf{y})\big) - \big(L(\mathbf{0},\mathbf{y}) - L(\mathbf{x},\mathbf{y})\big) && \text{by (4), (5)} \\
&= L(\mathbf{x},\mathbf{0}) - L(\mathbf{0},\mathbf{y}) \\
&\le 0 && \text{by (3)}
\end{aligned}$$

Furthermore if $d\phi/dt$ is to be zero, then from (3), strictness of L in $\mathbf{x}$ implies $\mathbf{x} = \mathbf{0}$, and similarly for $\mathbf{y}$. So the proposed Lyapunov function reduces along trajectories satisfying (1) and is only zero at the equilibrium. This proves asymptotic stability.

To obtain *global* asymptotic stability, we need a little more; that the level sets of ϕ are bounded. From [5] Corollary 14.2.2, this is so if and only if $\mathbf{0} \in \operatorname{int}(\operatorname{dom}\overline{\phi})$, for $\overline{\phi}$ the convex conjugate of ϕ. But, we know that $\mathbf{0}$ is in E and hence in $C \times D$ and so in the interior of $\operatorname{dom} F \times \operatorname{dom} G$. This is the domain of $\overline{\phi}$, giving $\mathbf{0} \in \operatorname{int}(\operatorname{dom}\overline{\phi})$ as required.

3.2 Proof: General Equilibrium Point

To prove stability about a general point we can shift the origin of the energy functions and prove that these shifted functions define a Lyapunov function of the original trajectories. Thus L in Theorem 1 is assumed to be strictly convex concave, but with arbitrary saddle point. The energy functions F and G are assumed concave and convex respectively, but without the restrictions on the location of their stationary points, or their value at the stationary points.

We introduce a new Lagrangian L', and new energy functions F' and G' defined as follows

$$L'(\mathbf{x},\mathbf{y}) = L(\mathbf{x}+\mathbf{x}^*,\mathbf{y}+\mathbf{y}^*) \tag{18}$$
$$F'(\mathbf{x}) = F(\mathbf{x}+\mathbf{x}^*) - F(\mathbf{x}^*) - \nabla F(\mathbf{x}^*)\cdot\mathbf{x} \tag{19}$$
$$G'(\mathbf{y}) = G(\mathbf{y}+\mathbf{y}^*) - G(\mathbf{y}^*) - \nabla G(\mathbf{y}^*)\cdot\mathbf{y} \tag{20}$$

where $(\mathbf{x}^*,\mathbf{y}^*)$ is, for the moment, some arbitrary reference point. We require F and G to be differentiable at $(\mathbf{x}^*,\mathbf{y}^*)$ but this is automatic if $(\mathbf{x}^*,\mathbf{y}^*)\in E$. Now,

$$\frac{d}{dt}\big(\nabla F'(\mathbf{x})\big) = \frac{d}{dt}\big(\nabla F(\mathbf{x}+\mathbf{x}^*) - \nabla F(\mathbf{x}^*)\big) = \frac{d}{dt}\nabla F(\mathbf{x}+\mathbf{x}^*)$$
$$\partial_x L'(\mathbf{x}) = \partial_x L(\mathbf{x}+\mathbf{x}^*,\mathbf{y}+\mathbf{y}^*)$$

and so the conditions on the trajectories (1) are invariant with respect to this type of transformation.

The functions F' and G' fulfill the energy function requirements (14)-(15) since they are respectively concave and convex, and

$$F'(\mathbf{0}) = F(\mathbf{x}^*) - F(\mathbf{x}^*) = 0, \qquad \nabla F'(\mathbf{0}) = \nabla F(\mathbf{x}^*) - \nabla F(\mathbf{x}^*) = \mathbf{0} \tag{21}$$

and similarly for G'.

Finally, by choosing $(\mathbf{x}^*,\mathbf{y}^*)$ to be the saddle point of L, the new Lagrangian L' satisfies (3)-(5), so all of the conditions of the previous section are met, and we know that a Lyapunov function exists that ensures the trajectories converge to the saddle point.

4 Applications to Network Control

We start by presenting a standard formulation of the network congestion control problem that has been widely studied in the literature (for example [7]). We will then show how our techniques can reproduce several algorithms that have been proposed for solving this problem.

Consider a set of n users who are allocated rates x_i by the network subject to capacity constraints K_j at each of m resources. This leads to a Lagrangian formulation

$$L(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{n} U_i(x_i) + \sum_{j=1}^{m} y_j\big(K_j - \sum_{l\in j} x_l\big) \tag{22}$$

with the constraint that $x, y \geq 0$. Here we abuse notation and write $l \in j$ to mean values of $l \leq n$ such that user l is using resource j. The strictly concave functions U_i represent the utilities for rates x_i that the n users of the system receive. The dual variables y_j are the Lagrange multipliers associated with the capacity constraints in the optimisation problem.

Note that this Lagrangian is convex but not *strictly* convex in its dual variables. This leads to problems in the dynamics; typically it is not possible to control the system to avoid overload (congestion). Thus it is conventional to add a barrier function to convexify the Lagrangian. This can be considered to be a way of improving the network control. It can also be considered to represent some real costs that arise as the network approaches capacity (e.g. delay or packet loss).

With this modification the Lagrangian becomes:

$$L(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} U_i(x_i) + \sum_{j=1}^{m} \Big(y_j \big(K_j - \sum_{l \in j} x_l\big) - \pi_j(y_j) \Big) \tag{23}$$

where the functions $\pi_j(y_j)$ are concave and positive. This Lagrangian can be made as close as we require to the original one by suitable choice of the barrier functions, for example by using $\pi_j(y) = \epsilon \log(y)$ and letting ϵ tend to 0. However, it may be more realistic for π_j to remain non-negligible. Then it moves the equilibrium away from overload until the costs of near-congestion are balanced by the loss of utility from the reduced rate.

An optimisation problem capturing the tradeoff between maximising utility while minimising the cost of the congestion can be recovered from (23) by performing the minimisation over y_j given by

$$\min_{y_j} \sum_{j=1}^{m} \big(y_j (K_j - X_j) - \pi_j(y_j) \big), \quad X_j = \sum_{l \in j} x_l \tag{24}$$

This is the Legendre transform of $\pi_j^K(y_j) = K_j y_j - \pi_j(y_j)$, with $X_j = \sum_{l \in j} x_l$ the dual variable of y_j. Writing this transform as $\overline{\pi}_j^K(X_j)$, the desired optimisation problem becomes

$$\max_{x_j} \sum_{i=1}^{n} U_i(x_i) + \sum_{j=1}^{m} \overline{\pi}_j^K \Big(\sum_{l \in j} x_l \Big) \tag{25}$$

This compares directly with the formulation in [7] (where $\overline{\pi}_j^K(X) = \int_0^X p_j(y) dy$).

The solution of the above optimisation and Lagrangian problems prescribe the required static configuration of the network. To find a distributed algorithm that the network can use to find this solution, various dynamic equations have been formulated over this same Lagrangian. We now demonstrate how some of these can be represented in our approach by choosing appropriate energy functions for both the primal and dual variables.

Example 1. For our first example we cast the dynamics discussed in [7] into the form prescribed in (1). There Kelly chooses $U_i(x_i) = w_i \log(x_i)$. To obtain his dynamics we need to choose

$$F(\mathbf{x}) = -\frac{1}{\kappa}\sum_{i=1}^{n}\log(x_i) \qquad\qquad G(\mathbf{y}) = \frac{1}{2\nu}\sum_{j=1}^{m} y_j^2 \tag{26}$$

Then (1) applied to the Lagrangian (23) gives (since L is differentiable)

$$\frac{-1}{\kappa x_i}\dot{x}_i = -w_i/x_i + \sum_{k\in i} y_k \qquad \Rightarrow \qquad \dot{x}_i = \kappa\Big(w_i - x_i\sum_{k\in i} y_k\Big) \tag{27}$$

As before we abuse notation here and write $k \in i$ for values of $k \leq m$ such that resource k is used by user i. This is the primal dynamics specified in [7] (equation 5).

For the dual dynamics we obtain

$$\frac{1}{\nu}\dot{y}_j = -\big(K_j - \sum_{l\in j} x_l - \pi_j'(y_j)\big) \quad \Rightarrow \quad \dot{y}_j = \nu\Big(\sum_{l\in j} x_l - \big(K_j - \pi_j'(y_j)\big)\Big) \tag{28}$$

which is equivalent to the dual dynamics in [7] (equation 9), where they write $q_j(y_j)$ for our $K_j - \pi_j'(y_j)$.

Having obtained dynamic equations (27) and (28) as instances of (1), we know automatically from Theorem 1 that a joint primal-dual algorithm with these dynamics is Lyapunov stable. In this respect we have a generalisation of [7]. To recover the primal-only and dual-only algorithms of [7] we use the fact that the system remains stable for arbitrary values of the ratio $\kappa : \nu$. In particular, in the limit as either the primal or dual dynamics become much faster than the other, we can consider the fast part of the system to be always at equilibrium.

Accordingly, to obtain the primal dynamics we consider the equilibrium of (28). Again using $X_j = \sum_{l\in j} x_l$, the equilibrium condition can be written as

$$X_j = \frac{\partial}{\partial y_j}\big(K_j y_j - \pi_j(y_j)\big) = \frac{\partial}{\partial y_j}\pi_j^K(y_j) \quad \Rightarrow \quad y_j = \frac{\partial}{\partial X_j}\overline{\pi}_j^K(X_j) \tag{29}$$

using the properties of the Legendre transform (as also used in (12)) to invert. In the notation of [7] this becomes $y_j = p_j(\sum_{l\in j} x_l)$ as required.

Alternatively, to obtain only dual dynamics we use the equilibrium condition of (27) to give $x_i = w_i/\sum_{j\in i} y_j$ which is equation 10 in [7], also as required.

Note here how our general approach automatically implies the stability of the primal-dual, primal-only, or dual-only algorithms, directly from the dynamic equations, all without the need to find separately motivated Lyapunov functions.

Example 2. Our second example considers dynamics presented in [8]. Here we retain the general concave utility function in the Lagrangian (23), and define energy functions as follows

$$F(\mathbf{x}) = \sum_{i=1}^{n}(-1/\kappa_i)\int_0^{x_i} U_i(x)\,dx \qquad\qquad G(\mathbf{y}) = \sum_{j=1}^{m}\frac{1}{2\nu_j}y_j^2 \tag{30}$$

Then we obtain dynamics

$$\dot{x_i} = \kappa_i\big(1 - \frac{1}{U_i'(x_i)}\sum_{k\in i} y_k\big) \quad \dot{y_j} = \nu_j\big(K_j - \sum_{l\in j} x_l - \pi_j'(y_j)\big) \tag{31}$$

If the U_i are strictly increasing functions (a reasonable assumption for flow utility) then F is concave, and our result demonstrates convergence of a combined primal-dual algorithm. To recover the primal-only dynamics in [8] we assume the dual variables (or shadow prices) reach their equilibrium quickly, and substitute (29) into (31). The dual-only dynamics in [8] discards the barrier functions $\pi_j(y_j)$, and assumes the flows instantaneously reach an equilibrium with respect to the dual variable.

Conditional on the assumption that the energy function $F(\mathbf{x})$ is built from the utility function according to (30) it is possible to reverse the above derivation: to start with a flow dynamics of the form in (31) and to infer the corresponding utility function. This line of argument has been used to associate utility functions with different versions of TCP flow control. For example, under those assumptions, the flow dynamics for TCP-Reno corresponds to a utility function $U(x) = (\sqrt{2}/\tau)\arctan(x\tau/\sqrt{2})$ [8].

Example 3. Finally we can apply our approach to the formulation in [9], which uses the (non-strict) Lagrangian in (22). We choose strictly concave (resp. convex) twice differentiable energy functions

$$F(\mathbf{x}) = \sum_{i=1}^{n} f_i(x_i) \qquad G(\mathbf{y}) = \sum_{j=1}^{m} g_j(y_j) \tag{32}$$

which, for $x_i > 0$, $y_j > 0$, give dynamics as in [9]:

$$\dot{x_i} = \frac{-1}{f_i''(x_i)}\big(U_i'(x_i) - \sum_{k\in i} y_k\big) \qquad \dot{y_j} = \frac{1}{g_j''(y_j)}\big(\sum_{l\in j} x_l - K_j\big) \tag{33}$$

However, under the assumption that the U_i (and hence also the Lagrangian) are differentiable at $x_i = 0$, the dynamics are extended in [9] onto the boundary $x_i = 0$ and $y_i = 0$ by restricting the terms on the right hand sides of (33) to be non-negative there.

These potentially awkward cases are easily handled by the use of the subgradients in Theorem 1. For $x_i = 0$ the primal sub-gradient is defined as

$$\mathbf{r} \in \partial_x L(\mathbf{x},\mathbf{y}) \Rightarrow r_i \geq \frac{\partial L(\mathbf{x},\mathbf{y})}{\partial x_i}\Big|_{x_i=0} = U_i'(0) - \sum_{k\in i} y_k \tag{34}$$

This inequality is straightforwardly satisfied by the dynamics in [9], which can be interpreted as choosing $r_i = U_i'(0) - \sum_{k\in i} y_k$ if this is greater than 0 and $r_i = 0$ otherwise. A similar argument holds for y_j.

In this example stability is not immediate because $L(\mathbf{x},\mathbf{y})$ is not strict in $\mathbf{y}$. However, by LaSalle's theorem [6] the Lyapunov function ϕ ensures convergence to a set in which $\dot{\phi} = 0$ and, as before, strictness in $\mathbf{x}$ requires $\mathbf{x}$ to take its equilibrium value there. The *invariant* set for which $\dot{\phi} = 0$ then requires $\dot{x}_i = 0$. If solving (33) for $\dot{x}_i = 0$ gives unique solutions for y_j, then stability is proven. This is assured by the full rank condition assumed in [9].

5 Conclusion

We have identified a general approach to specifying the dynamics of a distributed optimisation which comes with an implicit convergence result. This form offers quite a wide design space, within which further results, intuition (and optimisations) could be developed. In particular, it allows primal and dual algorithms to be combined easily, and boundary conditions to be handled straightforwardly, extending other results in the literature.

Our assumptions concerning F, G and L lead to a particularly direct and interesting stability proof, in which the transformation (19) and (20) is key to reusing the concave and convex functions appearing in the dynamic equations to construct the Lyapunov function. Analogous results might be similarly derived for non-differentiable concave-convex functions F, G, and for processes with discrete time steps. It also appears that extensions are possible in which the strictness of L can be relaxed in some of its variables.

References

1. Chiang, M., Low, S.H., Luo, Z., Shroff, N.B., Yu, W.: Nonlinear optimization of communication systems. IEEE Journal on Selected Areas in Communications **24**(8) (August 2006)
2. Wen, J.T., Arcak, M.: A unifying passivity framework for network flow control. IEEE Transactions on automatic control **49**(2) (February 2004) 162–174
3. Vinnicombe, G.: On the stability of networks operating TCP-like congestion control. In: Proc. IFAC. (2002)
4. Walker, N., Wennink, M.: Interactions in transport networks. Electronic Notes in Theoretical Computer Science **141**(5) (December 2005) 97–114
5. Rockafellar, R.T.: Convex Analysis. Princeton University Press (1970)
6. Khalil, H.: Nonlinear Systems. 3rd edn. Prentice Hall (2000)
7. Kelly, F., Maulloo, A., Tan, D.: Rate control in communication networks: shadow prices, proportional fairness and stability. In: Journal of the Operational Research Society. Volume 49. (1998)
8. Low, S., Srikant, R.: A mathematical framework for designing a low-loss low-delay internet. Network and Spatial Economics **4**(1) (March 2004) 75–102
9. Liu, S., Basar, T., Srikant, R.: Controlling the internet: A survey and some new results. In: Proceedings of 42nd IEEE Conference on Decision and Control. Volume 3. (Dec 2003) 3048–3057

NCRS: A Network RAM-Based Computational Resource Sharing Grid*

Yiming Zhang, Dongsheng Li, Rui Chu, and Xicheng Lu

School of Computer, National University of Defense Technology,
Changsha 410073, Hunan, China
Phone: +86-731-4575816
ymzhang@nudt.edu.cn

Abstract. In this paper we propose NCRS, a Network Computational Resource Sharing grid based on network RAM. In NCRS, the computing node regards the free memory of other nodes in networks as a complement of local memory and uses it to store the large amount of intermediate data during computation. When the remote data is required, rather than blocks and gets it from remote nodes (or local disks), the computing node sends the related instructions to the remote nodes where the data locates. We refer to the memory on the remote nodes in networks as Network Intelligent Memory (NIM). NIM carries out the received instructions, and by this means the computing node reduces the page-swaps with local disks and the instructions locally carried out.

Keywords: NCRS; NIM; Pseudo memory; Speculative execution; Instruction analyses.

1 Introduction

In order to exploit the vast wide-area distributed memory resources, we introduced RAM-Grid [1] in our previous work, which provides a universal memory service for other nodes on the Internet. RAM-Grid swaps obsolete local memory pages to remote memory service instead of local disk, which may lead a performance boost for memory-intensive applications when local physical memory is inadequate.

However, further experiments were made and results showed that RAM-Grid could not adapt to computational-intensive applications with very frequent page-swaps [1]. We analyze the results and find that RAM-Grid and other proposals for memory sharing [2,3] can **ONLY** reduce the overhead of each page-fault during the process of computation and can **NOT** reduce the total number of page-faults at all.

By extending the concept of RAM-Grid, in this paper we present NCRS, a Network Computational Resource Sharing grid. As in RAM-Grid, in NCRS the computing node regards the free memory of other nodes in networks as a complement of local memory and uses it to store the large amount of intermediate data during

* The work was partially supported by the National Natural Science Foundation of China under Grant No. 60673167 and No. 90412011, and the National Basic Research Program of China (973) under Grant No. 2005CB321801.

computation. However, in NCRS when page-fault occurs, rather than blocks and gets pages back from remote nodes as in RAM-Grid, the computing node sends the related instructions (r-instructions for short) to the remote nodes where the data locates. We refer to the memory on the remote nodes in networks as Network Intelligent Memory (NIM). NIM carries out the r-instructions for the computing node and returns the results, while the local computing node can jump over them and carries out the following instructions simultaneously. Like local page-swap schemes, all pages swapped to NIM also have a local-disk copy. When page-fault occurs, the local copy will be fetched from disk to memory simultaneously with remote execution, and the r-instructions will be executed locally if the remote result doesn't come back yet when the local copy has swapped-in to local memory. By this means we ensure that at worst NCRS has the same performance as local page-swap schemes for computational-intensive nodes.

NCRS reduces the overhead of each page-fault as in RAM-Grid, and the number of page-faults as well as the number of instructions carried out locally.

2 Network Intelligent Memory

We extend the concept of RAM-Grid by regarding the free memory in other nodes on the Internet as fast memory resources with intelligence (execution ability), and then the network memory has the characteristics as follows:

(1) The network memory in other nodes has the computing ability and can carry out instructions. For example, a local instruction such as "add eax, ebx" can add the data of eax register to that of the ebx register and save the result in eax register. If we can send an instruction similar to that of "add memA, memB" (both memA and memB are the memory addresses of the remote node) to the idle nodes where the data locates, then the idle node can add the data of memA to that of memB automatically.

(2) The speed of network memory is close to that of the local memory, while the total capacity is close to infinity. The usual latency of network ranges from hundreds of ms to several seconds [4], and if we select memory services carefully to control the latency less than one ms, then we can improve the system performance greatly.

We call these network memory resources Network Intelligent Memory (NIM), which are provided by the idle nodes on the Internet and can partly replace the local memory. Therefore, computing nodes can not only regard NIM as a complement to local physical memory and store the initial and intermediate data in it, but also send the related instructions to NIM. NIM carries out these instructions instead of the computing node and returns the results, while the local computing node can jump over these instructions and carries out the following instructions simultaneously until meets instructions that need the results of the remote execution. Compared with RAM-Grid, NIM reduces the number of the instructions carried out at the local computing node and the number of page-faults in the computational-intensive applications.

The physical memory is not hot-pluggable in today's computer architectures, and there are no operating systems supporting dynamic changes of memory capacity. However, the total capacity of NIM that can be used by the computing node is changeable. To solve this problem, we propose a new policy named Pseudo Memory Policy (PMP) for memory management in NCRS.

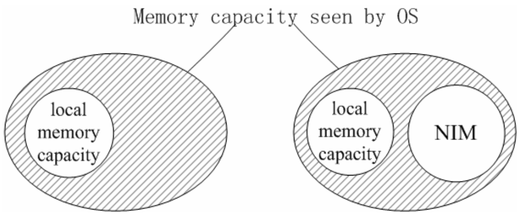


Fig. 1. Pseudo Memory Policy

As shown in figure 1 (a), during start-up of the OS, PSP let the OS believe that the computing node has huge memory capacity (closed down by the ellipse) with only "local memory capacity" free, and all other part in the ellipse is used. When using NIM, as shown in figure 1 (b), PSP inform the OS that some memory covered by "NIM" circle are freed and there are "local memory capacity" + "NIM" of memory that can be used by the OS, while the total memory capacity seen by the OS remains the same (closed down by the ellipse). As shown in figure 2, by this means PSP "cheats" the OS of the computing node and realizes the universal image of local memory and network memory.

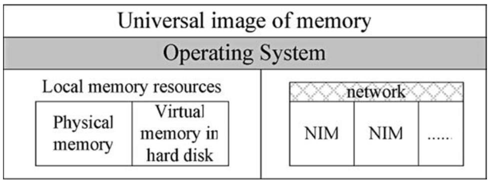


Fig. 2. Image of local and network memory

Because of the relativity and complexity of instructions, the computing node can't send all instructions to NIM for execution. It is complicated to decide which to send, when to send, and what to do after sending these instructions.

The key problem of instruction analyses is to analyze the operation characteristics of the instructions, the location of the operands and the correlation between the instructions being analyzed and the next ones. On one hand, because of the uncertainty of data location and program behavior, it's impossible to completely decide the location of the operands and the relationship of the instructions during compile phase; on the other hand, for efficiency it makes no sense to decide whether to carry out at NIM (where the data locates) completely at runtime. Therefore, considering both accuracy and efficiency of the decision, we need to combine compiling analyses and executing analyses as follows.

(1) The computing node maintains a pattern table to record the remote-executable instruction patterns. We discuss these patterns in the next sub-section.

(2) During the compile phase, the computing node matches patterns with the instructions and analyzes which instructions may be sent to NIM.

(3) When page-faults occurs at runtime, if the data locates in local page-swap area in disk the computing node swaps in the corresponding pages as usual, else the data must locates in one or more NIMs providing page services [1] for it. It checks whether the following instructions at the page-fault point are in the pattern matching records. Turn to step (6) if the matching fails.

(4) The computing node decides whether to send the matched n instructions to NIM according to the following conditions.

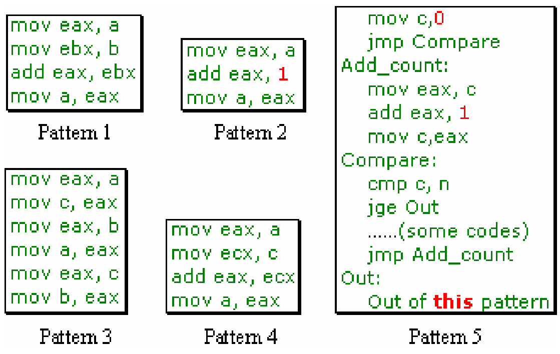
- The location of all data corresponding to the matched n instructions.
- Whether the instructions next to the matched n instructions are correlated with the execution result of the n ones.

Turn to step (6) if the computing node decides not to send.

(5) The computing node sends out the n instructions to the NIM where the data locates and carries out the instructions next to the n ones until meeting the instruction that needs the execution result at NIM.

(6) The computing node waits until the needed data is fetched back from the NIM where the data locates.

The remote-executable instruction patterns are important to NCRS. Currently we analyze the most common patterns as shown in figure 3 and we will study and define more complicated patterns for NCRS in our future work.



```
mov eax, a
mov ebx, b
add eax, ebx
mov a, eax
```
Pattern 1

```
mov eax, a
add eax, 1
mov a, eax
```
Pattern 2

```
mov eax, a
mov c, eax
mov eax, b
mov a, eax
mov eax, c
mov b, eax
```
Pattern 3

```
mov eax, a
mov ecx, c
add eax, ecx
mov a, eax
```
Pattern 4

```
    mov c,0
    jmp Compare
Add_count:
    mov eax, c
    add eax, 1
    mov c,eax
Compare:
    cmp c, n
    jge Out
    ......(some codes)
    jmp Add_count
Out:
    Out of this pattern
```
Pattern 5

Fig. 3. Remote-executable instruction patterns

Pattern 1: if the purpose of the instructions is to add one value at address *b* to another at address *a*, and both values and the result are stored at the same NIM (remote node), then the computing node may send the instructions to the NIM.

Pattern 2: if the purpose of the instructions is to add a constant to the value at address *a*, and both the value at address *a* and the result are stored at the same NIM (remote node), then the computing node may send the instructions and the constant to the NIM.

Pattern 3: if the purpose of the instructions is to exchange one value at address *b* and another at address *a*, and both values are stored at the same NIM (remote node), then the computing node may send the instructions to the NIM.

Pattern 4: if the purpose of the instructions is to add one value at address *c* to another at address *a*, and the value at address *a* and the result are stored at the same NIM (remote node), the value at address *c* is at the computing node (local node), then the computing node may send the instructions and the value at address *c* to the NIM.

Pattern 5: if the instructions are a loop and the correlated operands in the loop are at the same NIM, then the computing node may send the instructions to the NIM.

NIM carries out the r-instructions for the computing node and returns the results, while the local computing node can jump over them and carries out the following instructions speculatively. The speculative execution relies on two facts. First, programs can correctly predict the result of many operations such as loop test and consistency checks. Second, the latency of checkpoints is much less than network RTT to NIM, so substantial work can be done while waiting for the remote-execute results.

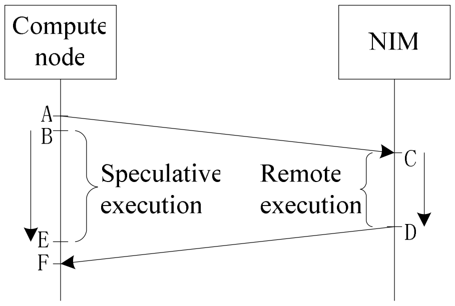


Fig. 4. Speculative execution

As shown in figure 4, if the following instructions need the results of the remote execution, the local computing node predicts the result of the r-instructions, checkpoints the state of current process (from time A to time B) and speculatively continue the execution of the following instructions based on the predicted result (from time B to time E).

The speculative execution blocks (from time E to time F) if it will influence others, such as invoking other processes or interacting with human users, until the speculations prove to be correct (time F).

When the result returns (time F), if the prediction of the remote execution results is correct, the checkpoint is discarded; otherwise the process state is restored to the checkpoint and the following instructions after the r-instructions are re-executed.

Like local page-swap schemes, all pages swapped to NIM also have a local-disk copy. When page-fault occurs, the local copy will be fetched from disk to memory simultaneously with remote execution, and the r-instructions will be executed locally if the remote result doesn't come back yet when the local copy has swapped-in to local memory. By this means we ensure that at worst NCRS has the same performance as local page-swap schemes for computational-intensive nodes.

Detailed speculative execution implementation is one of our future work.

3 Analysis and Evaluation

The ultimate goal of NCRS is to construct a computational resource sharing system. However, because the network latency and bandwidth vary greatly, NCRS works better than local execution only under some conditions. In the following analyses we assume the memory demand of computation is much higher than the local memory capacity. Note that in this section we omit the speculative execution introduced in section 2.3 due to lack of methods to trace the speculation accuracy, which will be studied in our future work.

In traditional local computing environments, when the computing node lacks memory greatly, a mass of page swap operations occur and take much more time than execution of instructions, so we can ignore the impact of the number of instructions actually executed. When performing a page swap operation with n successive pages, the latency is given by formula (1):

$$Latency_D = T_s + T_L + (n-1)\times T_W + n\times\frac{S_p}{B_d} \tag{1}$$

In formula (1), T_S denotes the average seek time, T_L denotes the average latency time, T_W is the average waiting time between two successive readings, S_P is the page size, and B_d is the average disk bandwidth.

In NCRS environments, there are two kinds of latency. If the local instructions following the remote-executing ones don't need the remote-executing result, the latency is the time to send instructions and could be ignored compared to IO latency. Otherwise the local instructions have to wait the remote-executing result and the latency is given by formula (2):

$$Latency_N = 2\times T_U + T_{RTT} + T_{remote-exec} + S_{result} / B_N \tag{2}$$

In formula (2), T_u is the start-up time, T_{RTT} is the round trip time, $T_{remote\text{-}exec}$ is the remote execution time, S_{result} is the remote result size, and B_N is the network bandwidth.

We trace an actual meteorological application [8] and modify the Linux kernel 2.4 to record its page swap history, including swap time (in microseconds), swap type (swap in or swap out), and the page location in local disk. During the monitoring, the standard setting is a PC with Intel Pentium 2.4GHz processor and 1GB physical memory, the operating system is RedHat AS 3 Linux and totally 523920 swaps is recorded. To simplify the simulation, we assume EISP can always find enough computing service providers for the consumer, and ignore the circumstances that the provider withdraws its services in use.

We build the simulator for NCRS with 1000 different nodes based on our previous work [1]. For local computing simulation, the disk parameters are: T_S=4.9ms, T_L=3.0ms, T_W=0.2ms, S_p=4KB, and B_d=80MB/s.

For NCRS simulation, the amount of physical memory of all nodes is uniformly distributed between 128 MB and 1024 MB. The memory capacity of the computing node is 240MB. The parameters in formula (2) are: T_u=5ms, T_{RTT} =2ms, B_N=2MB, and $T_{remote\text{-}exec}$ is equal to the local execution time. For simplicity we assume 10% of the local instructions following the remote-executing ones don't need the remote-executing result, other 90% instructions need to wait and S_{result} (the size of remote executing result to send back to the local computing node) is one half the size of data sent to NIM.

As shown in figure 5, the performance of NCRS is much better than that of RAM-Grid and the disk IO, and the execution time of NCRS goes down slowly as the bandwidth increases.

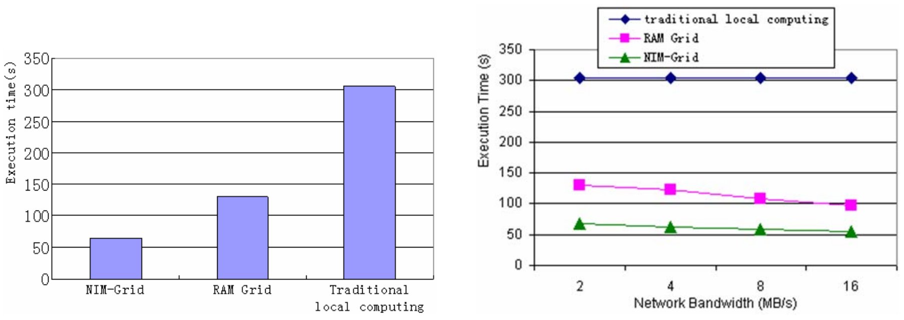


Fig. 5. Execution time of NIM-Grid, RAM Grid and local computing pattern

4 Conclusion

In this paper we propose a network RAM-based computational resource sharing grid, NCRS. In NCRS, the computing node regards the free memory of other nodes in networks (NIM) as a complement of local memory and uses it to store the large mount of intermediate data. During the process of computation the computing node sends the related instructions to NIM and NIM executes the instructions instead of it.

References

1. Chu, R., et al. A Distributed Paging RAM-Grid System for Wide-area Memory Sharing. in 20th International Parallel and Distributed Processing Symposium. 2006. Rhodes Island, Greece.
2. Feeley, M.J., et al. Implementing Global Memory Management in a Workstation Cluster. in Symposium on Operating Systems Principles. 1995. Copper Mountain Resort, Colorado.
3. Acharya, A. and S. Setia, The Utility of Exploiting Idle Memory for Data-Intensive Computations. Technical Report: TRCS98-02, 1998.
4. D. A. Patterson, "Latency lags bandwith" Communications of the ACM, vol. 47, pp. 71-75, 2004.
5. I. Foster, C. Kesselman, J. Nick, and S. Tuecke, "Grid services for distributed system integration," IEEE Computer, vol. 35, 2002.
6. I. Foster, "Service-Oriented Science," Science, vol. 308, pp. 814-817, 2005.
7. I. Foster, J. Frey, S. Graham, S. Tuecke, and K. Czajkowski, "Modeling Stateful Resources with Web Services," Globus Alliance, Argonne National Laboratory.
8. www.vce.org.cn/skyhawk

Tracing an Optical Buffer's Performance: An Effective Approach

W. Rogiest, D. Fiems, K. Laevens, and H. Bruneel

SMACS Research Group, Ghent University (UGent); Sint-Pietersnieuwstraat 41; B-9000 Gent; Belgium
Wouter.Rogiest@UGent.be

Abstract. Optical Burst Switching proposes a future-proof alternative to the current electronic switching in the backbone. The involved optical buffers are implemented with a set of Fiber Delay Lines, and suffer serious performance loss, when compared to RAM. Various existing models trace this loss, but either lack generality, accuracy, or effectiveness.

The optical buffer model we constructed is valid for general line lengths and burst sizes. An effective approach allowed to strongly reduce the solution's complexity, while remaining exact. This document presents the key formulas and performance graphs. The obtained model serves as a basic optimization tool, yielding results fast.

Keywords: OBS, optical buffer, FDL, fiber delay lines, discrete-time optimization.

1 Introduction

In a decade where bandwidth-consuming web services (with YouTube.com as a fashionable example) and peer-to-peer data exchange are working their way into everyday life, backbone infrastructure needs to be prepared for this ever-increasing bandwidth demand. Although data packets travel through the backbone in the form of light, they are still converted into electricity at every hop, in order to extract header information, buffer them, convert them back to light and transmit them to the next hop. Since this conversion is expected to be the bottleneck in the near future, the research community proposes alternative switching approaches, such as Optical Burst Switching (OBS) [1] and Optical Packet Switching (OPS) [2].

Just like conventional switches, optical switches need to resolve contention. Even without internal blocking, this arises inevitably, whenever two or more bursts (or packets) head for the same output at the same time. In general, wavelength conversion and buffering offer the most viable solutions to date.

Since light cannot be frozen, optical buffering is implemented by sending the bursts through sufficiently long pieces of fiber, often referred to as Fiber Delay Lines (FDLs). An optical buffer is thus a set of FDLs, each with different lengths. This way of implementing, though implementable with off-the-shelf components, has two drawbacks if compared to electronic RAM memory. On the one hand, optical buffers cannot realize all possible delays, which results in performance loss. Incoming bursts can only undergo delays equal to the length of one of the lines. For this reason, some capacity will be lost on the outgoing channel because, even when some bursts (or packets) are present in the buffer, they may not be available yet for transmission. The ensuing periods during which the outgoing channel remains unused, despite it's availability, are referred to as voids. They account for additional waiting time for the burst, and thus lead to increased loss. On the other hand, optical buffers have a large physical size. For typical OBS specifications (10 Gbps link, 100 kbit burst sizes), it takes about 2 km of fiber to apply a delay for the duration of a burst, and the physical size of a buffer thus grows quickly with the number of fibers. As such, an optical buffer typically has a smaller storage capacity than a RAM buffer, which leads to further increased loss. Therefore, quantifying this loss by means of an analytic model, and tuning the design parameters for optimal performance, are main questions in the ongoing research.

A prime contribution in the study of optical buffer performance, is [3]. There, and in [4, 5], Callegati studied an optical buffer with fiber lengths that are a multiple of a basic unit D, that is, $0, D, 2D, 3D, \ldots$(that is, equidistant line lengths). The type of buffer he called a degenerate buffer, while for the basic unit D he coined the term granularity. The analysis explored an approximation based on the iteration of a classic M/M/1/N buffer model, and yields approximate results for memoryless burst sizes. Extending the analysis to general burst sizes, and improving accuracy, [6] presents an accurate performance model for a degenerate buffer in discrete time, that is extended to continuous time in [7]. Murata and his co-authors [8] extend the approach of Callegati to multiwavelength systems, while in [9, 10], a method is provided to account for non-equidistant FDL lengths, that are not necessarely a multiple of the granularity. This approach is examined more thoroughly by Lambert and her co-authors in [11], and also, alternatively, in [12].

While much of the previous work is approximate [3, 8, 6, 7], here, exact performance measures are obtained. The model we constructed yields exact formulas, that offer the benefit of being valid for a broad range of parameter values at once. Like in [11], we worked in a discrete-time setting, with arbitrary sets of FDLs, and made no restrictive assumptions on the burst size distribution. Main difference with [11] is that we obtained a more effective modeling approach, by focusing on the waiting time only. In [11], the analysis is based on the evolution of the scheduling horizon, and computational complexity depends on the length of the longest delay line. By ruling out the scheduling horizon as a measure, our model's complexity is independent thereof, and depends solely on the number of delay lines, which is small in all practical cases. Our modeling thus allows to easily compute results for any burst size distribution, and any size, and this also in the case of very long delay lines. Therefore, it allows to approximate the continuous-time case arbitrary close without augmenting the calculation time.

In Sect. 2, we present the FDL buffer setting and modeling variables. The analysis of this model will be given in Sect. 3, proposing a simple Markov chain, and showing how to derive the waiting time and loss characteristics from this. Some numerical examples follow in Sect. 4. The latter show how the optical buffer's performance is strongly impacted by the buffer size, in a way that depends much on the burst size distribution.

2 Stochastic Model

Here, we first focus on the FDL buffer setting, to then move to the assumptions of the traffic model. Consequently, we look at the system equations, that are expressed in terms of the waiting time of an arbitrary burst.

2.1 Optical Buffer Setting

The optical buffer under consideration is a set of $N + 1$ Fiber Delay Lines, one of them with length $\omega_0 = 0$, and N lines with different lengths ω_i, $i = 1 \dots N$. As the set of lines are intented to resolve contention, it is necessary that contending bursts undergo different delays. Therefore, a useful FDL set never contains the same length twice, $\omega_i \neq \omega_j$ for $i \neq$j . For notational convenience, we denote the set of FDL lengths as $\Omega = \{\omega_0, \omega_1, \dots \omega_N\}$, and we sort the line lenghts ascendingly, $\omega_0 < \omega_1 < \dots < \omega_N$. Note that although a common choice is to choose equidistant lengths ($\omega_i = i \times D$, $i = 0 \dots N$), the analysis is done for arbitrary lengths.

This buffer is located at the output of a backbone switch, and is dedicated to a single outgoing wavelength. We consider bursts arriving at the buffer randomly, and possibly overlapping in time. Since there is only one wavelength to queue for, all overlap during transmission should be prevented. By means of a switching matrix that allows to send any burst to any of the $N + 1$ delay lines, buffer control exercises a FIFO (First-In-First-Out) scheduling discipline, and sends every burst to a sufficiently long delay line, so as not to overlap with the one-but-last burst. If such a (sufficiently long) delay line is present, the burst is accepted and enters; if not, the burst is dropped. The periods during which the system can accept any possibly arriving burst, are called available periods; the periods during which the system drops arriving bursts are called unavailable periods.

2.2 Arrivals and Acceptances

We work in a discrete-time setting, which implies that all random variables (rv's) and parameters (such as the ω_i) are expressed in multiples of a (generic) time slot length. We assume the bursts arrive in the system according to a Bernoulli arrival process, with parameter p. This implies that at the most one arrival occurs during a slot, and p gives the probability of such a burst arrival, for an arbitrary slot.

Arriving bursts are either accepted upon arrival (during available periods), or dropped (unavailable periods). We number the bursts in the order at which they arrive, but only assign an index to those bursts that are accepted. With each accepted burst k, we associate an inter-arrival time T_k, that captures the time between the kth arrival and the next burst arrival, being the arrival of (i) burst $k + 1$, if this next burst is accepted or (ii) a burst without number, if this next burst is dropped. For the assumed Bernoulli arrival process, these inter-arrival times T_k form a sequence of identical and independently distributed (iid) random variables (rv's) that have a common geometric distribution, with parameter p

$$\Pr[T_k = n] = t(n) = p \cdot \bar{p}^{n-1}, \quad n \geq 1. \tag{1}$$

Here, $\bar{p}$ is the commonly adopted shorthand for $1 - p$. The inter-acceptance time A_k is defined as the time between the kth acceptance and the $(k + 1)$th, and is in general larger (and never smaller) than T_k.

To track the system's performance in an easy way, we consider two mutually exclusive events, for an arbitrary accepted burst k.

`Next-Accept`. The burst that arrives just after the kth burst is accepted, and counted as burst $k+1$. The inter-acceptance time A_k is identical to T_k, and thus, it follows from (1) that

$$\Pr[A_k = n | \texttt{Next-Accept}] = t(n) = \bar{p} \cdot p^{n-1}, \quad n \geq 1. \tag{2}$$

`Next-Drop`. The burst that arrives just after burst k is dropped. Now, the burst following burst k is not assigned an index, and possibly, even more bursts are dropped before burst $k+1$ is accepted. The inter-acceptance time A_k clearly differs from the inter-arrival time, and has a more complicated probability distribution. Luckily, the latter need not be tracked, and we rely on an additional measure: the reactivation time $\widetilde{A}_k$, defined as the time between the end of the unavailable period following the kth burst, and the arrival of burst $k+1$. Note that the reactivation time is only relevant if the associated burst k effectively causes burst loss, by driving the system into an unavailable state. Invoking the memoryless nature of the arrival process, the reactivation time is easy to trace, and intimately linked to the inter-arrival times (1),

$$\Pr[\widetilde{A}_k = n | \texttt{Next-Drop}] = \widetilde{t}(n) = \bar{p} \cdot p^{n}, \quad n \geq 0. \tag{3}$$

The complementary use of the series of random variables A_k and $\widetilde{A}_k$ suffices to capture the timing aspects of arrival and acceptance, relevant for our analysis.

2.3 General Burst Sizes

For the characterization of the burst sizes, we adopt the mentioned numbering of bursts, and so the kth burst has a burst size B_k. The burst sizes, just like the inter-arrival times, form a sequence of iid rv's with a common distribution, but have no further restriction on their distributions. Therefore, we consider general probabilities

$$\Pr[B_k = n] = b(n), \quad n \geq 1, \tag{4}$$

that are arbitrary, except for the conditions that any useful probability mass function has to comply with: $0 \leq b(n) \leq 1$, $\sum_n b(n) = 1$.

2.4 System Equations

As mentioned, the system's evolution can be captured by means of the waiting time of a burst only. Still using the same numbering, we associate the waiting time W_k with the kth burst, and define it as the time between the acceptance of burst k, and the start of it's transmission. Again considering that either the next burst is accepted or dropped, we have

`Next-Accept`. The condition for this to happen, in terms of waiting times and FDL lengths, is that the burst that arrives just after the kth burst can be provided a sufficiently long delay, that is,

$$W_k + B_k - T_k \leq \omega_N.$$

Then, A_k equals T_k, and

$$W_{k+1} = \lceil W_k + B_k - A_k \rceil_{\Omega}, \tag{5}$$

where we adopted the notation $\lceil x \rceil_{\Omega} = \inf\{y \in \Omega, y \geq x\}$, $x \leq \omega_N$.

Next-Drop. Now, the burst that arrives just after the kth burst can not be provided a sufficiently long delay, and

$$W_k + B_k - T_k > \omega_N.$$

As a result of this (and of the memoryless nature of the arrival process), the waiting time of burst $k+1$ no longer relates to W_k, and W_{k+1} is characterized by the reactivation time, through

$$W_{k+1} = \lceil \omega_N - \widetilde{A}_k \rceil_{\Omega}. \tag{6}$$

These two system equations (5) and (6), together with their respective probability mass functions (2) and (3), provide the input for the analysis.

3 Analysis

In this section, the limited set of waiting times serves a state variable for a Markov chain, of which we trace the transition probabilies. Inversion yields the waiting time probabilities of accepted bursts, and this in turn allows to extract the loss ratio.

3.1 Markov Chain for Waiting Time

Before delving into the analysis, we note that the waiting time can only take on $N+1$ different $\omega_i \in \Omega$. Therefore, it is an attractive state variable for a Markov chain approach to the system.

The Markov chain we consider consists of $N+1$ states, that correspond to $N+1$ possible waiting times ω_i, $i = 0 \ldots N$. It is characterized by a transition matrix with probabilities β_{ij},

$$\beta_{ij} = \Pr[W_{k+1} = \omega_j | W_k = \omega_i], \quad 0 \leq i, j \leq N.$$

For ease of notation, we introduce $\omega_{-1} = -\infty$. We split β_{ij} in two separate contributions, that correspond to the events discussed in Sect. 2.4.

$$\begin{aligned}\beta_{ij} &= \Pr[\omega_i + B_k - T_k \leq \omega_N, \omega_j = \lceil \omega_i + B_k - A_k \rceil_{\Omega}] \\ &\quad + \Pr[\omega_i + B_k - T_k > \omega_N, \omega_j = \lceil \omega_N - \widetilde{A}_k \rceil_{\Omega}] \\ &= \Pr[\omega_{j-1} - \omega_i < B_k - T_k \leq \omega_j - \omega_i] \\ &\quad + \Pr[B_k - T_k > \omega_N - \omega_i] \Pr[\omega_N - \omega_{j-1} > \tilde{A}_k \geq \omega_N - \omega_j]. \end{aligned} \tag{7}$$

Since the burst sizes B_k and inter-arrival times T_k only occur as $B_k - T_k$, we introduce the series of random variables $U_k = B_k - T_k$, and it's cumulative distribution function

(CDF) $U(n)$. Taking into account the distribution of B_k and A_k, calculations show that $U(n)$ simplifies to

$$U(n) = \Pr[B_k - A_k \leq n] = \sum_{i=1}^{n} b(i)(1 - \bar{p}^{i-n-1}) + \bar{p}^{-n-1}B(\bar{p}), \tag{8}$$

where the sum over n disappears if $n \leq 0$, and $B(z)$ is the probability generating function of B_k, defined as $B(z) = \mathrm{E}[z^{B_k}] = \sum_{n=1}^{\infty} b(n)z^n$. Similarly, we consider (3) to obtain the CDF of $\widetilde{A}_k$, $\widetilde{A}(n)$, as

$$\widetilde{A}(n) = \Pr[\widetilde{A}_k \leq n] = 1 - \bar{p}^{n+1}, \quad n \geq 0, \tag{9}$$

and zero when $n < 0$. Adopting these notations, (7) can be stated as

$$\begin{aligned}\beta_{ij} &= U(\omega_j - \omega_i) - U(\omega_{j-1} - \omega_i) \\ &\quad + [1 - U(\omega_N - \omega_i)][\widetilde{A}(\omega_N - \omega_{j-1} - 1) - \widetilde{A}(\omega_N - \omega_j - 1)]\end{aligned} \tag{10}$$

Combining the last three expressions, we obtain an explicit expression for the coefficients β_{ij}, in terms of the (given) ω_i, $b(n)$ and p.

From here, we obtain the vector of the waiting times as the normalized left eigenvector $[w(n)]$ of the matrix $[\beta_{ij}]$, associated with the eigenvalue 1, that is to satisfy $w(n) = \sum_{j=0}^{N} w(j)\beta_{jn}$, $0 \leq n \leq N$. This eigenvector can easily be obtained numerically, posing no problem for the small N we are interested in. It contains the $N+1$ different steady-state waiting time probabilities

$$\lim_{k\to\infty} \Pr[W_k = \omega_n] = \Pr[W = \omega_n] = w(n), \quad 0 \leq n \leq N. \tag{11}$$

From this, we can also define a mean waiting time $\mathrm{E}[W] = \sum_{i=1}^{N} w(i)\omega_i$.

3.2 Loss Ratio

Up to now, we considered only bursts that were accepted, and even chose to leave the dropped bursts unnumbered. We now focus on the burst loss ratio (LR), defined as the fraction of arriving bursts that is dropped, and study the unavailable period, associated with an accepted burst k, in two cases. If on the one hand, the arrival of burst k does not push the system into unavailability, then the unavailable period following burst k equals zero. This implies that, the time slot after the arrival of burst k, a new arrival can already be accepted. In terms of the involved rv's, this means that $W_k + B_k - 1 \leq \omega_N$.

On the other hand, if the unavailable period following burst k is larger than zero, then $W_k + B_k - 1 > \omega_N$. Now, it takes the system a number of slots equal to $W_k + B_k - \omega_N - 1$, to become available again. The last measure is the unavailable period following burst k, under the condition $W_k + B_k - 1 > \omega_N$. Combination of both cases lead to the conclusion that the unavailable period, following burst k, is given by $(W_k + B_k - \omega_N - 1)^+$, where $(x)^+$ is shorthand for $max\{0, x\}$. Invoking the memoryless nature of the arrival process, we can write down an expression for $\mathrm{E}[X_k]$, the average number of lost bursts during the unavailable period following burst k,

$$\mathrm{E}[X_k] = p \cdot \mathrm{E}[(W_k + B_k - \omega_N - 1)^+]. \tag{12}$$

With (4) and (11), this becomes

$$\mathrm{E}[X_k] = p \cdot \left(\mathrm{E}[B] + \mathrm{E}[W] - \omega_N - 1 - \sum_{i=0}^{N} w(i) \sum_{j=1}^{\omega_N - \omega_i} b(j)(j + \omega_i - \omega_N - 1) \right).$$

Now, it suffices to note that, with every accepted burst, a number of $\mathrm{E}[X_k]$ bursts on average is dropped, resulting in a burst loss ratio (LR)

$$\mathrm{LR} = \mathrm{E}[X_k]/(1 + \mathrm{E}[X_k]).$$

4 Numerical Results and Discussion

With the above results at hand, one can easily study the impact of the various design parameters on loss performance. More specifically, one wants to determine optima for the granularity, which are values that yield a minimal burst loss ratio. While similar curves already occur in [6] for $N = 20$, the approximation applied there lost accuracy for small N. As such, the examples given here yield additional information, for the case of smaller (more realistic) buffer sizes. We look at four examples.

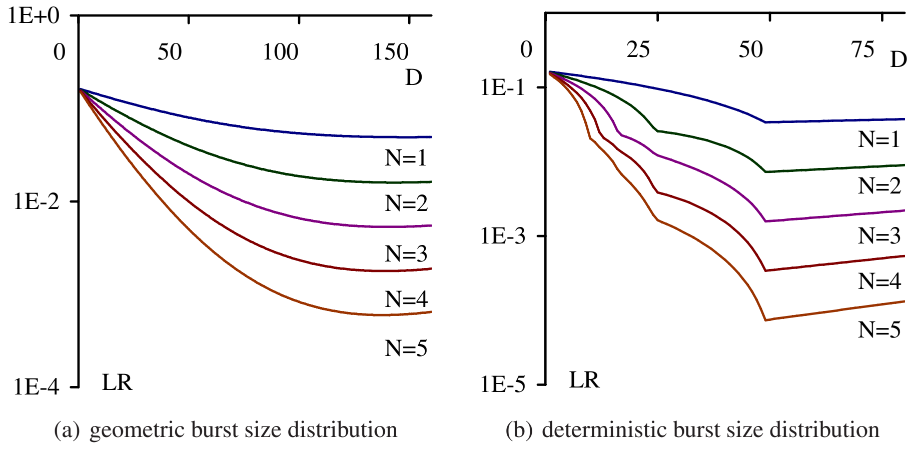
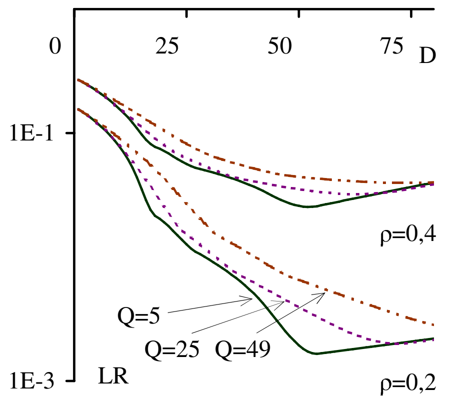


Fig. 1. Loss ratio as function of the FDL granularity (slots), for various buffer sizes N, burst size distributions (a) and (b), both with $\mathrm{E}[B] = 50$ slots, for a load $\rho = 0.2$

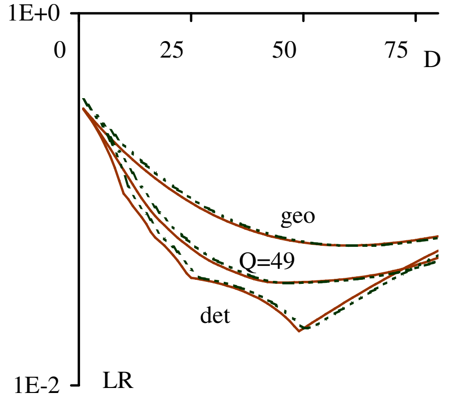
In both panes of Fig. 1, we assess the impact of a small buffer size on loss performance. They show the burst loss ratio (LR) as a function of the granularity D, for an equidistant delay line setting, $\mathrm{E}[B] = 50$, and a load of $\rho = 0.2$. The left pane displays the situation for geometric burst size distribrution, with the expected rise of the loss ratio, when the buffer size N decreases. Also, it can be seen that the LR lowers for increasing granularity, and only starts to rise again for D larger than about 150 slots.

Considering the five curves together, the figure suggests that the optimal granularity, for a given load, only slightly increases when the buffer size approaches its minimum of $N = 1$. Results for higher loads, not included here, confirm that the influence of diminishing buffer size on optimal granularity is but weak, especially if compared to the impact of variations in the load. The latter is illustrated in [6], and is much stronger than the influence observed here.

On the right pane of Fig. 1, we have the same setting, for fixed burst sizes, $\mathrm{E}[B] = B = 50$. Again, a smaller buffer suffers more loss, but now the optima alter in a more surprising way. More precisely, the "notches" at $D = (B-1)/n, n = 1, 2, \ldots$, known from previous work, are not uncountable (as was the case in [6] for an infinite-sized buffer), but are limited in number to N. Exactly N notches occur for each curve, at $D = (B-1)/n, n = 1, 2, \ldots N$. Since it is known from [6] that these values correspond to optima (with $D = B - 1$ being the optimum for low load, and $D = (B-1)/2$, then $(B-1)/3$,... for increasing load), we verified and found that the set of potential optima, for a load $0 \leq \rho \leq 1$, is indeed limited to (at the maximum) the number of fibers. Also, it was found that the same optimum shift as known from [6] takes place, but now only over the available values: first $D = B - 1$ for low load, then $D = (B-1)/2$ for increasing load if $N \geq 2$, and so on if $N \geq 3$...



(a) various uniform burst size distributions



(b) equidistant (continuous) or not (dashed)

Fig. 2. Loss ratio as function of the FDL granularity (slots), for $\mathrm{E}[B] = 50$ slots, with (a) uniform burst size distributions (various radii Q) and (b) different burst size distributions for an equidistant and non-equidistant setting, load $\rho = 0.5$

To verify if the optima for deterministic burst sizes also apply to varying burst sizes, we consider a uniform burst size distribution with radius Q, that has a mean burst size $\mathrm{E}[B] = 50$, and is uniform within the range $[50-Q, 50+Q]$. For small Q, this distribution resembles the deterministic distribution. As such, this setting allows to verify what influence variances on the burst size have on loss performance. In Fig. 2, the left pane compares the performance of an optical buffer of size $N = 3$ for three uniform distributions, having a narrow range ($Q = 5$, range $[45, 55]$), an intermediate range ($Q = 25$, range $[25, 75]$) and a broad range ($Q = 49$, range $[1, 99]$), and this for load $\rho = 0.2$, $\rho = 0.4$, respectively. For the narrow-ranged one, the curves look very similar to those of the deterministic distribution, and the same optimum around $D = B - 1$ shows. The curves for the intermediate-ranged and broad-ranged case show that increasing Q makes the granularity optimum shift toward higher values, at least for load $\rho = 0.2$ and $\rho = 0.4$. Curves not included here, for higher load, show that the optima for a narrow range (small Q) concentrate around the limited set $D = (B - 1)/n$, $n = 1, 2, \ldots N$, known from the deterministic distribution, while for larger Q, the optimum only gradually decreases.

Choosing non-equidistant lengths for the delay lines can in some cases provide better performance. As is shown in [11] for deterministic burst sizes, a non-equidistant set of FDLs can outperform an equidistant set of the same size. However, it turns out that this happens only when the load rises above a certain value (for example 60.17%, for $N = 10$, $B = 20$, $D = 19$). Further, even when the non-equidistant one outperforms the equidistant one, the performance gain is rather small. This said, non-equidistant settings remain interesting, since for more general assumptions (correlated arrivals, multi-wavelength output), the performance gain might be larger. For the right pane of Fig. 2, we chose non-equidistant FDL lengths in a way similar to [11], with shortened lengths for the largest lines. An equidistant set (continuous curves) and non-equidistant set (dashed curves) are considered, for a buffer size $N = 5$, load $\rho = 0.5$, and the burst size distributions geometric, deterministic and uniform ($Q = 49$). The non-equidistant set has FDL lengths D, $2D - 2$, $3D - 3$, $4D - 4$, $5D - 8$. The curves show how the non-equidistant set just outperforms the equidistant one, for geometric and uniform burst size distribution, while the opposite is true for a deterministic burst size distribution. Although not included here, figures for the same setting, for a load of $\rho = 0.3$, $\rho = 0.6$ and $\rho = 0.8$ resp., show the same qualitative result, while the performance difference itself always remains small.

5 Conclusions

Unlike previous work, we have studied an optical buffer by considering only the waiting times, that correspond to the lengths of the Fiber Delay Lines (FDLs). This straightforward approach allowed us to obtain exact results for a small computional load, especially for small buffers. Without posing restrictions on the lengths of these lines, we constructed a model valid for a Bernoulli arrival process, and a general burst size distribution, based on the analysis of the involved Markov chain. The results of the latter we used to obtain (i) the steady-state waiting time probabilities and, by considering the unavailable periods, (ii) the loss ratio.

The presented performance graphs show the impact of reduced buffer size on performance, and the associated optimal value for the granularity. While the optimum for geometric burst size distributions hardly changes when we lower the buffer size, the optimum for a deterministic burst size distribution does change. Also, a study of a uniform distribution showed how smaller and larger variation around an average burst size impacts performance. Finally, performance results for a non-equidistant set were compared to those of an equidistant set.

Concluding, the model presented here is rather elementary in it's approach, and therefore provides a basic tool, ready to use for optimization studies. The extension of the model toward (i) multi-wavelength systems, (ii) correlated arrivals, we consider challenging, and would surely lead to a more profound insight in the operation of optical buffers.

References

1. C. Qiao and M. Yoo. Optical burst switching–a new paradigm for an optical internet. *Journal on High-Speed Networks*, 8:69–84, 1999.
2. T. El-Bawab and J.-D. Shin. Optical packet switching in core networks: between vision and reality. *IEEE Communications Magazine*, 40:60–65, September 2002.
3. F. Callegati. Optical buffers for variable length packets. *Communications Letters, IEEE*, 4(9):292–294, 2000.
4. F. Callegati. On the design of optical buffers for variable length packet traffic. In *Proceedings of the Ninth International Conference on Computer Communications and Networks (Las Vegas)*, pages 448–452, 2000.
5. F. Callegati. Approximate modeling of optical buffers for variable length packets. *Photonic Network Communications*, 3(4):383–390, October 2001.
6. K. Laevens and H. Bruneel. Analysis of a single-wavelength optical buffer. In *Proceedings of Infocom 2003 (San Francisco)*, 2003.
7. W. Rogiest, K. Laevens, D. Fiems, and H. Bruneel. A performance model for an asynchronous optical buffer. *Perform. Eval.*, 62(1-4):313–330, 2005.
8. M. Murata and K. Kitayama. Ultrafast photonic label switch for asynchronous packets of variable length. In *INFOCOM 2002 (New York)*, 2002.
9. R. C. Almeida, J. U. Pelegrini, and H. Waldman. Optical buffer modelling for performance evaluation considering any packet inter-arrival time distribution. In *IEEE International Conference on Communications, 2004*, volume 3, pages 1771–1775, 2004.
10. R. C. Almeida, J. U. Pelegrini, and H. Waldman. A generic-traffic optical buffer modeling for asynchronous optical switching networks. *Communications Letters, IEEE*, 9(2):175–177, 2005.
11. J. Lambert, B. Van Houdt, and C. Blondia. Single-wavelength optical buffers: non-equidistant structures and preventive drop mechanisms. In *Proceedings of NAEC 2005 (Riva Del Garda)*, pages 545–555, 2005.
12. J. Lambert, B. Van Houdt, and C. Blondia. Queues with correlated inter-arrival and service times and its application to optical buffers. *Stochastic Models*, 22(2):233–251, 2006.

A New Necessary Condition for Shortest Path Routing

Mats Petter Pettersson and Krzysztof Kuchcinski

Lund University, Sweden
{matsp,kris}@cs.lth.se

Abstract. In shortest path routing, traffic is routed along shortest paths defined by link weights. However, not all path systems are feasible in that they can be realized in this way. This is something which needs to be taken into account when searching for a set of paths that minimize capacity consumption.

In this paper, we discuss a new necessary condition that can be used during search to prune infeasible path systems. The condition can be expressed using linear inequalities, or used in constraint programming, where its simple definition is convenient for the efficient implementation of propagation. Experiments on networks from the SNDLib benchmark show that this condition has strong pruning capabilities.

1 Introduction

In shortest path routing, e.g., the widely used OSPF protocol, traffic flows are controlled by assigning a weight to each link in the network. Point-to-point traffic is routed along shortest paths, as defined by these weights. Different settings of the link weights will lead to different path systems. We say that a path system is feasible if there is a weight system in which every path in the path system is a shortest path. Depending on the amount of bandwidth required between different node pairs and link capacities, some path systems are preferable to others, e.g., by minimizing the maximum percentage of bandwidth used on each link. In general this optimization problem is NP-hard [1].

Many approaches have been used for finding a weight system that uses the network in an optimal way, given demands and capacities. Here, we consider constraint-based methods, which model the problem using variables and constraints, e.g., mixed integer programming (MIP) and constraint programming (CP) [2]. They can both be used as a basis for complete search methods, guaranteed to find an optimal solution. Such model-based approaches typically include a set of variables describing the path system, as the optimization criteria can be conveniently expressed on these variables.

The model also needs to constrain the path system to be feasible. In this paper we discuss a necessary condition for feasibility, which we call the *4-node condition*, that is well suited for use in a complete model-based method using systematic search. This condition assumes two further restrictions on the path system: unique paths and symmetry. Path uniqueness means that there should be only one (unique) shortest path between each node pair. Symmetry means that traffic from node s to node t should use the same (undirected) path as traffic from node t to node s. We have previously used the condition in a CP-LP hybrid method for solving the problem with these restrictions [3].

In this paper, we look specifically at the constraints dealing with feasibility, of the CP part of the hybrid method. We present experimental work showing that the condition is very useful for pruning subtrees without feasible solutions.

The paper is organized as follows. In Sect. 2, we give a more formal definition of the problem and discuss its modelling. Section 3 defines the 4-node condition and describes some of its properties. In Sect. 4, we describe a CP model using the 4-node condition, followed by experimental work in Sect. 5. Conclusions and discussion of future work are given in Sect. 6.

2 Problem Definition and Modelling

2.1 The Problem

We are considering shortest path routing optimization with the extra requirement that all paths should be unique. Furthermore, we consider only undirected demands, i.e., we do not distinguish between demands from s to t and t to s. Due to the undirectedness of the demands, the network topology can be modelled with an undirected graph (V, E). We use set notation, $\{i, j\}$, to denote the undirected edge between i and j.

Apart from the graph, an instance of the unique undirected shortest path routing problem specifies the link capacity $c_{\{i,j\}}$ for each edge $\{i, j\} \in E$, and the sum of all bandwidth, $d_{\{s,t\}}$, demanded between every pair of nodes, s and t. If there are no demands between s and t, then $d_{\{s,t\}} = 0$.

A feasible solution for such an instance defines a positive integer weight $w_{\{i,j\}}$ for each edge, in such a way that these weights define unique shortest paths between all node pairs. We refer to the resulting set of paths as a feasible path system. The demands and the path systems define how much capacity is used on each link. The objective is to optimize some function of the capacity usage, e.g., to minimize the utilization of the maximally utilized link in the network.

2.2 Modelling of the Problem

We consider solution methods that model the problem using variables and constraints. Given a model, the problem is typically solved by a systematic search in the space of possible variable assignments. During the search process, the constraint model is used for pruning and computation of bounds. The quality of the constraint model lies in its ability to prune and provide tight bounds. Typical methods that fit in this framework are MIP and CP.

Constraint-based methods for shortest path routing need to express both constraints that guarantee feasibility, and optimization criteria. The latter are most easily expressed on flow variables that model the amount of flow on a specific link. These flow variables are linked by constraints to path variables, modelling the path system.

Modelling the path system can be done in several ways. What then remains is to enforce feasibility upon the model. The only known way for testing the feasibility of a path system uses a linear program. Its running time is polynomial, but can still be costly to compute. Using this approach introduces weight variables, one per link, and constraints linking them to the path variables. One problem with using such an approach with MIP seems to be that the linear relaxation of the model is weak. One potential way to deal with this problem is to introduce cuts, extra inequalities that help tighten the relaxation [4].

An alternative is to model using simpler conditions that are necessary but not sufficient for feasibility, meaning the model will allow some path systems that are not feasible. Several necessary conditions have been described in the literature [5,6]. Before a solution is finally accepted, it needs to be checked using the more expensive LP-based method that encodes a necessary and sufficient condition.

3 The 4-Node Condition for Path System Feasibility

We believe that problems similar to the weak linear relaxations for MIP apply to CP models using explicit weight variables. Therefore, our approach to the problem does not use weight variables, but the second approach outlined in the previous section, using necessary conditions to prune the search space as much as possible, followed by an LP-based test for each candidate solution. In this section, we introduce the 4-node condition that was used. In Sect. 4, we show how it was integrated into a CP method, and in Sect. 5, we give some experimental results.

3.1 The 4-Node Condition

Consider any four nodes, a, b, c, and d, in a network graph. Six node pairs that involve these nodes can be formed, and for each there should be a shortest path. Some of these paths may be subpaths of others, or may partly overlap. However, not all situations in which several paths overlap are feasible.

In Fig. 1, some feasible situations (Fig. 1(a-f)) and some infeasible ones (Fig. 1(g-h)) are shown. Only paths that include another of the four nodes are drawn, meaning that node pairs that have no path drawn between them have a path that does not pass either of the two other nodes. The situation in Fig. 1(g) is infeasible, since the path from a to d passes node b, but the path from a to c passes d without passing b. The six first situations are feasible, and basically the only feasible ones, up to permutation of the nodes.

We want to formalize this notion of feasibility of a path system with respect to sets of four nodes. To do this, we model a path system, using binary *node path variables.* The node path variable $y_{\{s,t\}u}$, has value 1 if and only if node u is used in the path between s and t. For each pair of nodes, s and t, we will have $|V|-2$ node path variables. Clearly, a path system defines a value for each node path variable. We will see that for a feasible path system, the reverse also

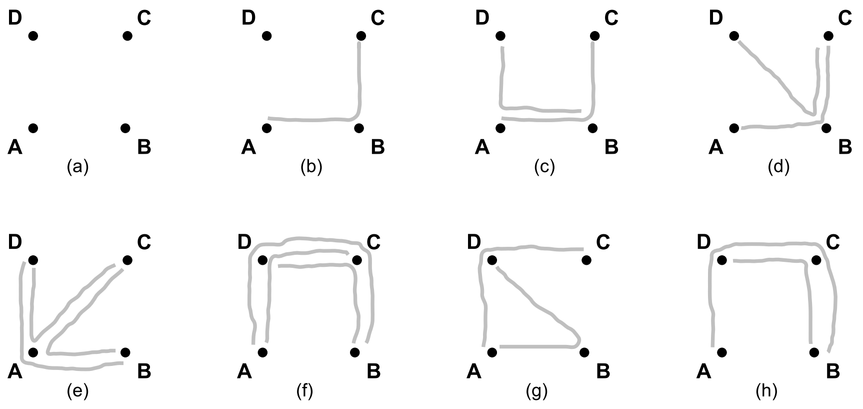


Fig. 1. Some situations for paths involving four nodes

holds, so that there is a one-to-one mapping between feasible path systems and assignments to node pair variables.

For four nodes, there are twelve node paths variables that are defined only on these nodes. Assuming that the path system is feasible restricts what combinations of values we can have for these twelve variables:

Definition 1. *We say that an assignment of the twelve node path variables only mentioning the same four nodes is 4-node consistent if there exists a feasible path system in which the variables would have these values.*

For twelve binary variables, we have 4096 possible assignments. Out of these, it turns out that only 53 are actually 4-node consistent. For example, the assignment with $y_{\{a,b\}c} = 1$, $y_{\{a,b\}d} = 1$, $y_{\{b,d\}c} = 1$, and all other node path variables zero is not 4-node consistent. This corresponds to the situation in Fig. 1(h). Also assigning $y_{\{a,c\}d} = 1$ would make the assignment 4-node consistent, corresponding to the situation in Fig. 1(f).

Definition 2. *We say that a path system is 4-node consistent if its representation by node path variables has all groups of twelve variables that are defined on the same four nodes being 4-node consistent. A partial path system is 4-node consistent if it can be extended to a fully defined 4-node consistent path system.*

It is important to note that 4-node consistency for a path system is only a necessary condition for feasibility, not a sufficient one. In other words, there are 4-node consistent path systems that are not feasible.

3.2 Linear Inequality Formulation

The 4-node condition for nodes a, b, c, and d, can be expressed using linear inequalities of the following three types:

$$y_{\{a,b\}c} + y_{\{a,c\}b} + y_{\{b,c\}a} \leq 1 \tag{1}$$

$$2y_{\{b,d\}c} + 2y_{\{a,d\}b} - y_{\{a,c\}b} - y_{\{a,d\}c} \leq 2 \tag{2}$$

$$2y_{\{c,d\}a} + 2y_{\{c,d\}b} - y_{\{b,d\}a} - y_{\{b,c\}a} - y_{\{a,d\}b} - y_{\{a,c\}b} \leq 2 \tag{3}$$

To encode the 4-node condition, we must include all versions of the above constraints where some or all of the node indices have been permuted. In all, there will be 4 unique constraints of the first type, 24 of the second type and 6 of the last type. Building a model where all these constraints are expressed explicitly for all groups of four nodes would be expensive. The alternative for MIP could be to use them as cutting planes in a branch-and-cut approach, or as will be shown in the next section, to express the 4-node condition implicitly in a CP approach.

3.3 Some Properties of the 4-Node Condition

Here we informally (due to limited space) discuss some properties of the 4-node condition. We assume that for a given network, we have a full 4-node consistent assignment to all node path variables.

Consider all nodes that appear on the path between nodes s and t, i.e., all u s.t., $y_{\{s,t\}u} = 1$. We show that 4-node consistency implies an order for the nodes on the path. Define $u \prec v \Leftrightarrow y_{\{u,t\}v} = 1 \wedge y_{\{v,t\}u} = 0$. If the node path variables are 4-node consistent, the relation $\prec$ will be a total order, defining the order of the nodes in the path between s and t. This implies that a 4-node consistent assignment to all node path variables uniquely defines a path system.

Consider the node s' adjacent to s on the path between s and t. The 4-node condition implies that the path from s' to t will contain the same nodes as the path from s to t, excluding s, and in the same order. Inductively, this implies that the path between any pair of nodes, u and v, on the path between s and t will follow the path between s and t. As a consequence, the 4-node condition implies a property called suboptimality, introduced in [5].

4 Using the 4-Node Condition in a CP Model

In this section we describe a CP model, originally introduced in [3], for the problem. This model uses the 4-node condition for pruning during search.

4.1 Constraint Programming

In CP, modelling is typically done with variables with finite discrete domains. Constraints on subsets of variables restrict what partial assignments are allowed, *consistent*, for those variables. The constraints collectively define the set of feasible solutions for the problem as a whole – full assignments consistent with every constraint. The set of constraints can be heterogeneous, unlike linear programming where only linear inequalities are allowed. Unlike LP, inference in CP is not global, not taking all constraints into account at once, but done by each constraint in isolation. Whenever a variable has its domain of possible values reduced, constraints involving that variable may infer domain reductions in other variables, which in turn may lead to further domain reductions in other variables. This process is known as *constraint propagation*, and will be triggered at each new search node by any domain reductions caused by the branching decision. Thus each constraint is equipped with a specific *filtering algorithm*, which is invoked whenever there is a domain reduction in any of the variables involving the constraint, and is responsible for the inference and enforcing of domain reductions in its other variables. When a domain has its domain emptied, we have discovered an inconsistency, and the search will backtrack.

4.2 Constraint Programming Model

We use two ways of modelling a shortest path for each node pair: as a set of edges and as a set of nodes. Binary *edge path variables*, $x_{\{s,t\}\{u,v\}}$, have value 1 iff the edge $\{u, v\}$ is used in the path between s and t. Binary *node path variables*, $y_{\{s,t\}u}$, have value 1 iff node u is used in the path. For convenience, here we include $y_{\{s,t\}s} = y_{\{s,t\}t} = 1$.

Below, we describe the conditions that are used in the constraint model. As the number of conditions used is very large, in the implementation we do not have an explicit constraint for each condition. Instead we have one single constraint, defined on all edge and node path variables, checking consistency and doing propagation for all the conditions. This is equivalent in effect, but makes the implementation more efficient.

The full model in [3] also includes variables modelling the flows on links, and capacity-based optimization criteria. As we are only concerned with feasibility, these are not included here.

Chanelling Condition. Both edge path and node path variables are by themselves sufficient to model unique paths between all node pairs. However, in the CP part of the model, some conditions are more naturally expressed on the node path formulation, some on the edge path formulation. Therefore, we keep both, and use channelling conditions to keep the two representations consistent. For each edge path variable, we have the following condition:

$$x_{\{s,t\}\{u,v\}} = 1 \leftrightarrow y_{\{s,t\}u} = 1 \wedge y_{\{s,t\}v} = 1 \wedge (\forall w. w \neq u \wedge w \neq v \rightarrow y_{\{u,v\}w} = 0) \tag{4}$$

Unique Path Condition. For each node pair $\{s, t\}$, the edge path variables $x_{\{s,t\}\{u,v\}}$ should define a unique simple path between s and t. This condition can be enforced partly by restricting the number of adjacent edges for each node. Nodes s and t have exactly one adjacent edge path variable set to one, and other nodes have either two adjacent edge path variables set to one (if they are included in the path), or none (if they are not included in the path). This condition will be enforced every time an edge path variable is set. However, it does not rule out disconnected cycles, so we will also need to enforce that no cycles are constructed. This is done whenever a path connecting s and t is finished, when we know that all edge path variables that are not included in the path should be set to zero.

4-Node Condition. Our constraint will impose that the partial assignment of node path variables should be 4-node consistent at any time during the search. It also does propagation, for each group of twelve variables detecting if an unassigned variable can have its value inferred, i.e., if it has the same value in all extensions of the partial assignment that are 4-node consistent with respect to the twelve variables.

The simple formulation of the 4-node condition in terms of node path variables is an advantage here, since it makes propagation convenient. Every time a node path variable is assigned, there is opportunity for further propagation. The implementation is basically a case analysis. When a variable is assigned, all possible consequences of this assignment according to the 4-node condition are considered. The structure of the condition can be exploited, so it is not necessary to explicitly consider all 53 possibilities for every group of twelve variables where the variable is included.

5 Experiments

5.1 Generating Full Search Trees

We first performed experiments to investigate the ability of the 4-node condition to prune search nodes from which no feasible path system can be completed. Pruning can take place either explicitly, by detecting inconsistency at a search node, or by propagation, which will set the value of a path variable, thus reducing the size of the remaining search space.

To measure the strength of the 4-node condition, we generated random network topologies and constructed the *entire* search tree, containing *all* feasible solutions. We did this using the constraint model presented in Sect. 4 (which we refer to as method 1), recording the number of feasible solutions found, and the total number of nodes in the search tree, including solution nodes.

As the 4-node condition is not a sufficient condition for feasibility, some leaf nodes will be generated that are consistent with all the constraints but still do not correspond to a feasible path system. To find out how many such nodes occur, we have also run our CP model with a post processing step, where every consistent leaf node is tested using a necessary and sufficient condition for feasibility (we refer to this as method 2). The number of leaf nodes passing this test is equal to the number of feasible path systems.

We also compared the performance of our constraint model to a model where the 4-node condition was replaced by an LP formulation of the weight setting problem (we refer to this as method 3). This LP formulation is a necessary and sufficient condition for feasibility, and is applied at every search node, where it is able to take assigned node path variables into account. This should detect at least all inconsistencies that the 4-node condition detects, but will not give any propagation. The main drawback comparing to the 4-node condition is that it is much

Table 1. Results for full search trees

$\|V\|$	$\|E\|$	$\|F\|$	$\|C\|$	$\|N_c\|$	$\|T_c\|$	$\|N_l\|$	$\|T_l\|$
4	**6**	53	53	105	0.08	105	0.11
5	7	92	92	183	0.02	219	0.21
5	**10**	2668	2668	5485	0.29	6799	9.63
6	9	618	618	1243	0.09	2189	2.94
6	9	986	986	2089	0.16	3479	4.68
6	12	37906	38290	81161	4.48	131597	227.04
6	**15**	583201	589921	1251361	78.45	-	-
7	10	368	368	765	0.11	1597	2.77
7	10	1736	1736	3593	0.36	8821	16.30
7	14	369499	380680	807927	62.31	-	-
7	14	360545	361627	756453	55.33	-	-
8	12	40120	42880	97203	8.72	270305	851.32
8	12	30291	30311	63597	5.75	193081	579.18
9	13	42408	42408	89621	11.18	385963	1848.00

Table 2. Results for randomly generated solutions

network	$\|V\|$	$\|E\|$	f_r	f_d
bwin	10	45	17	20
cost266	37	57	13	20
di-yuan	11	42	11	20
france	25	45	16	20
germany50	50	88	9	7
newyork	16	49	12	20
nobel-eu	28	41	19	20
nobel-germany	17	26	20	20
nobel-us	14	21	16	20
norway	27	51	19	20
pdh	11	34	16	20
pioro40	40	89	4	0
polska	12	18	19	20
zib54	54	80	8	18

slower. There are faster LP-based conditions [5], but these are less well-suited for taking assigned node path variables at interior nodes into account. Thus, what is interesting to look at for this method, comparing to the 4-node condition (method 1), is not so much the runtime as the number of generated nodes.

One complication with this kind of testing is that the number of solutions grows very fast with network size. Therefore we have only been able to test rather small randomly generated networks, using the same generation method as in [3]. We only show results for the networks where we were able to determine the number of feasible solutions, using method 2. Results are presented in Table 1. The first two columns indicate the number of nodes and edges (in bold if the graph is complete). $|F|$ denotes the number of feasible path systems, and $|C|$ the number of path systems consistent with the 4-node condition. $|N_c|$ is the total size of the search tree and $|T_c|$ the total time in seconds used for the pure constraint model (method 1). The last two columns give the corresponding information for the LP-based method (method 3). Note that a binary search tree with $|F|$ feasible solutions will have at least $|F|-1$ internal nodes, so the total number of nodes will be at least $2|F|-1$. The timeout was set to one hour. The experiments were done on an Intel Pentium M 1.6 GHz Linux machine, using the constraint solver JaCoP [7] together with the LP system lp_solve.

Using only the 4-node condition produces a few leaf nodes that are consistent with all constraints but still not define feasible path systems. However, this is a small percentage in all cases. We have $|N_c|$ close to $2|F|-1$, which indicates very good propagation. Furthermore, the pure constraint method is very fast, being able to handle thousands of nodes every second. In comparison, the complete third method generates more internal nodes, due to the fact that it does not do propagation, and, as noted above, is a lot slower.

5.2 Randomly Generated Solutions

As seen above, generating the entire search tree restricts the size of network that we can investigate. The advantage is that we can measure the number of internal nodes, not only the percentage of consistent leaf nodes that are actually feasible. However, restricting to small networks may exclude larger infeasible structures that are difficult for the 4-node condition, which is local in nature, to detect.

For larger networks, we instead run our CP method until it finds a first consistent solution and then break. The method's default search heuristic chooses an edge path variable to branch on at each internal node, and then first tries the branch where the variable is assigned to one. There is some randomness (random tie-breaking) in the choice of which variable to branch on, so different runs of the method will find different first consistent solutions. This makes it possible to estimate the percentage of 4-node consistent path systems that are infeasible.

The default search heuristic is designed to try paths using few links first, since we are looking for path systems that lead to as little capacity consumption as possible. As a result, our random sampling of path systems will also favor such path systems, which is reasonable since these are the path systems that the condition is most likely to be applied on. However, we also test a more even random sampling, which is achieved by randomly choosing which branch to try first at each internal node: assigning the edge path variable to 0 or 1.

We tested both variants of the search heuristic on a number of network topologies from the SNDLib benchmark set [8]. Results are presented in Table 2. For each network, we performed 20 tests each for the two search heuristics. In column f_r is the number of those runs that produced a first consistent path system that was also feasible for the random branch ordering. Column f_d gives the same result for the default branch ordering.

In general, the default value ordering works better, and produces feasible solutions with high probability. This is probably a consequence of the default value ordering's preference for short simple paths, while the random variable ordering may produce more complex path systems. The notable exceptions to this pattern are networks 'germany50' and 'pioro40'. The size of the networks may explain their bad performance, but it is more unclear why the random value ordering works better than the default one.

6 Conclusions

We introduced a necessary condition for the feasibility of path systems in unique shortest path routing for undirected demands, and showed some of its properties. A CP model using this condition was described and experiments using this model shows that the condition is able to prune most infeasible path systems for small and mid-size networks.

Several directions exist for future work. A similar approach could be applied to the same problem with directed demands. One of the drawbacks of modelling path systems with node path variables is the large number of variables needed.

Possibly, techniques similar to column generation could be used in CP, introducing variables only if necessary. Finally, the linear inequality formulation of the condition could be tried with an MIP model for the problem.

Acknowledgements

The authors would like to thank Michał Pioro for helpful discussions during this work. The first author is funded by the Swedish National Graduate School in Computer Science (CUGS).

References

1. Pióro, M., Medhi, D.: Routing, Flow, and Capacity Design in Communication and Computer Networks. Morgan Kaufmann (2004)
2. Rossi, F., van Beek, P., Walsh, T., eds.: Handbook of Constraint Programming. Elsevier (2006)
3. Pettersson, M.P., Szymanek, R., Kuchcinski, K.: A CP-LP approach to network management in OSPF routing. In: Proceedings of the 2007 ACM Symposium on Applied Computing (SAC), Seoul, Korea, March 11-15, 2007, ACM (2007) to appear
4. Tomaszewski, A., Pióro, M., Dzida, M., Zagożdżon, M.: Optimization of administrative weights in IP networks using the branch-and-cut approach. In: Proceedings of the Second International Network Optimization Conference (INOC 2005), Lisbon. Volume 2. (March 2005) 393–400
5. Ben-Ameur, W., Gourdin, E.: Internet routing and related topology issues. SIAM J. Discrete Math. **17**(1) (2003) 18–49
6. Broström, P., Holmberg, K.: Determining the non-existence of a compatible OSPF metric. In: International Network Optimization Conference, INOC 2005. (2005) 106–113
7. Kuchcinski, K.: Constraints-driven scheduling and resource assignment. ACM Transactions on Design Automation of Electronic Systems (TODAES) **8**(3) (July 2003) 355–383
8. SNDlib 1.0–Survivable network design data library. `http://sndlib.zib.de` (2005)

Optimal Congestion Control with Multipath Routing Using TCP-FAST and a Variant of RIP

Enrique Mallada and Fernando Paganini

Universidad ORT
Montevideo, Uruguay

Abstract. This paper discusses an optimization-based approach for congestion control together with multipath routing in a TCP/IP network. In recent research we have shown how natural optimization problems for resource allocation can be solved in decentralized way across a network, by traffic sources adapting their rates and routers adapting their traffic splits, all using a common congestion measure. We present here a concrete implementation of such algorithms, based on queueing delay as congestion price. We use TCP-FAST for congestion control, and develop a multipath variant of the distance vector routing protocol RIP. We demonstrate through ns2-simulations the collective behavior of the system, in particular that it reaches the higher transfer rates available through multiple routes.

1 Introduction

The use of multiple paths between a source and destination of traffic is a natural choice to enhance the performance of a network, especially for bulk transfers that care mainly about the overall throughput. Flow could stream over many channel paths inside the network, adapting to whatever capacity is available at the moment. This ideal calls, of course, for a more active role for the network layer than that of current practice. In the prevailing situation, IP routing is insensitive to congestion and the only real-time control is done by the transport layer, designed to be single path; even if multiple TCP streams are used, there is limited control from the source as to how traffic travels inside the network. To fully explore the multipath options from the *source* (or an overlay at the edge of the network) as recently investigated in [7,4], is not scalable given the exponential number of end-to-end paths.

A scalable alternative requires participation of the routers. Two main arguments against involving routers in real-time adaptation are (i) to keep them simple, and (ii) that congestion-based adaptation leads to instabilities such as route flaps. Tackling the second one first, it is well recognized that the main reason for such flaps is the use of single path routing, in which large bulks of traffic are suddenly switched. For multipath routing, stable methods based on gradual adaptation of the split of traffic have been known for a long time [3]. A recent proposal along these lines is [5], which uses a heuristic rule based on backpressure signals to adapt routing splits. Other related work for wireless networks is [1,12].

In [9] we proposed a way to combine multipath adaptive routing with the congestion control of TCP. We showed how to formulate global resource allocation optimization problems, and how to design local, scalable laws at sources and routers to solve them. In particular, we were able to give theoretical proofs of global convergence for arbitrary network topologies. These results are briefly reviewed in Section 2. Key to the overall solution is the propagation of a universal "congestion price" across the network.

The purpose of this paper is to develop an implementation of the ideas in [9]. We take as congestion price the queueing delay, which TCP variants such as FAST [6] estimate and use for congestion control, and routers can locally measure. This brings us back to the first point raised above: how complicated would it be for an IP router to perform multipath routing, and to handle and propagate these congestion signals? In Section 3 we show that the multipath routing algorithms of [9] fit well with distance vector routing protocols such as RIP (see [10]); we thus develop an enhancement of RIP to perform the necessary tasks. In Section 4 we present simulation work in ns2 to demonstrate the feasibility and properties of this implementation. We use the ns2 distribution of TCP-FAST [2], and developed our variant on the ns2 version of RIP. Conclusions are given in Section 5.

2 Combined Congestion Control and Multipath Routing

This section summarizes the ideas of [9] as to how to combine methods of congestion control with multipath routing, and the theory relating these algorithms to optimization.

2.1 Flow Variables

Consider a set of nodes $\mathcal{N}$, indexed by i, j, connected by a set of directed links $\mathcal{L}$, each denoted by l or by (i,j). The network supports various flows indexed by $k \in \mathcal{K}$, between a source node $s(k)$ and a destination node $d(k)$, following possibly multiple paths. We introduce the following variables: x^k, external rate entering the network at the source; y_l^k, rate of flow k through link l; x_i^k, total rate of flow k coming into node i. These quantities are subject to the natural flow-balance constraints. The total flow on link l is denoted by y_l, and its capacity by c_l.

2.2 Multipath Routing

The router at node $i \in \mathcal{N}$ must decide on which of its outgoing links $(i,j) \in \mathcal{L}$ it will forward incoming packets with destination d. We introduce, for this purpose, routing fractions or " split ratios" $\alpha_{i,j}^d$ satisfying

$$\alpha_{i,j}^d \geq 0, \qquad \sum_{(i,j)\in\mathcal{L}} \alpha_{i,j}^d = 1.$$

This means the rates of flow k satisfy $y_{i,j}^k = \alpha_{i,j}^{d(k)} x_i^k$ for each $(i,j) \in \mathcal{L}$.

2.3 Congestion Measure

Rates and routes will be controlled in response to a common congestion measure (called "price"). The basic price p_l is a scalar variable that measures the congestion state of each link $l \in \mathcal{L}$, depending on its total traffic y_l. In our implementation of Section 3, p_l will be the delay at the queue of link l.

The key to a scalable solution is the ability to summarize in a simple variable the congestion state of a portion of the network, using current routing patterns. We define, for this purpose, the node prices q_i^d, $i \in \mathcal{N}$, each representing the average price of sending packets from node i to destination d. Node prices must thus satisfy the recursion

$$q_d^d = 0, \qquad q_i^d = \sum_{(i,j)\in\mathcal{L}} \alpha_{i,j}^d [p_{i,j} + q_j^d], \quad i \neq d. \tag{1}$$

At the source node of flow k, the node price summarizes the congestion cost of the network for this flow. We denote it by

$$q^k := q_{s(k)}^{d(k)}.$$

2.4 Control

There are two main things to control in the network, source rates and router split ratios, based on the common congestion measure.

Control of source rates is typically done via the congestion window; a more macroscopic flow model takes the form of a "demand function" $x^k = f^k(q^k)$, in which rate responds to the congestion price q^k. In the case of TCP-FAST, this function takes the form

$$x^k = \frac{K_k}{q^k};$$

this is equivalent to the source maximizing its "consumer surplus" $U_k(x^k) - q^k x^k$ for the "utility function" $U_k(x^k) = K_k \log(x^k)$.

Routers must update the split ratios $\alpha_{i,j}^d$, based on congestion information, obtained locally or from its neighbors. Rather than instantaneously choosing the single least congested route, which causes route flaps, an idea that goes back to [3] is to gradually shift traffic to less congested routes. We impose the following conditions on the vector of changes $\{\Delta\alpha_{i,j}^d\}$:

- $\{\Delta\alpha_{i,j}^d\}$ depends on current ratios $\{\alpha_{i,j}^d\}$ and congestion prices $\{p_{i,j} + q_j^d\}$.
- $\{\Delta\alpha_{i,j}^d\}$ is negatively correlated with the route prices, and maintains node balance:

$$\sum_{(i,j)\in\mathcal{L}} \Delta\alpha_{i,j}^d (p_{i,j} + q_j^d) \leq 0, \qquad \sum_{(i,j)\in\mathcal{L}} \Delta\alpha_{i,j}^d = 0.$$

- Equilibrium is only reached when all outgoing links in use have equal price, $q_i^d = p_{i,j} + q_j^d$, and the rest have $\alpha_{i,j}^d = 0$.

2.5 Optimization Interpretation

The above strategy for control of rates and routes can be interpreted in terms of distributed optimization. Indeed, in [9] the equilibrium points of such algorithms are shown to solve one of the following problems, which generalize those in [7,8] to the case of arbitrary multipath routing.

Problem 1 (WELFARE). Maximize $\sum_k U_k(x^k)$, subject to link capacity constraints $y_l \leq c_l$, and flow balance constraints.

Problem 2 (SURPLUS). Maximize $S := \sum_k U_k(x^k) - \sum_l \phi_l(y_l)$ subject to flow balance constraints.

Both these problems optimize aggregate utility of all sources, in the second case discounting a "traffic engineering cost". Which optimization applies depends mainly on the method used to generate link prices: Problem 2 applies to the case where the congestion measure is a marginal cost,

$$p_l = \phi_l'(y_l); \tag{2}$$

Problem 1 applies to a congestion measure satisfying the dynamics

$$\dot{p}_l = \gamma_l [y_l - c_l]^+_{p_l}. \tag{3}$$

In addition to an interpretation at equilibrium, dynamic properties of these algorithms are studied in [9]; it is shown that under certain assumptions of time-scale in the control, they converge globally to the optimal points.

Both of the models in (2-3) have been applied to queueing delay; the first (a static function of link rate) follows from queueing theory in steady state; the second from fluid-flow considerations. Regardless of these considerations, the main point is that taking queueing delay as our notion of congestion, the proposed methods for adaptation of routes and source rates have a rationale in terms of solving a global optimization.

3 Implementation

The formalism described in the previous section can be taken as a basis for more than one implementation, depending on the choice of the link congestion measure, the source utility function, and the method for sharing congestion information between routers and with traffic sources. In [9] we outlined some of the implementation issues in general terms, a discussion we continue here.

3.1 Discussion and Strategy

- A first point concerns the formation of node prices, which are used by routers to make decisions on traffic splits. Given link prices generated at each router, the corresponding node prices that satisfy (1) can be found iteratively: each node periodically updates q_i^d to the right-hand side of (1), based on announcements of neighboring nodes and its own link prices, and then announces its new price to its neighbors. Under the assumption of continued connectivity, it can be shown this recursion converges. The message passing is exactly the same as in distance-vector protocols such as RIP. The main change is that instead of taking hop-count as the default metric in routing announcements, we replace it by congestion price. Specifically, when router i announces to its neighbors it can reach destination d, it attaches as metric the corresponding congestion price q_i^d.

- Update of router split ratios: an implementation difficulty here, already identified in [3], refers of the possibility of generating routing loops during the transient phase (these disappear in equilibrium due to optimality). A blocking method was proposed in [3] to avoid this, and can be adapted here as follows: a loop can only occur if some packets are going "uphill" in price q_i^d, interpreted as a potential. This "improper" behavior cannot be completely banned without potentially leading to discontinuities in $\alpha_{i,j}^d$; however it can be flagged, and communicated to neighbors, warning them not to *start* routing new traffic in the direction of the improper-behaving node. In this way, starting from a loop-free configuration, this property is preserved.
- Communication of the price to the sources. A major point of discussion (see [11]) among congestion control implementations is whether to introduce *explicit* congestion signals between routers and sources, or to rely exclusively on implicit measures which can be estimated by sources. The latter alternative is usually favored for practical reasons of incremental deployment.

  In this paper we hit a middle ground on this issue. On one hand, we believe that explicit congestion control at the fastest time scale (a price in every packet) would be both burdensome to the routers, and not very useful in the multipath context. After all, for routers to find their correct node prices based on the RIP protocol takes some time, so there is no point in providing fast feedback of a quantity that may not have the correct value. So we will favor a congestion signal that sources can implicitly estimate, such as loss probability or queueing delay. In this way, sources that probe the network with large amounts of packets may be able to infer the current congestion measure faster than the time it takes the routers to become aware of it.

  Still, these implicit estimation methods may have biases and it is essential for the correct functioning of the overall system that, over the long run, sources and routers use compatible prices. Therefore we include, at the slower time-scale of RIP announcements, an explicit portion of message passing between sources and routers which sources can use to calibrate their estimation. This requires the IP layer of a source node to listen to the RIP announcements, and pass the corresponding price q^k up to its TCP layer which is doing the price estimation.
- When choosing an implicit measure of congestion, to be useful in this scenario it should respect the recursion (1). From an end-to-end perspective, the price q^k must equal the average price experienced by packets over their respective routes, according to their corresponding routing fractions. In [9] we outlined

how loss probability (or ECN marking probability) satisfied this requirement, to first order. Here we focus on queuing delay: if p_l is the delay of each link's queue, then q_i^d in (1) is the average or expected queueing delay experienced by a packet between node i and destination. So the price q^k at the source is the average queueing delay experienced by packets when probing all routes. What sources can explicitly measure is, however, not queueing delay but the average round-trip-time of their packets,

$$\overline{RTT} = \overline{D} + q^k,$$

where $\overline{D}$ is the average propagation/processing delay over all routes.

In single-path implementations such as TCP-FAST [6], the propagation delay is estimated through "BaseRTT", i.e. the minimum observed RTT, which assumes that the source has encountered an empty queue at least for one packet. This assumption can be criticized, especially for a source that starts transmitting on an already congested path, leading to biases and unfairness. In any event, this solution is not available for a multipath setting: if the paths have different propagation delays, the required "average BaseRTT" will be different from the minimum. The correct tracking of this quantity is one compelling reason for the use of some explicit prices from the IP layer at a slow time-scale, as described below.

3.2 Details and ns2 Implementation

Multipath Distance Vector Protocol. This protocol is based on the Bellman-Ford distance vector algorithm, and its most well-known implementation, RIP (see e.g. [10]). The protocol learns routes to an IP destination address from its own locally connected networks, and from routes received from neighboring routers. But, as compared to RIP, our multipath protocol does not discard a route if it has a shorter (or cheaper) alternative; rather, it maintains in its routing table all possible next hops for a given destination. Each row in the routing table is accompanied with its metric, $p_{i,j} + q_j^d$, where $p_{i,j}$ is the queueing delay of the link, measured as the link queue divided by its capacity, and q_j^d is the metric learned from the downstream router. Also, each row has a variable that keeps track of the routing fraction $\alpha_{i,j}^d$ using this outgoing link. Forwarding decisions are made by choosing a pseudorandom number between 0 and 1, and going through the list of next hops until the sum of the $\alpha_{i,j}^d$ exceeds that number.

When the algorithm starts, it learns the routes from directly connected destinations, and assigns them cost $q_j^d = 0$. Since these are the first routes to be learned, they are assigned $\alpha_{i,j}^d = 1$: all traffic for this destination will initially be routed through this path. Analogously, every time a new destination is discovered it is assigned $\alpha_{i,j}^d = 1$; on the other hand, new routes learned for an already known destination are assigned $\alpha_{i,j}^d = 0$.

Routing announcements of the form (destination, metric, flag) are sent from each node to its neighbors. These are sent every Δ_t^r seconds, or also asynchronously if the node has received a notification that changes its routing table or metric. The first two fields are similar to RIP's, the metric is the weighted average q_i^d from (1). The additional flag indicates whether this is a proper or improper route, used for blocking loops: a route is announced as improper if at least one of the next hops j with $\alpha_{i,j}^d > 0$ satisfies either (i) $q_j^d > q_i^d$, or (ii) node j's last announcement had the improper flag on. We also included a version of RIP's "poison reverse" method: if node i is sending all its traffic to node j ($\alpha_{i,j}^d = 1$), the announcement from i to j carries infinite metric. This helps avoid trivial loops in the initial stages of the algorithm.

Periodically, with a separate period Δ_t^a , the adaptation of $\alpha_{i,j}^d$ takes place at node i. We describe this procedure, denoting for simplicity $\pi_j := p_{i,j} + q_j^d$, the metric seen by the node for each of its outgoing links:

– Identify the minimum metric $\pi_{min} = \min_j \pi_j$.
– For links which are more expensive than the minimum, update

$$\alpha_{i,j}^d(t + \Delta_t^a) := \alpha_{i,j}^d(t) + \beta\Delta_t^a(\pi_{min} - \pi_j);$$

  β is a system parameter that controls the speed of adaptation.
– The sum of all the decrements in $\alpha_{i,j}^d$ above is compensated by distributed increments in the cheapest links, except those which are *blocked*; a node is blocked if has the improper flag on and is receiving no traffic.

Modifications to TCP-FAST. Associated with source nodes are TCP-FAST agents. These estimate average RTT and BaseRTT by running, for each received ACK, the updates

$$\overline{RTT} := (1 - a) * \overline{RTT} + a * currentRTT \tag{4}$$

$$BaseRTT := (1 - b) * BaseRTT + b(\overline{RTT} - q^k) \tag{5}$$

The RTT averaging equation (4) is standard: the parameter a can be made inversely proportional to the current window size. This makes the time constant of the filter to correspond to a certain number of RTTs. Equation (5) to estimate BaseRTT is modified from FAST, it is not based on the minimum RTT. Instead, we apply a lowpass filter, with parameter $b << a$ (so $BaseRTT$ evolves at a slower time-scale than $\overline{RTT}$), driven by the values ($\overline{RTT} - q^k$), where the prices q^k are computed by the routing agent. In ns2, every time the routing agent at a node computes a new price q_i^d, it checks whether there are any FastTCP agents at the same node with that destination. If so, it updates the q^k parameter used by that agent.

Another modification required on FastTCP is that, consistently with multi-path routing, it no longer makes sense to consider the ordering of packet arrivals in decisions about congestion control. In particular, the duplicate ACK feature should be removed, and RTT averaging should be performed on all packets, not just those which come in order. In our current implementation, we sidestep these issues through the following means: on the receiving end, the receiver sends as ACK the sequence number of the received packet, rather than the usual of stating the next expected packet. All these become valid ACKs and are used in the RTT computation by the FastTCP source. On the other hand, we emulated on the source's side the receptor logic, so that the rest of the protocol did not require changes.

4 Simulation Results

We present some results for the simple network topology shown in Figure 1. There are two sources at nodes 1 and 2, a common destination at node 4; both sources use TCP-FAST with parameter $K_1 = K_2 = 250$. This parameter represents the number of packets to be stored in network queues in equilibrium. There are two bottleneck links: (2,4) with capacity 37.5 Mbps, and (3,4) with capacity 25 Mbps; both have one-way propagation delay of 25ms. Links (1,2) and (1,3) have capacity of 100Mbps, and 50ms delay. Capacities and delays are the same in the reverse path. Packet size is 1040 bytes.

The following parameters were used in routers: $\beta = 0.5$, $\Delta_t^r = 500ms$, $\Delta_t^a = 500ms$. In TCP sources, we used $a^{-1} = 4 * cwnd$, $b^{-1} = 3000$ for the averaging of RTT and BaseRTT.

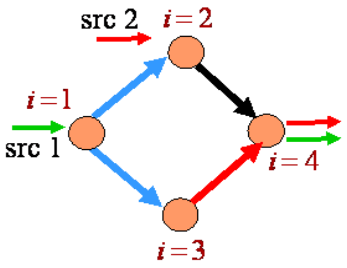


Fig. 1. Simulated network

The simulation results are depicted below. Figure 2 contains split ratios and metrics (prices) for nodes 1 and 3. Figure 3 contains queueing delays for bottleneck links (2,4) and (3,4), and the rates of both sources in packets/second.

In the initialization process, for random reasons all nodes (1,2,3) discover first the top route, via node 2 to destination node 4. In particular, node 3 sets its split ratios initially to route via node 1, the longest route, and blocks the use of link (3,4) for node 1. Therefore, initially all traffic from both sources travels on the top route and reaches the destination through link (2,4). For about 7 seven seconds, both TCP-FAST sources ramp up their rates, source 2 having an initial advantage due to its smaller RTT. At that time, link (2,4) saturates and the sources react, converging around 30 seconds to sharing the bottleneck fairly.

By 30 seconds, node 3 has completed its transfer of routing to link (3,4). This unblocks the bottom route for node 1, and allows for source 1 to increase its total rate, eventually filling the bottom link as well around 45 seconds. At the same time, this allows source 2 to send more traffic through link (2,4).

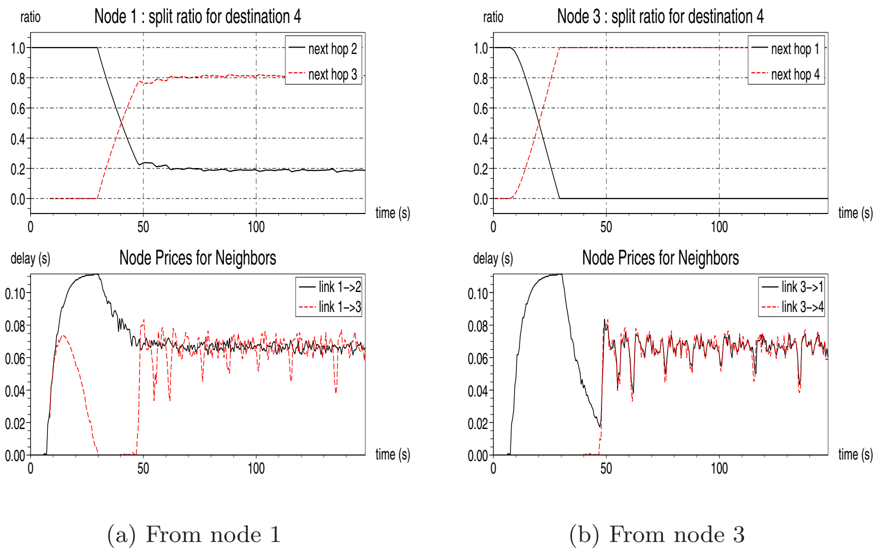


Fig. 2. Split ratios and prices for destination 4

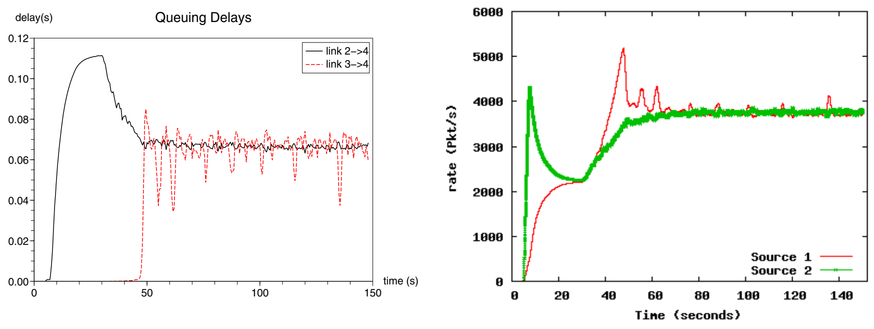


Fig. 3. Bottleneck queueing delays and source rates

After about 60 seconds, the system reaches an equilibrium with node 1 routing 20% of traffic from source 1 through the top path, and 80% through the bottom path. Node 2 routes all traffic from source 2 through link (2,4). Queueing delays (prices) at both bottlenecks equalize, so the sources with same demand curves equalize their rates to $(c_1 + c_2)/2 = 31.25$ Mbps or 3750 packets/second.

5 Conclusions

We have implemented distributed algorithms, at sources and network routers, which solve a distributed optimization problem, combining traffic engineering with congestion control as proposed in [9]. The implementation is based on queueing delay as a congestion price; routers measure local prices and exchange information with neighbors, following a multipath variant of a distance-vector routing protocol. Fast-TCP sources estimate this delay from their RTT measurements in real time, calibrating their propagation delay through periodic interactions with the IP layer. Our ns2 simulations over a simple network verify the expected behavior from the theory. Future work will involve more extensive tests and larger networks. Remaining challenges are partial deployment of this protocol and its compatibility with IP address summarizing.

Acknowledgment. This work was partially supported by grant FA9550-06-1-0511 of the US AFOSR, Latin American Initiative.

References

1. L. Chen, S. H. Low, M. Chiang and J.C. Doyle, "Cross-layer congestion control, routing and scheduling design in ad-hoc wireless networks", IEEE INFOCOM 2006.
2. T.Cui and L. Andrew, "FAST TCP simulator module for ns-2, version 1.1", available from `http://www.cubinlab.ee.mu.oz.au/ns2fasttcp`
3. R. G. Gallager, "A minimum delay routing algorithm using distributed computation", *IEEE Trans. on Communicactions*, Vol Com-25 (1), pp. 73-85, 1977.
4. H. Han, S. Shakkottai, C. V. Hollot, R. Srikant and D. Towsley, "Multi-Path TCP: A Joint Congestion Control and Routing Scheme to Exploit Path Diversity on the Internet", in *IEEE/ACM Trans. on Networking*, Vol. 14, Issue 6, Dec. 2006.
5. I. Gojmerac, T. Ziegler, F. Ricciato, P. Reichl, "Adaptive Multipath Routing for Dynamic Traffic Engineering", Proc. GLOBECOM'03, San Francisco, Nov. 2003.
6. C. Jin, D. X. Wei and S. H. Low, "FAST TCP: motivation, architecture, algorithms, performance"; *IEEE Infocom*, March 2004.
7. F. P. Kelly, A. Maulloo, and D. Tan, "Rate control for communication networks: Shadow prices, proportional fairness and stability", *Jour. Oper. Res. Society*, vol. 49(3), pp 237-252, 1998.
8. S. H. Low and D. E. Lapsley, "Optimization flow control, I: basic algorithm and convergence", *IEEE/ACM Trans. on Networking*, vol.7, no.6, pp 861-874, 1999.
9. F. Paganini, "Congestion control with adaptive multipath routing based on optimization", *Conf. on Information Sciences and Systems*, Princeton, NJ, Mar 2006.
10. L. L. Peterson and B. S. Davie, *Computer Networks: a Systems Approach*, Morgan-Kauffman, San Francisco, 2003.
11. S. Shalunov, L.Dunn, Y. Gu, S. Low, I. Rhee, S. Senger, B. Wydrowski, L. Xu, "Design Space for a Bulk Transport Tool", http://transport.internet2.edu/
12. Y. Xi and E. Yeh, "Node-Based Distributed Optimal Control of Wireless Networks", *Conf. on Information Sciences and Systems*, Princeton, NJ, Mar 2006.

Grid Brokering for Batch Allocation Using Indexes

Vandy Berten[1] and Bruno Gaujal[2]

[1] FNRS, Département d'Informatique
Université Libre de Bruxelles, Belgium
[2] INRIA and Lab. LIG (CNRS, INPG, UJF)
51, Av. J. Kunztmann, Montbonnot, France

Abstract. In this paper we show how dynamic brokering for batch allocation in grids based on bi-dimensional indices can be used in practice in computational grids, with or without knowing the sizes of the jobs. We provide a fast algorithm (with a quadratic complexity) which can be used off-line or even on-line to compute the index tables. We also report numerous simulations providing numerical evidence of the great efficiency of our index routing policy as well as its robustness with respect to parameter changes. The value of information is also assessed by comparing the performance of indexes when the sizes of the jobs are known and when they are not known.

1 Introduction

The aim of this paper is to propose an efficient but simple routing (or meta-scheduling, or brokering) strategy for computational grids having a centralized resource broker based architecture, such as EGEE (Enabling Grids for E-sciencE [1]), GridPP [2] or Grid 5000 [3]. The functioning of such a system is rather simple: a set of resources (clusters, ...) is available to a *resource broker* (RB). When a user wants some work to be executed, (s)he sends a parallel job to this resource broker, which is in charge to choose a resource, and to send (or route) the job to the selected resource. Two cases are studied in this paper. We consider the case where users announce the size of the job (*i.e.* the number of processes composing the job) so that the broker can take its decisions accordingly, and the case where the sizes of the jobs are not known *a priori* and follow some general distribution. A job never waits in the RB (except for the small time needed for choosing the resource), and is queued into the chosen servers immediately, where it will be scheduled thanks to classical well known cluster scheduling techniques.

The resource broker should not base its routing decisions on a static off-line strategy, since the input traffic is usually highly dynamic (the sizes and the arrival time of jobs are unknown and vary over a wide range of values). Therefore, most brokering strategies use the *current state* of the grid to allocate incoming jobs to resources. For instance, in EGEE [1], the routing of jobs may be based on the current *number of jobs* waiting in all the queues or on the *waiting times* of recent jobs having been processed.

Here, we model brokering problems in grids as Markov Decision Process (MDP) problems. While these problems are known to be P-space hard [4] in general, Whittle has shown that *index policies* are good candidates as sub-optimal solutions to these problems [5]. Whittle indices have been used successfully in [6] to control M/M/1 queues with breakdowns. This approach has also been extended in various directions in [7,8,9], for example.

The goal of this paper is to show that index policies provide a valid candidate to solve the brokering problems in grids which are seen as $G/M/k/B$ queues and that they can be used in practice because they very easy to compute and exhibit very good performances. The case of $M/M/k/B$ queues (with no batches) has been considered in full details in a long version of this work available as a research report [10]. Here we show how the approach can be used for batch arrivals. Most proofs which differ only slightly from the non-batch case are only sketched here.

The first objective of this paper is to design fast algorithms to compute the indices so that they can be used on-line (*i.e.* recomputed whenever major changes occur in the system). Unlike in most previous studies, the indexes that are used here are bi-dimensional functions: they depend of the size of the queue as well as the size of the incoming job. The naive approach (given in the paper) has a $O(eB^4K^2)$ complexity while our most advanced algorithm has a $O((e + \log(B))B^2K^2)$ complexity (here B is the buffer size , K is the maximal size of jobs and e is the required number of precision digits). This is done by exploiting several structural properties of our mathematical model (threshold optimality, monotony, univocal and convex functions) and the final complexity becomes sub-quadratic in the state space.

The second objective of this paper is to show the efficiency of our index policies, as a mathematically well founded alternative to intuitive policies such as "Join The Shortest Queue" (JSQ), for task allocation in grids. The numerical experiments provided in Section 4 show how well it behaves in terms of performance, and robustness. In all our experiments, the performance (average sojourn time) of our index based routing policy stays within 2% from the optimal policy and is always better than all classical policies (such as JSQ). Furthermore, our policy is very robust to parameter changes. For example, combining our index policies computed for loads 0.5 and 0.9 is enough to achieve very good performances (less than 2% loss) over the whole range of loads from 0 to 1. Also, a rough approximation of the job width distribution gives excellent results.

Finally, we assess the value of information, by comparing the performances of the multidimensional indexes used when the job sizes are known and of the one-dimensional indexes used for unknown job sizes.

2 Index Policies for Grids

In this paper a grid computing systems is seen as an heterogeneous set of independent clusters. All computing nodes in the same cluster are identical. In one cluster, all processes are queued in FIFO order into one finite buffer and are allocated to free nodes in an arbitrary fashion.

This is an approximation of the actual behavior of current grid computing systems. The main simplifications used here are the independence between all processes within one job (they usually are parallel processes and have dependencies) and the FIFO queuing policy in each cluster (they usually have priority and/or reservation features).

More formally, our system can be described as follows: a router gets a Poisson stream of jobs at rate of λ. It chooses one queue amongst N, and sends the job to the chosen queue, or reject it, if all queues are full.

A job is a set of independent processes, routed together through the same cluster, but scheduled independently on the cluster once they are in the cluster queue. The maximum number of processes in a job is K ($K \leq B$), and the probability for a job to be composed of k processes is denoted by P_k. The number of processes of a job is called the *job width*. The average job width ($\sum_{k=1}^{K} kP_k$) is denoted W.

Each cluster is seen as a queue of type $G/M/s_i/B_i/\text{FIFO}$ with service rate μ_i.

When the number of processes per job are not known at routing time, we need a new interpretation of the buffer capacity. If the number of processes in a queue i is higher or equal to its capacity B_i, a new job cannot be routed towards this queue. But one job can make the number of processes become greater than B_i. Once the number of processes is higher or equal to B_i, this number can only go down until $B_i - 1$ is reached.

In addition, the following notations will be used throughout the paper.

- x_i is the *queue i state*, or the number of processes currently present in the queue (waiting and running). $x_i \in \{0, \ldots, B_i + K - 1\}$.
- $x = \{x_1, \ldots, x_N\}$ is the *system state*.
- S is the *state space*: $S = \{0, \ldots, B_1 + K - 1\} \times \cdots \times \{0, \ldots, B_N + K - 1\}$.
- U is the *action space*, or the set of actions that the router can choose. An action is either i (if the queue i is chosen), or 0 (if the action is the rejection of a job). Rejection is only allowed when all queues are "full". Then, $U = \{1, \ldots, N\} \cup 0$ iff $x_i \geq B_i \ \forall i$.

In the following, we will focus on routing policies minimizing the expected discounted workload in infinite horizon[1]. This can be seen as a Markov Decision Problem in discrete time after uniformization by the constant $\Lambda = \lambda + \max_{i\in\{1,\ldots,N\}} s_i\mu_i$, and immediate cost $\sum_{i=1}^{N} c_i((x_i^{(m)} + \delta_{\{u_m(x^{(m)})=i\}}))$ where c_i is the unit cost in the i-th queue per customer.

As the cost is uniformly bounded over the state space, we will only consider time independent routing policies. A *routing policy* u is a function which says which action to take in each state. Then it is a function $u : S \to U$.

Computing the optimal policy boils down to solving a Bellman fixed point equation (see [11]). The problem here is that the size of the state space increases

[1] A similar question without discounting can also be considered. The approach used the rest of the paper also applies for long run average costs with minor adaptations. These extensions will not be discussed further in this paper.

exponentially with the number of queues. This problem, often called the "curse of dimensionality", is acute here. Indeed, the problem becomes too large to be handled by modern computers as soon as $N \geq 4$ (for queues of size 100).

This is why one needs to tackle the problem using a scalable approach.

2.1 Optimal Index Policies

Here is the way to construct a local cost function for each queue. We consider each cluster (*i.e.* a multi-server queue) in isolation and use a free parameter R, as a rejection cost. The first step is to construct optimal admission policy in each queue which is of threshold type. The optimal threshold is a function $\Theta_i(R)$ of the rejection cost. When the job sizes k are known, The optimal threshold also depends on k and is denoted $\Theta_i(R, k)$. In the following, we will mainly focus on the case where the job sizes are known. The equations in the unknown case are similar (by dropping the argument k).

Then, the local cost or *index* for this queue is the inverse function of $\Theta_i(R, k)$: $L_i(x_i, k) = \sup\{R | \Theta_i(R, k) \leq x_i\}$, and the policy is

$$u(x_1, \ldots, x_N, k) = argmin_{x_i < B_i}\{L_1(x_1, k), \ldots, L_N(x_N, k)\}$$

An arriving job of size k is then sent towards the queue which has the smallest current local cost. If the results of the argmin is empty, the incoming job is rejected.

In the following sections, devoted to the computation of $\Theta(R, k)$, we will focus on one specific queue. In order to simplify our notations, the subscript i of the queue will be dropped.

For one queue, the optimal control problem is to find the optimal control u (accept or reject each incoming customer) in order to minimize the long run discounted cost with initial state x_0, after uniformization by this total rate $\Lambda = \lambda + \max_i \mu_i s_i$. The α-discounted cost for the optimal policy with initial state x will be denoted $J^*(x, k)$.

Theorem 1. *The optimal policy minimizing the α-discounted cost in infinite horizon $J^*(x, k)$ for one $M^P/M/s/B$ queue (or one $M^P/M/s/\infty$ queue) is of threshold type.*

This result is rather classical (see for example [11]) for the case where the size of the jobs are unknown. In the case where $J^*(x, k)$ actually depends on k, the proof is similar to the unknown case, and uses induction on the number of steps of the policy iteration. Actually, we will see later (in Lemma 2) how the thresholds for the infinite and the finite cases are related.

2.2 Computing the Optimal Threshold

The objective of this section is to design an algorithm finding the optimal threshold policy in a $M^P/M/s/B$ queue (where s is the number of servers, and B is the queue size). When the size of an arriving job is known, a policy $u_{\theta(k)}$ with *threshold* $\theta(k) \in \{0, \ldots, B\}$ is defined as follows. When a job of size k arrives while x processes are currently present in the system (x is the *queue state*),

– if $x < \theta(k)$, the new job is accepted, and every processes of this job will cost c per time step (time between two events) spent in the system,
– otherwise ($x \geq \theta(k)$) the job is rejected, and costs kR (only once).

Bellman's equation for the α-discounted cost of one queue with rejection, can be adapted for threshold policy u_θ. In the unknown size case,

$$\begin{aligned} J(x) = \alpha\Big(& \lambda' \textstyle\sum_k P_k J(x + k\delta_{x<\theta}) + \mu' \min\{x,s\} J(x-1) \\ & +(1 - \lambda' - \mu' \min\{x,s\}) J(x)\Big) + cx + \lambda' RW\delta_{x\geq\theta}, \end{aligned} \tag{1}$$

where $J(x)$ is the infinite horizon α-discounted cost starting with x jobs in the queue of policy u_θ. When job sizes are known, we get the cost of $u_{\theta(k)}$:

$$\begin{aligned} J(x,k) = \alpha\Big(& \lambda' \textstyle\sum_{k'=1}^{K} P_{k'} J(x + k\delta_{x<\theta(k)}, k') + \mu' \min\{x,s\} J(x-1,k) \\ & +(1 - \lambda' - \mu' \min\{x,s\}) J(x,k)\Big) + cx + \lambda' Rk\delta_{x\geq\theta(k)}. \end{aligned} \tag{2}$$

Equations (1) and (2) can easily be rewritten in a matrix form (For $J(x,k)$, the multidimensional variable is converted into one by setting $X = x \cdot K + k - 1$). Let F and S be appropriate matrices. Then, Equations (1) and (2)can now be written as:

$$J = \alpha F J + S. \tag{3}$$

Finding the threshold θ or $\theta(k)$ which gives the smallest cost corresponds to looking for a θ solution of the following systems, which is the same as Equations (1) and (2) with a min on the right hand side: $J = \min_{\theta\in\{0,\ldots,B\}}(\alpha F J + S)$ and $J = \min_{\theta\in\{0,\ldots,B\}^{\{0,\ldots K\}}}(\alpha F J + S)$, respectively.

We will call $\Theta(R)$ (resp. $\Theta(R,k)$) the function which gives the optimal threshold for a rejection cost R (resp. for a rejection cost R and for jobs of size k), assuming that other parameters are constant.

Using policy iteration, one can compute the optimal policy and its cost, when the job sizes are known. When the job sizes are not known, one only needs to remove the argument k everywhere.

The worse case complexity is $O(KB^3)$ (because bounds are known for $\Theta(R,k)$).

This allows us to compute the function $\Theta(R,k)$ for all k from $\mathbb{R}^+$ into $\{0,\ldots,B\}$. This function will be the base of our index $\mathtt{I}(x,k)$,which goes from $\{0,\ldots,B\} \times \{0,\ldots K\}$ into $\mathbb{R}^+$. $\mathtt{I}(x,k)$ is then a bi-dimensional array. $\mathtt{I}$ is defined as the inverse of Θ, or, more exactly, $\mathtt{I}(x,k)$ gives the larger rejection cost R such as $\Theta(R,k) \leq x$. Or, $\forall k, \quad \mathtt{I}(x,k) = \sup\{R | \Theta(R,k) \leq x\}$.

3 Algorithmic Improvements and Complexity

In this section, several improvements of the algorithm used to compute Θ (and thus $\mathtt{I}$) are explained. They are mostly based on Lemmas 1 and 2. They involve both algebraic simplifications and reductions of the problem size. The complexity of our final algorithm is quadratic on the state space. Other techniques (not based on the policy iteration) could be used to compute the optimal index. One possibility is to use the average cost equation and its derivatives, as done in [6]. All our attempts have proved to be numerically unstable for bi-dimensional indexes, which makes this approach unusable in practice. Other techniques based on polytope problems [7] also have a high complexity for bi-dimensional indexes.

Here, the following lemma is helpful to bound the initial search for thresholds.

Lemma 1. *Under the foregoing notations, the functions $\Theta(R,k)$ and $\mathtt{I}(x,k)$ have the following properties:*
i- $\Theta(R,k)$ and $\mathtt{I}(x,k)$ are non-decreasing in both parameters.
ii-For all $R \geq \frac{c}{1-\alpha}$, and all k, $\Theta(R,k) = B$.

Proof. (sketch) Point i is a direct consequence of the structure of the cost function. As for point ii, it is direct consequence of the fact that $J^*(x+k,k') - J^*(x,k) \leq k\frac{c}{1-\alpha}$, uniformly over all values of k.

Algorithm 1 computes the threshold when the job sizes are known.

Algorithm 1. Computes the best threshold $\Theta(R,.)$ for a rejection cost R

```
Data: $R$
Result: Best thresholds for this $R$ ($\Theta(R)$)
$\theta^* \leftarrow$ first estimation; // $\theta$, $\theta^*$ and $\theta'$ are vectors
repeat
    $\theta \leftarrow \theta^*$, $gain^* \leftarrow 0$;
    Find $J_\theta$, solution of $LJ = R$;
    foreach $k \in [1, \dots, K]$ do
        foreach $\theta'(k) \in \{0, \dots, B-k+1\} \setminus \theta(k)$ do
            foreach $x \in [0, \dots, B]$ do
                if $J_{\theta,\theta'}(x,k) > J_\theta(x,k)$ then
                    next $\theta'(k)$;
                else
                    $gain += J_\theta(x,k) - J_{\theta,\theta'}(x,k)$;
            if $gain > gain^*$ then
                $\theta^*(k) = \theta'(k)$;
                $gain^* = gain$;
until $gain^* < \epsilon$ ;
return $\theta^*$;
```

Lemma 2. *Let θ^* be the optimal threshold for a system with a queue size of $B > \theta^*$. Then, θ^* is the optimal threshold for any identical system with queue size $B' > \theta^*$ and B' is the optimal threshold for any identical system with queue size $B' \leq \theta^*$.*

This is a rather direct corollary of the form of the functionnal J used in the proof of Theorem 1.

This result will greatly help us to further improve the complexity of linear solving calls (solving of J_θ), which represents the main cost of our algorithm. If $\Theta(R,k)$ has been computed and if $R' < R$, then $\Theta(R',k) \leq \Theta(R,k)$. We can then compute $\Theta(R',k)$ as if the buffer size were $B' = \Theta(R,k)$, which reduces the search space for the new threshold. The same can be done for k. This method strongly reduces the time spent in solving $J_\theta(x,k)$. Indeed, solving $J_\theta(x,k)$ goes from a problem in $O((B+K)\cdot K)$ to $O((B'+K)\cdot K)$.

In the dichotomy algorithm we will present below, we can show that in a large majority of cases, the value of $\Theta(R,k)$ can only take 2 consecutive values, we can then use B' equal to the largest one.

3.1 Computing the Index Function

In order to compute the index vector $\mathtt{I}(x,k)$ with precision ϵ, one need to compute $\Theta(R,k)$ for every R such that $\Theta(R,k) = \Theta(R+\epsilon,k) - 1$. Each such R will give us the corresponding value of the index ($\mathtt{I}(\Theta(R,k)) = R$). The algorithm will be composed of two parts:

1. First, for each k, find (at least) one point on each step of $\Theta(R,k)$, meaning that $\forall x \in \{0,...,B\}$, we have a value R_x such as $\Theta(R_x,k) = x$. This is done by dichotomy. While we have R_1 and R_2 with $R_1 < R_2$ and $\Theta(R_2,k) - \Theta(R_1,k) > 2$, for which we do not have any point between them, we use Θ for some value between R_1 and R_2, for instance $\frac{R_1+R_2}{2}$.
2. Then a second dichotomy is used to get the values of the jumps of Θ. For every couple R_1, R_2 found at the previous step such as $\Theta(R_1,k)+1 = \Theta(R_2,k)$, we search for a value $R_1 < R_m < R_2$ such that $\Theta(R_m,k)+1 = \Theta(R_m+\epsilon,k)$.

Theorem 2. *Computing the index table* $\mathtt{I}$ *with precision* ϵ *can be done in* $O((B+K)K^2(eB+\log B))$ *in the worst case, where* $e = -log(\epsilon)$.

Proof. (*sketch*) The first phase of the algorithm needs to identify $B+1$ values of R, one for each possible threshold.

For given R and k, computing $\Theta(R,k)$ can be done by using a dichotomy over the values of θ in the set $\{0,\cdots,B\}$. The cost for each θ corresponds to solving a "quasi triangular" system of size smaller than $B+K$ and has a known solution in $O((B+K)\cdot K)$ so that the complexity to get $\Theta(R,k)$ is $O((B+K)\cdot K^2 \log B)$. Since B values have to be computed, the overall complexity of phase one is $O((B+K)K^2 \log B)$.

As for phase 2, the dichotomy imposes to compute less than eB values $\Theta(R)$. Each of them can be done in time $O((B+K)K)$ because only two values for $\Theta(R)$ are possible at this point (see the previous comments).

Hence, the overall complexity is $O(eB(B+K)K^2 + (B+K)K^2 \log B)$.

4 Numerical Experiments

In this section, we will present some simulation results where we compare several routing strategies. We compare our index strategy with some classical strategies.

The input rate varies between 0 and 100% of the total service capacity. The plots display the ratio between the *average sojourn time* in the stationary regime for a given strategy[2]. As stated in the previous section, several strategies have been compared. Most of them are already available in EGEE. Here are the names we use in the plots.

`Random` : Bernoulli routing (weight: $\mu_i s_i$)	`JSQ` (Join the Shortest Queue): $\mathtt{I}_i(x) = x$
`JSQ2` : $\mathtt{I}_i(x) = \min(0, x - s_i)$	`JSQ-mu` : $\mathtt{I}_i(x) = x/(\mu_i s_i)$
`JSQ2-mu` : $\mathtt{I}_i(x) = \min(0, x - s_i)/(\mu_i s_i)$	`Batch` : "our" strategy.
`JSW` (Shortest Waiting time): $\forall x \leq s_i,\ \mathtt{I}_i(x) = 1/\mu_i$ and $\forall x > s_i,\ \mathtt{I}_i(x) = \dfrac{x+1}{\mu_i s_i}$.	

Each curve required between 8 and 10 hours of computations. Therefore, each plot below required approximately 3 days of computation at 3.4 GHz. This was however highly parallelizable, which allowed us to obtain a plot in a few hours on a cluster. All our simulation were performed on the Grid5000 platform [3] giving a smooth and easy access to several thousands of CPU spread across France.

4.1 Impact of Information

In order to assess the value of information, we have compared the performance of our policies when the job sizes are known and when they are not known. The gain for knowing the job sizes remains uniformly below 1 %, as long as K the bound on job sizes is smaller than $B/10$. When K grows, the gap increases as expected and becomes more sensible (up to 10 %). However, the computation burden of computing bi-dimensional indexes increases sharply (several weeks over a large cluster). In the following, all the experiments are run under the assumption that the job sizes are unknown at routing time.

4.2 Impact of Job Width Distribution

We study here the impact of the job width distribution on the performance. As a simple example, we have chosen a platform with 4 queues, with various sizes. We have tried 3 job width distributions: firstly, a simple sequential distribution, meaning with one-process jobs (Fig. 1, left). Then, a distribution where the job widths are power of 2 up to 16 (1, 2, 4, 8, and 16), with decreasing probability (Fig. 1, middle). And finally, we use a real job width distribution, extracted from the DAS2 workload from the Feitelson's Parallel Workloads Archive [12] (Fig. 1, right).

We mainly notice that even if the comparison between metrics greatly varies, the gain of `Batch` over other strategies is still between 20 and 40% in some configurations (except `Random` which is worse). Moreover, a wider distribution seems to

[2] Notice that when the discounting factor is close to one and $c = 1$, this is very close to the average discounted cost.

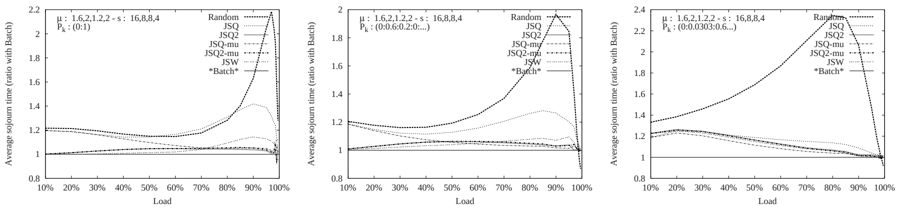


Fig. 1. We show the same platform with several job width distributions. On the first plot, we only have sequential jobs (non batch). On the last, we use job width distribution extracted from a real workload.

conduct to a larger gain, mainly at low load. On the right plot of Fig. 1, even the best strategy is 20% slower than `Batch` at low load. This is of course not surprising, as `Batch` is the only strategy taking the job width distribution into account.

4.3 Robustness on Load Variations

The main practical difficulty induces by our index strategy is that it depends on the input rate: in real systems, the input rate can vary which would require to obtain an estimation of this new load, and to recompute our indices. This leads to the following question: how robust is index routing with respect to to the input rate variation ? Does an index computed for a load of 50% perform efficiently if the actual load is 20% or 80% ?

We show on Figure 2 the behavior of a system where three different indices are used. The first index (`Batch`) is the classical one, which means that this index is computed according to the actual load. The second index (resp. `Batch.4` and `Batch.15` for left and right plots of Fig. 2) is an index computed for a load

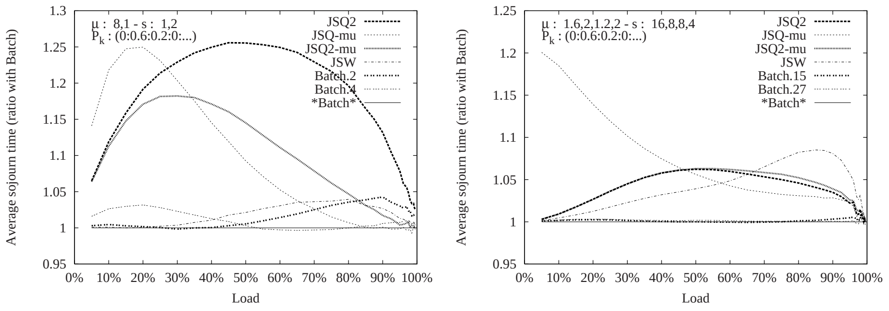


Fig. 2. This plot shows this impact of performance if the load used in the indices computations is not known, or not precisely known. Theses plots show that, depending of the configuration, a incorrect knowledge of the load could have either a little or even no impact at all.

of $\sim$ 50%, and used for every load between 0 and 100%. The third one (resp `Batch.2` and `Batch.27`) is similar, for a load of $\sim$ 90%. The number following the dot in the notation is the input rate (λ) used for the index computation.

On the first plot, it appears that the 50%-index (resp. 90%-index) is still better or equivalent than any other classical strategies for a load varying between 0 and 70% (resp. 40% and 100%). This shows that our indices are really robust, and that they do not require to known the exact input rate.

In the second plot, the robustness is much more important: the load (λ) used for the computation of index seems to be of little impact; whatever the parameter λ, `Batch` is better than any other strategy.

A strategy where two sets of indices $\mathcal{I}_1 = \{\mathtt{I}_1, \ldots, \mathtt{I}_N\}$ and $\mathcal{I}_2 = \{\mathtt{I}'_1, \ldots, \mathtt{I}'_N\}$ are computed, and where the resource broker uses $\mathcal{I}_1$ or $\mathcal{I}_2$ according to an estimation of the actual load would be very efficient in all cases.

4.4 Robustness on Job Width Distribution

In order to estimate the impact of the correctness of the job distribution knowledge, we show two scenarii in Fig. 3 in which we add two curves (`Batch.geom` and `Batch.exp`) for which the index was computed assuming the job width distribution was respectively geometrical between 1 and K, and geometrical only for powers of two (other values have a null probability). For the two plots, we use the same architecture (4 queues), but the job width distribution is different: the distribution extracted from Feitelson's archive on the left plot, with $K = 64$, and a simple distribution with $K = 8$ in the right plot. Notice that in both cases, the "powers of 2" distribution is a better approximation of the actual distribution.

On the first plot, the index using geometrical approximation is rather far from the `Batch` curve, especially for low loads, but is still better that any other strategy. The strategy based on "powers of two" approximation is almost indistinguishable from `Batch`. On the second plot, both approximations are almost indistinguishable from `Batch`. This shows that our policy is also very robust to the job width distribution.

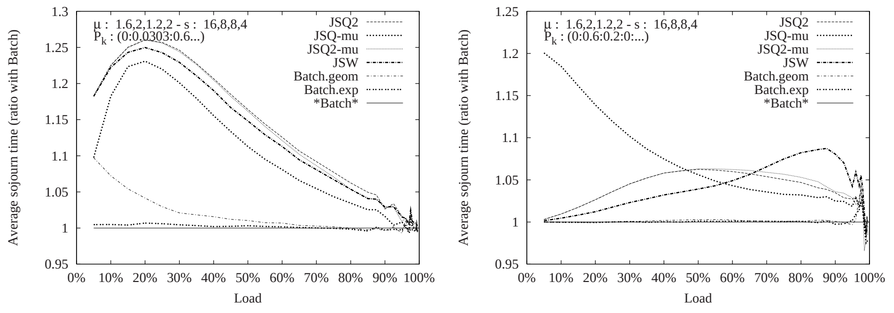


Fig. 3. Situation where the indices were computed with an approximation of the job width distribution. We observe very good performances, even for rough approximations.

References

1. EGEE: Grids for E-sciencE. [Online]. Available: http://www.eu-egee.org
2. GridPP. [Online]. Available: http://www.gridpp.ac.uk
3. Grid 5000, a large scale nation wide infrastructure for grid research. [Online]. Available: https://www.grid5000.fr/
4. C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of optimal queueing network control," *Math. Oper. Res.*, vol. 24, pp. 293–305, 1999.
5. P. Whittle, *A celebration of applied probability*, j. gani ed. J. Appl. Probab. Spec., 1988, vol. 25A, ch. Restless bandits: activity allocation in a changing world, pp. 287–298.
6. J. Palmer and I. Mitrani, "Optimal and heuristic policies for dynamic server allocation," *Journal of Parallel and Distributed Computing*, vol. 65, no. 10, pp. 1204–1211, 2005, special issue: Design and Performance of Networks for Super-, Cluster-, and Grid-Computing (Part I).
7. J. Niño-Mora, "Dynamic allocation indices for restless projects and queueing admission control: a polyhedral approach," *Mathematical Programming, Series A*, vol. 93, no. 3, pp. 361–413, 2002.
8. R. R. Weber and G. Weiss, "On an index policy for restless bandits," *Journal of Applied Probability*, vol. 27, pp. 637–648, 1990.
9. P. Ansell, K. D. Glazebrook, J. Nino-Mora, and M. O'Keeffe, "Whittle's index policy for a multi-class queueing system with convex holding costs," *Mathematical Methods of Operational Research*, vol. 57, pp. 21–39, 2003.
10. V. Berten and B. Gaujal, "Index routing for task allocation in grids," INRIA, Tech. Rep. RR-5892, 2006.
11. M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. New York: Wiley, 1994.
12. H. Li, D. Groep, and L. Wolters, "Workload characteristics of a multi-cluster supercomputer," in *Job Scheduling Strategies for Parallel Processing*, D. G. Feitelson, L. Rudolph, and U. Schwiegelshohn, Eds. Springer Verlag, 2004, pp. 176–193, lect. Notes Comput. Sci. vol. 3277.

Load Shared Sequential Routing in MPLS Networks: System and User Optimal Solutions

Gilles Brunet[1], Fariba Heidari[2], and Lorne G. Mason[2]

[1] Videotron Ltd., Montreal, QC, Canada
[2] Department of Electrical and Computer Engineering, McGill University, Montreal, QC, Canada
Gilles.Brunet@videotron.com, fariba.heidari@mail.mcgill.ca, lorne.mason@mcgill.ca

Abstract. Recently Gerald Ash has shown through case studies that event dependent routing is attractive in large scale multi-service MPLS networks. In this paper, we consider the application of Load Shared Sequential Routing (LSSR) in MPLS networks where the load sharing factors are updated using reinforcement learning techniques. We present algorithms based on learning automata techniques for optimizing the load sharing factors both from the user equilibrium and system optimum perspectives. To overcome the computationally expensive gradient evaluation associated with the Kuhn-Tucker conditions of the system optimum problem, we derive a computationally efficient method employing shadow prices. The proposed method for calculating the user equilibrium solution represents a computationally efficient alternative to discrete event simulation. Numerical results are presented for the performance comparison of the LSSR model with the user equilibrium and the system optimum load sharing factors in some example network topologies and traffic demands.

1 Introduction

In the early days of packet switching much attention was given to the routing problem. See [1] for an early survey of routing algorithms where an application of learning techniques to packet routing in data networks was considered. The event dependent routing method described in reference [1] can be applied to either datagram or virtual circuit data networks, however in the case of datagram networks the received packets can be miss ordered. With the appearance of the Internet, destination based IP routing was widely adopted for reasons of scalability and stability in spite of the fact that destination based routing gives the user little control over how his/her traffic is routed. This in turn means that traffic may be routed over congested links (paths) while at the same time alternative less congested paths are available.

The need for better control of traffic routing, also referred to as “traffic engineering”, gave rise to the MPLS standard. Multi Protocol Label Switching (MPLS) is a connection oriented framework proposed by the IETF to improve traffic engineering, congestion management and QoS provisioning in traditional IP networks [2]. In this framework, constraint-based routing and label swapping replaces the hop-by-hop destination-based routing mechanism used in traditional IP networks. With MPLS, route selection can employ either hop by hop routing or explicit routing. In the explicit routing method, a single Label Switching Router (LSR), usually the ingress LSR, specifies all (or some of) the hops in the Label Switched Path (LSP). Explicit routing gives the designer the ability to control the traffic load distribution in the network.

The algorithms proposed for LSP routing in MPLS networks are mostly state dependent ([3]). In state dependent routing, the information about the status of the network is flooded through the network and routing tables are updated using this information. Event dependent routing algorithms, on the other hand, use the observed events to update their knowledge about the status of the network. Different event dependent routing schemes have been proposed and successfully used in TDM networks. Reference [4] presents an event dependent routing scheme for destination-based routing and shows the convergence with probability one of the proposed algorithm to the set of approximate Cesaro-Wardrop equilibria. An application of event dependent routing schemes in the MPLS networks has been presented in [5, 6]. To the best of our knowledge, the performance of the routing scheme studied in [5, 6] can only be derived from discrete event simulation and there is no analytical approach for evaluating the performance of the algorithm. In reference [7], we present an alternative event dependent routing scheme with the application to explicit source routing in MPLS networks. The proposed algorithm is based on the Load Shared Sequential Routing (LSSR) where load sharing factors are updated using reinforcement learning techniques.

In this paper, we study the load share optimization problem both from user and system optimization perspectives in the LSSR model and give a computationally efficient method for solving these problems. The solution approach uses a recursion, governing the expected behavior of an ϵ-optimal learning automata, to converge to the point where the Kuhn-Tucker conditions of the optimization problem are satisfied. The application of learning automata techniques in solving the load share optimization problem for the single class circuit-switched networks with fully connected topology and 2-hop alternate paths has been studied in [8]. The solution approach proposed in this paper can be used in the multi-rate traffic case with general network topology where some links may be shared among two or more alternate paths. Numerical results are presented comparing network blocking probabilities obtained from the user equilibrium and the system optimum load sharing factors in some example network topologies and traffic demands.

2 Methodology

This section briefly reviews the methodology and the algorithms that are used in this paper to compute the user equilibrium and the system optimum load sharing factors when LSSR is applied to route Label Switched Paths (LSPs) in a multi-rate MPLS network.

Consider a learning automaton with K actions and the following updating scheme [9]:

$$p_j(t+1) = \begin{cases} p_j(t) + G[\delta_{ji} - p_j(t)]x(t) & j = 1,..,K-1 \\ 1 - \sum_{i=1}^{K-1} p_i(t+1) & j = K \end{cases} \tag{1}$$

where at time t, the ith action is selected; δ_{ji}, is Kronecher delta function and $x(t) \in [0,1]$ is the reward associated with the selected action. In the context of this paper, a reward is associated with a completed call attempt. In addition to being ϵ-optimal [10], when the covariance between each pair (p_i, p_j) ($\sigma^2_{p_i p_j} = E[p_i p_j] - E[p_i]E[p_j]$) is negligible, this learning scheme has the following expected behavior:

$$E[p_j(t+1)] \approx \begin{cases} E[p_j(t)][1 + G(s_j(t) - \sum_{k=1}^{K} s_k(t)E[p_k(t)])] & j = 1,...,K-1 \\ 1 - \sum_{j=1}^{K-1} E[p_j(t+1)] & j = K \end{cases} \tag{2}$$

where,

$$s_i(t) = E[x(t)|a(t) = a_i].$$

This recursion governing the expected behavior will be used in solving optimization problems described in the next sections. This methodology has been previously employed to compute the user equilibrium and the system optimum routing solutions in datagram networks [1]. It was shown that the recursion given in Equation (2), has fixed points which are in one-to-one correspondence with stable user equilibrium solutions, when the datagram network is modeled as in Gallager's classic paper [11]. In references [9] and [10] the same recursion was derived to approximate the expected behavior of the action probabilities of the cross-correlation algorithm of Equation (1) in a stationary environment under slow learning conditions. The same cross-correlation learning algorithm and the recursion governing its expected behavior under slow learning conditions were applied to a variety of routing and flow control problems in data and ATM networks in a series of papers [12], [13] and [14]. Recently Alanyali has provided an analysis of the behavior of distributed learning algorithms controlling a Markov process in [15].

The present paper differs from the results previously reported in that the network model used here is different, namely a general mesh topology supporting multi-rate LSPs. The multi-rate performance model of Greenberg and Srikant [16] is used where the LSPs are characterized by their Effective Bandwidth [17].

3 RL-Based Load Shared Sequential Routing

Load Shared Sequential Routing (LSSR) randomly partitions the class s traffic load associated with origin 'o' and destination 'd' ($\lambda^{o,d,s}$) into n sub-streams using the set of load sharing factors (sub-stream selection probabilities), $\{\alpha_1^{o,d,s}, ..., \alpha_k^{o,d,s}\}$. Each sub-stream is then offered to a route tree which consists of one or more alternate paths. The alternate paths of each route tree are

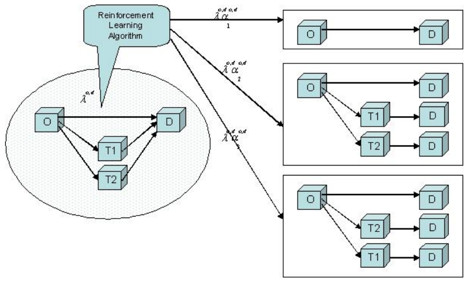


Fig. 1. LSSR Model

tried sequentially. If there is not enough bandwidth available on at least one link of one path, the request is forwarded to the next alternate path. This process is repeated until all alternate paths in the route tree have been tried sequentially. If all paths have been tried unsuccessfully, the request is lost and rejected from the network. A pictorial representation of the LSSR model is provided in Figure 1.

The LSSR model imposes no restriction on the load sharing factors other than non-negativity and

$$\sum_k \alpha_k^{o,d,s} = 1. \tag{3}$$

3.1 Load Shared Sequential Routing, User Equilibrium Solution

User equilibrium can be explained in terms of Wardrop equilibrium [18]. In the LSSR context, let $\lambda^{o,d,s}$ represent the total class s traffic load between origin 'o' and destination 'd' and let $L(\underline{\alpha}^{o,d,s})$ be the cost of allocating the traffic $\lambda^{o,d,s}$ according to the load sharing factors $\underline{\alpha}^{o,d,s}$. The set of load sharing factors $\underline{\alpha}$ is at the Wardrop equilibrium if for each $\alpha_i^{o,d,s}, \alpha_j^{o,d,s} > 0$, we have $L_i^{o,d,s} = L_j^{o,d,s}$ and that if there exits a route tree with $\alpha_\ell^{o,d,s} = 0$, then $L_i^{o,d,s} \leq L_\ell^{o,d,s}$. The load sharing factors $(\underline{\alpha})$ at the Wardrop Equilibrium are the solution to the following minimization problem:

$$\min \overline{Z(\underline{\alpha})} = \sum_{o,d,s,k} \overline{L_k^{o,d,s}} \tag{4}$$

$$\text{subject to} \qquad \sum_{k=1}^{K_{o,d,s}} \alpha_k^{o,d,s} = 1 \qquad\qquad (v^{o,d,s}) \tag{5}$$

$$\alpha_k^{o,d,s} \geq 0 \qquad\qquad (u_k^{o,d,s}), \tag{6}$$

where $u_k^{o,d,s}$ and $v^{o,d,s}$ are Lagrangian multipliers and $\overline{L_k^{o,d,s}}$ is defined as:

$$\overline{L_k^{o,d,s}} = \int_0^{\alpha_k^{o,d,s}\lambda^{o,d,s}} L_k^{o,d,s} d\alpha. \tag{7}$$

From the Kuhn-Tucker conditions we have

$$L_k^{o,d,s} = -\frac{v^{o,d,s}}{\lambda^{o,d,s}} \qquad \forall \quad \alpha_k^{o,d,s} > 0. \tag{8}$$

From this last condition, the user equilibrium solution is one that equalizes the cost on all the route trees on which traffic is offered ($\alpha_k^{o,d,s} > 0$).

3.2 Load Shared Sequential Routing, System Optimization Problem Formulation

In this case, the optimization problem has the following form:

$$\min Z(\underline{\alpha}) = \sum_{o,d,s,k} \lambda^{o,d,s} \alpha_k^{o,d,s} L_k^{o,d,s} \tag{9}$$

$$\text{subject to} \sum_{k=1}^{K_{o,d,s}} \alpha_k^{o,d,s} = 1 \qquad (v^{o,d,s}) \tag{10}$$

$$\alpha_k^{o,d,s} \geq 0 \qquad (u_k^{o,d,s}). \tag{11}$$

From the Kuhn-Tucker (KKT) conditions we have

$$\frac{\partial Z}{\partial \alpha_k^{o,d,s}} = -v^{o,d,s} \qquad \forall \quad \alpha_k^{o,d,s} > 0, \tag{12}$$

which means that at the system optimum solution, for each (o, d, s), the partial derivatives of network cost with respect to load sharing factors on all the route trees with $\alpha_k^{o,d,s} > 0$ are equal.

3.3 The Application of Learning Automata in Solving User and System Optimization Problems

Assume there is a cross-correlation learning automata engine, $A^{o,d,s}$, associated with each (o, d, s) with the set of actions of $A^{o,d,s}$ being the set of route trees available for routing bandwidth requests of class s between pair (o, d). Let $\alpha_k^{o,d,s}$ be the load sharing factor associated with kth route tree of (o, d, s).

As discussed in the previous section, from KKT conditions of the user and system optimization problems, in the optimum solution, the partial derivatives of the cost function with respect to load sharing factors of those route trees with load sharing factor greater than zero are equal and less than or equal the partial derivative of the cost function with respect to load sharing factor of other route trees.

The recursion governing the expected behavior of cross-correlation learning automata is used for solving the optimization problems. First, the load sharing factors are arbitrarily initialized and traffic is distributed according to these load sharing factors over the set of route trees. The resulting blocking probabilities are calculated using the method of [16]. The load sharing factors are then updated using the recursion formula of Equation (2) with the following possible choices for $\underline{s}$ parameters:

$$\text{For user optimization problem :} \qquad s_k^{o,d,s}(t) = 1 - \frac{L_k^{o,d,s}}{\sum_{r=1}^{K_{o,d,s}} L_r^{o,d,s}}, \tag{13}$$

$$\text{For system optimization problem :} \qquad s_k^{o,d,s}(t) = 1 - \frac{\frac{\partial Z}{\partial \alpha_k^{o,d,s}}}{\sum_{r=1}^{K_{o,d,s}} \frac{\partial Z}{\partial \alpha_r^{o,d,s}}}. \tag{14}$$

The traffic distribution and resulting blocking probabilities are then updated and the process is repeated until $|\alpha^{\mathbf{o,d,s}}(t+1) - \alpha^{\mathbf{o,d,s}}(t)|$ is sufficiently small.

As seen from Equation (14), for the case of the system optimization problem, the calculation of $\underline{s}$ parameters implies the calculation of the partial derivatives of total blocking probability of the network with respect to each load sharing factor. The analytical formulation for these partial derivatives is complex and the numerical methods are subject to approximation errors which will affect the accuracy of the final results. To overcome this drawback, we consider a reformulation of the problem that leads to a more efficient approach for calculating $\underline{s}$ parameters.

3.4 System Optimization Problem, Alternative Formulation

In this formulation, we consider two sets of auxiliary variables: the average arrival rate for class s on link (i,j) ($\mathbf{m} = [m_{i,j,s}]$) and the class s blocking probability of link (i,j) ($\mathbf{B} = [B_{i,j,s}]$). The reformulated system optimization problem then becomes:

$$\min_{\alpha,\mathbf{m},\mathbf{B}} Z(\alpha,\mathbf{m},\mathbf{B}) = \sum_{o,d,s,k} \lambda^{o,d,s} \alpha_k^{o,d,s} L_k^{o,d,s} \tag{15}$$

subject to

$$m_{i,j,s} = \beta(\alpha,\mathbf{B}) \qquad (\eta_{i,j,s}) \tag{16}$$

$$B_{i,j,s} = Pb(m_{i,j,s}) \qquad (\omega_{i,j,s}) \tag{17}$$

$$\sum_k \alpha_k^{o,d,s} = 1 \qquad (v^{o,d,s}) \tag{18}$$

$$\alpha_k^{o,d,s} \geq 0 \qquad (u_k^{o,d,s}), \tag{19}$$

where $\beta(\alpha,\mathbf{B})$ can be derived using the method presented in [16] and $Pb(\mathbf{m})$ can be calculated using the Kaufman-Roberts recursion method.

The Kuhn-Tucker conditions are obtained by setting the derivative of Lagrangian of the problem (H) with respect to α, $\mathbf{m}$ and $\mathbf{B}$ equal to zero. This yields the following equations:

$$\frac{\partial H}{\partial m_{i,j,s}} = 0 \Rightarrow \eta_{i,j,s} = \sum_{r=1}^{S} \omega_{i,j,r} \frac{\partial Pb(m_{i,j,r})}{\partial m_{i,j,s}} \tag{20}$$

$$\frac{\partial H}{\partial B_{i,j,s}} = 0 \Rightarrow \omega_{i,j,s} = \sum_{p,q,s} \eta_{p,q,s} \frac{\partial \beta_{p,q,s}}{\partial B_{i,j,s}} - \sum_{o,d,k} \lambda^{o,d,s} \alpha_k^{o,d,s} \frac{\partial L_k^{o,d,s}}{\partial B_{i,j,s}} \tag{21}$$

$$\frac{\partial H}{\partial \alpha_k^{o,d,s}} = 0 \Rightarrow v^{o,d,s} = \sum_{(i,j)\in(o,d,s,k)} \eta_{i,j,s} \frac{\partial \beta_{i,j,s}}{\partial \alpha_k^{o,d,s}} - \lambda^{o,d,s} L_k^{o,d,s} - u_k^{o,d,s} \tag{22}$$

$$\alpha_k^{o,d,s} u_k^{o,d,s} = 0 \qquad \alpha_k^{o,d,s} \geq 0 \tag{23}$$

$$(i,j),(o,d) \in \{1,...,N\}^2, s = 1,...,S, k = 1,...,K_{o,d,s} \tag{24}$$

From these equations, for all route trees with $\alpha_k^{o,d,s} > 0$, the terms $g_k^{o,d,s} = L_k^{o,d,s} - \sum_{(i,j)\in(o,d,s,k)} \eta_{i,j,s} \frac{1}{\lambda^{o,d,s}} \frac{\partial \beta_{i,j,s}}{\partial \alpha_k^{o,d,s}}$ are equal.

So if the values of $g_k^{o,d,s}$ can be calculated at each iteration of the recursion method discussed in the previous section, the load sharing factors can be updated using:

$$s_k^{o,d,s}(t) = 1 - \frac{g_k^{o,d,s}}{\sum_{r=1}^{K_{o,d,s}} g_r^{o,d,s}}. \tag{25}$$

To do so, the value of $\eta(i,j,s)$ need to be calculated. These values can be derived from the following compact set of equations:

$$\begin{cases} \underline{\eta} = \underline{\dot{\mathbf{P}}\mathbf{b}\omega} \\ \underline{\omega} = \underline{\dot{\beta}\eta} - \underline{\dot{\mathbf{L}}}\lambda_k \end{cases} \Rightarrow \quad \underline{\eta} = (\underline{\dot{\beta}}^{-1} - \underline{\dot{\mathbf{P}}\mathbf{b}})\underline{\dot{\mathbf{L}}}\lambda_k \tag{26}$$

This set of equations models a hierarchical routing architecture where one centralized processor would be interconnected to the learning automata associated with every (o,d,s). On a regular basis, the centralized processor collects the network information, updates the $\underline{\eta}$ parameters and distributes the updated parameters to the learning automata engines. Such a mechanism is capable of generating and maintaining a performance level equivalent to the one expected by the system solution. A pictorial representation of this architecture is given in Figure 2.

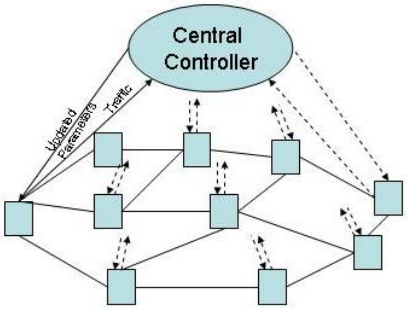


Fig. 2. Hierarchical Control Architecture

4 Numerical Results

In this section, the performance of LSSR algorithm with user and system optimized load sharing factors are compared in an example 4-node network 2 different classes of service under full sharing assumption. Here, the capacity of each link is 150 trunks. The effective bandwidth for class 1 is equal to 1 trunk and for class 2 is equal to 2 trunks. For each (o, d) pair, there are 5 sets of route-trees; one with only the direct path; two with the direct path and one of the alternate paths and two with the direct path and two other alternate paths. The order of the alternate paths is different in the last two route-trees. The stopping condition is when L1 norm of successive iterations differs by less than 10^{-5}. The performance comparison of user and system optimized load sharing factors with normal traffic load (for each (o, d, s), $\lambda^{o,d,s} = 41$) and heavy traffic load (for each (o, d, s), $\lambda^{o,d,s} = 50$) with different $\underline{\eta}$ updating intervals are presented in Table (1) and Table (2) repectively.

Table 1. 4-node 2-traffic class network under normal traffic

System/ User	T Update Interval	Blk. Prob.	No. of Iteration
Sys	T = 1	.00152	12547
Sys	T = 20	.00155	14764
Sys	T = 200	.00154	14921
Sys	T = 1000000	.00161	16356
User		.00161	446

Table 2. 4-node 2-traffic class network under heavy traffic

System/ User	T Update Interval	Blk. Prob.	No. of Iteration
Sys	T = 1	.0727	2582
Sys	T = 20	.0741	2150
Sys	T = 200	.0784	3563
Sys	T = 1000000	.0912	5327
User		.1232	737

In the next set of experiments, a 9-node network topology is considered with fully isolated maximum allocation bandwidth constraint model. As different classes are fully isolated, we consider only one of the classes of service. For each (o, d) pair, sets of route-trees compose of the direct path and one or two alternate paths. The stopping condition is when the L1 norm of the load sharing factors in successive iterations differs by less than 10^{-5}. Here again, for the system optimization problem, the parameter $\underline{\eta}$ is updated once every T iterations. The blocking probability results for 3 different updating intervals with normal and heavy traffic loads are summarized in Table 3 and Table 4.

As seen from the presented results, for the case of normal traffic load, the user equilibrium and the system optimum solutions give similar network performance while in the case of heavy traffic load, the system optimum solution gives better performance in terms of blocking probability. Moreover, the interval between updating $\underline{\eta}$ parameters has a negligible impact on the final blocking probability of the network. This in turn means that the system optimal solution can be obtained using a few number of updating $\underline{\eta}$ parameters. One should note that for the system optimum solution, the computational cost depends on the number of iterations and the computation cost of deriving the $\underline{\eta}$ parameters.

Table 3. 9-Node Network Nominal Rate

System/ User	T Update Interval	Blk. Prob.	No. of Iteration
Sys	T = 10	.00448	17363
Sys	T = 100	.00448	19483
Sys	T = 1000000	.00451	21436
User		.00452	1849

Table 4. 9-Node Network Heavy Traffic

System/ User	T Update Interval	Blk. Prob.	No. of Iteration
Sys	T = 10	.01868	8679
Sys	T = 100	.01868	10376
Sys	T = 1000000	.01875	12543
User		.01879	2385

5 Summary and Conclusion

In this paper, an event dependent routing method based on load shared sequential routing for MPLS networks was presented and the problem of optimizing the load sharing factors with the objective of minimizing the blocking probability either in full sharing case or the case where MAM is used as the bandwidth constraint model was discussed. A new method for solving the optimization problem both from the user and the system optimization perspectives was given.

In general, the user equilibrium and the system optimum solutions yield different routing solutions. Since the user solution is derived solely from local information, the resulting network blocking probability will generally be higher than that of the system optimum solution. However, in some cases, such as the cases studied in this report for the networks operating in nominal traffic loads, the difference can be relatively small.

While the system solution can only be derived with the global information, this paper has shown that it is possible to decompose the centralized optimization process into relatively smaller sub-processes. The centralized operations are restricted to the evaluation of the shadow prices. All other operations can be performed through decentralized sub-processes.

References

[1] Mason, L.G.: Equilibrium flows, routing patterns and algorithms for store-and-forward networks. Journal of Large Scale systems **8** (1985) 187–209

[2] LeFaucheur, F., Lai, W.: Requirements for Support of Differentiated Services-aware MPLS Traffic Engineering. RFC 3564 (2003)

[3] Marzo, J.L., Calle, E., Scoglio, C., Anjah, T.: QoS online routing and MPLS multilevel protection: a survey. IEEE Communications Magazine (2003) 126–132

[4] Borkar, V., Kumar, P.: Dynamic Cesaro-Wardrop equilibration in networks. IEEE Trans. on Automatic Control (2003) 382–396

[5] Ash, G.R.: Performance evaluation of QoS-routing methods for IP-based multi-service networks. Computer Communications **26** (2003) 817–833

[6] Ash, G.R.: Traffic Engineering and QoS Optimization of Integrated Voice & Data Networks. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2006)

[7] Heidari, F., Mannor, S., Mason, L.G.: Reinforcement learning-based load shared sequential routing. IFIP-Networking, to appear (2007)

[8] Brunet, G.: Optimisation de l'acheminement séquentiel non hiérarchique par automates intelligents. Master's thesis, INRS-Telecommunication (1991)

[9] Mason, L.G.: An optimal learning algorithm for S-model environments. IEEE Trans. on Automatic Control **18** (1973) 493–496
[10] Mason, L.G.: An optimal learning algorithm employing cross-correlation. In: Conference Record, Joint Automatic Control Conference. (1973) 572–578
[11] Gallager, R.: A minimum delay routing algorithm using distributed computation. Communications, IEEE Trans. on **25** (1977) 73–85
[12] Cotton, M., Mason, L.G.: Adaptive isarithmic flow control in fast packet switching networks. IEEE Trans. on Communications **43** (1995) 1580–1590
[13] Pelletier, A., Cotton, M., Mason, L.G.: Combined adaptive routing and flow control in fast packet switching networks. In: Proceedings of COMCON 4, Rhodes, Greece. (1993) 557–569
[14] Vazquez-Abad, F.J., Mason, L.G.: Decentralized adaptive flow control of high-speed connectionless data networks. Journal of Operations Research **47** (1999) 928–942
[15] Alanyali, M.: Learning automata in games with memory with application to circuit-switched routing. In: Decision and Control, 2004. CDC. 43rd IEEE Conference on. Volume 5. (2004) 4850– 4855
[16] Greenberg, A.G., Srikant, R.: Computational techniques for accurate performance evaluation of multirate, multihop communication networks. IEEE/ACM Trans. on Networking **5** (1997) 266–277
[17] Kelly, F.P.: Effective bandwidths at multi-class queues. Queueing Systems: Theory and Applications **9** (1991) 5–15
[18] Wardrop, J.G.: Some theoretical aspects of road traffic research. In: Proceedings of the Institution of Civil Engineers, Part II. (1952) 325–378

Pricing for QoS Provisioning Across Multiple Internet Service Provider Domains

Soheil Saberi[1], Roland P. Malhamé[1], and Lorne G. Mason[2]

[1] École Polytechnique de Montréal and GERAD, Montréal Canada H3T 1J4
[2] McGill University, Montréal Canada H3A 2K6

Abstract. In this paper we introduce a pricing scheme to be employed between a group of Internet service providers (ISPs) and a customer who wishes to initiate a packet flow from a fixed origin to a fixed destination. The ISPs are transparent to the customer who relies on a third party company for both the choice of the relevant ISPs and the unit flow price negotiated. The customer pays only for that portion of the traffic, which meets a predefined maximum tolerable total delay within the ISP networks. After taking in a fixed percentage of total profit, the third party redistributes the remaining benefits to the ISPs according to a sharing mechanism, which reflects both, the QoS the ISPs declare they will meet, as well as their real performance. The pricing emerges as the result of a Stackelberg game with the third party as the leader and the ISPs as the followers.[1]

Keywords: Multiple Domain Internet Pricing, Game theory, Statistical Quality of Service, Stackelberg Games.

1 Introduction

With the advent of new Internet applications for which more quality guarantees are expected from Internet service providers, existing flat rate charging schemes have become more and more inappropriate [1]. As a result, Internet pricing is currently a very active area of research. Based on the notion of effective bandwidth, a statistically founded tool for the evaluation of quality constrained bandwidth requirements for certain types of traffic in data networks [2,3], as well as different results from both cooperative and non cooperative game theory [4], various pricing approaches have been proposed.

In many schemes, along with the basic objective of pricing which is to recover the incurred costs, other goals have been considered among which, congestion control and fair allocation of resource to users [2,5], admission control and QoS provisioning [6], allocating the resource to users who value it most by selling the service in an online auction [7,8]. As argued in [9], the profit of ISPs as major players in Internet, has been neglected in many pricing schemes; therefore in

[1] This research was supported by a strategic grant from NSERC, CANADA, grant: STPGP269449-03.

this paper we are also interested in the interaction between ISPs and the outcome of the non-cooperative game between them. However, our model differs in a number of ways from that in [9]. We have assumed the case of only one data flow that passes through designated ISPs, and the end user who initiates the process is assumed to be willing to pay only for that portion of the traffic that meets a specific delay bound. On the other hand, an ISP reward structure is defined whereby each ISP obtains a share of customer payments which depends on both its initially declared individual quality of service goals, as well as on a statistical measure of how successful this ISP is in meeting the goals in question. Furthermore, the setup here is not one of guaranteed quality of service, but rather statistical quality of service. Such a choice was made for at least two reasons: firstly, deterministic quality of service guarantees can be quite wasteful in terms of bandwidth requirements. Secondly, when involving multiple ISP domains, guaranteed qualities of service tend to require a high degree of end to end coordination, and thus the complexity and overhead communications requirements of such schemes can quickly reach unmanageable levels as network size increases. Instead here, the setup is such that the enforcement of quality of service is an affair left as entirely internal to each independent network. If a particular network complies with a high degree of success rate relative to its declared goals, it will be rewarded accordingly. If not, it will not. This way, the control scheme for quality enforcement can be left as *decentralized* as possible.

A third party company herein referred to as *TP* has been introduced as a coordinator between the end user and ISPs. In return, it receives a fixed portion of customers payments. We adopt a Stackelberg game environment, in which *TP*, is the leader, and ISPs form the group of followers.

Overprovisioning of capacities may be the solution for many network operators to deal with delay and congestion issues, but as discussed in [2], while this looks like the right choice in backbones of the network, it may not be so for its metropolitan part, and even less so in the access part of the network. This stems from the fact that overdimensioning in the latter parts requires a lot more investment and this would raise the costs as edge nodes are approached. Based on this observation, we have assumed that each ISP involved in our model has at least one congestion node along the chosen route, and the imposed delay caused by this node, dominates that of any other route link within the ISP domain. In summary, each ISP is represented by a single bottleneck node along the chosen route.

The organization of the paper is as follows. In Section 2, we describe our modelling framework. In Section 3 we specify the utility functions associated with all of the active agents. In Section 4 we present our success-rate based pricing scheme, and we establish existence of a unique Nash equilibrium for the ISP part of the Stackelberg game. This is followed by a set of examples in Section 5, while Section 6 summarizes our conclusions and plans for future work.

2 Model Description

The proposed model involves three types of agents: a *customer* herein referred to as C, TP, and a collection of *ISPs* to be selected by TP. In our model, C is an end user with a potentially large volume of traffic to be sent on a regular basis from a given destination A to a destination B, and who initiates contacts with TP for that purpose. However, C specifies a maximum end to end tolerable delay for those transmitted packets for which it is willing to pay a per unit premium. We denote the maximum delay tolerated by C as T_{max}. An example of traffic type particularly relevant to the context here is VoIP. This is because in VoIP one can sustain the high loss probabilities that may occasionally result from the organization scheme to be proposed. Furthermore, there does already exist market regulators in the VoIP context and they can readily be identified as potential TPs in our model. Indeed the Telecom Decision CRTC 2005-28, which has been set by Canadian Radio-Television and Telecommunications Commission is a clear example of a set of regulations, upholding rather identical regulatory framework as extant traditional phone services for VoIP [10].

Division of revenues amongst telephone companies is based on mutual agreements between pairs of service providers. In the case of a large number of such providers of different hierarchical levels e.g. trunk providers and access network providers, the task of revenue sharing is currently performed by a third party company. Exchanges of balances, and information about each traversing telephone call between service providers are based on annual calculations. In the current model, TP plays an enhanced role, as compared to the case of telephone networks, in that a real-time information and revenue sharing mechanism is adopted.

TP together with C, agree on an *offered traffic versus unit flow price curve*, whereby offered traffic levels increase as bandwidth unit price decreases. This curve is a form of commitment on the part of the customer that it will pay a fixed bandwidth unit price per unit time for sending a given ultimately agreed to traffic level, unless it can demonstrably establish failure by TP to meet the QoS requirements at that traffic level. In the latter case, C's per unit time payment is reduced by the fraction of its total traffic inadequately transmitted. As a consequence of this arrangement, it is in C's best interest to constantly probe performance by sending traffic (useful or otherwise) at the agreed to level.

TP selects a number of ISPs along the route who are willing to be solicited in offering the service to C. At this stage, TP gathers from the candidate ISPs the parameters which specify the rules of the game they have to play and whose outcome will be their individual share of the income.

In the practical context, we assume that packet end to end delays, and within ISP domains, can be monitored for performance verification. However, all optimization decisions are founded on specific modelling assumptions. In the current context, we have settled for a simple M/M/1 queueing model of each network. We have assumed constant packet lengths, which without further loss of generality are taken to be of length unity, so that the probability of meeting the delay requirement can be expressed as $P(t \leq T_i) = 1 - e^{(\lambda - \mu_i)T_i}$, where λ is the rate of the source, t, μ_i and T_i are random delay, service rate and *declared* maximum transit time in network$i$, respectively.

The need to calculate success probabilities in each network, stems from the fact that we wish to reflect the customer payment mechanism on the ISPs involved in the negotiation. More specifically, the fraction of total revenue dedicated to an ISP directly depends on the probability of meeting the declared delay within its network. Moreover, as mentioned earlier, C pays according to the probability that its packet reaches the destination in time; the latter probability can be derived from the probability distribution of individual network delays.

The per unit time cost for the customer will be: $\Pr(t \leq T_{\max})C_v(\lambda)\lambda$, where $C_v(\lambda)$ is the unit cost versus traffic λ, dependency curve , herein referred to as the *customer response* curve. For convenience here, it is taken to be a decaying exponential. Indeed, anticipating a decreasing function of demand versus price is standard (see [9] for example). With all active agents and their declared parameters thus defined, we are ready to formulate the rules of a Stackelberg game whose outcome is the traffic rate submitted by C to the ISPs, the corresponding premium unit flow price paid by C, and the revenue obtained by each of the candidate ISPs.

3 Utility Functions and Game Framework

3.1 Third Party *TP*

TP, is a company responsible for all negotiations with the ISPs, with the understanding that the negotiation process must remain transparent to the customer. *TP*s unit time revenue is a fixed fraction of the total unit time payments made by C. The utility function of *TP* is considered to be:

$$TP_U(\lambda) = M \ \Pr(t \leq T_{\max})C_v(\lambda)\lambda \, . \tag{1}$$

where $M \in [0;1]$ is the fraction of total benefit reserved for *TP* . The only decision variable of *TP* is λ, and it is chosen to maximize *TP*'s revenue, or equally *total* customer payments to the ISPs, so that in a formulation of the game where ISPs cannot acquire more bandwidth, this corresponds to the social welfare optimization problem. We also assume an upper bound λ_{max} for the rate of data transfer.

3.2 Service Providers

We assume each network involved in the transaction to have a certain amount of bandwidth μ_i , naturally available for C's traffic. Furthermore, we assume that this initial bandwidth is sufficient to insure that the maximum possible source rate λ_{max} can be satisfied by any of the μ_i's ($\lambda < \mu_i \quad \forall i$). The ISPs have the option of increasing the amount of bandwidth they dedicate to C's traffic, via a specified cost of c_i per unit of added bandwidth. Let $\Delta\mu_i$ be the added bandwidth with an upper bound $\Delta\mu_i^{\max}$, so that the actual bandwidth that network i can allocate to the flow becomes: $\mu_i + \Delta\mu_i$. For each potential λ, the fraction of profit, which is not taken by *TP*, is assumed to be available in its entirety to the participating ISPs. However, for each fixed λ, ISPs are pitted against each other in a game, the rules of which will be defined in what follows. The idea is to reflect the payment mechanisms at *TP*'s level all the way down to the ISPs. More specifically, ISPi is asked to provide a (hypothetical) maximum delay T_i that it declares itself ready to aim at meeting. This $T_i \in [0; T_i^{\max}]$ is very instrumental in determining ISP i's share of total income available after *TP*'s payment, in that it is proposed that the fraction of that total allocated to ISPi be given by:

$$S_i = \frac{(1 - e^{-(\mu_i + \Delta\mu_i - \lambda)T_i})}{T_i^{\beta}} \left[\sum_{j=1}^{n} \frac{(1 - e^{-(\mu_j + \Delta\mu_j - \lambda)T_j})}{T_j^{\beta}} \right]^{-1} , \qquad (2)$$

with β as a coefficient between 0 and 1 (inclusively), and n as the number of ISPs.

Also note that, the larger the declared time, the less margin is left for other providers to accommodate their own delays along the packet route. From that point of view, fairness would dictate that a large declared T_i should correspondingly penalize the declarer (this explains the T_i^{β} in the denominator in (2)). The latter penalty prevents ISPs from letting their own declared T_i's go to infinity in an effort to maximize their chances of success. Also, note that for an adequate choice of β the optimal choice of declared T_i may well become the mean delay in the network. In addition, as alluded to earlier, the ISP has the option of either buying for a given unit price extra bandwidth, or equivalently freeing, albeit at the cost of some loss of revenue per unit bandwidth, a given amount of bandwidth, thus modulating its effective service rate μ_i. As a consequence ISPi, must provide two decision variables: T_i, and the extra amount of bandwidth $\Delta\mu_i$ it wishes to buy . Note that if we fix $\Delta\mu_i = 0$ (no bandwidth buying allowed), it is not difficult to see that, modulo a reward shift by an appropriate constant, the game is equivalent to a *zero-sum* game. Using this allocation rule, we define the utility function as:

$$ISP_{U_i} = (1 - M)C_v(\lambda)\Pr(t \le T)\lambda S_i - c_i\Delta\mu_i \, . \qquad (3)$$

where $(1 - M)C_v(\lambda)\Pr(t \le T)\lambda$ represents the revenue after payment of *TP*, and c_i is the extra per unit bandwidth equivalent cost.

3.3 Formulation of the Game

While we have specified different agents utility functions, we have not thus far specified the sequence in which the game is played. Given the predominant role of *TP* as the main organizer, we suggest that *TP* be considered as the higher level of the hierarchy within a Stackelberg game, i.e. *TP* is the leader. All participating ISPs are followers, and thus, for each fixed value of customer traffic rate

λ decided by the leader *TP*, we shall be looking for potential Nash equilibria. We also assume a perfect information environment, whereby each player knows all extra bandwidth unit buying costs, initial networks dedicated bandwidths to C's traffic, as well the customer response curve. This strong assumption is made in order to investigate the feasibility of the ideal game. However, more relaxed versions of the game where ISP's costs per unit bandwidth are assumed unknown to TP as well as to other competing ISPs, are possible and indeed workable.

Having the position of the leader in this game, *TP* can predict the outcome of the non-cooperative game among the followers, for any λ. By exploiting this fact, *TP* can specify the customer traffic level which best suits its interests.

Remark 1. Considering the expression of ISP_i's utility function in (3), we note that except for the share term S_i, the utility does not depend on the choice of declared maximum transit time T_i. Also for given $(\mu_i + \Delta\mu_i)$, T_i can be selected independently of other decision variables to maximize S_i, leaving $\Delta\mu_i$ as the unique decision variable of ISP_i. Furthermore, for the special case where the coefficient β in (2) is equal to 1, the optimum choice is $\forall i, T_i = 0$.

Remark 2. The fact that, at least for the $\beta = 1$ case, the optimal choices of declared maximum network transit times T_i for the ISPs correspond to the highly unrealistic value of *zero*, justifies their characterization as *declared* values. This leads to a reasonable rule for sharing benefits among ISPs. Indeed, for $\beta = 1$ as T_i goes to zero, L'Hôpital's rule yields:

$$S_i/S_j = (\mu_i + \Delta\mu_i - \lambda)/(\mu_j + \Delta\mu_j - \lambda)\,. \tag{4}$$

(4) in fact indicates that customer payments after commission are shared among ISPs in *inverse proportion to the mean packet transit time* in each of the networks. Also, it can be shown that choosing a β different from 1 is equivalent to a sharing rule where shares become proportional to $(\mu_i + \Delta\mu_i - \lambda)^\beta$. Thus as β decreases, ISPs could become more reluctant to buy bandwidth.

However, more relaxed versions of the game where ISP's costs per unit bandwidth are assumed unknown to *TP* as well as to other competing ISPs, are possible and indeed workable. In the next section, analysis is focused on the $\beta = 1$ case. For that special case, we establish the existence of Nash Equilibrium (NE) for the followers game corresponding to any admissible λ.

4 Properties of the Followers Game for $\beta = 1$

In the telecommunication literature the throughput of the data stream $(1 - e^{-(\mu_i+\Delta\mu_i-\lambda)T_i})\lambda$ over mean delay T_i is defined as the *power* factor. Thus for $\beta = 1$ the sharing mechanism presented in (2) can be regarded as a function of each ISP's power factor P_i. More specifically:

$$S_i = P_i / \sum_{j=1}^{n} P_j \quad \text{where: } P_i = (1 - e^{-(\mu_i+\Delta\mu_i-\lambda)T_i})\lambda/T_i. \tag{5}$$

In [11] an approach based on maximization of product of power factors to allocate a fair division of flows to users, has been introduced. Indeed, this corresponds to a so-called Nash bargaining solution. Instead, in the current model, each ISP tries to maximize its power factor and has an interest in securing a high overall success rate in meeting end to end QoS constraints.

Theorem 1. *Under an inequality detailed in Lemma 2 in Appendix, in the Stackelberg game defined by leader utility function (1) and followers utility functions (3) with $\beta = 1$, for every admissible λ set by the leader, the follower game admits a Nash Equilibrium.*

Proof. To prove the existence of NE's we use a paraphrase of the following theorem [4]:

Theorem 2. *For each player, assuming the sets of decision variables are closed, bounded and convex, and assuming that each player's utility function is continuous in all decision variables associated with all players, and strictly concave in the entries associated with its own decision variables, for every admissible combination of decisions of other players, the associated n-person nonzero-sum game admits a Nash Equilibrium in pure strategies.*

The theorem above can be easily shown to hold if strict concavity is replaced by the assumption of existence of a unique maximizer for each player's utility function for arbitrary decisions made by other players. Existence of a unique maximizer is satisfied, provided utility functions can be shown to be strictly *log concave* in their own decision variables. See Appendix for the proof. Since, $\Delta\mu_i \in [0; \Delta\mu_i^{\max}]$, the set of decision variables are both convex and compact. The continuity of utility functions on the admissible decision variable set is also obvious. Therefore a Nash Equilibrium exists. $\square$

5 Numerical Results for a Two ISP Game

We consider the case of two competing ISPs and associate arbitrary bandwidth unit costs to them. The inputs are: $\mu_1 = 1.1$,$\mu_2 = 1.2$ packet/ms, $C_v(\lambda) = e^{-(\lambda/0.75)}, \lambda_{\max} = 1$ packet/ms, $M = 20\%$, $T_{\max} = 6$ ms, $\Delta\mu_1, \Delta\mu_2 \in [0; 1]$ and $c_1 = 0.075, c_2 = 0.055$. Although, a mathematical proof of the existence of

Table 1. Simulation results of two competitive ISPs for $\beta = 1$ and $\beta = 0.5$

	$\beta = 1$	$\beta = 0.5$
Optimal λ(packet/ms)	0.750	0.708
$(\Delta\mu_1, \Delta\mu_2)$ at NE (packet/ms)	(0.442,0.603)	(0.275,0.360)
(T_1, T_2) at NE (ms)	(0,0)	(1.88,1.47)
(ISP_{U_1}, ISP_{U_2}) at NE	(0.0588,0.0891)	(0.0701,0.0961)
TP_U at NE	0.0536	0.0517
$Pr(t \leq T_{max})$ %	97.06	93.75

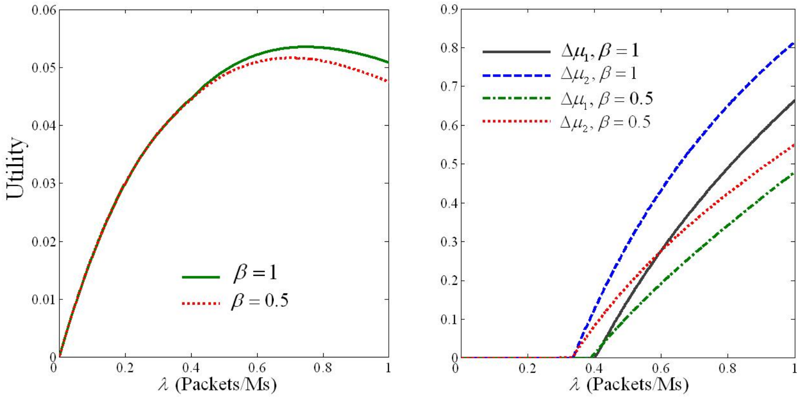


Fig. 1. Left: TP's utility versus the rate of transfer for $\beta = 1$ and $\beta = 0.5$, Right: $\Delta\mu_1$ and $\Delta\mu_2$ at Nash equilibria for $\beta = 1$ and $\beta = 0.5$

Nash equilibria for values of β other than 1, has not been established as yet, we numerically investigate the two cases of $\beta = 1$ and $\beta = 0.5$. Simulation results are shown in Table 1 and Fig.1. From Table 1, one sees that when β changes from 1 to 0.5, both ISP utilities increase, but more so for the ISP with less initial bandwidth. This comes at the price of decreasing the incentives of ISPs in buying more bandwidth. This in turn lowers the QoS to the customer who has to contend with a lower probability of success.

6 Conclusion and Future Work

Along with the growth of VoIP and other delay sensitive Internet applications, pricing and accounting of the new services, demand new techniques and methods to better reflect each provider's performance. In this article we have proposed a scheme for rewarding Internet provider companies, which can provide low delay communications. However, no performance guarantees are given.

The global end to end performance (or equivalently the probability of meeting total delay requirements) is a result of all agents efforts to cut transit time in their own networks. This points to the importance of fair revenue sharing rules between ISPs. To deal with this issue we have investigated a class of sharing rules, parameterized by the β variable. Setting β at a value less than 1, tends to reduce the financial advantage that a given ISP gets from an increase in bandwidth relative to other ISP's along the route. While this results in lower QoS, it can help offset unfair competitive advantages enjoyed by some ISP's along the route. Finding the β that makes declared transit times equal to mean transit times, and existence and uniqueness of NE in the followers game for $\beta \neq 0$, are other future areas of investigation. Also in the future we will consider repeated forms of the game to account for the possibility of imperfect information, and online utility parameter estimation. Finally ISPs along the route could be divided into subgroups in which competition is deemed fairer, insofar as the cost of acquiring bandwidth is concerned.

References

1. DaSilva, L.A.: Pricing for QoS-enabled networks: A survey,. IEEE Communications Review **3**(2) (2000)
2. Courcoubetis, C., Weber, R.: Pricing Communication Networks: Economics, Technology and Modeling. Wiley, New York (2003)
3. Kelly, F., Muallo, A., Tan, D.: Rate control for communication networks: Shadow prices, proportional fairness, and stability. J.Oper. Res. Soc. **49** (1998) 237–252
4. Basar, T., Olsder, G.J.: Dynamic Non cooperative Game. Academic Press, London and San Diego (1995)
5. Srisankar, R., Kunniyur, S.: Analysis and design of an adaptive virtual queue (AVQ) algorithm for active queue management. SIGCOMM (2001) 123–134
6. Li, T., Iraqi, Y., Boutaba, R.: Pricing and admission control for QoS enabled internet. Computer Networks **46** (2004) 87–110
7. Lazar, A., Semret, N.: Design and analysis of the progressive second price auction for network bandwidth sharing. Telecommunication Systems, Special Issue on Network Economics (2000)
8. Maillé, P., Tuffin, B.: Multi-bid auctions for bandwidth allocation in communication networks. INFOCOM (2004)
9. He, L., Walrand, J.: Pricing and revenue sharing strategies for internet service providers. IEEE/JSAC **24**(5) (2006) 942
10. Canadian Radio-Television and telecommunications commission: Regulatory framework for voice communication services using Internet Protocol. (2005)
11. Mazumdar, R., Mason, L.G., Douligieris, C.: Fairness in network optimal flow control. IEEE/ACM Transactions on Networking (1990)
12. Kleinrock, L.: Queuing Systems. Volume 1. Wiley Interscience, New york (1975)
13. Avriel, M., Diewert, W.E., Schaible, S., Zang, I.: Generalized Concavity. Plenum Press, New York (1988)

Appendix

Lemma 1. *The global success probability function $Pr(t \leq T_{max})$ is strictly concave with respect to each ISP decision variable $\Delta\mu_i$, regardless of $\Delta\mu_j, j \neq i$.*

Proof. The probability density function (pdf) of waiting time t in a simple M/M/1 queue is [12] : $g(t,x) = xe^{-xt}$. where $x = \mu + \Delta\mu - \lambda$. The total delay T that is imposed on each packet, is the sum of individual delays within each ISP's network. Thus, the pdf of T $(f(T,X))$, is the result of a convolution of all component pdf's.

$$f(T,X) = g(t_1,x_1) * g(t_2,x_2) * \cdots * g(t_n,x_n) \text{ where: } X = [x_1, x_2, \cdots, x_n]\,. \quad (6)$$

Defining $F(T,X)$ as the probability distribution function (PDF) of T, and $X_{-i} = [x_1, \ldots, x_{i-1}, x_{i+1}, \cdots, x_n]$, the pdf of total transit time when the time spent in

ISPi is excluded, can be defined as: $h_{-i}(T, X_{-i}) = g(t_1, x_1) * \cdots * g(t_{i-1}, x_{i-1}) * g(t_{i+1}, x_{i+1}) * \cdots * g(t_n, x_n) > 0$

The probability that the total packet delay be less than T_{max} is given by:

$$F(T_{\max}, X) = \int_0^{T_{\max}} h_{-i}(t, X_{-i}) * g(t, x_i)dt = \int_0^{T_{\max}} \int_0^{t} h_{-i}(\tau, X_{-i})g(t-\tau, x_i)d\tau dt\,. \tag{7}$$

Using Fubini's theorem to change the order of integration in (7), we will have:

$$F(T_{\max}, X) = \int_0^{T_{\max}} \int_{\tau}^{T_{\max}} (g(t-\tau, x_i)\,dt \quad)h_{-i}(\tau, X_{-i})\,d\tau\,, \tag{8}$$

where $G(x,t)$ is the PDF of $g(x,t)$. Our goal is to show that $\forall X_{-i}, \quad \frac{\partial^2 F}{\partial x_i^2} < 0$.

Using Lebesgue's dominated convergence, the differentiation can be carried across the integral:

$$\frac{\partial^2 F}{\partial x_i^2} = \int_0^{T_{\max}} h_{-i}(\tau, X_{-i})\frac{\partial^2}{\partial x_i^2}G(T_{\max}-\tau, x_i)d\tau\,. \tag{9}$$

Note that $\frac{\partial^2}{\partial x_i^2}G(T_{\max}-\tau, x_i) = -(T_{\max}-\tau)^2 e^{-(x_i)(T_{\max}-\tau)} < 0$ and $h_{-i} > 0$; hence (9) is always negative, and as a result the global success probability is strictly concave in x_i or equally in $\Delta\mu_i$. □

Lemma 2. *For any admissible values of decision variables X_{-i}, and assuming the following threshold for the total cost paid by the customer:*

$$AF(x_i, X_{-i}, T_{\max}) > max\{c_i\}\sum_{j=1}^{n} x_j, \text{ where: } A = (1-M)C_v(\lambda)\lambda, \tag{10}$$

$ISP_{U_i}(x_i, X_{-i})$ has a unique maximizer with respect to x_i.

Proof. Our goal is to show that:

$$ISP_{U_i}(x_i, X_{-i}) = \frac{x_i}{\sum\limits_{j=1}^{n} x_j}AF(x_i, X_{-i}, T_{\max}) - c_i(x_i - \mu_i + \lambda)\,, \tag{11}$$

always admits a unique maximizer. In Lemma 1, the strict concavity of $F(x_i, X_{-i}, T_{\max})$, with respect to x_i was established. On the other hand the function $-c_i \sum\limits_{j=1}^{n} x_j$ is a linear function in x_i, thus the function:

$$AF(x_i, X_{-i}, T_{\max}) - c_i \sum_{j=1}^{n} x_j\,. \tag{12}$$

is also strictly concave in x_i. ISP_{U_i} is assumed to have positive value for all ISPs and as a result, (12) is always positive. Assumption (10) ensures a positive value for (12) for all ISPs. Using Mangasarian's theorem [13], the log of (12) is a strictly concave function in x_i, and (12) will be strictly *log concave*. Furthermore $x_i(\sum_{j=1}^{n} x_j)^{-1}$ is also a strictly log concave function in x_i. Since log concavity is preserved under multiplication, and in view of the strictly increasing nature of the log function, the utility function in (11) has a unique maximizer in x_i. $\square$

Robust Wardrop Equilibrium

Fernando Ordóñez[1] and Nicolás E. Stier-Moses[2]

[1] Department of Industrial & Systems Engineering, USC
fordon@usc.edu
[2] Graduate School of Business, Columbia University
stier@gsb.columbia.edu

Abstract. Agents competing in a network game typically prefer the least expensive route to their destinations. However, identifying such a route can be difficult when faced with uncertain cost estimates. We introduce a novel solution concept called *robust Wardrop equilibria* (RWE) that takes into account these uncertainties. Our approach, which generalizes the traditional Wardrop equilibrium, considers that each agent uses distribution-free robust optimization to take the uncertainty into account. By presenting a nonlinear complementary problem that captures this user behavior, we show that RWE always exist and provide an efficient algorithm based on column generation to compute them. In addition, we present computational results that indicate that RWE are more stable than their nominal counterparts because they reduce the regret experienced by agents.

Keywords: Network Games, Wardrop Equilibrium, Robust Optimization, Robust Shortest Path, Robust Game Theory.

1 Introduction

Network games model the interaction between agents who select routes to go from their origins to their destinations. The most common applications can be found in modeling telecommunication, transportation, and logistic systems. Although in some cases decisions are dictated by a system manager, most often agents select routes on their own, giving rise to a competition for the network resources. It is typically assumed that agents are independent and wish to optimize some individual performance measure—such as utility, delay, cost, or profit—until they all collectively achieve an equilibrium situation in which no agent has any incentive to deviate. Realistically, performance measures are rarely known exactly prior to making a decision. Agents may have estimates of these performance measures based on past experience, but even if all decisions could be accurately forecasted, there could still be external factors that create uncertainty. A way to resolve this deficiency is to consider an imperfect information game in which actual performance measures can vary from the deterministic nominal values. This allows agents to take the uncertainty into account at the time of making their decisions.

This article generalizes the notion of Wardrop equilibria by proposing a solution concept for network games with uncertain performance measures in which each agent solves a robust optimization problem. In network games, cost functions (also known as latency, link-performance or congestion functions) that estimate the cost of using a link create interdependencies between agents' decisions. We decompose these costs into two terms: a *nominal* term that depends on decisions made by agents, and an *error* term which accounts for random effects not modeled directly. In the context of telecommunication networks, the error term can be due to defective equipment, noise, interference, or signal degradation, while in the context of transportation, it can be caused by accidents, traffic signals, weather or varying traffic conditions.

As agents are typically rationally bounded, we do not assume that they know (or can make use of) the distribution of errors. We only assume that they know the support for every link, which is given by the maximum deviation from the nominal cost. This simple uncertainty model only requires the estimation of a single parameter per link. We assume that all agents use robust optimization to solve their problems with uncertain information, which leads to an outcome that we call a *robust Wardrop equilibrium* (RWE). Robust optimization has become a popular paradigm in mathematical programming, gaining a wide acceptance in a number of applications such as portfolio optimization, supply chain management, and network design, to name a few. This paradigm addresses optimization problems with uncertain parameters by finding a solution that has optimal worst-case cost. Alternative methods assume that uncertain elements have known distributions, require the ability to sample from a distribution, represent the uncertainty through scenarios that may lead to large-scale problems, or use simple uncertainty models such as only considering expectations. The robust optimization approach represents the uncertainty by considering that the uncertain parameters belong to a bounded convex set. Such sets can represent the estimation confidence intervals of the uncertain parameters and also model interactions or correlations between them. Intuitively, these sets should prohibit all parameters from taking their worst-case values simultaneously since that event is extremely unlikely. Indeed, although we take a worst-case perspective, to avoid being overly pessimistic, we omit unlikely situations where a large number of links encounter large deviations from their nominal costs.

Main Contributions and Roadmap. This paper proposes a novel model and a solution concept for network games that incorporate uncertainty in forecasts of costs. The model proposed is such that there always exists an equilibrium for the network game and, moreover, it is unique when cost functions are strongly monotone. We present a computational procedure for the solution concept that puts together several existing pieces: we use the framework of nonlinear complementary problems [1], a column generation algorithm for nonadditive cost functions [12], and the robust shortest path problem [7]. Our computational results show that RWE are closer to an equilibrium with respect to the realized costs than the nominal counterpart. In other words, RWE reduces the regret that players experience after the uncertainty is revealed, which makes it a very appealing solution concept.

The paper is organized as follows. Section 2 defines a robust Wardrop equilibrium and shows its existence. In Section 3, we discuss an efficient column generation algorithm to find RWE. Section 4 presents computational results that compare the quality of robust Wardrop equilibria to equilibria that disregard the uncertainty. Finally, we present our conclusions in Section 5.

Related Work. Our model generalizes the traditional concept of a Wardrop equilibrium of network games. Wardrop postulated that users in a transportation network select routes of minimal cost [24]. Wardrop equilibria and many extensions have since become widely used by practitioners; see [3, 22]. Most network models developed to date assume that delays can be predicted accurately. However, it has been recognized that this is not necessarily the case in practice [20, 19]. We propose an approach in which users do not know the distributions of errors and select routes by solving robust optimization problems.

The robust optimization approach was introduced by [6] and provides a solution with the best objective value for the worst-case scenario. An attractive feature of a robust solution is that it behaves well for all likely uncertainty. In addition, in many settings finding this solution is no harder than solving the deterministic problem. Robust optimization has provided solutions that are insensitive to the uncertainty and thus efficient in practice to problems arising in diverse applications [5, 10, 13, 7].

Game theorists have long considered that players may not have complete information at the time of making their decisions. Unknown information is modeled with probability distributions, and players compute their expected payoffs using them [14, 15]. A shortcoming of this model is that it is not obvious how players can estimate the prior distribution. A few recent papers have explored the application of robust optimization to game theory. [16] characterizes *robust Nash equilibria* in simple games as solutions to a second-order cone complementarity problem. [2] also considers robust games and proves that robust Nash equilibria always exist. These articles, however, consider finite number of players and do not concentrate on robust equilibria in network settings.

Finally, there are transportation models that consider a different type of uncertainty. *Stochastic user equilibria* incorporate uncertainty from variations in the perception of costs by different users [9, 8]. This definition assumes that costs are well determined but different users extract different utility from it. Basically, the difference is that there is one realization of the random variable for each user as opposed to the same realization for all users. This approach and ours are complementary to each other, and both sources of uncertainty could be modeled together.

2 Definition and Existence of RWE

We consider a directed graph $G = (N, A)$ together with a set of origin-destination (OD) pairs $K \subseteq N \times N$. For each terminal pair $k = (s_k, t_k) \in K$, let $\mathcal{P}_k$ be the set of directed (simple) paths in G from s_k to t_k, and let $d_k > 0$ be the demand rate associated with commodity k. We refer to the set of all paths by $\mathcal{P} := \bigcup_{k \in K} \mathcal{P}_k$. A feasible flow f assigns a nonnegative and possibly fractional value f_P to every path $P \in \mathcal{P}$ such that $\sum_{P \in \mathcal{P}_k} f_P = d_k$ for all $k \in K$. The total flow along arc $a \in A$ can be easily computed by summing over paths: $f_a := \sum_{a \in Q \in \mathcal{P}} f_Q$. Whether we refer to an arc- or path-based flow will be clear from the subscript and context. The cost of each arc a is separated in two components: a deterministic load-dependent nominal value and an additive random value. The nominal value is expressed by the nonseparable function $\ell_a : \mathbb{R}^A_{\geqslant 0} \to \mathbb{R}_{\geqslant 0}$, assumed to be positive, continuous and strictly monotone. To model uncertainty, we consider an additive deviation from the nominal value given by $Z_a \gamma_a$, where Z_a is random variable with support in $[0, 1]$ and γ_a is an upper bound on the possible deviation from the nominal value. We assume that users do not know the distribution of Z_a; they only have information about γ_a. The actual cost on each arc a is, therefore, $\ell_a(f) + z_a \gamma_a$, where z_a is a realization of Z_a.[1]

Congestion gives rise to competition among users. The resulting flow is given by the solution to a nonatomic network game in which there are an infinite number of players who control an infinitesimal amount of demand and seek the path with minimum cost. The novel element is that users use robust optimization and take *robust shortest paths* [7]. As it is unlikely that extremely high costs are realized in many arcs, we give each user an uncertainty budget of Γ to constrain the total deviation from the nominal cost. (Note that the model extends directly to more general cases such as an uncertainty budget that depends on the user type.) Each user seeks the shortest path considering the possibility of facing some deviations from the nominal costs. In other words, a user traveling between OD pair k selects the path that has the best worst-case cost:

$$\min_{P \in \mathcal{P}_k} \max \Big\{ \sum_{a \in P} (\ell_a(f) + z_a \gamma_a) : \sum_{a \in P} z_a \leqslant \Gamma,\ 0 \leqslant z_a \leqslant 1 \Big\}, \tag{1}$$

where f is the flow that encodes the collective decisions made by all users. We let the nominal cost of a path $P \in \mathcal{P}$ under a given flow f be $\ell_P(f) := \sum_{a \in P} \ell_a(f)$, and the maximum deviation from it be $\gamma_P^\Gamma := \max \big\{ \sum_{a \in P} z_a \gamma_a : \sum_{a \in P} z_a \leqslant \Gamma,$ $0 \leqslant z_a \leqslant 1 \big\}$. With these definitions, (1) can be expressed as finding the path $P \in \mathcal{P}_k$ that minimizes $\ell_P(f) + \gamma_P^\Gamma$.

Wardrop postulated that users in a network game select routes of minimal cost [24]. This leads to the Wardrop equilibrium, under which users do not have an incentive to deviate because all selected shortest paths. We generalize this solution concept by incorporating the uncertain cost to the model. The following definition states that a robust Wardrop equilibrium optimizes the users' objective introduced above for all users simultaneously.

[1] Note that all users will experience the same cost. This is in contrast to stochastic user equilibrium models, where each user extracts a different utility from the same travel time requiring one realization of the random variable for each user.

Definition 1. *A flow f is a* robust Wardrop equilibrium *(RWE) if and only if*

$$\ell_P(f) + \gamma_P^{\Gamma} \leqslant \ell_Q(f) + \gamma_Q^{\Gamma} \quad \textit{for all } P, Q \in \mathcal{P}_k,\ k \in K \textit{ with } f_P > 0\ .$$

In the case of Wardrop equilibrium, [1] showed that the equilibrium condition can be expressed as a nonlinear complementarity problem (NCP) in the space of arc-flows. For the robust case, this can also be done, although the difference is that we have to consider a path formulation because maximum deviations γ_P^{Γ} depend on the whole path. The following proposition, whose proof follows from results for path formulations of NCPs in [1], characterizes a RWE. The complementarity condition is the key element of this formulation as it says that paths can only route flow if they are shortest with respect to the users' objective.

Proposition 2. *A flow f is a RWE if and only if it solves*

$$0 \leqslant f_P \ \perp\ \ell_P(f) + \gamma_P^{\Gamma} - u_k \geqslant 0 \quad \text{for all } P \in \mathcal{P}_k, k \in K\,, \tag{2}$$

where the notation $\perp$ means that at least one of the two constraints on either side must be tight, and the free variable $u_k \in \mathbb{R}$ represents the minimal objective function values for the users' robust objective. Moreover, a RWE always exists, and it is unique when functions ℓ_a are strictly increasing.

In the case of separable functions $\ell_a(\cdot)$ for which the cost on an arc $a \in A$ depends only on the flow f_a on the same arc, the robust Wardrop equilibrium problem can be cast as a convex optimization problem with path variables (see, e.g., [3] for a background on potential functions).

3 A Column Generation Algorithm for Computing RWE

A central difficulty in computing a RWE is that the users' objective function is not separable. Hence, arc formulations are unsuitable and we have to resort to path variables and the NCP shown in (2). This section outlines an algorithm that uses a column generation procedure. We follow the approach of [12], which studied a column generation algorithm for solving NCPs arising from network equilibrium problems with non-additive costs.

The algorithm maintains an *active* set of paths $\mathcal{P}' = \cup_{k\in K}\mathcal{P}'_k$ that it works with, where $\mathcal{P}'_k \subseteq \mathcal{P}_k$. Players are restricted to selecting paths in that set. At each step the algorithm solves a problem similar to (2), referred to as the *master problem*, and gets a restricted equilibrium solution. This consists on finding a flow f and vector u of minimum costs such that $0 \leqslant f_P \ \perp\ \ell_P(f) + \gamma_P^{\Gamma} - u_k \geqslant 0$ for all $P \in \mathcal{P}'_k,\ k \in K$. If f is also at equilibrium over the complete set of paths $\mathcal{P}$, we can stop; otherwise we add more paths to $\mathcal{P}'$. This happens when there is no path in $\mathcal{P} \setminus \mathcal{P}'$ that is shorter than u_k for the objective induced by f. In other words, we solve the following shortest path problem with non-additive costs (but linear when formulated in terms of paths), which we refer to by $SP(f)$:

$$v(f) := \min\Big\{ \sum_{P\in\mathcal{P}} \big(\ell_P(f) + \gamma_P^{\Gamma}\big)x_P : \sum_{P\in\mathcal{P}_k} x_P = d_k \text{ for } k \in K, (x)_{P\in\mathcal{P}} \geqslant 0\Big\}. \tag{3}$$

This problem is easily solved because it decomposes in independent robust shortest path problems for each OD pair. The robust shortest path problem, which was first studied by [7], can be solved by calling a regular shortest path routine m times. The following proposition provides a restatement of the termination condition. Its proof can be found in the full version of this article [21].

Proposition 3. *A flow f^* is a RWE if and only if f^* is an optimal solution to $SP(f^*)$, or equivalently, $v(f^*) = \sum_{P\in\mathcal{P}} \left(\ell_P(f^*) + \gamma_P^{\Gamma}\right) f_P^*$.*

Note that if f^* is the flow found by solving the master problem, it can be extended to a full flow in $\mathcal{P}$ by setting $f_P^* = 0$ for all $P \in \mathcal{P}\backslash\mathcal{P}'$. Then, by summing only over $\mathcal{P}'$, the condition of Prop. 3 becomes $v(f^*) = \sum_{P\in\mathcal{P}'} \left(\ell_P(f^*) + \gamma_P^{\Gamma}\right) f_P^*$, which is what we use in the algorithm. We are ready to state the full algorithm.

Algorithm 1. Column Generation

```
1: Initialize: Add arbitrary paths to $\mathcal{P}'$ (at least one per OD pair).
2: Set $f^* = 0$ and $v(f^*) = -\infty$.
3: while $v(f^*) < \sum_{P\in\mathcal{P}'} \left(\ell_P(f^*) + \gamma_P^{\Gamma}\right) f_P^*$ do
4:    Solve the master NCP. Let $f^*$ be the optimal solution.
5:    Solve Problem $SP(f^*)$. Let $x^*$ be the optimal solution and $v(f^*)$ be its value.
6:    Add paths used in $x^*$ to $\mathcal{P}'$.
7: stop. The flow $f^*$ is a RWE.
```

This algorithm finishes in finite time, since in the process of checking whether the candidate flow f^* is a global solution or not, we identify at least one new path to add to the active set $\mathcal{P}'$. Hence, in the worst case we iterate until $\mathcal{P}' = \mathcal{P}$, in which case (2) and the master problem would be identical. It is important to note that, in general, the column generation algorithm converges in a small number of iterations and it does not enumerate all paths, as it succeeds in quickly identifying the typically small number of paths used in the equilibrium solution.

4 Evaluation of RWE

In this section we present computational experiments that compare RWE with the standard Wardrop equilibria that assumes deterministic costs. For brevity, we refer to the latter as *nominal equilibria*. Our main goal is to show that RWE are an attractive alternative for networks with cost uncertainties.

We performed extensive computational experiments with problems ranging from small to moderately sized, for various uncertainty sets. All instances have separable cost functions because in that case the master problem reduces to a convex optimization problem, which is easier to solve. We use AMPL [11] to implement the column generation procedure described by Algorithm 1, LOQO [23] to solve the master problem, and CPLEX 9.1 [17] to solve the robust shortest path problem.[2] Due to lack of space, we only describe the results corresponding to Sioux Falls, which is a well-studied network in the transportation literature that is normally used to test various equilibrium concepts and algorithms to compute them. The outcome of the other experiments can be found in the full version of this article [21].

Albeit unknown to users, in practice there must exist random variables for each deviation Z_a from the nominal cost on arc $a \in A$. For simplicity, we assume that each Z_a is independently and uniformly distributed on the range $[0, \gamma_a]$; similar results are obtained for other distributions. Nevertheless, due to the users and planners' bounded rationality and limited resources, they do not know these distributions, they just know the support. We use Monte Carlo simulation to evaluate different solutions. For a given flow f, we obtain an empirical distribution of the cost faced by users by drawing random numbers z_a for the actual deviation of cost Z_a. Note that there are two sources of randomness that contribute to the empirical distribution: a user of OD pair $k \in K$ selects a path $P \in \mathcal{P}_k$ with probability f_P/d_k, and the cost faced by a user that selected path P is $\sum_{a \in P}(\ell_a(f_a) + z_a\gamma_a)$.

We evaluate solutions by computing their distance to an equilibrium under the realized costs on the network. Following [18], we measure that distance with the *unfairness* of the flow, defined as the maximum over $k \in K$ of the ratio between the largest and the smallest cost for users of OD pair k. The unfairness thus measures how much users regret the decisions they made. Note that in general the unfairness is strictly greater than one for any solution as users will always suffer from some variability in the experienced costs because of uncertainty.[3] A flow with minimal unfairness is more attractive as an equilibrium solution because users' costs are closer to being constant. To avoid the influence of small amounts of flow that may take extreme values possibly due to numerical issues with the nonlinear optimization problem, the computational study computes the unfairness with the 95th and the 5th percentile of the empirical distribution of costs instead of the maximum and minimum. When talking about a particular user, its unfairness is defined as the ratio of its actual cost to the smallest cost among all users.

Sioux Falls with Simplified Demands. To benchmark the robust equilibrium solutions, we use a simplified version of the Sioux Falls instance, in which we only consider 5 OD pairs. The purpose of this simplified example is to be able to analyze the solutions in detail to develop insights for the trade-offs and benefits of robust equilibria. The complete version of the article also presents results for the original Sioux Falls instance [21].

Since the Sioux Falls instance that is publicly available does not contain uncertainty parameters [4], and considering that longer links tend to be more

[2] The bottleneck of the computation is solving the master problem so we do not expect significant changes in the running times had we implemented the more efficient algorithm to solve the robust shortest path problem described in [7].

[3] In other words, an ex-post equilibrium—a flow that is at equilibrium for all realization of uncertainty—does not exist.

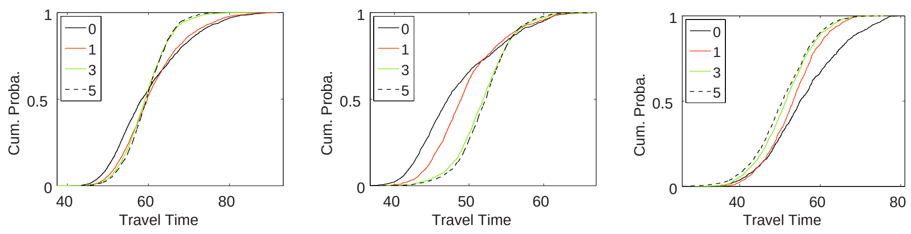


Fig. 1. Probability distribution functions of cost for different values of Γ. Each graph depicts a different OD pair.

uncertain, we created initial parameters γ_a proportional to the free-flow cost. Next, we modified those initial values depending on the proximity to the core of the network. In the context of transportation networks, the downtown area of a city is more congested because more cars use metered parking, enter and exit parking lots, look for parking places, there are more cabs that stop or circulate slowly, etc. We can thus argue that the cost of a link close to downtown tends to be more uncertain than in the periphery. We designated an area of the network as downtown, and assigned a high value of γ_a to arcs inside it, and medium values to arcs in the proximity with the intention of modeling a transition area.

Comparing the RWE and the nominal equilibrium, RWE with larger values of Γ tend to make more use of arcs in the periphery as they have low uncertainty while less flow is routed along arcs close to the downtown area. Fig. 1 summarizes these flows by showing the cumulative probability distributions of total cost for 3 representative OD pairs. Cost distributions for robust solutions are steeper for all OD pairs, meaning that they have less variability and are invariably fairer than their nominal counterparts. Distributions of robust solutions do not always dominate the distribution of a nominal equilibrium, but that is not a cause of concern because it is well known that equilibria need not be efficient.

Figure 2 concentrates on the unfairness of RWE. The graph on the left shows the unfairness corresponding to users of the five OD pairs. The graph is piecewise constant because as all users of the same OD pair are indistinguishable, they are all subject to the same unfairness. Different series correspond to different values of Γ, and users are sorted to make the curves nondecreasing. This means that users in each curve are not in one-to-one correspondence. On the right graph, we plot the improvement in unfairness of a robust equilibrium with respect to the nominal equilibrium. To compute it, we subtract one from each unfairness value because an unfairness value of one corresponds to a true equilibrium and is a lower bound. The maximum improvement possible is 100%, which occurs when the improved instance is at equilibrium with respect to the experienced costs. Because improvements are positive everywhere, all robust equilibria are better than their nominal counterpart. For the majority of users, solutions with higher Γ are better; moreover, for more than two thirds of the users, the improvement in unfairness with respect to the nominal equilibrium is more than 40% when $\Gamma = 5$.

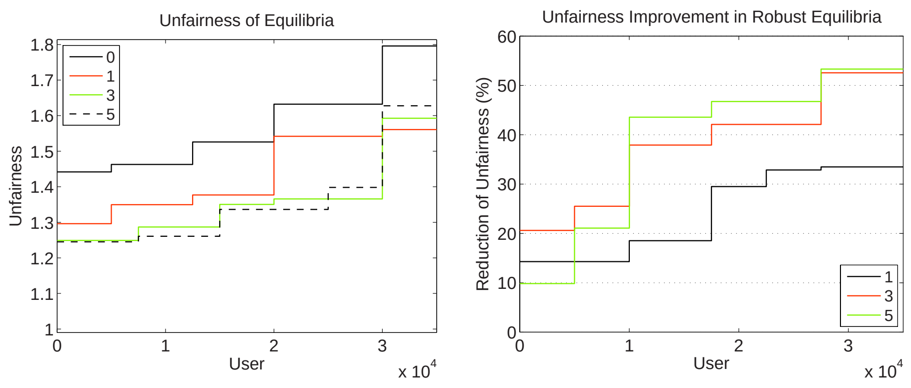


Fig. 2. Unfairness of equilibria. On the left, we plot the unfairness experienced by the users. On the right, we plot the unfairness improvement of robust equilibria. Both graphs include curves for different values of Γ, and users for each curve are sorted independently by increasing value of unfairness.

5 Conclusions

Our computational study has indicated that robust equilibria are indeed more fair than nominal ones, which is reflected in the reduced variability of costs computed by Monte Carlo simulations. This implies that using a robust optimization approach seems to be a better model of user behavior than one that ignores uncertainty completely. Indeed, this allows users to factor the variability in their plans to obtain an increase on the accuracy of the cost estimates before starting their trips. In the full version of this paper, we also provide results for a large instance called Friedrichshain [18]. This shows that our algorithm scales well when the instance size grows and that the conclusions remain valid for larger realistic instances.

We remark that the solution concept introduced here is valid for more general models, such as nonatomic congestion games or users with varying degrees of risk-aversion. Extending the solution concept presented here to atomic games, elastic or stochastic demands, and uncertainties that depend on user behavior is a matter of future work.

Acknowledgments. The authors are grateful to Michele Aghassi, whose constructive comments helped to improve this paper, and to one anonymous referee for suggesting the convex optimization formulation of RWE.

References

[1] H. Z. Aashtiani and T. L. Magnanti. Equilibria on a congested transportation network. *SIAM J. on Algebraic and Discrete Methods*, 2:213–226, 1981.
[2] M. Aghassi and D. Bertsimas. Robust game theory. *Mathematical Programming B*, 107:231–273, 2006.

[3] E. Altman, T. Boulogne, R. El-Azouzi, T. Jiménez, and L. Wynter. A survey on networking games in telecommunications. *Computers and Operations Research*, 33:286–311, 2006.

[4] H. Bar-Gera. Transportation network test problems, 2002.

[5] A. Ben-Tal and A. Nemirovski. Robust truss topology design via semidefinite programming. *SIAM J. on Optimization*, 7:991–1016, 1997.

[6] A. Ben-Tal and A. Nemirovski. Robust convex optimization. *Mathematics of Operations Research*, 23:769–805, 1998.

[7] D. Bertsimas and M. Sim. Robust discrete optimization and network flows. *Mathematical Programming B*, 98:49–71, 2003.

[8] C. Daganzo and Y. Sheffi. On stochastic models of traffic assignment. *Transportation Science*, 11:253–274, 1977.

[9] R. B. Dial. A probabilistic multi-path traffic assignment algorithm which obviates path enumeration. *Transportation Research*, 5:83–111, 1971.

[10] L. El-Ghaoui and H. Lebret. Robust solutions to least-square problems to uncertain data matrices. *SIAM J. Matrix Anal. Appl.*, 18:1035–1064, 1997.

[11] R. Fourer, D. M. Gay, and B. W. Kernighan. *AMPL: A Modeling Language for Mathematical Programming*. Brooks/Cole, 2002.

[12] S. Gabriel and D. Bernstein. The traffic equilibrium problem with nonadditive path costs. *Transportation Science*, 31:337–348, 1997.

[13] D. Goldfarb and G. Iyengar. Robust portfolio selection problems. *Mathematics of Operations Research*, 28:1–38, 2003.

[14] J. C. Harsanyi. Games with incomplete information played by "Bayesian" players. Part I. The basic model. *Management Science*, 14:159–182, 1967.

[15] J. C. Harsanyi. Games with incomplete information played by "Bayesian" players. Part II. Bayesian equilibrium points. *Management Science*, 14:320–334, 1968.

[16] S. Hayashi, N. Yamashita, and M. Fukushima. Robust Nash equilibria and second-order cone complementarity problems. *J. of Nonlinear and Convex Analysis*, 6:283–296, 2005.

[17] ILOG. CPLEX, 2005. `http://www.ilog.com`.

[18] O. Jahn, R. H. Möhring, A. S. Schulz, and N. E. Stier-Moses. System-optimal routing of traffic flows with user constraints in networks with congestion. *Operations Research*, 53:600–616, 2005.

[19] H. Liu, X. Ban, B. Ran, and P. Mirchandani. An analytical dynamic traffic assignment model with stochastic network and travelers perceptions. *Transportation Research Record*, 1783:125–133, 2002.

[20] P. B. Mirchandani and H. Soroush. Generalized traffic equilibrium with probabilistic travel times and perceptions. *Transportation Science*, 21:133–152, 1987.

[21] F. Ordoñez and N. E. Stier-Moses. Robust wardrop equilibria. Columbia University Working Paper DRO-2006-04, 2006.

[22] Y. Sheffi. *Urban Transportation Networks*. Prentice-Hall, 1985.

[23] R. J. Vanderbei. LOQO: An interior point code for quadratic programming. *Optimization Methods and Software*, 12:451–484, 1999.

[24] J. G. Wardrop. Some theoretical aspects of road traffic research. *Proceedings of the Institution of Civil Engineers, Part II, Vol. 1*, 325–378, 1952.

Hierarchical Game and Bi-level Optimization for Controlling Network Usage Via Pricing

Yezekael Hayel

LIA/University of Avignon,
339, chemin des Meinajaries,
84911 Avignon, France
yezekael.hayel@univ-avignon.fr

Abstract. Using the inherent relation between price and demand, a usage-based pricing mechanism can be an efficient tool for controlling the usage of scarce resource like bandwidth in access networks and wireless communication, where there is big competition between users. I propose here to describe my contributions in this field of network pricing and particularly dealing with hierarchical game. This kind of multi-level game is well adapted for optimizing the system manager's decision taking into account the follower game between users. We propose the use of hierarchical games for studying different networking scenarios : optimal scheduling in a DiffServ router and uplink transmissions in a CDMA cell.

Keywords: hierarchical game, pricing, queueing theory.

1 Introduction

Optimization of protocols parameters in networking applications induce different behavior of users. In a context of usage-based pricing, in order to control system usage and also to optimize provider's revenue, the decision of the access price has a big influence on the user's behavior(for a thorough overview of pricing issues in telecommunication networks, see e.g.[7]). One way of studying such bi-level systems is the *Hierarchical Games* [28] approach. In this paper, we consider a game model based on a *Stackelberg Games* for the study of networking systems like an Ingress DiffServ router and an access point using CMDA access mechanism.

A Stackelberg game is a particular case of a hierarchical game with 2 players, where one is the *leader* and the other one is the *follower*. This kind of game has been used in [27] for studying a revenue maximization problem taking into account a non-cooperative flow control game with finite number of users. In our formulation, the number of users is not fixed and we propose a condition for each user to leave or join the system based on his satisfaction.

2 Model

We consider a system with M class of users. For each class $i = 1, \ldots, M$, each user sends its traffic with a constant rate λ_i. The access price is p_i. When the system manager sets his parameters (access price in our context) and the competition between users has a stable point (an equilibrium), the revenue of the system manager is expressed by:

$$R(p_1, p_2, \ldots, p_M) = \sum_{i=1}^{M} \lambda_i N_i^* p_i, \tag{1}$$

where N_i^* is the number of users of class i at equilibrium in the system.

2.1 User Decision

Assume an infinite population of heterogeneous potential users. They differ from their sensitivity to their perceived quality of service (QoS). We consider that the main QoS parameter is the mean packet delay $\mathrm{I\!ED}$. The *utility* for a class i user depends not only on the mean packet delay $\mathrm{I\!ED_i}$, but also on the price per packet p_i, in the following way:

$$u_i(\mathrm{I\!ED_i}) = f_i(\mathrm{I\!ED_i}) - \mathrm{p_i}. \tag{2}$$

We consider, as proposed in [8], that each user will join (leave) the system if its utility is positive (negative). Hence, we can model the competition between each users as a competition (non-cooperative game) between each class of users. Indeed, each (super-)users $i \in \{1, \ldots, M\}$ wants to maximize the following utility function :

$$U_i(N) = N_i \mathbb{1}_{u_i \geq 0},$$

with $N = (N_1, N_2, \ldots, N_M)$ the vector of the number of users of each class.

2.2 System Manager Decision

The leader's objective is to maximize his revenue. Then he has to solve the following optimization problem:

$$\max_p R(p),$$

where $p = (p_1, p_2, \ldots, p_M)$. The objective function described in Equation (1) depends on the Nash equilibrium N^* of the follower's non-cooperative game. Then, before optimizing his objective function, the leader has to compute the Nash Equilibrium N^* depending on his decision parameters. This kind of problem has been explored in transport networks and is called *bi-level optimization* [10,11].

We propose in the following sections to apply this hierarchical game model for designing optimal mechanisms in communication networks. First, we study an optimal pricing and scheduling mechanism in a wired network with service differentiation and second, we propose an optimal power and pricing mechanism in a wireless CDMA network.

3 Optimal Pricing and Scheduling

The notion of service differentiation is an important key for providing QoS in communication networks and has been proposed as the key concept for designing the *DiffServ* architecture. The term “service differentiation” usually carries the implicit meaning of offering enhanced services.

3.1 Less-Than-Best Effort Services

Less-than-best-effort (LBE) has been proposed as a service for carrying non-critical traffic. Some examples of application scenarios [1] where a LBE service may prove useful are:

– content mirroring and news distribution; new distributed applications that can take advantage of spare network capacity;
– non-time-critical, bulk-data transfer based on TCP;
– isolating production traffic from test traffic; isolating mission-critical traffic from other kinds of production traffic that may be disruptive (e.g., traffic generated from a student dormitory in a university campus).

The LBE concept has already been tested in academic research networks like Internet2 [2] and GÉANT [3]. Such studies have focused mainly on the impact of LBE on more-critical traffic, and on practical issues like router configuration.

Two kinds of schedulers have been proposed to handle LBE traffic [4,1,2]: strict, non-preemptive priority queueing (PQ) and weighted fair queueing (WFQ) or one of its variants, like for instance weighted round-robin [5]. From a theoretical standpoint, a WFQ-like scheduler can be regarded as a packet-level version of a Generalized Processor Sharing (GPS) server [6].

We assume that the network under consideration offers only two services: best-effort and less-than-best-effort, and that users are charged on a per-packet basis. The network is modeled as a single bottleneck node, here either a PQ server or a GPS server is used to handle two queues, one for packets marked as “BE” and another for packets marked as ”LBE”. Considering an heavy traffic regime, we approach GPS with several independent queues like in the PMP mechanism [9]. Then, there is no competition between classes, each type of traffic joins a queue depending only on the access price and the number of sources for class i is :

$$N_i^* \quad \text{such that} \quad u_i(N_i) = 0.$$

The system manager is then able to find the optimal scheduling in terms of profit. Our main conclusion described in details in [18] in a work in collaboration with D. Ros from the ENST Bretagne, is that a network offering two different services (i.e., BE and LBE) may yield higher revenues than a network with no service differentiation, and also that the type of scheduler used may play an important role in maximizing revenues. In particular, we have shown that:

1. Priority Queueing is more efficient, in economic terms, than both a GPS scheduler and a simple FIFO queue,
2. Revenues are lower with a GPS scheduler than with a FIFO queue.

3.2 Pricing TCP Sessions

Another important scheduling proposed in the networking control literature is Discriminatory Processor Sharing (DPS). Indeed, it provides a more flexible way of giving priority than the PQ discipline. It has been introduced by Kleinrock [12] for a single server. It consists in serving classes in proportions controlled by weights, while packets within a given class are served according to a standard processor sharing strategy (which constitutes the main difference with respect to GPS, where packets within a class are served according to a FIFO scheme) [13]. A theoretical charm of DPS is that, unlike GPS, a closed-form solution of steady-state delay exists [14]. DPS has been justified in practice, at the flow level, as a fluid approximation of some weighted round-robin scheme [20]. It has also been shown in [15,16] that a queue with DPS discipline is well adapted, as an approximation, to the way TCP connections can share bandwidth. It uses the fact that, at the session level, TCP can be analyzed using a PS approach [17]. We thus use DPS at the flow level here in order to fit the TCP modelling [15,16].

We consider Poisson flow arrivals, the number of flows in progress behaves like the number of customers in an M/M/1 processor sharing queue [12]. Moreover, as we assume two different classes of traffic, the model behaves like an M/M/1 discriminatory processor sharing queue. Let μ be the service rate of the server/router.

There exists a nonnegative parameter γ representing the *relative priority* of data customers and $1-\gamma$ for voice customers. Still, when packets of one class are not present in the queue, the server is fully allocated to the other class, but flows within a class are served according to a processor sharing (PS) scheme. A closed-form formula for the average delays in such M/M/1 queues are given in [19, page 86] by

$$\mathbb{E}\mathrm{D_v} = \frac{\left(1 + \frac{\lambda_\mathrm{d}\mathrm{N_d}(2\gamma-1)}{\mu-(1-\gamma)\lambda_\mathrm{v}\mathrm{N_v}-\gamma\lambda_\mathrm{d}\mathrm{N_d}}\right)}{\mu - \lambda_\mathrm{v}\mathrm{N_v} - \lambda_\mathrm{d}\mathrm{N_d}} \quad \text{and} \quad \mathbb{E}\mathrm{D_d} = \frac{\left(1 - \frac{\lambda_\mathrm{v}\mathrm{N_v}(2\gamma-1)}{\mu-(1-\gamma)\lambda_\mathrm{v}\mathrm{N_v}-\gamma\lambda_\mathrm{d}\mathrm{N_d}}\right)}{\mu - \lambda_\mathrm{v}\mathrm{N_v} - \lambda_\mathrm{d}\mathrm{N_d}}. \quad (3)$$

The steady-state average numbers of sessions of each type has to ensure that utilities are positive or null, otherwise the number of sessions naturally decreases. Following the same line, this average number increases until the utility approaches 0. It means that each type of application naturally adapts its steady-state number of sources in the sense the maximum (mean) number of sources of a given type cannot make negative its residual utility. Hence, we have a game between the different types of applications, for the maximum mean number of sessions, potentially leading to a Nash equilibrium. Let us now investigate the existence and uniqueness of this equilibrium.

We obtain that, at equilibrium, N_v and N_d must verify

- If $N_d, N_v > 0$, then $u_d(N_d, N_v) = u_v(N_d, N_v) = 0$,
- If $N_d > 0,\ N_v = 0$, then $u_d(N_d, N_v) = 0,\ u_v(N_d, N_v) \leq 0$,
- If $N_d = 0,\ N_v > 0$, then $u_d(N_d, N_v) \leq 0,\ u_v(N_d, N_v) = 0$.

Figure 1 illustrates these equations, where the maximum number of customers of each type in the network is necessarily on the "minimum" of curves

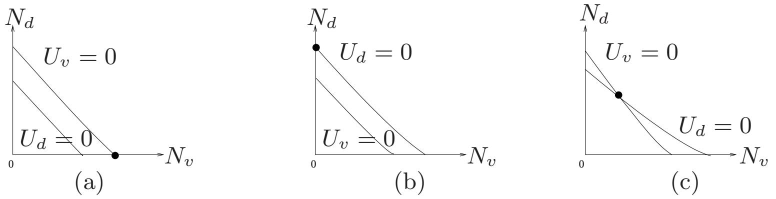


Fig. 1. Curves for the (maximum) number of customers of each type, and the associated Nash equilibrium. The three possible cases are displayed. The resulting Nash equilibrium is displayed by the thick point.

$u_d(N_d, N_v) = 0$ and $u_v(N_d, N_v) = 0$. Indeed, the mean number of sources of each type increases, decreasing then the utilities until a residual utility reaches zero. Figure 1 depicts the three situations that will be used later on: either the curves cross each other in the domain $\{(N_d, N_v) : N_d, N_v \geq 0\}$, or one curve is always under the other in this domain.

It is important to note that at optimal prices, the optimal γ is then 0, giving then strict priority to voice traffic. Actually, whatever the choices of parameters we have, this result has been verified by extensive simulations that cannot be reported here for lack of space. This is somehow in accordance with [18] where we have shown for a simple M/M/1 queue that PQ has to be preferred to GPS to optimize the revenue. In the present case, a formal proof of this conjecture has still to be found.

This study is detailed in [26]. In the next section, we consider this model in a wireless context of a UMTS network based on CDMA.

4 Control of Wireless CDMA Transmissions

Pricing has also been used in CDMA networks ([21,22,23]) by using their specificities: the price charged to a user is computed in terms of the QoS degradation imposed to others by this user, the so-called externality. This can be shown to directly depend on the transmission powers through the interferences. This generally leads to a game-theoretical analysis and price optimization. We consider here a different view of CDMA network control where prices do not depend on power or interference levels, but simply on the volume of transmitted data.

The model we propose is inspired by the one in [24], where an optimal resource allocation scheme was obtained among different classes of users, but for a fixed and pre-determined number of users in each class. In [24], power is controlled to reach the given thresholds of signal-to-interference plus noise ratio (SINR) for which QoS requirements are met. A processing gain exhibiting good performance is computed. We consider here that the processing gain is fixed for each class of service.

We focus on the reverse link of a single cell. We consider a DS-CDMA network where the chip rate R_c is assumed to be equal for all users. We assume a multiclass system, with C classes, where a user is characterized by class i. When packets are sent, they enter a buffer after error control coding through forward error correction (FEC), and they are converted to a DS-CDMA signal at symbol rate R_c/n_i, with n_i being the processing gain (which should not be larger than $R_c/(\lambda_i L_i)$). L_i is the length in terms of symbols of packet of class i and λ_i is the rate of the (Poisson) arrival stream of packets. We assume perfect power control, that is, for each i, the signal transmission power is controlled by the base station such that it is received at level P_i. The choice of n_i and received power P_i at the base station affects packet delay and transmission rate. This has been extensively discussed in [24]. Note that this also affects the performance of other classes of users. So, we fix the values of n_i to those giving good performance in [24]. We consider that a new packet is generated as soon as the preceding one is successfully delivered. This is referred to as *continuously active users*, which might represent the transmission of long files for instance.

In DS-CDMA, a key parameter is the received signal-to-interference plus noise-ratio (SINR). QoS metrics such as delay and bit error probability depend directly on it. For class i users, the SINR is

$$SINR_i = \frac{P_i n_i}{\gamma \left(\sum_{k=1}^{N_i - 1} P_i + \sum_{j \neq i}^{M} \sum_{k=1}^{N_j} P_j \right) + \sigma^2}, \tag{4}$$

where γ is a constant which depends on the shape of DS-CDMA chips, N_j is the number of class j connections and σ^2 is the background noise power.

For all classes of traffic, we assume that channel coding includes forward error correction (FEC). We assume that the bit error probability (BEP) is an exponentially decaying function of the SINR. Specifically, we assume that for a user in class i, the BEP is

$$p_{b_i} = \mathcal{F}(SINR_i),$$

with $\mathcal{F}(x) = \exp(-\beta x)$. Similarly, thanks to the assumption that each user's on/off indicator is independent from symbol to symbol, the probability of retransmission is

$$p_{r_i} = 1 - [1 - \mathcal{F}(SINR_i)]^{L_i r_i} \tag{5}$$

with r_i the FEC code rate for class i.

Performance measures can be directly expressed in terms of the SINR. Consider for instance the mean packet delay $\mathrm{I\!ED_i}$ for type-i traffic. It is composed of the mean waiting time in the queue $\mathrm{I\!EW_i}$ and the mean retransmission time $\mathrm{I\!ES_i}$, $\mathrm{I\!ED_i} = \mathrm{I\!EW_i} + \mathrm{I\!ES_i}$. It is shown in [24] that

$$\mathrm{I\!ED_i} = \frac{\mathrm{L_i n_i}}{\mathrm{R_c(1 - p_{r_i})}}. \tag{6}$$

On the other hand, base stations also have constraints on capacity. As stated in [25], for dynamic range limitations on the multi-access receiver and to guarantee system stability, the total received noise plus interference power to background noise ratio is limited for a class-i user to

$$\frac{\gamma \left(\sum_{k=1}^{N_i - 1} P_i + \sum_{j \neq i}^{M} \sum_{k=1}^{N_j} P_j \right) + \sigma^2}{\sigma^2} \leq \frac{1}{\eta}, \tag{7}$$

where η is typically 0.25 or 0.1. This inequality provides an upper-bound on the number of users for each class, for fixed received powers.

Selfish class i users apply for service as soon as their utility u_i, described in Equation (2), is positive. Demand is thus directly controlled by prices and reception powers, so that it potentially leads to a (Nash) equilibrium on the number of active users. Then, for each class, either the number of sources is zero with negative utility (meaning that no user has interest in participating), or is equal to capacity with positive utility (meaning that no more users are allowed to enter for physical reasons), or the number of sources is positive and less than capacity, with null utility (meaning that the users' cost reach their valuation and no other user has interest in entering, since it would lead to a negative utility). Formally, an equilibrium is a tuple (N_v^*, N_d^*) such that $N_v^*, N_d^* \geq 0$ and $\forall i, j \in \{d, v\}$, $j \neq i$:

- Either $N_i^* = 0$ and $u_i(1, N_j^*) < 0$;
- or N_i^* has reached the capacity constraint (7) (so the inequality becomes an equality) and $u_i(N_v^*, N_d^*) > 0$;
- or $N_i^* > 0$, under the capacity constraint (7), and $u_i(N_v^*, N_d^*) = 0$ so that no other user has an incentive to join (a potentially leaving user being immediately replaced by a new one).

Considering only one class of traffic, the number of users at equilibrium is:

$$N^* = \begin{cases} 0, & \text{if} \quad p > p_{max} = f(\frac{nP}{\sigma^2}), \\ 1 - \frac{\sigma^2}{\gamma P} - \frac{n}{\gamma f^{-1}(p)}, & \text{if} \quad f(\frac{nP\eta}{\sigma^2}) < p < f(\frac{nP}{\sigma^2}), \\ 1 + \frac{1-\eta}{\eta}\frac{\sigma^2}{\gamma P}, & \text{otherwise.} \end{cases}$$

For several classes of traffic, the equilibrium and the leader optimization problem are more complicated and details results are described in [29] in a work in collaboration with V. Ramos from the UAM.

5 Conclusion and Perspectives

Hierarchical games are (surprisingly) rarely used for telecommunication modeling. In this paper we have shown, however, that in a context of usage-based pricing mechanism, this type of games give important results for the optimization of network control parameters. The only statement which used this approach is the model presented in [27]. Our bi-level game is different because it is based on a typical user behavior which determines the equilibrium. It should be important to consider other user dynamics like population dynamics proposed in the Evolutionary Game Theory [30]. It may also be interesting to consider bi-level optimization algorithms like ones proposed in [31] in a context of communication networks.

Acknowledgement

I would like to thank my thesis's advisor, Dr. B. Tuffin, who has directed my works summarized in this paper.

References

1. T. Chown, T. Ferrari, S. Leinen, R. Sabatino, N. Simar, and S. Venaas, “Less than Best Effort: Application Scenarios and Experimental Results,” in *Proceedings of QoS-IP 2003*, ser. Lecture Notes in Computer Science, no. 2601, 2003, pp. 131–144.
2. “QBone Scavenger Service (QBSS),” Internet2 QBone Initiative, `http://qbone.internet2.edu/qbss/`.
3. “Analysis of Less-than-Best-Effort Services,” TF-NGN LBE working group, `http://www.cnaf.infn.it/~ferrari/tfngn/lbe/`.
4. R. Bless, K. Nichols, and K. Wehrle, “A Lower Effort Per-Domain Behavior for Differentiated Services,” Internet Draft draft-bless-diffserv-pdb-le-01.txt, work in progress, Nov. 2002.
5. Z. Wang, *Internet QoS — Architectures and Mechanisms for Quality of Service.* Morgan Kaufman Publishers, 2001.
6. A. K. Parekh and R. G. Gallager, “A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single Node Case,” *IEEE/ACM Transactions on Networking*, vol. 1, no. 3, pp. 344–357, June 1993.
7. C. Courcoubetis and R. Weber, *Pricing Communication Networks—Economics, Technology and Modelling.* Wiley, 2003.
8. M. Mandjes, “Pricing Strategies under Heterogeneous Service Requirements,” in *proceedings of IEEE Infocom*, 2003.
9. A. Odlyzko, “Paris Metro Pricing for the Internet,” in *ACM Conference on Electronic Commerce (EC’99)*, 1999, pp. 140–147.
10. J. Bard, *Practical Bilevel Optimization: Algorithms and Applications*, Kluwer, 1999.
11. G. Anandalingam and T. Friesz, ”Hierarchical Optimization: an introduction”, *Annals of Operations Research*, vol. 34, 1992.
12. L. Kleinrock, ”Time-shared Systems: A Theoreticol Treatment” *Journal of the Association of Computing Machinery*, vol. 14, 1967.
13. K. Rege, and B. Sengupta, ”Queue-length Distribution for the Discriminatory Processor-Sharing Queue”, *Operations Research*, vol. 44, 1996.
14. G. Fayolle, I. Mitrani, and R. Iasnogorodki, ”Sharing a Processor Among Many Job Classes”, *Journal of the Association of Computing Machinery*, vol. 27, 1980.
15. E. Altman, T. Jimenez, and D. Kofman, ”DPS Queues with Stationary Ergodic Service Times and the Performance of TCP in Overload”, in *proceedings of IEEE Infocom*, 2004.
16. T. Bu, D. Towsley, ”Fixed Point approximations for TCP Behavior in an AQM Network”, *in proceedings of ACM Sigmetrics*, 2001.
17. S. Ben Fredj, T. Bonald, A. Proutiere, G. Regnie, and J. Roberts, ”Statistical Bandwidth Sharing: a Study of Congestion at Flow Level”, *in proceedings of IEEE Sigcomm*, 2001.
18. Y. Hayel, D. Ros and B. Tuffin, ”Less-than-Best Effort Services: Pricing and Scheduling”, *in proceedings of IEEE Infocom*, 2004.
19. R. Hassin, and M. Haviv, ”To Queue or Not To Queue”, Kluwer’s, 2003.
20. D. Mitra, and A. Weiss, ”A Closed Network with a Discriminatory Processor-Sharing Server”, *in proceedings of ACM Sigmetrics*, 1989.
21. P. Marbach, and R. Berry, ”Downlink Resource Allocation and Pricing for Wireless Networks”, *in proceedings of IEEE Infocom*, 2002.
22. C. Saraydar, N. Mandayam, and D. Goodman, ”Efficient Power Control via Pricing in Wireless Data Networks” *IEEE transactions on Communications*, vol. 50, no. 2, 2002.

23. V. Siris, and C. Courcoubetis, "Resource Control for Loss-Sensitive Traffic in CDMA Networks", *in proceedings of IEEE Infocom*, 2004.
24. J. Kim, and M. Honig, "Resource Allocation for Multiple Classes of DS-CDMA Traffic", *IEEE Transactions on Vehicular Technology*, vol. 49, no. 2, 2000.
25. A. Viterbi, "CDMA. Principles of Spread Spectrum Communication", Addison-Wesley, 1995.
26. Y. Hayel, and B. Tuffin, "Pricing for Heteregeneous at a Discriminatory Processor Sharing Queue", *in proceedings of IFIP/Networking Conference*, 2005.
27. T. Basar, and R. Srikant, "A Stackelberg Network Game with a Large Number of Followers", *Journal of optimization theory and applications*, vol. 115, no. 3, 2002.
28. A. Charnes, and S. Sorensen, "Hierarchical Games", Research Report, Texas University, 1971.
29. Y. Hayel, V. Ramos, and B. Tuffin, "Optimal Static Pricing of Reverse-link DS-CDMA Multiclass Traffic", *in proceedings of QEST*, 2006.
30. J. Weibull, "Evolutionary Game Theory", MIT Press, 1997.
31. J. Clegg, and M. Smith, "Cone Projection versus Half-space Projection for the Bilevel Optimisation of Transportation Networks", *Transportation Reserach B*, vol. 35, 2001.

Transit Prices Negotiation: Combined Repeated Game and Distributed Algorithmic Approach*

Dominique Barth[1] , Johanne Cohen[2], Loubna Echabbi[3,**], and Chahinez Hamlaoui[1]

[1] PRiSM, 45, avenue des etats-Unis 78035 Versailles Cedex, France
[2] lORIA, campus scientifique, BP 239, F-54506 Vandœuvre les Nancy, France
[3] INPT, Madinat Al Irfane, avenue allal el Fassi, Rabat, Maroc
echabbi@inpt.ac.ma

Abstract. We present both a game theoretic and a distributed algorithmic approach for the transit price negotiation problem in the interdomain routing framework. The analysis of the centralized transit price negotiation problem shows that the only one non cooperative equilibrium is when the lowest cost provider takes all the market. The perspective of the game being repeated makes cooperation possible while maintaining higher prices. We consider then the system under a realistic distributed framework and simulate its behaviour under a simple price adjustment strategy and analyse whether it matches the theoretical results.

Keywords: interdomain routing, repeated games, distributed algorithmic.

1 Introduction

Today inter-domain market plays on two different time scales: A long term time scale (months or even years) where economic contracts are negotiated and a short term (seconds) where routing decisions are made based on the concluded business relationships. Some recent works [1,4,5] propose to couple those two processes more tightly by enabling a more dynamic interaction between transit price propositions and routing decisions. In order to capture the dynamic aspect of such interaction, authors of these papers propose to employ a repeated game approach. The repeated game framework enables to capture how the threat of a future behaviour can impact the current actions of players.

In [1], the repeated routing game is introduced and a price matching strategy is proposed and analysed. The difference between the analysis in this work and our proposal is that we consider that the traffic dedicated to a given destination can be routed only through a single provider. This assumption is made in order to maintain a coherence with the Border Gateway Protocol where only one path

* This work was supported by The RNRT ACTRICE and the ANR SOGEA french projects.
** Corresponding author.

is chosen to each destination. In our proposal, we still consider an interaction between transit price negotiation and routing decision process. The bilateral economic nature of the Internet is still maintained by a cascade like pricing where each agent negotiates low prices only with his immediate neighbours. This is different from the source based pricing approach taken in [1]. We propose an adequate model to capture the different dimensions of the problem then we focus on the analysis of the related game on some specific scenario mainly by considering the simple but not simplistic case of one source and one destination. In the game analysis, we assume a full knowledge of the different parameters of the problem, which is not very realistic but gives an idea about the nature of the game. Further, we will analyse the problem from a distributed point of view taking into account realistic considerations.

2 The Transit Price Negotiation Model

The network is given by a graph $G(N, E, cost)$ where N represents the ASs, $cost$ is the unitary cost related to managing the transit over the AS intra-network and E are the physical inter-domain links. We will focus on an only one traffic flow between a node S which is the source of the traffic, it can be an access network and the node $Dest$ its destination. The rest of the nodes are providers. A random variable P represents the period on which inputs (graph, traffic Matrix) are stable. We assume that P follows an exponential law with mean D, that can be obtained by some statistical knowledge or stochastic analysis. We consider that the source has an upper bound on price under which she accepts to send the traffic. Otherwise, she does not send the traffic. We will denote it p_{max}. We consider discrete transit prices. Price discretisation depends on the encoding format in control packets, for instance here we take a unit discretization. That is provider transit price can take values as $1, 2, 3, \ldots$.

During the period P, the inputs of the problem are stable and a stationary environment game can model interactions between ASs during the transit price negotiation. Each AS announces its transit price to his neighbours with the corresponding route into the destination. When an AS decides to buy a route from its neighbour, he can itself announce this route to his own neighbours while proposing an adequate transit price. Thus, the negotiation follows a cascade like model from the destination backward to the source, where each AS in the path plays both the customer and the provider role. The objective of each provider is clearly to maximize its own benefit by proposing attractive transit prices but also by choosing itself the lowest providers. In case of identical announces, an AS can choose a provider following a pre-order on his providers. Our goal is to analyse equilibrium situations where ASs do not have the incentive to deviate from their proposed prices and to check whether such situations are beneficial to the sender (the source).

The game proceeds in series of stages of identical duration d a constant of common knowledge. Then d/D models the probability of the game coming to an end. We consider δ such that $d/D = 1 - \delta$ the probability that the game is still taking place. Hence, the game arrives to stage k with probability δ^k. At the start of each stage : Each player advertises its per packet price . We suppose that each AS is aware of the history of the game; that is after each stage all ASs are aware of proposed prices and consequent outcomes on the previous stages.

A powerful notion while analysing a repeated game is the *subgame perfect equilibrium*[6], where the played strategies represent a Nash equilibrium in each subgame. That is given any history of the game given by past plays, the adopted strategies still represent a Nash equilibrium trough the rest of the game. A set of strategies can be proved to induce a subgame perfect equilibrium if they satisfy the **one deviation principle**. This principle ensures that no player can increase its utility by deviating from its original strategy at a single stage. The intuition behind this principle is that improving the utility of a player supposes that at least at one stage the pay-off obtained by deviating is greater than the one in the original strategy. Thus, in order to prove that the set of strategies form a subgame perfect equilibrium, it is sufficient to prove that they satisfy the one deviation principle.

Now, let us analyse our game under these considerations on a simple scenario. First, we consider the simple case where there is a single communication in a network of 4 vertices, one source and one destination and two intermediate providers. We consider that providers have identical costs. There exists an analogy with the Bertrand game [3]. Bertrand game models interactions between duopoly firms that propose homogeneous products and compete only on price. The consumers buy all products from the cheaper firm or half at each when the price is equal. In the Bertrand game, firms are supposed to have the same marginal cost, when the customer demand is supposed to be linear in the price. A monopoly price p is given and represents the price that the firm will charge if she had the monopole on the market. In our simple scenario providers have identical costs when demand is constant. The monopole price is given by p_{max} since it is the maximum price that can be charged.

There are two possible outcomes in Bertrand competition : Both firms decide to not cooperate and price the only non-cooperative Nash equilibrium which is to charge the marginal cost c. Indeed for each price p_1 proposed by firm1, the best response of firm2 consists for every $p_1 > c$ in lowering slightly the price to win the market. The only one equilibrium is (c,c). Note that when marginal costs are different, the firm with lower marginal cost can win all the market. Otherwise, both firms can cooperate and charge the monopoly price p and thus share the market. Since BGP requires a single routing, splitting the traffic can be done on time by alternating $(p_{max}, p_{max}+1)$ announces. Hence, for instance player 1 wins over even stages and player 2 over odd ones. In order to make this threat credible a punishment should be added to avoid deviations. Thus, a possible strategy can be to alternate $(p_{max}, p_{max}+1)$ announces and if player 1 for instance deviates by playing $p' = p_{max} - 1$[1] in stage $2k$ then player 2 plays $p' - 1$ in stage $2k + 1$. This strategy satisfies the one stage deviation principle for sufficiently patient players.

[1] p' should be lower than p_{max} in order to win the game but the higher possible to make the maximum benefit. Given the unit discretisation, that price is $p_{max} - 1$.

Indeed, suppose without loss of generality that player 1 decides to deviate by playing $p_{max} - 1$ in an odd stage $2k$ then player 2 will play $p_{max} - 1$ in the stage $2k + 1$. Considering an only one stage deviation, the game will continue with the original strategy. Then pay offs will differ only in stage $2k$ and $2k + 1$. In stage $2k$ player 1 wins with price $p - 1$ and in stage $2k + 1$ he will loose which give him a total benefit of $p_{max} - 1$.[2] When in the original strategy he would get a total pay-off of p_{max}. The expected improvement is given by discounting the pay off by the probability of the game taking place at the corresponding stage. Hence the deviation is profitable iff: $(p_{max} - 1) * \delta^{2k} > p_{max} * \delta^{2k+1}$ that is when $\delta \geq \frac{p_{max}-1}{p_{max}}$ the one deviation principle is satisfied for sufficiently patient players and the proposed strategy is then a subgame perfect equilibrium. Note, that this strategy is profitable to both providers but is not profitable to the source since it induces a flip-flop like routing. This behaviour can easily be generalized to the case of n providers with identical costs.

Let us consider now the case of one source and one destination with n possible intermediate providers having different costs. For the purpose of simplicity, let us assume that $cost_1 < cost_2 < \ldots < cost_n$ where $cost_i$ is the cost of provider i.[3] The utility of the provider is the difference between its price and its cost if he wins and 0 otherwise. Again, similarly with the Bertrand game the only one non cooperative equilibrium is when the lowest cost provider (here provider 1) takes all the market by proposing $cost_2 - 1$. Again the perspective of the game being repeated makes cooperation possible in order to maintain higher prices and strategies can be constructed with the same intuition to prevent deviations. However, given that costs are different, many cooperations are possible. For example, provider 1 can announce a price in $[cost_2..cost_3 - 1]$[4] in order to invite provider 2 to join him and share the market. Actually, they can cooperate by both setting the price at $cost_3 - 1$ the maximum price such that they can get all the market. In such a situation, we will talk about coalition and denote the set of providers joining it $coalition_2 = \{1, 2\}$. And so on, provider 1 can set a price in $[cost_{i-1}..cost_i - 1]$ in order to invite provider $i - 1$ to join him and share the market then forming $coalition_i = \{1..i\}$.

Obviously $coalition_i \subset coalition_j$ iff $i < j$ that is if a provider j can join a given coalition then every provider $i < j$ can do. Also, each provider i can only join a $coalition_j$ where $j >= i$. Note that if providers $i = 1, \ldots, j$ decide to cooperate thus forming $coalition_j$ and given that they are all utility maximisers they should announce $cost_{j+1} - 1$. Hence, we will talk about strategy of joining coalition j when the strategy consists on setting price equal to $cost_{j+1} - 1$.

Now, the question that arises is which coalition will be chosen and would all providers necessary to form it have actually incentive to join it. Let denote s_i^j

[2] Of course multiplied by the amount of traffic, but here we consider without loss of generality a unit of traffic.

[3] When $cost_1 \leq cost_2 \leq \ldots \leq cost_n$, we obtain the same results, we need just to consider class of providers having the same cost.

[4] Integer values in the corresponding interval.

the strategy of player i that consists in choosing to join $coalition_j$ where $j \geq i$. This can be done by provider i setting a price $p_i^j = cost_{j+1} - 1$.

The utility of a player i when choosing $coalition_j$ is given by:

$$u_i(s_i^j, s_{-i}) = \begin{cases} 0 & \text{if } \exists i', j' < j \text{ s.t } s_{i'} = s_{i'}^{j'} \\ \frac{p_i^j - cost_i}{|\{i'/s_{i'} = s_{i'}^j\}|} & \text{otherwise} \end{cases}$$

That is, the utility of provider i is 0 if another provider proposes a lower price, otherwise he shares the market with the other providers that have proposed the same price as him. Hence the utility of player i choosing a $coalition_j$ given that all other providers $i' \leq j$ have also join that coalition is $(p_i^j - cost_i)/j$. Each player is expected to choose the coalition that maximizes such utility. We will denote the corresponding strategy (price announced) s_i^*.

For instance for provider 1 $s_1^* = max\{cost_2 - cost_1 - 1, \ldots, \frac{cost_n - cost_1 - 1}{n-1}, \frac{p_{max} - cost_1}{n}\}$ and we denote $coalition_{j^*}$ the corresponding coalition. Now, the question is whether providers $\{2, \ldots, j^*\}$ will choose to join the same coalition. That is $coalition_{j^*}$ is the coalition that maximizes their utility too. The answer is given by the following theorem:

Theorem 1. *If $coalition_{j^*}$ is the coalition that maximizes the first provider utility then it maximizes providers $2, \ldots, j^*$ utilities:*

$$\forall i' \in \{2, \ldots, j^*\} \quad s_{i'}^* = s_{i'}^{j^*}$$

The proof skipped due to space limit can be found in the research report [2].

That is the lowest cost provider chooses his preferred coalition and the involved providers follow him. When different best coalitions are possible (with the same utility) a problem of coordination can arise. A dominant strategy for the lowest cost provider is to choose the lowest coalition and for the other members to follow him. When a player k deviates players $1, .., k-1$ punish him by playing according to $coalition_{k-1}$. For sufficiently patient players, this is a subgame perfect equilibrium. The intuition, is that the punishment will exclude the deviator from the coalition for the rest of the game.

Let us consider now the case where instead of direct connection, providers are connected via disjoint routes to the source. Let us denote i the direct provider connected to the destination, i' the corresponding provider connected to the source and l_i the length of the route between i and i'. Without loss of generality we consider $l_1 < l_2 < \ldots < l_n$. The benefit of a provider is the difference between the price at which he has bought the route (the price of his provider) and the price at which he proposes the route to his customer. The net benefit is obtained by subtracting the transit cost. For simplicity assume first that there are no transit costs. That is a customer is interested to buy a route if at least he can make a benefit of 1.

The game can be separated into two different games: the sequential game that each provider i plays with his predecessors on the route to s and the simultaneous game that players $i = 1 \ldots n$ are playing in order to fix their price.

This separation is possible and relevant only because paths are disjoint. Indeed on each route, players between i and i' including i' are completely dependent on i the owner of the route toward the destination. This game is known as the ultimatum game [6] where some value is to be divided between some players. A given player (called the first mover) proposes a division of the value and the others can only accept the division or refuse it inducing utility of 0 to everyone. The optimal strategy for the player proposing the division is to take the maximum portion and let to the others the minimum such they are still interested (in a continuous setting ϵ and in a discrete frame setting 1) which is an Nash equilibrium. The player who is proposing the division has an advantage because he is the first mover in the sequential game. In our case, if there is for example only one route, the provider 1 is the first mover because he announces a route to the destination first and he should propose the route at $p_{max} - l_1$ letting each of the other intermediate providers get a benefit of 1 (the route is proposed then to the source at p_{max}). The following providers will then accept since they prefer to get a benefit of 1 rather than to lose the market.

When there are several routes the provider i has to fix his price depending on the simultaneous game he is playing with the other direct providers. Providers directly connected to the destination ($\{1, \ldots, n\}$) have to take into account that each provider on the correspondent path should at least make a benefit of 1. Hence each of the n possible routes can be proposed to the source at least at $l_i + 1$ by each corresponding i'. The problem can be viewed then as n providers proposing to connect directly the source and the destination as in the former case with each provider i having a cost $l_i + 1$. The lowest cost provider who has the market power is the one on the shortest path. As we have argued above, he chooses his optimal coalition and the other involved providers follow him. Intermediate providers have to propose the price at which they have bought the route $+1$, otherwise their route will not be chosen by the source. Note that when internal transit costs are not null then we can obtain the same results by considering as metric the sum of transit costs. We give an example that help to understand how the situation can be different in the general case from the special cases.

Example 1. Let us consider the network given in Fig. 1. Suppose that the source has a $p_{max} = 8$. Provider 1 prefers to join *coalition*$_2$ since $(6/2 > 8/3)$ and provider 2 follows him. They will then propose $p_1 = 6$ and $p_2 = 5$. Provider 4

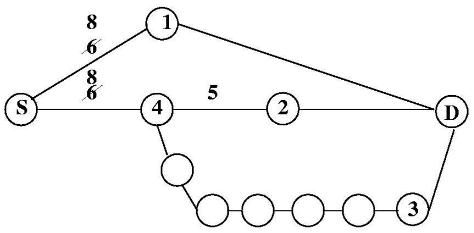


Fig. 1. Cases of non disjoint routes between the source and the destination

should announce 1 to insure that the second route will be chosen. His advantage is that he blocks the third route. he can propose a coalition to provider 1 where they can improve their benefit. The first and the second route still shares the market but at a higher price (p_{max}) as depicted in Fig 1. Provider 2 conserves his benefit and have non incentive to punish provider 1 or 4. Provider 4 acts as a stopper of the third route and thus can propose a second coalition that improve his benefit without decreasing benefit of provider 2.

This is the intuition we have used to propose an algorithm that computes prices in the general case [2]. It consists in computing successive coalitions to improve intermediary providers benefit while respecting precedent coalitions.

3 The Dynamic Distributed Game

In this section we try to analyse how the system behaves in a distributed framework. Indeed, in reality nodes have only a local view of the game including the topology and thus the nature and the length of the possible routes. We simulate the distributed game and investigate if some specific local strategies can lead to a similar results than the one expected by the theoretical analysis. For this purpose, we need to introduce first the distributed algorithmic model. The system is still modeled by a graph linking the source S and the destination $Dest$ with m nodes labeled $i \in \{1 \dots n \dots m\}$ where the direct nodes are $\{1 \dots n\}$. We denote $Successor(i)$ the set of possible providers of node i and $Predecessor(i)$ the set of possible customers of the node i. A Node i is characterized by the following variables: **Current price** per unit of traffic denoted p_i which is announced by the node i to his neighbours in $Predecessor(i)$, **Current Provider** denoted $provider(i)$ which is one of node's neighbours that can reach the destination. It is the one who proposes the best price to i. For $j = provider(i)$ we have $p_j \leq p_k \ \forall k \in Successor(i)$, **State** denoted $state(i)$ that indicates whether the node is crossed by the transit traffic (O) or not (N). That means that the node belongs to the chosen route. For the special case of the source the state indicates whether the source has received at least an acceptable route (its price is lower than p_{max}) or not. We define the set $Customer(j) = \{k / provider(k) = j\}$. We have $p_i > p_{provider(i)} \ \forall i$ that is a node proposes a route at a price higher than the price at which he has bought it. Every node chooses the provider that proposes the lowest route price. If a node chooses a provider j and thereafter it receives from a provider k a proposal of a route with lower price it switches toward k.

In a distributed setting, each node is informed of all the variables of its neighbours using traffic control. However all routes may not be visible at every node: the set of routes learned at one node depends on route selection at its provider. Node's state depends on the route chosen by the source. At the beginning each node's state is equal to N because routes are not established yet. Node's state is updated when he receives a state update message from its neighbours as following: state(i)= O if $\exists j \in Customer(i)$ such that $state(j) = O$ or if $i = S$ and $p_{provider(S)} \leq p_{max}$. and N otherwise. Hence when the source chooses an acceptable route, its state changes to O and then it sends an update message to its provider who in turns changes his state and so on until the destination. When the source switches on a new received route with a better price, the state of nodes on the new route is updated iteratively into O when the state of the nodes on the old route is updated iteratively into N.

We propose to test a simple strategy that all the nodes can use to update their price depending on their state: if $state(i) = O$ then $p_i \longleftarrow p_i + 1$ otherwise $state(i) = N$ and then if $(p_i - p_{provider(i)}) > 1$ then $p_i \longleftarrow p_i - 1$.

The intuition behind this strategy is that providers with no transit traffic decrease their prices in order to attract the traffic. Each provider accepts to transit the traffic if he has at least a benefit of 1 and does not decrease its price under this limit. When a provider gets the transit traffic, he tries to increase his price in order to reach the maximum possible benefit.

4 Simulation Analysis

Our objective here is to study the stabilizing behaviour of the distributed system under the above price adjustment strategy and whether it matches the theoretical results. Simulation is done using OMNET [7] simulator. We consider different topologies (Fig. 2). Links have the same propagation delay equal to 0.31 ms. Neither queuing nor scheduling delays are considered in the simulation. Node state messages are generated automatically when the node state is updated and are sent as traffic control messages. In our simulation, the stage game duration is $d = 50ms$. We implement the simple price adjustment strategy explained above and consider different scenarios:

Scenario 1. We consider topology 2 and simulate the price adjustment strategy when transit prices starts from a high price (chosen $> p_{max} = 20$ as depicted in Fig. 3 at the left side. Then prices are adjusted until t= 150 ms (stage 4) where routes proposed to the source become acceptable. Both routes share the market but at a higher price. Direct providers have an advantage over intermediate ones, the first one taking the maximum benefit.

When a direct provider chooses to start at a price lower than p_{max} as in the scenario depicted in Fig. 3 at the right side then his route is selected during

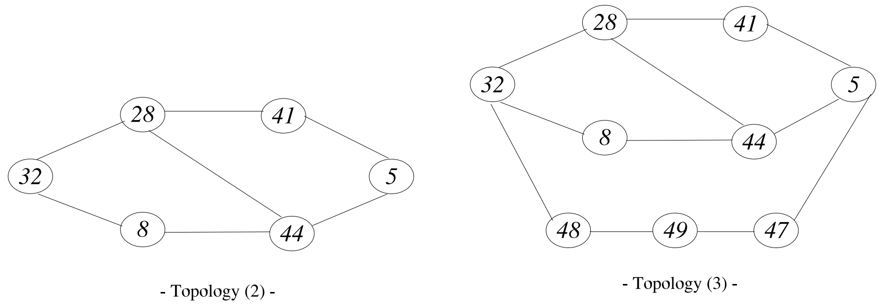


Fig. 2. The different simulated topologies

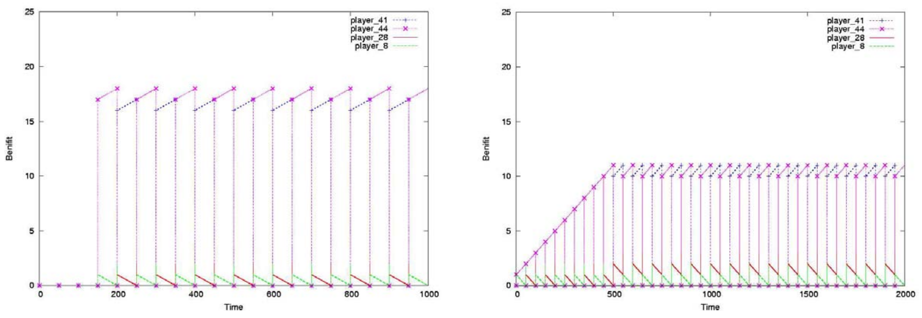


Fig. 3. Providers benefit in scenario 1

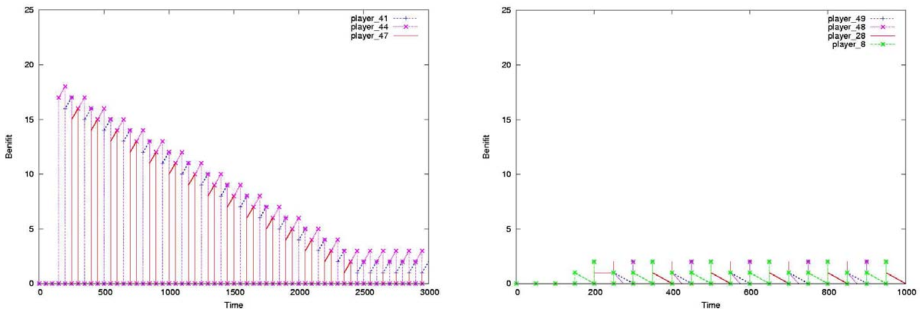


Fig. 4. Direct providers and intermediary providers benefits in scenario 2

few steps. Prices are then adjusted until a situation where both routes share the market. Note that provider 41 and 44 would have better benefit with the precedent scenario where they started both at p_{max}. In summary, this simple strategy gives similar results to those expected by the theoretical analysis using only local information. Indeed, when providers start from high prices they can share the market while maintaining higher prices than when they do not cooperate.

Scenario 2. We consider now topology 3, where three different direct providers compete for the market. We simulate the price adjustment strategy where all transit prices starting from $p_{max} = 20$. Direct providers benefits and intermediate providers benefits are depicted in Fig 4. As with topology 1, prices are adjusted until step = 150 ms (stage 4) where routes proposed to the source become acceptable. However, direct providers do not succeed in maintaining high prices. This can be explained by the way prices are updated. Indeed, the three routes share the market but each provider lowers it price at least two times (when the other routes are chosen) but increases only one time its price. This leads the prices to decrease drastically. We need a more elaborated strategy in order to obtain a behaviour which is similar to the theoretical expected behaviour.

5 Conclusion

We present a combined game theoretic and distributed algorithmic approach to the transit price negotiation problem. We highlight situations where cooperation is possible in order to maintain higher prices. However such situations lead to a flip flop routing. An interesting issue is to investigate how the source can avoid such behaviour for example by adding some penalties when its provider changes its price. A more elaborated local strategies are currently tested mainly based on stochastic learning of optimal strategies. Finally we are investigating how to generalize the proposed approach to a network where there are different sources and destinations for the traffic while considering coherent routing.

References

1. M. Afergan. Using repeated games to design incentive-based routing systems. In 25th Conference on Computer Communications, IEEE INFOCOM, 2006.
2. D. Barth, J. Cohen, L. Echabbi, and C. Hamlaoui. Transit price negotiation : a repeated game approach. In PRiSM Report 2006/76, www.prism.uvsq.fr, 2006.
3. J. Bertrand. Theorie mathematique de la richesse sociale,. Journal des Savants, 67:499 – 508, 1883.
4. J. Feigenbaum, C. Papadimitriou, R. Sami, and S. Shenker. A BGP-based mechanism for lowest-cost routing. In Proceedings of the 2002 ACM Symposium on Principles of Distributed Computing., 2002.
5. R. La and V. Anantharam. Optimal routing control: Repeated game approach. IEEE Transactions on Automatic Control, 47(3):437 – 450, 2002.
6. Martin J. Osborne. An Introduction to Game Theory. Oxford University Press, USA, 2003.
7. A. Varga. The omnet++ discrete event simulation system. In In European Simulation Multiconference, 2001.

Cost Minimisation in Multi-interface Networks*

Ralf Klasing[1], Adrian Kosowski[2], and Alfredo Navarra[1,3]

[1] LaBRI - Université Bordeaux 1 - CNRS, 351 cours de la Liberation, 33405 Talence cedex, France
{Ralf.Klasing,Alfredo.Navarra}@labri.fr

[2] Department of Algorithms and System Modeling, Gdańsk University of Technology, Narutowicza 11/12, 80952 Gdańsk, Poland
kosowski@sphere.pl

[3] Dipartimento di Matematica e Informatica, Universitá degli Studi di Perugia, Via Vanvitelli 1, 06123 Perugia, Italy
navarra@dipmat.unipg.it

Abstract. The paper focuses on the problem of minimisation of energy consumption by wireless devices. Since wireless communications are some of the main causes of battery drainage, connections must be carefully established. We study complexity issues of the so called Cost Minimisation in Multi-Interface Networks problem. Given a graph $G = (V, E)$ with $|V| = n$ and $|E| = m$, which models a set of wireless devices (nodes V) connected by multiple radio interfaces (edges E), the aim is to switch on the minimum cost set of interfaces at the nodes in order to satisfy all the connections. Every node holds a subset of all the possible k interfaces. A connection is satisfied when the endpoints of the corresponding edge share at least one active interface. We distinguish two main variations of the problem by treating the cost of maintaining an active interface as uniform (i.e., the same for all interfaces), or non-uniform. In general, we show that the problem is APX-hard while we obtain an approximation factor of $\min\{\lceil\frac{k+1}{2}\rceil, \frac{2m}{n}\}$ for the uniform case and a $(k-1)$-approximation for the non-uniform case. Next, we concentrate our attention on several classes of networks: with bounded degree, planar, with bounded treewidth and complete.

1 Introduction

While technology advances and more powerful devices are released, special effort is required for managing new kinds of communication problems. Nowadays wireless devices hold multiple radio interfaces, allowing switching from one communication network to another according to required connectivity and related quality. The selection of the "best" radio interface for a specific connection might depend on various factors. Namely, its availability in specific devices, the required communication bandwidth, the cost (in terms of energy consumption) of maintaining an active interface, the available neighbours and so forth. While

* The research was partially funded by the European projects COST Action 293, "Graphs and Algorithms in Communication Networks" (GRAAL) and, COST Action 295, "Dynamic Communication Networks" (DYNAMO).

managing such connections, a lot of effort must be devoted to energy consumption issues. Devices are, in fact, usually battery powered and the network survivability might depend on their persistence in the network. This introduces a challenging and natural optimisation problem that must take care of different variables at the same time. Generally speaking, given a set of k interfaces and a graph $G = (V, E)$, where V represents the set of wireless devices and E the set of required connections according to proximity of devices and the available interfaces that they may share, the problem can be stated as follows. What is the cheapest way, i.e., which subset of available interfaces in each node must be activated in order to satisfy (cover) all the connections described by E while minimising the overall cost? Note that a connection is satisfied when the endpoints of the corresponding edge share at least one active interface.

We call such a problem Cost Minimisation in Multi-Interface Networks (k-CMI for short). In this paper, we study the complexity of k-CMI in various scenarios. k-CMI turns out to be a very hard problem in general, hence we also consider possible approximation algorithms. We deal with two main variations of the problem: the case in which the cost of activating an interface is the same for each interface, and the more general case in which such a cost may be different. Indeed, the first model is equivalent to asking for the minimum total number of activated interfaces inside the network in order to cover all the connections. We also consider different graph classes that are of interest from both theoretical and practical points of view, namely: with bounded degree, since in real world scenarios users are normally connected to a limited number of nodes; planar, since the induced graph of joining users in a network is likely to be planar; trees, since MiddleWare strategies are heavily based on this kind of structure (see for instance [1]); complete graphs, since this is one of the main structures used for modelling P2P networks (see for instance [2]).

In contrast to its natural importance, to date there appear to be no known complexity results of k-CMI. Conversely, from a practical point of view, efficient algorithms dealing with cost minimisation in multi-interface networks are strongly required [3,4]. Indeed, our research starts from [4] where a slightly different model of k-CMI is introduced. That model considers also the possibility of having mutually exclusive interfaces, i.e., interfaces that, if activated, preclude the activation of some other interfaces. The motivation is quite technical, for instance the WiFi interface can operate in different modalities: Infrastructure and Ad-Hoc. If a device activates WiFi in the Infrastructure modality, it cannot satisfy connections that require the Ad-Hoc modality and vice versa. In this paper we have not introduced this further restriction since the problem is already of practical relevance and not easily solvable. Table 1 summarises our obtained results. Due to space limitations, some of the proofs have been omitted and will appear in the full version of the paper.

Outline: The next section provides some definitions and notation in order to formally describe the k-CMI problem. Section 3 contains the results concerning the hardness, and Section 4 gives approximation algorithms for the k-CMI problem in general graphs. Section 5 is devoted to the hardness and the

Table 1. Hardness and approximability of the k-CMI problem

Graph class	Interfaces	Complexity of k-CMI	
		non-uniform costs	uniform costs
General graphs	$k = 2$	$O(n^3)$	$O(nm)$
	$k \geq 3$	$(k-1)$-approx, APX-hard	$\min\{\lceil \frac{k+1}{2} \rceil, \frac{2m}{n}\}$-approx, APX-hard
Graphs of bounded Δ	$k \geq 3$	Δ-approx, APX-hard for $\Delta \geq 5$	$\frac{\Delta+1}{2}$-approx, APX-hard for $\Delta \geq 5$
Planar graphs	$k \geq 3$	NP-hard, $PTAS$	NP-hard, $PTAS$
Trees	any k	$O(n)$	$O(n)$
Complete graphs	any k	$O(n^2)$	$O(n^2)$

approximation factors of k-CMI with respect to various classes of graphs. In particular, we consider graphs with bounded degree, planar, with bounded treewidth and complete. Finally, Section 6 contains conclusive remarks and a discussion of interesting open problems.

2 Definitions and Notation

Unless otherwise stated, the network graph $G = (V, E)$ is always assumed to be simple (i.e., without multiple edges), undirected and connected. Moreover, we always denote by n and m the cardinality of the sets V and E respectively. The degree of node $v \in V$ is denoted by Δ_v and the set of its neighbours by $N(v)$. The minimum node degree of graph G is denoted by δ, and its maximum node degree by Δ.

A global characterisation of interfaces of respective nodes from V is given in terms of an appropriate interface assignment function W, according to the following definition.

Definition 1. *A function $W\colon V \to 2^{\{1,\ldots,k\}}$ is said to* cover *graph $G = (V, E)$ if for each $\{u, v\} \in E$ the set $W(u) \cap W(v) \neq \emptyset$.*

The cost of activating an interface for a node is assumed to be identical for all nodes and given by cost function $c\colon \{1, \ldots, k\} \to \mathbb{N}$, i.e., the cost of interface i is written as c_i. The considered k-CMI optimisation problem is formulated as follows.

k-CMI: Cost Minimisation in Multi-Interface Networks

***Input*:**	A graph $G = (V, E)$, an allocation of available interfaces $W\colon V \to 2^{\{1,\ldots,k\}}$ covering graph G, an interface cost function $c\colon \{1, \ldots, k\} \to \mathbb{N}$.
***Solution*:**	An allocation of active interfaces $W_A\colon V \to 2^{\{1,\ldots,k\}}$ covering graph G such that $W_A(v) \subseteq W(v)$ for all $v \in V$.
***Goal*:**	Minimise the total cost of the active interfaces, $c(W_A) = \sum_{v \in V} \sum_{l \in W_A(v)} c_l$.

In all further considerations, the problem instance is stated in the form of the triple (G, W, c).

3 Hardness in General Graphs

Theorem 1. *The optimal solution to* 2-CMI *can be found in* $O(n^3)$ *steps.*

Proof. Let (G, W, c) be the considered instance of 2-CMI. For any $I \subseteq \{1, 2\}$, let $V_I \subseteq V$ denote all those nodes whose set of available interfaces is exactly I, and let $V_{A\,I}$ denote all those nodes whose set of active interfaces in solution W_A is exactly I (formally, $V_I = W^{-1}(I)$ and $V_{A\,I} = W_A^{-1}(I)$). The sought solution W_A may be equivalently stated in the form of a partition $V = V_{A\{1\}} \cup V_{A\{2\}} \cup V_{A\{1,2\}}$. Conversely, a given partition $V = V_{A\{1\}} \cup V_{A\{2\}} \cup V_{A\{1,2\}}$ provides a correct and optimal solution to k-CMI if and only if the following conditions are fulfilled:

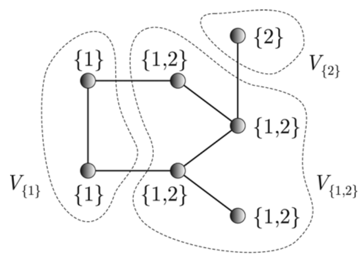
1. All nodes with only one interface available activate it, i.e. $V_{\{1\}} \subseteq V_{A\{1\}}$ and $V_{\{2\}} \subseteq V_{A\{2\}}$.
2. No two nodes $u \in V_{A\{1\}}$ and $v \in V_{A\{2\}}$ may be connected by an edge of the graph, $\{u, v\} \notin E$.
3. The value of the cost expression $c(W_A) = c_1|V_{A\{1\}}| + c_2|V_{A\{2\}}| + (c_1 + c_2)|V_{A\{1,2\}}|$ is minimised.

In particular, the fulfillment of condition 1 is equivalent to the requirement that $W_A(v) \subseteq W(v)$ for all $v \in V$, the fulfillment of condition 2 implies that all edges from E are covered by W_A, while condition 3 is a restatement of the k-CMI minimality requirement.

It now suffices to show an efficient algorithm for finding a partition $V = V_{A\{1\}} \cup V_{A\{2\}} \cup V_{A\{1,2\}}$ which fulfills conditions 1–3. In order to achieve this, consider the following modification of graph G, remembering that V can be partitioned into the disjoint union $V_{\{1\}} \cup V_{\{2\}} \cup V_{\{1,2\}}$ of nodes with different available sets of interfaces.

In the first step, identify all nodes of G belonging to $V_{\{1\}}$ into one distinguished node s_1, and all nodes of G belonging to $V_{\{2\}}$ into one distinguished node s_2, appropriately modifying all adjacent edges (if set $V_{\{1\}}$ or $V_{\{2\}}$ is empty, the respective node s_1 or s_2 is simply inserted as an isolated vertex). Next, for all the remaining nodes of G, $v \in V_{\{1,2\}}$, add two copies of v labelled v' and v'' as isolated nodes to the set of nodes of the graph, and insert the edges $\{v, v'\}, \{v', s_1\}, \{v, v''\}, \{v'', s_2\}$ into its edge set, Figure 1. The new graph is now denoted as $\overline{G} = (\overline{V}, \overline{E})$; for convenience, we write $\overline{V} = \{s_1, s_2\} \cup V_{\{1,2\}} \cup V'_{\{1,2\}} \cup V''_{\{1,2\}}$. Let us recall that a *node cutset* between nodes s_1 and s_2 in graph $\overline{G}$ is any set of nodes $S \subseteq \overline{V} \setminus \{s_1, s_2\}$, such that any path between s_1 and s_2 contains at least one node from S; we make the following claim.

Claim 1. In graph $\overline{G}$, there exists a node cutset S between s_1 and s_2, such that $|S \cap V_{\{1,2\}}| = p$, $|S \cap V'_{\{1,2\}}| = p'$, $|S \cap V''_{\{1,2\}}| = p''$, and $|S \cap \{v, v', v''\}| \leq 1$ for all $v \in V_{\{1,2\}}$, if and only if in graph G there exists a partition of V into



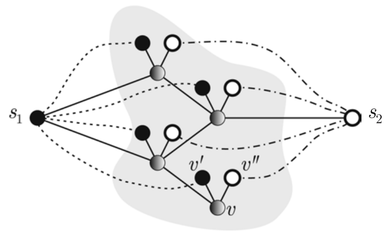


Fig. 1. On the left, an exemplary instance of 2-CMI, together with the node set partition $V = V_{\{1\}} \cup V_{\{2\}} \cup V_{\{1,2\}}$. On the right, the corresponding graph $\overline{G}$.

sets $V_{A\{1\}}, V_{A\{2\}}, V_{A\{1,2\}}$ *which fulfill conditions* 1 *and* 2 *and* $|V_{A\{1\}}| = |V_{\{1\}}| + p'', |V_{A\{2\}}| = |V_{\{2\}}| + p', |V_{A\{1,2\}}| = p$.

The claim implies that the problem of finding a partition of V with minimum cost $c(W_A)$ is equivalent to finding the specified node cutset in $\overline{G}$, for which the value of the following expression is minimised:

$$\begin{aligned} C(W_A) &= c_1|V_{A\{1\}}| + c_2|V_{A\{2\}}| + (c_1+c_2)|V_{A\{1,2\}}| = \\ &= c_1(|V_{\{1\}}| + p'') + c_2(|V_{\{2\}}| + p') + (c_1+c_2)p = \\ &= (c_1|V_{\{1\}}| + c_2|V_{\{2\}}|) + c_1p'' + c_2p' + (c_1+c_2)p \end{aligned}$$

Since the value of $(c_1|V_{\{1\}}| + c_2|V_{\{2\}}|)$ depends only on the problem instance and not on the choice of S, it may be discarded as constant, and a simpler cost function can be minimised instead:

$$\overline{c}(S) = c_1p'' + c_2p' + (c_1+c_2)p$$

However, this minimisation criterion can be imposed by a simple node weight function. Let $\mu : \overline{V}\backslash\{s_1, s_2\} \to \mathbb{N}^+$ be defined as follows: $\mu(v) = c_1+c_2$ for all $v \in V_{\{1,2\}}$, $\mu(v') = c_2$ for all $v' \in V'_{\{1,2\}}$, and $\mu(v'') = c_1$ for all $v'' \in V''_{\{1,2\}}$. Clearly, $\overline{c}(S) = \sum_{u\in S} \mu(u)$, and a cutset S minimising $\overline{c}(S)$ can be found by reduction to the weighted maximum flow problem [5]. In general, a solution may therefore be obtained in $O(n^3)$ time by applying the push-relabel algorithm [6], whereas in the case of unit cost interfaces $O(nm)$ time complexity can be achieved by means of the Ford-Fulkerson algorithm [5]. It now only remains to be shown that such a cutset S can be converted into a cutset fulfilling the condition $|S \cap \{v, v', v''\}| \leq 1$, for all $v \in V_{\{1,2\}}$, required in the claim, without affecting the value of $\overline{c}(S)$. Indeed, note that if v belongs to S, then both v' and v'' can always be removed from S without any consequences. On the other hand, if $v' \in S$ and $v'' \in S$, then the set $S^* = S \setminus \{v', v''\} \cup \{v\}$ is also a valid cutset, and $\overline{c}(S^*) = \overline{c}(S)$. By repeating the procedure, after a linear number of steps we obtain a cutset in $\overline{G}$, which, by applying the method from the proof of the claim, can in linear time be converted into a partition $V_{A\{1\}} \cup V_{A\{2\}} \cup V_{A\{1,2\}}$ of V. Knowing this partition, the final solution to k-CMI can be returned after an immediate linear-time post-processing step. □

Theorem 2. *For any $k \geq 3$, k-CMI is APX-hard even when restricted to instances of maximum degree $\Delta = 5$, and unit cost interfaces.*

Proof. We consider a polynomial transformation from the well-known Vertex Cover problem on subcubic graphs to k-CMI. On those instances Vertex Cover is known to be APX-hard [7]. Given a subcubic graph $G = (V, E)$, it is known that in general its chromatic number is at most three [8]. We can then partition its nodes into three subsets V_1, V_2 and V_3 according to an optimal coloring in such a way that $V_1 \bigcup V_2 \bigcup V_3 \equiv V$ and for each edge $e = \{x, y\} \in E$, x and y do not belong to the same subset V_i for every $i = 1$, 2 or 3.

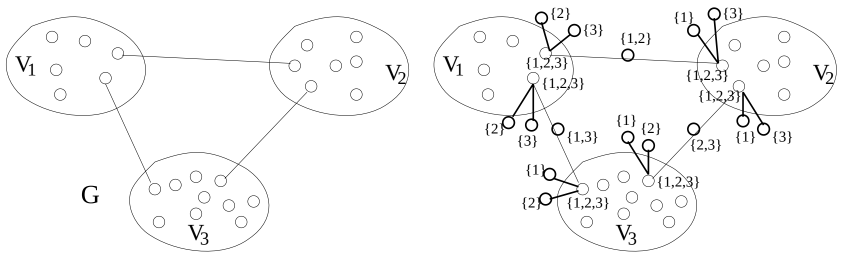


Fig. 2. On the left, the graph G subdivided into three node subsets according to a 3-coloring and the three possible kind of edges. On the right the modifications obtained for each kind of edge belonging to G and the interfaces associated to the related nodes.

As illustrated in Figure 2, with each node $v \in V$ we associate three interfaces, namely 1, 2, and 3. Moreover, to each $v \in V$ we connect two new nodes. Those new nodes have only one interface: 2 and 3 (1 and 3 or 1 and 2) respectively if $v \in V_1$ ($v \in V_2$ or $v \in V_3$). For each edge of G we add a further node. With such a node we associate two interfaces. If the considered edge connects V_1 and V_2 (V_1 and V_3 or V_2 and V_3) then we associate interfaces 1 and 2 (1 and 3 or 2 and 3) to the added node. Without loss of generality, let us consider an edge $e = \{x, y\} \in E$ such that $x \in V_1$ and $y \in V_2$. In order to solve k-CMI on the new graph of maximum degree 5 built from G, we necessarily have to activate interfaces 2 and 3 in x, and 1 and 3 in y. In order for both x and y to be able to communicate with the new intermediate node, either such a node must activate both its interfaces or one among x and y has to activate its third available interface. Since we are in the unit cost interface context, both the solutions are locally equivalent. On the other hand, activating the third interface for either x or y may lead to a decrease of the number of activated interfaces in the global solution. This is implied by the fact that the neighbourhood of the added intermediate node between x and y is constituted by only x and y, while both x and y may have many other connections. So, we can always look for solutions where one among x and y has all its three interfaces activated. In order to conclude the proof, it remains to show that if a solution for k-CMI can be found on the new graph, then a solution for Vertex Cover on G can be easily obtained. From the above discussion, in fact, we have shown that for each edge $e \in E$, at least one of its endpoints must have all its interfaces activated. When this happens we say the node belongs to the Vertex Cover and the claim holds. $\square$

4 Approximation Algorithms

Theorem 3. *Given a graph $G = (V, E)$, k-*CMI *is approximable within a factor of $k - 1$.*

Proof. In order to achieve such an approximation we describe a simple algorithm that greedily activates interfaces among the nodes. It starts from the cheapest interface 1, and it activates it in each node that has a neighbour holding that interface. Let $V_1 \subseteq V$ be the set of nodes in which the algorithm activated interface 1 and let $E(V_1)$ be the corresponding set of covered edges. Note that the optimal solution restricted to $E(V_1)$ (i.e., the set of activated interfaces of an optimal solution at the endpoints of the edges belonging to $E(V_1)$) clearly costs at least as much as the cost of our algorithm. In the second step, the same is done for the next cheapest interface 2 among the remaining connections $E \setminus E(V_1)$. Again, the cost of the optimal solution restricted to $E(V_2)$ is at least the cost paid by our algorithm. This is implied by the fact that any connection belonging to $E(V_2)$ cannot be covered by interface 1 otherwise the algorithm would have covered it in the previous step. This process is continued for all the interfaces in a non-decreasing cost order but for the last two interfaces. When the two most expensive interfaces remain, in fact, we can apply the optimal algorithm of Theorem 1. Since each step costs at most as much as the optimal solution, the claim then holds by observing that the whole process requires $k - 1$ steps. $\square$

Theorem 4. *Given a graph $G = (V, E)$, k-*CMI *is approximable within $\lceil \frac{k+1}{2} \rceil$ in the case of unit cost interfaces.*

Proof. Consider the following algorithm for assigning a set of active interfaces $W_A(v) \subseteq W(v)$ for each node $v \in V$.

1. Compute a minimum hitting set of interfaces $W_H(v)$ which is potentially sufficient for communication with all neighbouring nodes, i.e. a set $W_H(v)$ such that $W_H(v) \subseteq W(v)$, $\forall_{u \in N(v)} W_H(v) \cap W(u) \neq \emptyset$, and the cardinality of $W_H(v)$ is minimised.
2. If $|W(v)| > \lceil \frac{k+1}{2} \rceil$, then let $W_A(v)$ be any subset of $W(v)$ such that $W_A(v) \supseteq W_H(v)$ and $|W_A(v)| = \max\{|W_H(v)|, \lceil \frac{k+1}{2} \rceil\}$.
3. If $|W(v)| \leq \lceil \frac{k+1}{2} \rceil$, then let $W_A(v) = W(v)$.

Clearly, for each node v we have $|W_A(v)| \leq \max\{|W_H(v)|, \lceil \frac{k+1}{2} \rceil\}$, whereas in the optimal solution the number of active interfaces is at least equal to $W_H(v)$. Consequently, the obtained solution is a $\lceil \frac{k+1}{2} \rceil$-approximation. In order to prove the correctness of the solution, it suffices to show that for each edge $\{u, v\} \in E(G)$ we have $W_A(u) \cap W_A(v) \neq \emptyset$. Observe that:

– if $|W(u)| \leq \lceil \frac{k+1}{2} \rceil$ and $|W(v)| \leq \lceil \frac{k+1}{2} \rceil$, then $W_A(u) = W(u)$, $W_A(v) = W(v)$, and $W(u) \cap W(v) \neq \emptyset$ by definition of the instance of k-CMI;
– if $|W(v)| > \lceil \frac{k+1}{2} \rceil$ and $|W(u)| \leq \lceil \frac{k+1}{2} \rceil$, then $W_A(u) = W(u)$, $W_A(v) \supseteq W_H(v)$ and $W_H(v) \cap W(u) \neq \emptyset$;
– finally, if $|W(v)| > \lceil \frac{k+1}{2} \rceil$ and $|W(u)| > \lceil \frac{k+1}{2} \rceil$ then $|W_A(v)| \geq \lceil \frac{k+1}{2} \rceil$, $|W_A(u)| \geq \lceil \frac{k+1}{2} \rceil$ and $W_A(v) \subseteq \{1, \ldots, k\}$, $W_A(u) \subseteq \{1, \ldots, k\}$;

thus in all cases the sought condition $W_A(u) \cap W_A(v) \neq \emptyset$ is fulfilled. □

Theorem 5. *Given a graph $G = (V, E)$, k-CMI is approximable within $\frac{2m}{n}$ in the case of unit cost interfaces.*

Proof. The algorithm simply chooses one interface for each edge in order to satisfy the connection. This means that for each edge at most one interface in each endpoint is activated. It follows that for m edges it activates at most $2m$ interfaces for n nodes. □

5 Results for Various Graph Classes

5.1 Graphs with Bounded Maximum Degree

Let us recall that a graph is called *d-degenerate* for some value of parameter d if all its subgraphs have a node of degree at most d, $d \geq \min_{H \subseteq G} \delta(H)$.

Theorem 6. k-CMI *is approximable within $d + 1$ for d-degenerate graphs.*

Proof. By a simple characterisation [9], in linear time it is possible to order the node set of a d-degenerate graph in the form of a sequence $s = \{v_1, \ldots, v_n\}$, such that $\forall_{1 \leq i \leq n} |\{v_1, \ldots, v_{i-1}\} \cap N(v_i)| \leq d$. Consider an algorithm which assigns the set $W_A(v)$ to successive nodes of V according to sequence s. Initially, all the sets $W_A(v)$ are empty; in the first step, set $W_A(v_1)$ remains empty. In the i-th step, for $i \geq 2$, we define $W_A(v_i) \subseteq W(v_i)$ in such a way that $\forall_{1 \leq j < i, v_j \in N(v_i)} W(v_j) \cap W_A(v_i) \neq \emptyset$ and the value of the cost expression $\sum_{l \in W_A(v_i)} c_l$ is minimised; moreover, for all nodes $v_j \in N(v_i)$, $1 \leq j < i$, set $W_A(v_j)$ is augmented by at most one interface of minimum possible cost to guarantee that $W_A(v_j) \cap W_A(v_i) \neq \emptyset$. Once the process is complete, the obtained function W_A is clearly a correct solution to k-CMI. Observe that in the i-th step, $i \geq 2$, set $W_A(v_i)$ is assigned a cost value not greater than that of the optimal interface assignment for node v_i, and that the cost of only at most d other sets $W_A(v_j)$, $1 \leq j < i$, is additionally increased by a value not exceeding the cost of $W_A(v_i)$. The cost of the obtained solution is thus not greater than $(d + 1)(C_{\text{OPT}} - C_{\text{OPT}\,v_1})$, where C_{OPT} denotes the total cost of an optimal solution for graph G and $C_{\text{OPT}\,v_1}$ is the cost of some optimal solution for node v_1, and the proposed algorithm is clearly a $(d + 1)$-approximation. □

For any graph G of maximum degree $\Delta \geq 2$, there exists an edge $e = \{u, v\} \in E$ such that graph $G \setminus \{e\}$ is $(\Delta - 1)$-degenerate, with a corresponding sequence of nodes such that $u = v_1$ and $v = v_n$ [8]. The procedure from the proof of Theorem 6 can be applied for graph $G \setminus \{e\}$, leading to an interface assignment of cost at most $\Delta(C_{\text{OPT}} - C_{\text{OPT}\,v_1})$ for $G \setminus \{e\}$. Since the cost of additionally enabling communication on edge e using the cheapest possible interface shared by nodes u and v is at most $2C_{\text{OPT}\,v_1} \leq \Delta \cdot C_{\text{OPT}\,v_1}$, the total cost of the obtained solution remains not greater than $\Delta \cdot C_{\text{OPT}}$.

Corollary 1. k-CMI *is approximable within Δ for graphs of maximum degree Δ.*

For the case of unit cost interfaces the following theorem provides an improved approximation ratio.

Theorem 7. k-CMI *is approximable within $\frac{\Delta+1}{2}$ for graphs of maximum degree Δ, in the case of unit cost interfaces.*

5.2 Planar Graphs and Graphs of Bounded Treewidth

For planar graphs, the *NP*-hardness of k-CMI can be shown by analogy to the proof of Theorem 2, since the Vertex Cover problem remains *NP*-hard for planar graphs of maximum degree 3 [10]. However, the optimisation criterion of k-CMI is local, i.e. the validity and local optimality of a solution for a given node can be verified in polynomial time only by analysing its neighbourhood. Taking this into account, k-CMI for planar graphs is easily shown to admit a polynomial time approximation scheme (PTAS), constructed according to the general approach of Baker [11].

Corollary 2. *For planar graphs,* k-CMI *is NP-hard, but admits a PTAS.*

Similarly, the locality of the k-CMI optimisation criterion makes it possible to apply a dynamic programming technique described by Bodlaender [12] to solve k-CMI optimally and in linear time for the class of graphs with bounded treewidth, which includes trees, outerplanar graphs, and series-parallel graphs.

Corollary 3. *For any constant $t \in \mathbb{N}^+$,* k-CMI *can be optimally solved in $O(n)$ steps for graphs of treewidth t.*

5.3 Complete Graphs

Theorem 8. *Given a complete graph K_n of n nodes,* k-CMI *is optimally solvable in $O(n^2)$ steps.*

Proof. We divide the set of nodes into classes according to the available interfaces they have. In this way, there will be at most 2^k classes. Since the graph is complete, symmetry implies that every node belonging to the same class has the same subset of interfaces activated in the optimal solution. The maximum number of interfaces that a node can activate is of course k. Hence, by trying all the possible configurations (at most 2^k) for the available interfaces in each class, we can compute the optimal solution in $(2^k)^{2^k}$ steps by checking each time if all the edges are covered. □

6 Conclusion and Future Work

In this paper we have considered the Cost Minimisation in Multi-Interface Networks problem. After providing practical motivation for the study of the problem, we have concentrated our attention on problem hardness and approximation factors in general and more specific settings. The obtained results have shown that the problem is hard and approximation algorithms are highly relevant. This suggests the need for future study of better performing approximation algorithms or heuristics. Another very interesting issue would be to study the problem from a distributed point of view. Indeed, it is worth noting that the algorithm used to derive Theorem 7 can be easily implemented in a distributed setting. On the other hand, the algorithm proposed in Theorem 3 can be adapted to a k-approximation distributed algorithm. As also mentioned in the introduction, other slightly different models of the problem are of main interest in further investigations.

References

1. Caporuscio, M., Carzaniga, A., Wolf, A.L.: Design and evaluation of a support service for mobile, wireless publish/subscribe applications. IEEE Transactions on Software Engineering **29**(12) (2003) 1059–1071
2. Cilibrasi, R., Lotker, Z., Navarra, A., Perennes, S., Vitanyi, P.: About the lifespan of peer to peer networks. In: Proceedings of the 10th International Conference On Principles Of Distributed Systems (OPODIS). Volume 4305 of LNCS. (2006) 288–302
3. Adya, A., Bahl, P., Padhye, J., Wolman, A., Zhou, L.: Multi-radio unification protocol for IEEE 802.11 wireless networks. In: Proceedings of the 1st International Conference on Broadband Networks (BroadNets). (2004) 344–354
4. Caporuscio, M., Charlet, D., Issarny, V., Navarra, A.: Energetic Performance of Service-oriented Multi-radio Networks: Issues and Perspectives. In: Proceedings of the 6th International Workshop on Software and Performance (WOSP), ACM Press (2007) 42–45
5. Even, S.: Graph Algorithms. Computer Science Press (1979)
6. Goldberg, A.V., Tarjan, R.E.: A new approach to the maximum-flow problem. Journal of the ACM **35**(4) (1988) 921–940
7. Papadimitriou, C.H., Yannakakis, M.: Optimization, approximation, and complexity classes. Journal of Computer and System Sciences **43** (1991) 425–440
8. Brooks, R.L.: On coloring the nodes of a network. Proceedings of Cambridge Philosophical Society **37** (1941) 194–197
9. Matula, D.W., Beck, L.L.: Smallest-last ordering and clustering and graph coloring algorithms. Journal of the ACM **30**(3) (1983) 417–427
10. Garey, M.R., Johnson, D.S.: Computers and Intractability, A Guide to the Theory of NP-Completeness. W.H. Freeman and Company, New York (1979)
11. Baker, B.S.: Approximation algorithms for NP-complete problems on planar graphs. Journal of the ACM **41**(1) (1994) 153–180
12. Bodlaender, H.L.: Dynamic programming on graphs with bounded treewidth. In: Proceedings of the 15th International Colloquium on Automata, Languages and Programming (ICALP). Volume 317 of LNCS. (1988) 105–118

Minimum Transmission Energy Trajectories for a Linear Pursuit Problem

Attila Vidács[1] and Jorma Virtamo[2]

[1] Budapest University of Technology and Economics
Department of Telecommunications and Media Informatics
Magyar tudósok krt. 2, H-1117 Budapest, Hungary
vidacs@tmit.bme.hu

[2] TKK Helsinki University of Technology, Networking Laboratory
P.O. Box 3000, FIN-02015 TKK, Finland
jorma.virtamo@tkk.fi

Abstract. In this paper we study a pursuit problem in the context of a wireless sensor network, where the pursuer (i.e., mobile sink) trying to capture a pursuee (i.e., tracked object), moving with constant velocity, is always directly communicating with a sensor node in the very near proximity of the pursuee. Assuming that the sensor nodes can adjust their transmission power depending on the distance ρ between the pursuer and pursuee according to the usual power law $\rho^{-\alpha}$, the task is to find the optimal trajectory of the pursuer minimizing the total transmission energy. We approach this classical control theoretic problem by the method of dynamic programming. The cost function, describing the transmission cost with an optimal policy, factorizes into radial and angular functions. The partial differential equation governing the cost function can then be reduced to an ordinary differential equation for the angular function. This equation as well as the related optimal trajectories can be solved numerically. The qualitative behavior of the trajectories is also discussed. The trajectories are self-similar in the sense that any magnification of an optimal trajectory is also an optimal trajectory for different initial conditions.

1 Introduction

Pursuits are common in many areas, including predators that hunt for their preys, missiles that are heading towards their moving target, or a robot that is trying to reach (or at least get as close as possible to) its target to be monitored, etc. (see [1,2] and references therein). Technically speaking, in a pursuit one particle travels along a specified curve, while a second pursues it, with a motion directed towards the first. When the pursuer travels faster than the pursued, the question then becomes: “At what point do the two meet?” “What is the capture point?” Besides, an interesting study can be made if there is a cost associated with the pursuit, and the task is to reduce this cost as much as possible. For example, a trivial objective can be to catch the target as soon as possible. The answer is typically given by defining the optimal strategy to drive the pursuer, or equivalently, defining the optimal curve of pursuit that should be followed. The curve of pursuit is simply the trajectory traced by the pursuer.

A typical assumption is that the pursuer is always heading right towards the target, i.e., with the paths of the pursuer and pursuee parameterized in time, the pursuee is always on the pursuer's tangent. This is because the pursuee's trajectory is not known in advance, either because the pursuer is not able to predict it, or the pursuee is actively trying to avoid the pursuer by changing its direction and speed adaptively. However, we concentrate on the problem where the pursuee is moving constantly along a straight line irrespective to the pursuer's behavior, also called as *linear* pursuit. (An excellent overview of the history of pursuit curves is found in a series of articles written by Arthur Bernhart, among which the first discusses pursuit curves where the pursued moves along a straight line [3].) Moreover, in our model the cost rate is related with the actual distance of the pursuer from the pursuee during the chase. The total energy of the *entire* pursuit is to be minimized, with the consumed power at each step being proportional to a given power of the relative distance between the pursuer and pursuee.

We adopt this pursuit problem to a wireless sensor networking scenario. As an example, consider a sensor networking application where sensor nodes detect any moving object within their sensing range, and report on it to a sink node. (For a general description of wireless sensor networks, please refer to [4,5,6].) Here we assume a single-hop network where all sensors send radio packets directly to the sink. The most important source of energy leakage in this scenario is the energy needed for radio communication. We assume that the radio transmission power obeys the well-known power-law $\rho^{-\alpha}$ as a function of the distance ρ, and the nodes are able to adjust their transmission power as needed. Since alerted nodes report periodically, the consumed energy at each time is related to the distance between the sink and the moving object. A pursuit problem can be defined if we allow the sink node to move freely. The pursuer in this case is the mobile sink, while the tracked object takes the role of the pursuee. Since the mobile sink is constantly communicating with the sensor nodes that are sensing the object (or, less likely, directly with that object), in order to reduce energy consumption, the task is to find the optimal pursuit curve that leads to minimal energy and thus extended network lifetime. (For a detailed description on the energy consumption, sink mobility and network lifetime in wireless sensor networks, please refer to [7].)

The task leads to a classical (non-stochastic) control theoretic problem. Differential equations for a linear pursuit are sometimes applied, where the pursued starts at rest and then moves along a straight line. In the simplest case, where the pursuer is always heading directly towards the pursuee, the equation of motion for the pursuer is then solvable by first setting the first derivative equal to a particular point. However, in our case the cost function is nonlinear. We approach the problem with the dynamic programming approach of Bellman [8], [9]. The state of the system is defined by the relative position of the pursuer and the pursuee. Associated with the state there is the cost function which represents the minimal cost from that state to the end when optimal curve of pursuit is followed. An analogous, but for telecommunications people more familiar concept is 'distance vector' which represents the length of the shortest path from a given node to the destination. In this networking context, dynamic programming principle is well-known from the solution of the shortest-path problem using the Bellman-Ford algorithm [10]. In our pursuit setting, we derive a partial differential equation for the cost function. Assuming that the transmission power obeys the power-law $\rho^{-\alpha}$, the partial differential equation reduces to an ordinary differential equation that can be solved numerically. When the cost function is known, the optimal trajectories can easily be calculated. In particular, when the problem is to catch the pursuee in minimum time (i.e., $\alpha = 0$), one easily infers that the optimal policy is to head with full speed towards where the pursuee is going (the meeting point) along a straight line. When large distances are very costly in terms of transmission power (i.e., α grows), the nature of the trajectory changes. For very large α the optimal trajectory at any instant heads towards the pursuee's current position in order to decrease the distance as quickly as possible.

Another consequence of the power-law dependent transmission power is that the optimal pursuit curves are self-similar: given an optimal trajectory from a given initial point, magnifying the trajectory, i.e., multiplying the distance from the origin (pursuee) of each point of the trajectory by a constant yields the optimal pursuit curve starting from the point where the original initial point is sent by the magnification transformation.

The rest of the paper is organized as follows. In Section 2 the investigated linear pursuit problem is formulated, and a (non-linear) partial differential equation is derived for the optimal trajectory, using the dynamic programming method. The way how to solve this equation is also shown. Section 3 presents numerical results for different initial parameter settings. Finally, Section 4 concludes the paper.

2 Pursuit, Cost, Optimal Trajectory

2.1 Notation and Problem Formulation

Consider a linear pursuit game. Assume that the pursuee moves with constant speed v, and the maximal velocity of the pursuer is u. The pursuer is faster, thus the ratio $\nu = v/u$ is smaller than one. The positions of the pursuee and pursuer at time t are denoted by $\mathbf{s}(t)$ and $\mathbf{r}(t)$, respectively (see Fig. 1). In particular, we will assume that the pursuee moves along the x-axis at a constant velocity, i.e., $\mathbf{s}(t) = \mathbf{s}(0) + v\,t\,\mathbf{e}_1$. The *relative* position ($\boldsymbol{\rho}$) of the pursuer to the pursuee can be expressed as

$$\boldsymbol{\rho} = \mathbf{r} - \mathbf{s} = x\,\mathbf{e}_1 + y\,\mathbf{e}_2,$$

where $\mathbf{e}_1$ and $\mathbf{e}_2$ are the perpendicular unit vectors.

Assuming that the communication power (P) depends on the relative distance as

$$P = |\boldsymbol{\rho}|^{\alpha}, \quad \alpha \geq 0, \tag{1}$$

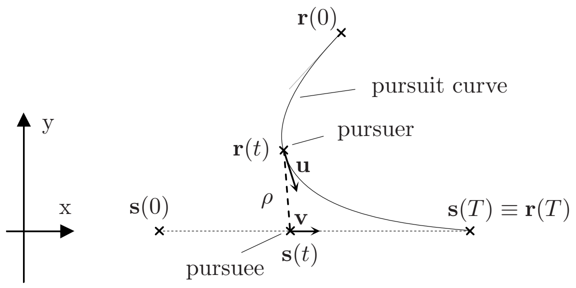


Fig. 1. Notations, pursuit curve

the task is to navigate the pursuer, i.e., specify the trajectory $\mathbf{r}(t)$ up to some time T when the pursuer catches the pursuee, so that the total energy consumed is minimized. Thus, the *minimal* energy consumption during the 'chase' is given by

$$\varepsilon(\boldsymbol{\rho}) = \min_{|\mathbf{u}(t)| \le u, \forall t} \int_0^T |\mathbf{r}(t) - \mathbf{s}(t)|^\alpha \, dt, \tag{2}$$

where $\mathbf{u}(t) = \frac{d}{dt}\mathbf{r}(t)$, $\boldsymbol{\rho} = \mathbf{r}(0) - \mathbf{s}(0)$ and $\mathbf{r}(T) = \mathbf{s}(T)$. Our task is to find $\mathbf{u}(t)$ that realizes the minimum of (2).

Recalling the wireless sensor networking application mentioned earlier, the pursuit can be interpreted as follows. Assume that a mobile object is moving across the sensor field with a constant speed (see Fig. 2). Sensor nodes within

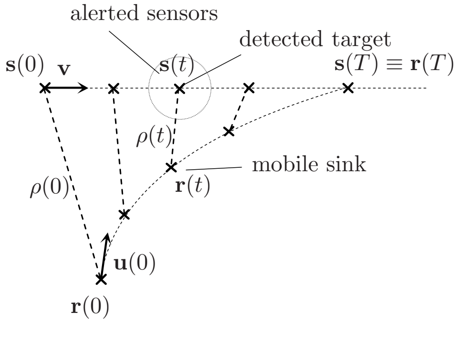


Fig. 2. Minimal energy 'chase' trajectory

sensing range that detect the object in its very near proximity send packets to the sink node via their radio interface. We assume that the nodes are aware of their actual distance from the sink, and are able to adjust their radio transmission power according to (1) to reduce energy consumption. (For example, since we do not have any restriction on the sink node, we can assume that it is capable of broadcasting its position periodically to every node. Another solution would be that—instead of receiving the coordinates from the pursuer—the received signal strength could be used at the node to estimate the pursuer's distance from the node.) Assuming a mobile sink, the task is to find an optimal trajectory for the sink to minimize the *overall* energy consumption (i.e., (2)) of the network.

2.2 Catch in Minimum Time

In the case $\alpha = 0$ the transmission power is constant and the objective reduces to catching the pursuee in minimum time. It is easy to see that then optimal strategy for the pursuer is to go with maximal speed along a straight line to the point where it reaches the pursuee. If the pursuee at time $t = 0$ is at the origin, then at time t it is at point $(vt, 0)$. This point is reached by a pursuer at time t from all the points that lie on a circle with center $(vt, 0)$ and radius ut, see Fig. 3. The cost function $\varepsilon(\boldsymbol{\rho})$ at point $\boldsymbol{\rho} = (x, y)$ is then the time t that solves the equation

$$\sqrt{(x - vt)^2 + y^2} = ut.$$

The solution is

$$u\,\varepsilon(\boldsymbol{\rho}) = \frac{\sqrt{x^2 + (1 - \nu^2)\,y^2} - \nu x}{1 - \nu^2} = \rho\,\frac{\sqrt{1 - \nu^2 \sin^2\theta} - \nu\cos\theta}{1 - \nu^2}, \tag{3}$$

where in the latter form we have used polar coordinates, where $\rho = |\boldsymbol{\rho}|$ and θ is the angle between $\boldsymbol{\rho}$ and $\mathbf{e}_1$. Note that the expression factorizes into radial and angular factors. Rescaling ρ with a constant factor multiplies $\varepsilon(\boldsymbol{\rho})$ by the same factor. It follows that a magnification of an equivalue contour yields another equivalue contour as depicted in Fig. 3.

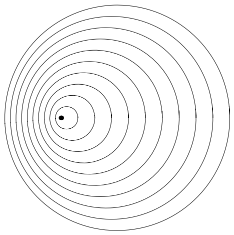


Fig. 3. Equivalue contours of $\varepsilon(\boldsymbol{\rho})$ are circles that are obtained by a magnification or contraction operation from each other

2.3 Dynamic Programming

For a general α we approach the problem with the method of dynamic programming. Let the pursuer choose the velocity $\mathbf{u}$ at time zero and proceed with this velocity over time interval dt. In order for the initial velocity to be optimal, we must have

$$\begin{aligned}
\varepsilon(\boldsymbol{\rho}) &= \rho^\alpha\, dt + \min_{|\mathbf{u}|\le u} \varepsilon(\boldsymbol{\rho} + (\mathbf{u}-\mathbf{v})\, dt) \\
&= \rho^\alpha\, dt + \min_{|\mathbf{u}|\le u} \{\varepsilon(\boldsymbol{\rho}) + (\mathbf{u}-\mathbf{v})\cdot \nabla\varepsilon(\boldsymbol{\rho})\, dt\} \qquad (4)\\
&= \rho^\alpha\, dt + \varepsilon(\boldsymbol{\rho}) - \mathbf{v}\cdot\nabla\varepsilon(\boldsymbol{\rho})\, dt + \min_{|\mathbf{u}|\le u} \mathbf{u}\cdot\nabla\varepsilon(\boldsymbol{\rho})\, dt.
\end{aligned}$$

The minimum of the last term with respect to **u** is obtained when the direction of **u** is opposite to $\nabla\varepsilon(\boldsymbol{\rho})$ and $|\mathbf{u}| = u$. The minimum value attained is $-u|\nabla\varepsilon(\boldsymbol{\rho})|\, dt$. This leads to the first order (non-linear) partial differential equation for the function $\varepsilon(\boldsymbol{\rho})$,

$$\rho^\alpha - \mathbf{v}\cdot\nabla\varepsilon(\boldsymbol{\rho}) - u|\nabla\varepsilon(\boldsymbol{\rho})| = 0.$$

In component form this reads

$$\rho^\alpha - v\frac{\partial\varepsilon}{\partial x} - u\sqrt{\left(\frac{\partial\varepsilon}{\partial x}\right)^2 + \left(\frac{\partial\varepsilon}{\partial y}\right)^2} = 0.$$

Next we focus on how the solution of this equation can be obtained.

2.4 Solving the Equation

Since the important parameter is the ratio (ν) of the speeds of the pursuee and pursuer, and not the absolute speeds, without loss of generality we may take $u = 1$. Then, with the notation $v = \nu u$, the equation reads

$$\rho^\alpha - \nu\frac{\partial\varepsilon}{\partial x} - \sqrt{\left(\frac{\partial\varepsilon}{\partial x}\right)^2 + \left(\frac{\partial\varepsilon}{\partial y}\right)^2} = 0. \qquad (5)$$

This is most easily solved using polar coordinates introduced above, i.e. we solve $\varepsilon = \varepsilon(\rho, \theta)$. A solution is obtained with the separable trial (cf. the form of (3))

$$\varepsilon(\rho,\theta) = \tfrac{1}{1+\alpha}\, \rho^{\alpha+1}\varphi(\theta), \qquad (6)$$

where the constant factor $\frac{1}{1+\alpha}$ is introduced for later convenience, and $\varphi(\theta)$ is an angular function yet to be found. With this trial we have

$$\begin{cases}
\dfrac{\partial\varepsilon}{\partial x} = \rho^\alpha\big(\cos\theta\,\varphi(\theta) - \sin\theta\,\varphi'(\theta)/(1+\alpha)\big),\\[2ex]
\dfrac{\partial\varepsilon}{\partial y} = \rho^\alpha\big(\sin\theta\,\varphi(\theta) + \cos\theta\,\varphi'(\theta)/(1+\alpha)\big).
\end{cases}$$

Upon substitution in (5) the factor ρ^α is canceled and we are left with an ordinary differential equation for the angular function $\varphi(\theta)$,

$$\nu\big(\cos\theta\,\varphi(\theta) - \sin\theta\,\varphi'(\theta)/(1+\alpha)\big) + \sqrt{\varphi(\theta)^2 + \varphi'(\theta)^2/(1+\alpha)^2} = 1. \qquad (7)$$

More explicitly, solved for $\varphi'(\theta)$ the differential equation reads[1]

$$\varphi'(\theta) = (1+\alpha)\,\frac{\nu \sin\theta - \sqrt{1 - 2\nu\varphi(\theta)\cos\theta - (1-\nu^2)\varphi^2(\theta)} - \nu^2\varphi(\theta)\sin\theta\cos\theta}{1-\nu^2\sin^2\theta}. \tag{8}$$

Because of symmetry, we have $\varphi'(0) = \varphi'(\pi) = 0$. The corresponding values $\varphi(0)$ and $\varphi(\pi)$ are readily solved from (7)[2],

$$\varphi(0) = \frac{1}{1+\nu}, \qquad \varphi(\pi) = \frac{1}{1-\nu}. \tag{9}$$

It is straightforward to check that the angular function, i.e. the coefficient of ρ, in (3) satisfies (8) for $\alpha = 0$, while an analytic solution for general α is not known. Equation (8) can, however, easily be solved numerically[3]. In Fig. 4 a family of solutions for $\varphi(\theta)$, corresponding to different values of α, $\alpha = 0$, 1, 2, 5, and 25, are depicted for a fixed value of $\nu = \frac{1}{2}$ (with this value of ν we have $\varphi(0) = \frac{2}{3}$ and $\varphi(\pi) = 2$).

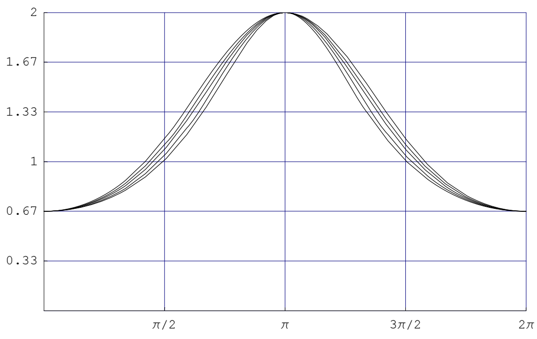


Fig. 4. Angular function $\varphi(\theta)$ for $\alpha = 1$, 2, 5, 25 (from top to bottom) with $\nu = \frac{1}{2}$

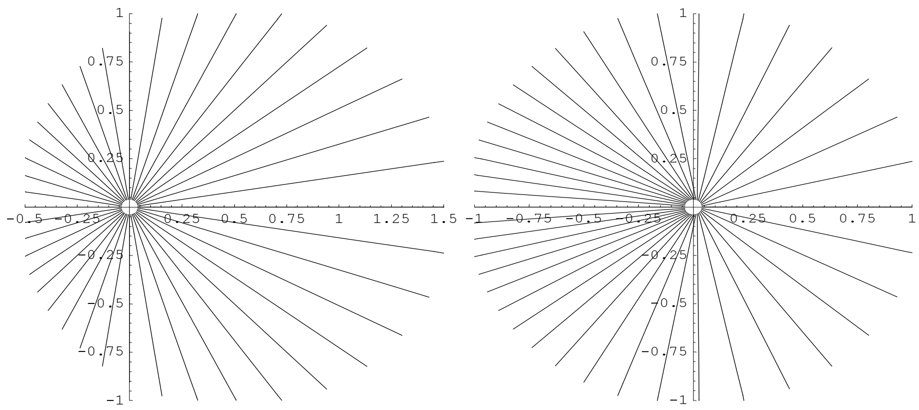
3 Numerical Results

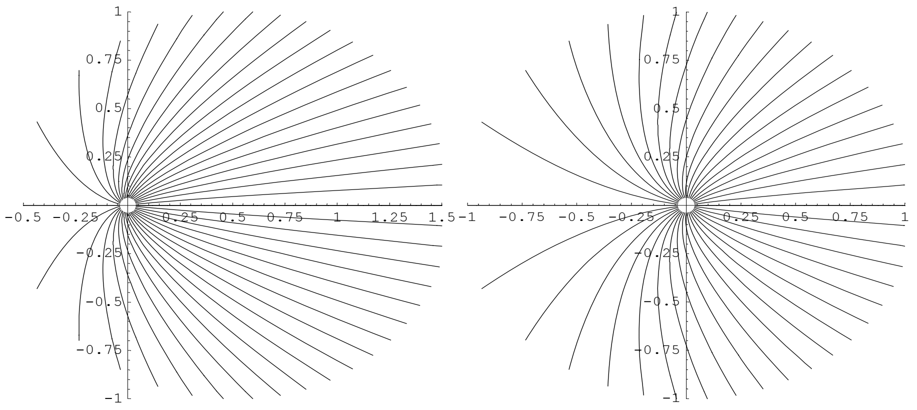
Recalling that the velocity vector $\mathbf{u}$ of the pursuer is opposite to the direction of $\nabla\varepsilon(\boldsymbol{\rho})$ and that the pursuer always uses the full speed $|\mathbf{u}| = 1$, the trajectory

[1] One has to choose the minus sign for the square root; plus sign would lead to an imaginary solution.

[2] These results can be derived also as follows. When $\theta = 0$ or π, the optimal strategy is obviously to go straight along the x-axis at full speed towards the pursuee. Then, $\varepsilon(x\,\mathbf{e}_1) = \int_0^{|x|/(1\pm\nu)} (|x| - (1\pm\nu)\,t)^\alpha\,dt = \frac{1}{1\pm\nu}\int_0^{|x|}(|x|-y)^\alpha\,dy$, where $\pm$ stands for sign x. The integration yields $\varepsilon(x\,\mathbf{e}_1) = \frac{1}{1\pm\nu}\frac{1}{1+\alpha}|x|^{1+\alpha}$ from which, in view of (6), result (9) follows.

[3] To guarantee numerical stability, the equation has to be solved backwards from π to 0; values in the range $\theta \in (\pi, 2\pi)$ are obtained by symmetry from those in range $\theta \in (0,\pi)$.


(a) $\alpha = 0$


(b) $\alpha = 2$

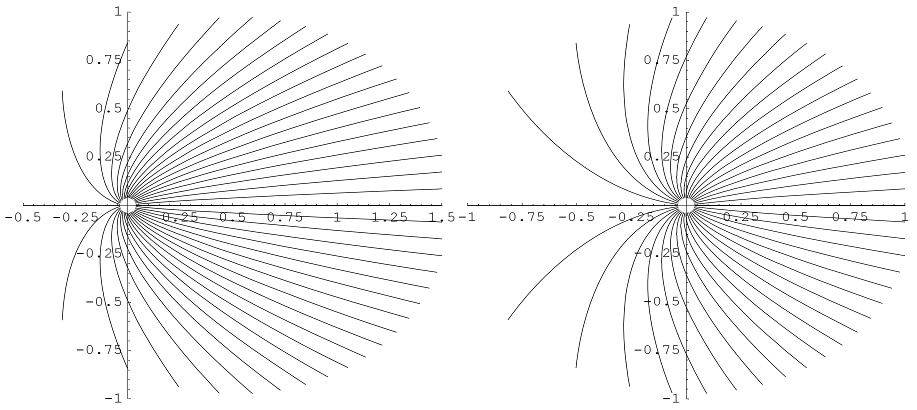

(c) $\alpha = 5$

Fig. 5. Trajectories in moving and fixed coordinates

of the pursuer can be solved when the function $\varepsilon(\boldsymbol{\rho})$ is known. In the sequel, we denote the unit vector in the direction of $-\nabla\varepsilon(\boldsymbol{\rho})$ by $\boldsymbol{\epsilon}(\boldsymbol{\rho})$.

It is useful to note some general properties of the trajectories. First, from the separable form (6) it follows that the direction of $\nabla\varepsilon$ is a function of the angle θ only. It then follows that if $f(x, y) = 0$ defines the path of a trajectory, then also $f(cx, cy) = 0$ is a path for all $c > 0$. In other words, an arbitrary magnification or contraction of an optimal path results in another optimal path.

The trajectories can be solved either in moving coordinates (moving with the pursuee) or in fixed coordinates, i.e. one can solve either $\boldsymbol{\rho}(t)$ or $\mathbf{r}(t)$. These are determined by the differential equations

$$\frac{d}{dt}\boldsymbol{\rho}(t) = \boldsymbol{\epsilon}(\boldsymbol{\rho}(t)) - \mathbf{v},$$

$$\frac{d}{dt}\mathbf{r}(t) = \boldsymbol{\epsilon}(\mathbf{r}(t) - \mathbf{v}\,t).$$

In Fig. 5 we give examples of the trajectories for three different values of α, $\alpha = 0, 2, 5$, with $\nu = \frac{1}{2}$. The trajectories are drawn for $t \in (0, 1)$ for a pursuer that reaches the pursuee at time $t = 1$. For $\alpha = 0$ the trajectories are straight lines as they should.

Looking at the trajectories in fixed coordinates (the right hand graphs), one notes an intuitively obvious behavior. Regarding that at time $t = 0$ the pursuee is at point $(-\frac{1}{2}, 0)$, we find that in the case $\alpha = 0$ the pursuer does not head to 'where the pursuee is' but directly to 'where the pursuee is going'. When the value of α increases the optimal trajectory more and more turns to the one that heads to 'where the pursuee currently is'. This happens in order to decrease the distance between the pursuer and the pursuee as quickly as possible; this is advantageous because for a large α the objective function decreases very rapidly as the distance decreases.

4 Conclusions

We studied a linear pursuit problem with an application example of target detection and tracking in a wireless sensor networking scenario using a mobile sink. We identified the optimal trajectory that should be followed by the sink to minimize the energy consumption in the network. The energy of radio transmission to be minimized is defined by a cost rate that obeys the well-known power-law $\rho^{-\alpha}$ as a function of the distance ρ between the mobile sink (pursuer) and the moving target (pursuee).

We approached the problem by the method of dynamic programming. We showed that the cost function, describing the radio transmission cost with an optimal policy, factorizes into radial and angular functions. The partial differential equation governing the cost function reduces to an ordinary differential equation for the angular function. This equation as well as the related optimal trajectories can be solved numerically.

Parameter α gives a great flexibility to this model. When α is set to zero, the problem reduces to the task of catching the target as soon as possible. The resulting optimal trajectories in this case are straight lines leading directly towards the rendezvous-point. On the other hand, when α is set to two or more, the cost function is a realistic model for the energy requirement of radio transmission. The resulting optimal trajectory ensures in this case the minimum overall energy consumption in the network. When α is large, the optimal pursuit is the one where the sink is always heading right towards the target's actual position, trying to reduce the relative distance as much as possible.

An interesting consequence is that, having the power-law dependent cost function, the optimal pursuit curves are self-similar in the sense, that any magnification of the curve results in an optimal trajectory as well, but for different initial conditions.

References

1. Tsao, L.P., Lin, C.S.: A new optimal guidance law for short-range homing missiles. Proc., National Science Council, ROC(A) **24**(6) (2000) 422–426
2. Croft, E.A., Fenton, R.G., Benhabib, B.: Optimal rendezvous-point selection for robotic interception of moving objects. IEEE Transactions on Systems, Man, and Cybernetics, Part B **28**(2) (1998) 192–204
3. Bernhart, A.: Curves of pursuit. Scripta Mathematica **20** (1954) 125–141
4. Akyildiz, I., Su, W., Sankarasubramaniam, Y., Cayirci, E.: A survey on sensor networks. IEEE Communications Magazine **40**(8) (August 2002) 102–116
5. Akyildiz, I., Su, W., Sankarasubramaniam, Y., Cayirci, E.: Wireless sensor networks: A survey. Computer Networks (Elsevier) Journal **38**(4) (March 2002) 393–422
6. I.F., A., Kasimoglu, I.: Wireless sensor and actor networks: Research challenges. Ad Hoc Networks Journal (Elsevier) **2**(4) (October 2004) 351–367
7. Vincze, Z., Vass, D., Vida, R., Vidács, A.: Adaptive sink mobility in event-driven clustered single-hop wireless sensor networks. In: Proc., 6th Int. Network Conference (INC 2006), Plymouth, UK (July 2006) 315–322
8. Bellman, R.E.: Dynamic Programming. Princeton University Press (1957)
9. Bertsekas, D.P.: Dynamic Programming and Stochastic Control. Academic Press (1976)
10. Bertsekas, D., Gallager, R.: Data Networks. 2nd edn. Prentice Hall (1992)

A Hybrid Energy Saving Mechanism for VoIP Traffic with Silence Suppression

Jung-Ryun Lee

Electronic Engineering, University of Incheon, 177,
Dohwa-dong, Nam-gu, Incheon, Korea
jungryunlee@incheon.ac.kr

Abstract. In this paper, we propose a hybrid energy saving mechanism (HESM) that combines power-saving mode (PSM) for real-time traffic with PSM for non real-time traffic, which is applicable to Voice over IP (VoIP) traffic with silence suppression. The proposed method uses power saving class (PSC) II during talk-spurt periods of parties A and/or B. On the other hand, during mutual silence periods, sleep interval placement is determined by the proposed probabilistic sleep interval decision (PSID) algorithm. Under the PSID algorithm, the length of sleep intervals in mutual silence period under the PSID algorithm is determined by the cumulative density function of the length of mutual silence period. We use Brady model that is an accurate model of the on-off characteristics of conversational speech, in order to obtain the CDF. The performances of HESM based on the PSID algorithm are evaluated by the energy consumption of an MS and the VoIP packet drop probability. Results show that the proposed HESMs reduce considerably the energy consumption of an MS compared to PSC II, while satisfying the QoS constraints on the VoIP packet drop probability for VoIP connection in the base station (BS). Specifically, the result shows that proposed HESM reduces the energy consumption of MSs by up to 25%.

1 Introduction

In wireless networks, power-saving mode (PSM) is a very important technology for prolonging the limited battery lifetime of mobile stations (MSs). In general PSM, MSs enter sleep mode during sleep intervals and wake up during predetermined intervals (wake-up interval), in order to verify whether there are any buffered packets waiting for it in the base station (BS). If there are no pending packets, the MSs return to sleep mode. Otherwise, the MSs and the BS initiate the procedure for data exchange. Since the packet arrived at the BS during sleep interval should be pended until wake-up interval starts, PSM inevitably involves the packet buffering delay in the BS. This is the main reason why PSM has previously been used for only non real-time traffic, such as web-browsing applications, which allow for longer delays than real-time traffic. However, the recently-defined IEEE 802.16e standard suggests that PSM may be used for real-time traffic, under the rubric power-saving class (PSC) II [1].

PSC II is different from PSC I, which is also defined in IEEE 802.16e, but for non real-time traffic. First, PSC I adopts a binary truncated exponent algorithm, which doubles the length of the sleep interval until it reaches the maximum. By contrast, PSC II

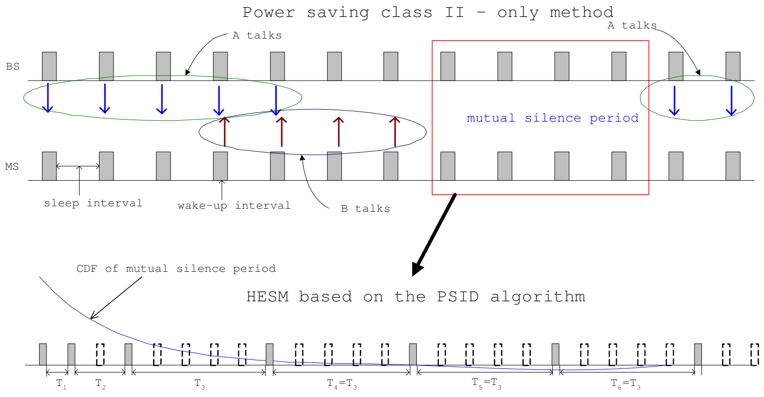


Fig. 1. The concept of hybrid energy saving mechanism

uses a fixed length of sleep interval because real-time traffic is generated periodically. Second, in PSC I, there is a traffic indication message that notifies an MS of the existence of pending packets in the BS. However, this message is not used in PSC II, in the interests of eliminating signaling overhead due to frequent packet exchange between MSs and the BS. Considering these properties of PSC II, if we apply PSM to VoIP service, which is a real-time application, PSC II, which can adjust the length of sleep interval according to the period of VoIP traffic generation, should be used.

The characteristics of VoIP packets, such as framing interval, data rate, and silence suppression, differ according to the voice codec used. Some voice codecs, such as G.723.1A, G.723.B or Adaptive Multi Rate (AMR) in 3GPP, use silence suppression functionality, which does not send VoIP packets during silence periods, in order not to waste system bandwidth. Suppose that PSM is applied to a VoIP service whose codec supports silence suppression. In this case, it is desirable to use PSC II instead of PSC I, since VoIP service is a real-time traffic generated periodically. However, using PSM II alone is not efficient, because PSC II uses a *fixed*-length sleep interval and does not discriminate between talk-spurt and mutual silence periods. Instead, in mutual silence periods, it would be more energy-efficient to use an algorithm that places its sleep intervals flexibly. Thus, in this paper, we suggest a hybrid energy saving mechanism (HESM), which adopts PSM differently between talk-spurt and mutual silence periods. The suggested HESM uses PSC II during talk-spurt periods, and probabilistic sleep interval decision (PSID) algorithm during mutual silence periods. The proposed PSID algorithm determines the length of the sleep interval according to the distribution function for the length of mutual silence periods. Fig. 1 shows the concept of the HESM based on PSID algorithm.

PSM has been widely studied. Since its main purpose is to reduce the power consumption of an MS, almost all previous work has focused on how a suitable sleep interval placement may be determined for non-real time traffic without deterioration of Quality of Service (QoS) index such as maximum allowable delay [2]. A method that minimizes energy consumption for wireless web access using the bounded slowdown (BSD) protocol is suggested by R. Krashinsky and H. Balakrishnan [3]. The BSD protocol minimizes energy consumption while providing a guaranteed bound on RTT slowdown for request/response network traffic. In [4], the authors proposed a smart PSM (SPSM) scheme in the context of an IEEE 802.11 WLAN system, which guarantees a desired delay performance per user with minimum energy consumption. This algorithm proposed a penalty function to express the delay constraints of each user and solved energy-consumption minimization problem subject to delay-performance constraints. However, to our knowledge, there has been no study of PSM considering hybrid usage.

The remainder of this paper is structured as follows. In Section 2, Brady model for the on-off property of biconversational traffic is introduced. Section 3 gives a detailed explanation of the proposed HESM algorithm. In Section 4, results for the energy consumption of an MS and the packet loss probability are provided as performance metrics. Section 5 concludes.

2 Brady Model

This paragraph gives a brief explanation of Brady model in [5]. The on-off characteristic of conversational speech was established by Brady who used a six-state Markov chain to represent talk-spurt and silent states of two users, A and B, as shown in Fig. 2. Fig. 2 is divided into quadrants. Each quadrant represents a different state for users A and B, who are engaged in conversation. The upper left quadrant (representing A speaking and B silent) and the lower right quadrant (representing B speaking and A silent) contain individually one state, while the lower left quadrant (A and B both silent) and upper right quadrant (A and B both speaking) contain individually two states to differentiate

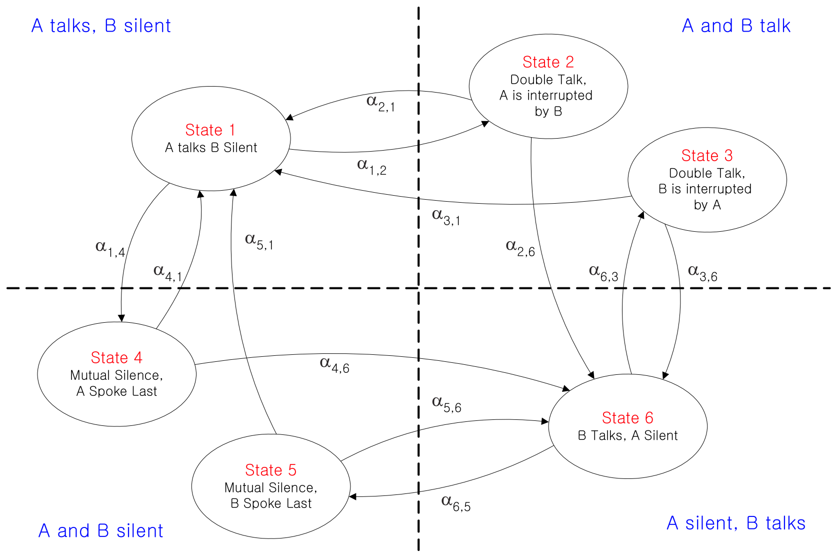


Fig. 2. Six-state Brady model

which of the two parties spoke last. The variable α followed by two subscripts is used to symbolize each state transition parameter, with the first subscript indicating the original state and the second subscript indicating the new state. For example, $\alpha_{1,4}$ refers to the transition from State 1 to State 4 in Fig. 2. Notice that the state transition parameters for parties A and B have the same value. For the details, interested readers are encouraged to read the article by Brady [5].

Let Y be a random variable representing the length of a mutual silence period. From the Brady model, the cdf of Y is calculated as

$$F_Y(t) = 1 - e^{-\lambda t} \tag{1}$$

where $\lambda = \alpha_{4,1} + \alpha_{4,6}$.

Let M be the matrix representing the state transition probability defined in Fig. 2. Solving the equilibrium equation of the Markov chain, $\Pi = \Pi \cdot M$ where $\Pi = (\pi_1, \pi_2, \ldots, \pi_6)$ and π_i denotes a portion of state i, indicates that the portions of mutual silence periods $(\pi_4 + \pi_5)$ against the duration of the entire conversation amount to around 19%. This result justifies the usage of the PSID algorithm during mutual silence periods, even for real-time VoIP traffic.

3 Hybrid Energy Saving Mechanism

When the voice codec of each party uses silence suppression functionality, VoIP packets are generated periodically during the talk-spurt period, but not generated at all during silent periods. Instead, a Silence Insertion Descriptor (SID) frame is generated at the beginning of a silent period [6]. Thus, the starting time of mutual silence period can be decided in the BS by verifying the acceptance of SIP frames of both parties A and B.

The basic concept of HESM is to use different PSM according to the voice activity of each party. During talk-spurt periods, PSC II with a fixed length of sleep interval is used. During mutual silence periods, we consider PSID algorithm, so as to save the energy consumption of an MS. Hereafter, we call the method that uses the PSID algorithm during mutual silence periods the HESM-PSID algorithm.

Here, we define the notation to be used in the remainder of this paper. Let the length of the i-th sleep interval and the length of the wake-up interval be T_i and T_l, respectively. Considering the wake-up interval, we denote the length of the j-th sleep interval with wake-up interval by $\overline{T}_j = T_j + T_l$. Then, assuming that PSM starts at time 0, the j-th sleep interval with wake-up interval is expressed as

$$\overline{S}_1 = [0, \overline{T}_1), \tag{2}$$

$$\overline{S}_j = [\sum_{i=1}^{j-1} \overline{T}_i, \sum_{i=1}^{j} \overline{T}_i) \text{ for } j \geq 2. \tag{3}$$

In addition, we define the fixed length of sleep interval when PSC II is applied to talk-spurt periods as T_f. T_f is determined according to the framing interval of the voice codec used.

The PSID algorithm uses the distribution of length of a mutual silence period derived in (1) during mutual silence periods. The key concept of the proposed PSID algorithm

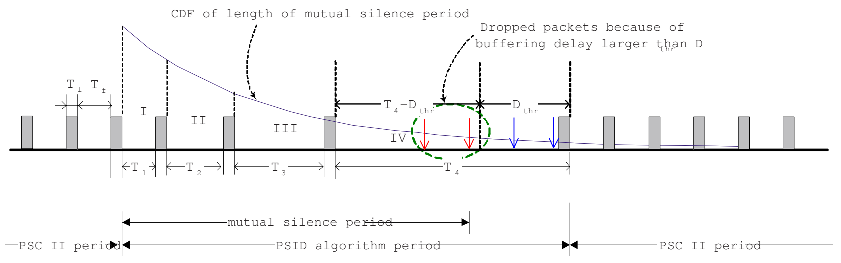


Fig. 3. Examples of the PSID-HESM algorithm

is to place sleep intervals such that the probability that a mutual silence period may end within a sleep interval is the same across all sleep intervals in the extension-allowed interval. As for the extension-allowed interval in the PSID algorithm, the length of the longest sleep interval should be controlled, to prevent them from being excessively long. The PSID algorithm determines the extension-allowed interval according to the distribution function for Y in order to reflect Y's behavior. Suppose that a mutual silence period starts at time 0. Let the extension-allowed interval be $[0, \beta]$. We determine the target probability $P_{tar,k}$ which denotes the probability that Y is included in the extension-allowed interval. In this paper, $P_{tar,k}$ is chosen in terms of the mean ($E\{Y\}$) and standard deviation (σ_Y) of Y as follows:

$$P_{tar,k} = F_Y(E\{Y\} + k \cdot \sigma_Y) = F_Y(\frac{1+k}{\lambda}) \tag{4}$$

where k is a real number. By the definition of $P_{tar,k}$,

$$\beta = \frac{1+k}{\lambda}. \tag{5}$$

To determine the exact position of each sleep interval in the extension-allowed interval, the extension-allowed interval, $[0, \beta]$, should be divided into appropriate subintervals each of which becomes the sleep interval. Let the number of these subintervals be N. The PSID algorithm divides $[0, \beta]$ into N mutually exclusive subintervals, $\overline{S}_1, \overline{S}_2, \ldots, \overline{S}_N$. Let $\overline{S}_j = [\alpha_i, \beta_i]$. The PSID algorithm determines β_i so as to satisfy the following equation:

$$F_Y(\beta_i) = i\frac{P_{tar,k}}{N} \quad (i = 1, 2, \ldots, N.) \tag{6}$$

By definition, $\alpha_1 = 0$. From the mutual exclusiveness of each $\overline{S}_j$, $\beta_i = \alpha_{i+1}$. In addition, $\beta_N = \beta$ from (4), (5) and (6). The above method completely determines $\{\overline{S}_i\}$ for $1 \leq i \leq N$. Further, the probability that Y is included in the extension-allowed interval should not be 1, because β goes to infinity when $P_{tar,k} = 1$. Thus, there is a probability of $1 - P_{tar,k}$ that the mutual silence period may end outside the extension-allowed interval. When the mutual silence period does not end within the extension-allowed interval, the PSID algorithm sets T_{N+j} to T_N for $j \geq 1$, and this length is

Table 1. Sleep Interval Placement for the HESM-PSID algorithm

$P_{tar,k}$	N	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	β_9	β_{10}
0.637 ($k = 0$)	2	120	310	500	690	880	1070	1260	1450	1640	1830
	3	75	170	310	450	590	730	870	1010	1150	1290
	4	55	120	200	310	420	530	640	750	860	970
	5	45	90	150	220	310	400	490	580	670	760
	6	35	75	120	170	230	310	390	470	550	630
	7	30	65	100	140	185	240	310	380	450	520
	8	30	55	85	120	155	200	250	310	370	430
	9	25	50	75	105	135	170	210	255	310	365
	10	20	45	65	90	120	150	180	220	260	310
0.866 ($k = 1$)	2	175	615	1055	1595	1935	2375	2815	3255	3695	4135
	3	105	265	615	965	1315	1665	2015	2365	2715	3065
	4	75	175	325	615	905	1195	1485	1775	2065	2355
	5	60	130	25	365	615	865	1115	1365	1615	1865
	6	50	105	175	265	395	615	835	1055	1275	1495
	7	45	90	145	210	295	415	615	815	1015	1215
	8	40	75	125	175	240	325	435	615	795	975
	9	35	70	105	150	205	265	345	450	615	780
	10	30	60	95	130	175	225	285	365	465	615

the same as that of the last (longest because of the *exponential* distribution of Y) sleep interval inside the extension-allowed interval. So, T_{N+j} is the same as T_N.

From (6), the probability that the mutual silence period ends within $\overline{T}_j$ becomes

$$\begin{aligned} P\{Y \in \overline{S}_j\} &= F_Y(\beta_i) - F_Y(\alpha_i) \\ &= F_Y(\beta_i) - F_Y(\beta_{i-1}) = \frac{P_{tar,k}}{N}. \end{aligned} \tag{7}$$

It means that the PSID algorithm divides the extension-allowed interval *evenly*, so that the probability that the mutual silence period ends within each $\overline{S}_j$ is the same across all j less than N. However, the length of sleep interval $\overline{T}_j$ differ from each other, because Y follows an exponential distribution (i.e., Y is not a constant function). For example, in Fig. 3 b), $\{\overline{T}_j\}$ is not the same, but increases as j increases from 1 to 4. However, the size of area I, II, III, and IV, each of which represents the probability that Y ends within $\overline{S}_1$, $\overline{S}_2$, $\overline{S}_3$ and $\overline{S}_4$ respectively, is the same. Notice that the actual placement of each $\overline{S}_j$ depends on $P_{tar,k}$ and N in the PSID algorithm.

4 Performance Evaluation

4.1 Simulation Setup

We evaluate the performance of the HESM-PSID algorithm in the context of IEEE 802.16e system. The granularity of the sleep interval length is set to $5ms$ according to the frame length[1] defined in [1]. The length of wake-up interval T_l is assumed to be one

[1] In [1], possible frame lengths for OFDMA (Orthogonal Frequency Division Multiple Access) are $2ms$, $2.5ms$, $4ms$, $5ms$, $8ms$, $10ms$, $12.5ms$, and $20ms$.

frame length, $5ms$. The packet generation ratio of VoIP codec is assumed to be $20ms$, so we set the length of sleep interval during PSC II periods, T_f, to $15ms$ considering T_l. The end-to-end delay threshold, to satisfy delay constraint in VoIP traffic, is assumed, as in [7], to be $270ms$. With this assumption, D_{thr} which is the maximum allowable delay in the BS is set to $197ms$ [8]. The sleep interval placement under the PSID-HESM algorithm is derived from $P_{tar,k}$ and N. Table 1 shows various sleep interval positions according to $P_{tar,k}$ and N with assumption that a mutual silence period starts at time 0. For evaluation of energy consumption of an MS, we define E_{slp} and E_{act} as the energy consumption of MS in sleeping and active modes, respectively. $1.5W$ and $0.045W$ are used as values of E_{act} and E_{slp}, respectively [9]. Monte Carlo method with $10,000$ trials is used for simulation runs and each run (conversation) lasts 100 secs.

4.2 Result for Energy Consumption of an MS

The energy consumption of the MS for the HESM-PSID algorithm compared to the PSC II-only method is shown in Fig. 4 (b). For each k, as N increases, the energy consumption increases due to shorter sleep intervals. Larger k results in a longer extension-allowed interval. Thus, given that the number of subintervals (N) is fixed, the increment of k results in long subintervals. So, the number of wake-up intervals experienced by the MS is reduced. That is why the energy consumption for $k = 1$ is smaller than that for $k = 0$. Fig. 4 shows that the HESM-PSID algorithm can reduce the energy consumption of an MS by up to 25%.

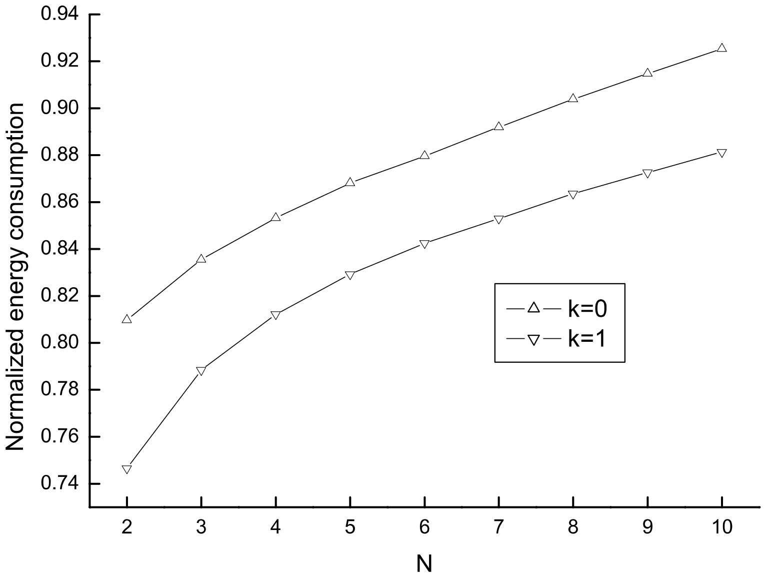


Fig. 4. Energy consumption for the HESM-PSID algorithm as a function of N

4.3 Result for VoIP Packet Drop Probability

When the length of sleep interval is longer than the maximum allowable delay D_{thr}, some VoIP packets arrived at the BS during the sleep interval may be missing. Fig. 5 presents the drop probability of VoIP packets due to the buffering delay in the BS when the HESM-PSID algorithm is applied in cases of $k = 0$ and $k = 1$. When $k = 0$ and $N = 2$, the lengths of sleep intervals are $120ms$ and $190ms$ (refer to Table 1), which are less than D_{thr}. So the packet drop probability due to the buffering delay in the BS becomes 0. Since the length of sleep interval decreases as N increases, the packet drop probability is 0 for all $N \geq 2$ when $k = 0$. By contrast, when $k = 1$, the smaller N causes longer sleep intervals, because the subinterval of the extension-allowed interval increases. Thus, the packet drop probability increases as N decreases. However, when N is large enough to cause all subintervals included in the extension-allowed interval to be smaller than D_{thr} (under the given conditions, $N > 6$), the packet drop probability becomes zero.

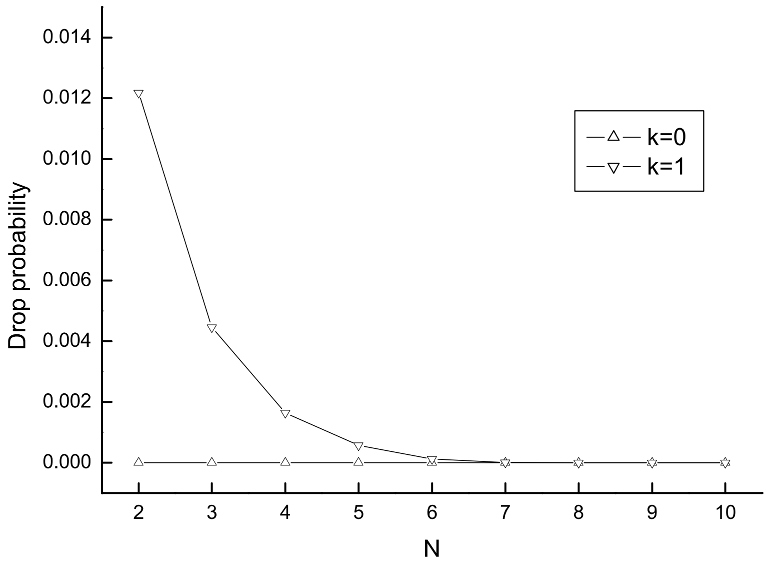


Fig. 5. VoIP packet drop probability in the BS for the HESM-PSID algorithm as a function of N

5 Conclusions

In this paper, we suggested the HESM algorithm for VoIP traffic with silence suppression. The proposed HESM algorithm uses PSC II during talk-spurt periods and the PSID algorithm during mutual silence periods. In the HESM-PSID algorithm, $P_{tar,k}$ and N determine the length of sleep intervals in mutual silence periods, in order that the probability that a mutual silence period may end within each sleep interval is the same across all sleep intervals in the extension-allowed interval. The performances for the energy consumption of an MS and packet drop probability were provided. The results show that the energy consumption of an MS can be reduced by up to 25% compared to PSC II-only method while satisfying the delay constraint of VoIP connection in the BS when the HESM-PSID algorithm with $k = 1$ is used.

References

1. IEEE P802.16e/D10, *"Draft IEEE Standard for Local and Area Networks-Part 16",* Sep. 2005.
2. C. E. Jones, K. M. Sivalingam, P. Agrawal, and J. C. Chen., "Survey of energy efficient network protocols for wireless networks", *ACM Wireless Networks*, vol. 7, no. 4, pp. 343-358, July 2001.
3. R. Krashinsky, H. Balakrishnan, "Minimizing energy for wireless web access with bounded slowdown", *MobiCom 2002*, Atlanta, Georgia, USA, pp.119-130, Sep. 2002.
4. D. Qiao and K. G. Shin., "Smart Power-Saving Mode for IEEE 802.11 Wireless LANs", *Proc. IEEE INFOCOM*, Miami, FL, Mar. 2006
5. P. T. Brady, "A model for generating on-off speech patterns in two-way conversation," *Bell Syst. Tech. J.*, vol. 48, no. 7, pp. 2445-2472, Sep. 1969.
6. A. Estepa, et al., "A New Approach for VoIP Traffic Characterization, *IEEE Commun. Letters*, vol. 8, no. 10, pp. 644-646, Oct. 2004.
7. ITU-T Recommendation G.114, "One-way transmissin time", May, 2003.
8. M. Yavuz, et al., "VoIP over cdma2000 1xEV-DO Revision A", *IEEE Commun. Magazine*, vol. 44, no. 2, pp. 88-95, Feb. 2006.
9. E. S. Jung, and N. H. Vaidya, "An energy efficient MAC protocol for wireless LANs", *INFOCOM 2002, Proceedings, IEEE*, vol. 3, pp.23-27, Jun. 2002.
10. Yang Xiao, "Energy saving mechanism in the IEEE 802.16e Wireless MAN", *IEEE Commun. Letters*, vol.9, no. 7, pp. 595-597, Jul. 2005.
11. Salkintzis, A. K. and Chamzas, C., "Performance analysis of a downlink MAC protocol with power-saving support", *Vehicular Technology, IEEE Transactions on*, vol. 49, no. 3, pp. 1029-1040, May. 2000.

Author Index