Available online at www.sciencedirect.com

SCIENCE DIRECT®

Computer Networks 50 (2006) 982–1002

Computer Networks

www.elsevier.com/locate/comnet

Pricing differentiated services: A game-theoretic approach ☆

Eitan Altman [a], Dhiman Barman [b], Rachid El Azouzi [c], David Ros [d,*], Bruno Tuffin [e]

[a] INRIA, B.P. 93, 2004 Route des Lucioles, 06902 Sophia-Antipolis Cedex, France
[b] 111 Cummington Street, Department of Computer Science, Boston University, Boston, MA 02215, USA
[c] Université d'Avignon et des Pays de Vaucluse (IUP), LIA—CERI, 339 Chemin des Meinajariès, BP 1228, 84911 Avignon Cedex 9, France
[d] GET/ENST Bretagne, Rue de la Châtaigneraie, CS 17607, 35576 Cesson Sévigné Cedex, France
[e] IRISA/INRIA Rennes, Campus de Beaulieu, 35042 Rennes Cedex, France

Received 16 November 2004; received in revised form 16 May 2005; accepted 26 June 2005
Available online 8 August 2005

Responsible Editor: J.C. de Oliveira

Abstract

The goal of this paper is to study pricing of differentiated services and its impact on the choice of service priority at equilibrium. We consider both TCP connections as well as noncontrolled (real-time) connections. The performance measures (such as throughput and loss rates) are determined according to the operational parameters of a RED (Random Early Discard) buffer management. The latter is assumed to be able to give differentiated services to the applications according to their choice of service class. We consider a service differentiation for both TCP as well as real-time traffic where the quality of service (QoS) of connections is not guaranteed, but by choosing a better (more expensive) service class, the QoS parameters of a session can improve (as long as the service class of other sessions are fixed). The choice of a service class of an application will depend both on the utility as well as on the cost it has to pay. We first study the performance of the system as a function of the connections' parameters and their choice of service classes. We then study the decision problem of how to choose the service classes. We model the problem as a noncooperative game. We establish conditions for an equilibrium to exist and to be uniquely defined. We further provide conditions for convergence to equilibrium from nonequilibria initial states. We finally study the pricing problem of how to choose prices so that the resulting equilibrium would maximize the network benefit.

☆ A shorter version of this paper appeared in the Proceedings of the Third IFIP-TC6 Networking Conference [1].

* Corresponding author. Tel.: +33 2 99 12 70 46.
E-mail addresses: Eitan.Altman@sophia.inria.fr (E. Altman), dhiman@cs.bu.edu (D. Barman), Rachid.Elazouzi@lia.univ-avignon.fr (R.E. Azouzi), David.Ros@enst-bretagne.fr (D. Ros), Bruno.Tuffin@irisa.fr (B. Tuffin).

1389-1286/$ - see front matter
doi:10.1016/j.comnet.2005.06.008

Keywords: TCP; RED/AQM; Nash equilibrium; Pricing; Service differentiation

1. Introduction

We study in this paper the performance of competing connections that share a bottleneck link. Both TCP connections with controlled rate as well as CBR (Constant Bit Rate) connections are considered. A RED active queue management (AQM) algorithm is used for the early dropping of packets. We allow for service differentiation between the connections through the rejection probability (as a function of the average queue size), which may depend on the connection (or on the connection class). More specifically, we consider a buffer management scheme that uses a single averaged queue length to determine the rejection probabilities (similar to the way it is done in the RIO-C (coupled RIO) buffer management [2]); for any given averaged queue size, packets belonging to connections with higher priority have smaller probability of being rejected than those belonging to lower priority classes. To obtain this differentiation in loss probabilities, we assume that the loss curve of RED is scaled by a factor that represents the priority level of the application. We obtain various performance measures of interest such as the throughput, the average queue size and the average drop probability.

We then address the question of the choice of priorities. Given utilities that depend on the performance measures on one hand and on the cost for a given priority on the other hand, the sessions at the system are faced with a noncooperative game in which the choice of priority of each session has an impact on the quality of service of other sessions. For the case of CBR traffic, we establish conditions for an equilibrium to exist. We further provide conditions for convergence to equilibrium from nonequilibria initial states. The game formulation of the problem arises naturally, since a classical optimization approach where a common objective function is maximized, is not realistic in IP networks; indeed, it is quite rare that users of a network collaborate with each other (or even “know” each other).

Finally we study numerically the pricing problem of how the network should choose prices so that the resulting equilibrium would maximize its benefit.

We briefly mention some recent work in that area. Ref. [3] has considered a related problem where the traffic generated by each session was modeled as a Poisson process, and the service time was exponentially distributed. The decision variables were the input rates and the performance measure was the goodput (output rates). The paper restricted itself to symmetric users and symmetric equilibria and the pricing issue was not considered. In this framework, with a common RED buffer, it was shown that an equilibrium does not exist. An equilibrium was obtained and characterized for an alternative buffer management that was proposed, called VLRED. We note that in contrast to [3], since we also include in the utility of CBR traffic a penalty for losses (which is supported by studies of voice quality in packet-based telephony [4]), we do obtain an equilibrium when using RED. For other related papers, see for instance [5] (in which a priority game is considered for competing connections sharing a drop-tail buffer), [6] as well as the survey [7]. In [8], the authors present mechanisms (e.g., AIMD of TCP) to control end-user transmission rate into differentiated services Internet through potential functions and corresponding convergence to a Nash equilibrium.

The approach of our pricing problem is related to the Stackelberg methodology for hierarchical optimization: for a fixed pricing strategy one seeks the equilibrium among the users (the optimization level corresponding to the “follower”), and then the network (considered as the “leader”) optimizes the pricing strategy. This type of methodology has been used in other contexts of networking in [9,10].

The structure of this paper is as follows. In Section 2 we describe the model of RED, then in Section 3 we compute the throughputs and the loss probabilities of TCP and of CBR connections for given priorities chosen by the connections. In Section 4 we introduce the model for competition between connections at given prices. In Section 5 we focus on the game in the case of only CBR connections or only TCP connections and provide properties of the equilibrium: existence, uniqueness and convergence. Remark that isolating elastic (i.e., TCP) flows from real-time (i.e., UDP/CBR) flows—that is, mapping TCP and UDP flows to two different service classes—is a fairly common way of protecting TCP traffic from UDP flows in a differentiated-services architecture. Note that, inside each service class, we consider that flows have different parameters (like, say, different round-trip times). In Section 6 we provide an algorithm for computing Nash equilibrium for the symmetric case. The optimal pricing is then discussed in Section 7. We present numerical examples in Section 8 to validate the model.

2. The model

The main goal of the Random Early Discard (RED) algorithm is to provide *congestion avoidance* (that is, an operating region of low delay and high throughput) by trying to control the average queue length at a router [11]. A RED-enabled router estimates the average queue length q by means of an exponentially-weighted moving average; this estimate is updated with every incoming packet as: $q \leftarrow (1 - w_q)q + w_q Q$, where Q denotes here the instantaneous queue length "seen" by the packet, and $w_q \in [0, 1]$ is the averaging weight (the lower the value of w_q, the longer the "memory" of the estimator). Here we assume that the time averaging parameters of RED are such that the average queue size, and hence the drop probabilities p_i's have negligible oscillations. We are aware of the fact that for some RED parameters this may not be the case, and that the interaction between RED and TCP can lead to instabilities if the parameters are not chosen correctly.

This average queue value is then compared to two thresholds $q_{\min}$ and $q_{\max}$, with $q_{\min} < q_{\max}$, in order to decide whether or not the incoming packet should be dropped. The drop probability is 0 if $q \leqslant q_{\min}$, 1 if $q \geqslant q_{\max}$, and $p_{\max}(x - q_{\min})/(q_{\max} - q_{\min})$ if $q = x$ with $q_{\min} < x < q_{\max}$; the latter is the congestion avoidance mode of operation. $p_{\max}$ is the value of the drop probability as the average queue tends to $q_{\max}$ (from the left). This is illustrated in Fig. 1. In a best-effort network, the value of $p_{\max}$ is the same for all flows sharing the buffer, whereas in a network implementing service differentiation packets may "experience" different values of $p_{\max}$, according to the service class they belong to—as we will see below, it is the latter case which we are focusing on.

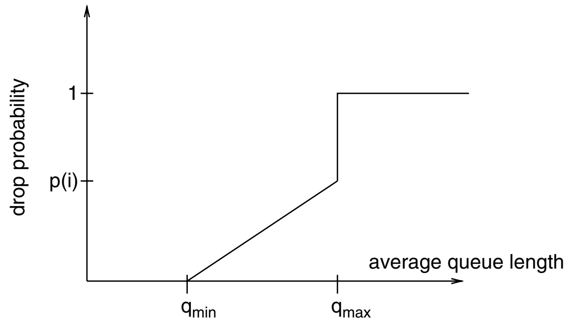


Fig. 1. Drop probability in RED as a function of q.

The purpose of the early discarding of packets (i.e., dropping a packet before the actual physical queue is full) is to signal the sources that implement congestion-control mechanisms—like TCP sources—to reduce their sending rates, in order to prevent heavy congestion. The random nature of drops aims at avoiding synchronization of flows having similar round-trip times [11], i.e., all sources increasing and decreasing their congestion windows in unison, leading to strong oscillations of queue lengths and lower throughput.

We consider a set $\mathcal{N}$ containing N TCP flows (or aggregate of flows) and a set $\mathcal{I}$ containing I real-time CBR flows that can be differentiated by RED; they all share a common buffer yet RED handles them differently.[1] We assume that they all share common values of $q_{\min}$ and $q_{\max}$ but each flow i may have a different value $p(i)$ of $p_{\max}$, leading to a differentiated treatment. In other words, the slope t_i of the linear part of the curve in Fig. 1 depends on the flow i:

[1] RED punishes aggressive flows more by dropping more packets from those flows.

$$t_i = \frac{p(i)}{q_{\max} - q_{\min}}.$$

Denote $\mathbf{t} = (t_i, i \in \mathscr{I} \cup \mathscr{N})$. We identify t_i as the priority class of a connection. The service rate of the bottleneck router is given by μ.

2.1. Practical considerations

Let us add a few remarks concerning practical issues of this proposal, like scalability and implementation complexity. First, in a DiffServ-like architecture [12], users may select a specific QoS treatment on a packet-per-packet basis, and that treatment corresponds precisely to a RED-like AQM policy that may drop packets with a probability that depends on a tag carried by the packet (this is how the Assured Forwarding per-hop behavior [13] operates)—the tag may well be set by the user to signal how the packet should be treated by the core routers. So, in the context of our proposal, from a practical (i.e., implementation) viewpoint, a user choosing her own $p(i)$ in the router requires just a straightforward setting of the QoS tag she puts on her packets.

On the other hand, letting a user choose the thresholds $q_{\min}$ and $q_{\max}$ does not seem realistic: the (feasible) values of the thresholds depend on link speeds and on the actual, "physical" capacity of router queues, which may vary from a link/router to another [11].

The fact that each source may choose a different value for the slope could cause problems in scaling our approach to large networks or to a large number of flows. This scaling problem can, however, be solved by using the following distributed approach: the RED queue could restrict to put on each packet the value of q at a given time. Then the decision of whether the packet would be dropped or not, depending on the slope that corresponds to the source of the packet, can be delegated to the edge router that corresponds to that connection. (In a differentiated service environment, there are edge routers that behave as policers, i.e. they can mark or drop packets that do not comply with the user's type.) The edge routers are directly connected to the corresponding sources so it is much easier to take the dropping decisions there. Note that our analysis does not depend on how exactly congestion signals are conveyed to a given source so using the above approach does not change our results.

3. Computing the throughputs

We use the well-known relation for TCP rate:

$$\lambda_i = \frac{1}{R_i}\sqrt{\frac{\alpha}{p_i}}, \quad i \in \mathscr{N}, \tag{1}$$

where R_i and p_i are TCP flow i's round-trip time and drop probability, respectively. α is typically taken as 3/2 (when the delayed-ACKs option is disabled) or 3/4 (when it is enabled). We shall assume throughout the paper that the queueing delay is negligible with respect to R_i for the TCP connections.

In contrast, the rates λ_i, for $i \in \mathscr{I}$, of real-time flows are not controlled and are assumed to be fixed. If $\mathscr{N} = \emptyset$ we assume throughout the paper that $\sum_{j\in\mathscr{I}}\lambda_j > \mu$ (unless otherwise specified), otherwise the RED buffer is not a bottleneck. Similarly, if $\mathscr{I} = \emptyset$ we assume that TCP senders are not limited by the receiver window.

In the model above, we assume that the number of flows is constant over time. This corresponds to a scenario of long-lived flows in which, for instance, TCP connections are used for the transfer of large files in storage networks or in backup of disks (so that we may assume that the square-root throughput formula (1) holds) and UDP flows are associated to the streaming of long CBR-encoded multimedia flows. Furthermore, we assume that the short-lived TCP flows, even if more numerous than long lived flows, do not affect the performance of long-lived TCP flows. (This assumption is compatible with the natural scaling that is expected to occur as the Internet grows, see [14].)

In general, since the bottleneck queue is seen as a fluid queue, we can write

$$\sum_{j\in\mathscr{I}\cup\mathscr{N}} \lambda_j(1 - p_j) = \mu.$$

If we operate in the linear part of the RED curve then this leads to the system of equations:

$$\begin{cases} \sum_{j\in\mathscr{I}\cup\mathscr{N}} \lambda_j(1-p_j) = \mu, \\ p_i = t_i(q - q_{\min}), \quad \forall i \in \mathscr{I}\cup\mathscr{N} \end{cases}$$

with $(N+I+1)$ unknowns: q (average queue length), and p_i, $i \in \mathscr{I}\cup\mathscr{N}$, where λ_i, $i\in\mathscr{N}$ is given by (1). Substituting (1) and

$$p_i = t_i(q - q_{\min}) \quad \forall i, \tag{2}$$

into the first equation of the above set, we obtain a single equation for q:

$$\sum_{j\in\mathscr{N}} \frac{1}{R_j}\sqrt{\frac{\alpha}{t_j(q-q_{\min})}}(1 - t_j(q-q_{\min})) + \sum_{j\in\mathscr{I}} \lambda_j(1 - t_j(q - q_{\min})) = \mu. \tag{3}$$

If we write $x = \sqrt{q - q_{\min}}$, then (3) can be written as a cubic equation in x:

$$Z(x) = z_3x^3 + z_2x^2 + z_1x + z_0 = 0, \tag{4}$$

where

$$z_3 = \sum_{j\in\mathscr{I}} \lambda_j t_j, \quad z_2 = \sum_{j\in\mathscr{N}} \frac{1}{R_j}\sqrt{\alpha t_j},$$
$$z_1 = \mu - \sum_{j\in\mathscr{I}} \lambda_j, \quad z_0 = -\sum_{j\in\mathscr{N}} \frac{1}{R_j}\sqrt{\frac{\alpha}{t_j}}.$$

Note that this equation has a unique positive solution if there are only TCP or only real-time connections; in either case, it becomes a quadratic equation.

Proposition 1. *Fix the values of t_j, $j \in \mathscr{I}\cup\mathscr{N}$. The cubic Eq. (4) has a unique real positive solution. Assume that the solution lies in the linear region of RED. Then the average queue size is given as $q_{min} + x^2$ where x is the unique positive solution of (4) and the loss probability for session i is given by $p_i = t_i(q - q_{min})$.*

Proof. Assume first that $\mathscr{I}$ and $\mathscr{N}$ are both non-empty. Since the coefficients of the cubic equation are real, it has either a single real solution and two other conjugate complex solutions, or it has three real solutions [15]. Consider first the case in which all solutions are real. Then since the product of solutions is positive (it equals $-z_0/z_3$), there are either one or three positive solutions. But the latter is excluded since the sum of solutions is positive (it equals $-z_2/z_3$).

Next consider the case of a single real solution. Since the two other solutions are conjugate, their product is positive. Then since the product of all solutions is positive (it equals $-z_0/z_3$), the real solution is positive. □

Note that, in the case of only real-time connections ($\mathscr{N} = \emptyset$) operating in the linear region, we have

$$q = q_{\min} + \frac{\sum_{j\in\mathscr{I}}\lambda_j - \mu}{\sum_{j\in\mathscr{I}}\lambda_j t_j} \tag{5}$$

and

$$p_i = t_i \frac{\sum_{j\in\mathscr{I}}\lambda_j - \mu}{\sum_{j\in\mathscr{I}}\lambda_j t_j}. \tag{6}$$

(Recall that, throughout the paper, when considering this case we shall assume that $\sum_{j\in\mathscr{I}}\lambda_j > \mu$.)

In the case of only TCP connections ($\mathscr{I} = \emptyset$) operating in the linear region, we have

$$q = q_{\min} + \frac{\left(-\mu + \sqrt{\mu^2 + 4\alpha\sum_{j\in\mathscr{N}}\left(\frac{1}{R_j\sqrt{t_j}}\right)\sum_{j\in\mathscr{N}}\left(\frac{\sqrt{t_j}}{R_j}\right)}\right)^2}{4\alpha\left(\sum_{j\in\mathscr{N}}\frac{\sqrt{t_j}}{R_j}\right)^2} \tag{7}$$

and

$$p_i = t_i \frac{\left(-\mu + \sqrt{\mu^2 + 4\alpha\sum_{j\in\mathscr{N}}\left(\frac{1}{R_j\sqrt{t_j}}\right)\sum_{j\in\mathscr{N}}\left(\frac{\sqrt{t_j}}{R_j}\right)}\right)^2}{4\alpha\left(\sum_{j\in\mathscr{N}}\frac{\sqrt{t_j}}{R_j}\right)^2}. \tag{8}$$

4. Utility, pricing and equilibrium

We denote a strategy vector by $\mathbf{t}$ for all flows such that the jth entry is t_j. By $(t_i, [\mathbf{t}]_{-i})$, we define a strategy where flow i uses t_i and all other flows $j \neq i$ use t_j from vector $[\mathbf{t}]_{-i}$.

We associate to flow i a utility U_i. The utility will be a function of the QoS parameters and the price payed by flow i, and is determined by the actions of all flows. More precisely, $U_i(t_i, [\mathbf{t}]_{-i})$ is given by

$$a_i\lambda_i(1 - p(t_i, [\mathbf{t}]_{-i})) - b_i p(t_i, [\mathbf{t}]_{-i}) - d(t_i),$$
$$a_i > 0, \ \ b_i \geqslant 0$$

where the first term stands for the utility for the goodput, the second term stands for the dis-utility for the loss rate and the last term corresponds to the price $d(t_i)$ to be paid by flow i to the network.[2]

In particular, we find it natural to assume that a TCP flow i has $b_i = 0$ (as lost packets are retransmitted anyhow, and their impact is already taken into account in the throughput). Moreover, since λ_i for TCP already includes the loss term $p_i(t_i, [t]_{-i})$, the utility function of TCP is assumed to be

$$U_i(t_i, [\mathbf{t}]_{-\mathbf{i}}) = a_i\lambda_i(1 - p(t_i, [\mathbf{t}]_{-i})) - d(t_i).$$

We assume that the strategies or actions available to session i are given by a compact set of the form:

$$t_i \in S_i \quad \text{where } S_i = \left[t^i_{\min}, t^i_{\max}\right], \quad i \in \mathcal{I} \cup \mathcal{N}.$$

Here we assume that $t^i_{\min} > 0$ for all $i \in \mathcal{I} \cup \mathcal{N}$.

Each flow of the network strives to find its best strategy so as to maximize its own objective function. Nevertheless its objective function depends upon its own choice but also upon the choices of the other flows. In this situation, the solution concept widely accepted is the concept of Nash equilibrium.

Definition 1. A Nash equilibrium of the game is a strategy profile $\mathbf{t} = (t_1, t_2, \ldots, t_M)$ where $M = I + N$ from which no flow has any incentive to deviate. More precisely, the strategy profile $\mathbf{t}$ is a Nash equilibrium if the following holds true for any i

$$t_i \in \arg\max_{\bar{t}_i \in S_i} U_i(\bar{t}_i, [\mathbf{t}]_{-i}).$$

t_i is the best strategy that flow i can use if the other flows choose the strategies $[\mathbf{t}]_{-i}$.

Note that the network income is given by $\sum_{i \in \mathcal{I} \cup \mathcal{N}} d(t_i)$. Since the $p_i(t_i, [\mathbf{t}]_{-i})$ are functions of t_i and $[\mathbf{t}]_{-i}$, d can include pricing per volume of traffic successfully transmitted. In particular, we allow for d to depend on the uncontrolled arrival rates of real-time sessions (but since these are constants, we do not make them appear as an argument of the function d).

We shall sometimes find it more convenient to represent the control action of connection i as $T_i = 1/t_i$ instead of as t_i. Clearly, properties such as existence or uniqueness of equilibrium in terms of t_i directly imply the corresponding properties with respect to T_i.

5. Equilibrium for only real-time sessions or only TCP connections

We assume throughout that $t^i_{\max} \leqslant 1/(q_{\max} - q_{\min})$ for all connections. The bound for $t^i_{\max}$ is given so that we have $t^i_{\max}(q_{\max} - q_{\min}) \leqslant 1$. From (2) we see that $p_i \leqslant 1$ with equality obtained only for the case $t_i = 1/(q_{\max} - q_{\min})$.[3]

In our analysis, we are interested mainly in the linear region. For only real-time sessions or only TCP connections, we state the assumptions and describe the conditions for linear region operations and we show the existence of a Nash equilibrium.

Theorem 1. *A sufficient condition for the system to operate in the linear region is that for all i*:

1. *For only real-time connections*:

$$\lambda > \mu \quad \text{and} \quad t^i_{\min} > \frac{\lambda - \mu}{\lambda(q_{\max} - q_{\min})}. \tag{9}$$

[2] Linear utilities are commonly used for their tractability (see e.g. [16]), but they also have some mathematical justification: a utility that is given as the sum of (weighted) performance measures can be interpreted as the Lagrange relaxation of constraints that are imposed on the average delays, average loss probabilities, etc.

[3] Note that if the assumption does not hold then for some value $q' < q_{\max}$ we would already have for some i, $p_i = 1$ so one could redefine $q_{\max}$ to be q'. An important feature in our model is that the queue length beyond which $p_j = 1$ should be the same for all j.

2. *For only TCP connections*:

$$t^i_{\min} > \left(\frac{-\mu + \sqrt{\mu^2 + 4\alpha(\sum_{j\in\mathcal{N}} \frac{1}{R_j})^2}}{4\sqrt{\alpha\Delta q}\sum_{j\in\mathcal{N}} \frac{1}{R_j}} \right)^2, \tag{10}$$

where $\lambda = \sum_{j\in\mathcal{I}} \lambda_j$ *and* $\Delta q := q_{max} - q_{min}$.

Proof. The condition (9) (respectively (10)) will ensure that the value of q obtained in the linear region (see (5) and (7), respectively) is not larger that $q_{\max}$. Indeed, for real-time connections, (9) implies that

$$\sum_{j\in\mathcal{I}} \lambda_j t_j > \frac{\lambda - \mu}{q_{\max} - q_{\min}},$$

which implies together with (5) that $q < q_{\max}$.

Finally, the fact that the queue size is not below the lower extreme of the linear region (i.e., $p_i > 0$ for all i) is a direct consequence of $\lambda > \mu$.

The case of only TCP connections is proved in Appendix A.1. □

The following result establishes the existence of Nash equilibrium for only real-time sessions or only TCP connections.

Theorem 2. *Consider either the case of only real-time sessions or of only TCP connections. Assume that the system operates at the linear regime and the functions d are convex in* $T_i := 1/t_i$. *Then a Nash equilibrium exists.*

Proof. See Appendix A.2. □

5.1. Supermodular games

Let us now introduce the notion of a *supermodular* game, which will be used in Theorems 3–5 below. Supermodular games have the following appealing monotonicity property: for any user i and any fixed policy $[\mathbf{t}]_{-i}$ of the users other than i, the best response of user i to the other users' policy $[\mathbf{t}]_{-i}$ is *monotone* in $[\mathbf{t}]_{-i}$.

This implies the following properties of supermodular games.

- Several dynamic update schemes (for example, a round robin one) converge to a Nash equilibrium. For example, if we start with all players using their *smallest* available action and a round robin update scheme is used (where at each time period another player changes its action to a best response against the actions used by other players) then the sequence of actions will be monotone nondecreasing and hence will converge to a limit. (More details will be given below in the so called "Greedy Algorithm" that we shall introduce.)
- This limit turns out to be an equilibrium. Hence the monotonicity property of the best response sequence implies *existence* of an equilibrium.
- Using the same procedure when starting with the largest strategy of each user gives a monotone decreasing sequence whose limit is again a (possibly different) Nash equilibrium. For more details see [17].

Definition 2. The game $(S_1, \ldots, S_M, U_1, \ldots, U_M)$ is supermodular if for all i

- S_i is a sublattice,[4]
- U_i is upper semi-continuous in t_i and $[\mathbf{t}]_{-i}$,
- U_i has nondecreasing differences in $(t_i, [\mathbf{t}]_{-i})$, i.e., for all $t_i \geqslant t'_i$ and $[\mathbf{t}]_{-i} \geqslant [\mathbf{t}]'_{-i}$,

$$U_i(t_i, [\mathbf{t}]_{-i}) - U_i(t'_i, [\mathbf{t}]_{-i}) \geqslant U_i(t_i, [\mathbf{t}]'_{-i}) - U_i(t'_i, [\mathbf{t}]'_{-i}),$$

where $S_i = [t^i_{\min}, t^i_{\max}]$, $S = S_1 \times S_2 \cdots \times S_M$ and $M = I + N$.

By nondecreasing differences in $(t_i, [\mathbf{t}]_{-i})$, we mean that it has the property that the incremental gain by choosing a greater t_i is greater when $[\mathbf{t}]_{-i}$ is larger. For example if the utility of user i has nondecreasing differences in the vector $\mathbf{t}$, user i increases her utility if she increases her slope in response to an increase in the slope of another user j. If U_i is twice differentiable, then the supermodularity is equivalent to

[4] S_i is a sublattice of $\mathbb{R}^M$ if $\mathbf{t} \in S_i$ and $\mathbf{t}' \in S_i$ imply that $\mathbf{t} \wedge \mathbf{t}' \in S_i$ and $\mathbf{t} \wedge \mathbf{t}' \in S_i$, where $\mathbf{t} \wedge \mathbf{t}' = (\max(t_1, t'_1), \ldots, \max(t_M, t'_M))$ and $\mathbf{t} \wedge \mathbf{t} = (\min(t_1, t'_1), \ldots, \min(t_M, t'_M))$.

$$\frac{\partial^2 U_i}{\partial t_i \partial t_j} \geqslant 0, \tag{11}$$

for all $\mathbf{t}$ in S. Applying Topkis' Theorem [17] in this context shows immediately that each flow's best response function is increasing in the action of the other flows. An useful propriety of super-modular games is that we can use monotonicity to prove the existence of equilibria and greedy algorithms. A greedy algorithm is a simple, so-called tatônnement of Round Robin scheme for best response that converges to the equilibrium.

Let us now introduce the following asynchronous dynamic greedy algorithm (GA).

Greedy Algorithm. Assume a given initial choice $\mathbf{t^0}$ for all flows. At some strictly increasing times τ_k, $k = 1, 2, 3, \ldots$, flows update their actions; the actions t_i^k at time $\tau_k > 0$ are obtained as follows. A single flow i at time τ_{k+1} updates its t_i^{k+1} so as to optimize $U_i(\cdot, [\mathbf{t}^k]_{-i})$ where $[\mathbf{t}^k]_{-i}$ is the vector of actions of the other flows $j \neq i$. We assume that each flow updates its actions infinitely often. In particular, for the case of only real-time sessions, we update t_i^{k+1} as follows:

$$t_i^{k+1} = \underset{t_i \in [t^i_{\min}, t^i_{\max}]}{\arg\max}\, a_i \lambda_i (1 - p_i) - b_i p_i - d(t_i), \tag{12}$$

where p_i in (12) is given by (6).

For the TCP-only case, we update t_i^{k+1} as follows:

$$t_i^{k+1} = \underset{t_i \in [t^i_{\min}, t^i_{\max}]}{\arg\max}\, \frac{a_i}{R_i} \sqrt{\frac{\alpha}{p_i}} (1 - p_i) - d(t_i), \tag{13}$$

where p_i in (13) is given by (8).

We assume that the duration of a stage is quite long, so that sufficient information can be obtained by the user in order to be able to estimate p_i.

Remark 1. For the case of real-time sessions, we may obtain a closed-form solution for t_j^{k+1} with specific cost function $d(t_i)$ such as $\frac{d}{t_i}$ which will lead to update of t_i^{k+1} as follows:

$$\delta_i^k = \frac{\sum_{j \neq i} \lambda_j t_j^k}{\sqrt{(a_i \lambda_i + b_i)(\sum_{j \in \mathscr{I}} \lambda_j - \mu)(\sum_{j \neq i} \lambda_j t_j^k)} - \lambda_i \sqrt{d}},$$

where δ_i^k is such that $\frac{\partial U_i}{\partial t_i}|_{t_i = \delta_i^k} = 0$ and U_i corresponds to the utility function of real-time session i. Then t_i^{k+1} is given by:

$$t_i^{k+1} = \begin{cases} t^i_{\min} & \text{if } \delta_i^k < 0, \\ t^i_{\max} & \text{if } \delta_i^k < t^i_{\min}, \quad \delta_i^k \geqslant 0, \\ t^i_{\min} & \text{if } \delta_i^k > t^i_{\max}, \quad \delta_i^k \geqslant 0, \\ \delta_i & \text{otherwise}. \end{cases}$$

Theorem 3. *For the case of only real-time connections we assume that* $\forall j$, $\lambda_{min} \leqslant \lambda_j \leqslant \lambda_{max}$, *and*

$$(I - 1)\lambda_{\min} t_{\min} \geqslant \lambda_{\max} t_{\max},$$

where $t_{\min} = \min_{i \in \mathscr{I}}\{t^i_{\min}\}$ *and* $t_{\max} = \max_{i \in \mathscr{I}}\{t^i_{\max}\}$. *Then there is smallest equilibrium* $\underline{t}$ *and largest equilibrium* $\overline{t}$, *and the* **GA** *dynamic algorithm converges to* $\underline{\mathbf{t}}$ (*respectively* $\overline{\mathbf{t}}$) *provided it starts with* $t^j_{\min}$ *for all* j (*respectively* $t^j_{\max}$ *for all* j).

Proof. Both statements will follow by showing that the game is super-modular, see [17,18]. A sufficient condition is that

$$\frac{\partial^2 U_i}{\partial t_i \partial t_j} = -(a_i \lambda_i + b_i) \frac{\partial^2 p_i}{\partial t_i \partial t_j} \geqslant 0.$$

We have

$$\frac{\partial p_i}{\partial t_i} = \left(\sum_j \lambda_j - \mu \right) \left(\frac{1}{\sum_{j \in I} \lambda_j t_j} - \frac{t_i \lambda_i}{(\sum_{j \in I} \lambda_j t_j)^2} \right),$$

leading to

$$\frac{\partial^2 p_i}{\partial t_i \partial t_k} = \lambda_k \left(\sum_{j \in \mathscr{I}} \lambda_j - \mu \right) \frac{-\sum_{j \in \mathscr{I}} \lambda_j t_j + 2 t_i \lambda_i}{(\sum_{j \in \mathscr{I}} \lambda_j t_j)^3}.$$

The latter is nonpositive if and only if $\sum_{j \neq i} \lambda_j t_j \geqslant \lambda_i t_i$. A sufficient condition is that $(I - 1)\lambda_{\min} t_{\min} \geqslant \lambda_{\max} t_{\max}$. Thus the game is super-modular. The result then follows from standard theory of super-modular games [17,18]. □

Theorem 4. *For the case of only real-time connections, we assume that* $\forall j$, $\lambda_{min} \leqslant \lambda_j \leqslant \lambda_{max}$, *and* $2 t^3_{\min} \lambda^2_{\min} > t^3_{\max} \lambda^2_{\max}$. *Under supermodular condition, the Nash equilibrium is unique.*

Proof. See Appendix A.3. □

Theorem 5. *For the case of only TCP connections, assume that* $\forall j$, $t_{min} \leqslant t \leqslant t_{max}$ *and*

$$(3 + p_i)\frac{\partial p_i}{\partial t_i}\frac{\partial p_i}{\partial t_j} \geqslant 2p_i(p_i + 1)\frac{\partial^2 p_i}{\partial t_i \partial t_j} \forall\ i, j,\ \ i \neq j. \tag{14}$$

Then the game is super-modular.

Proof. See Appendix A.4. □

Remark 2. It would also be interesting to consider a price per unit of received volume, i.e., of the form $d(t_i)\lambda_i(1 - p_i)$. However, looking at the super-modularity of the utility function gives a condition depending on $d'(t_i)$, $d(t_i)$ and the t_j that does not seem tractable. On the other hand, we can consider a pricing per unit of *sent volume*, i.e., of the form $d(t_i)\lambda_i$ (since λ_i is fixed), Conditions of Theorems 2,3 then hold to provide a Nash equilibrium.

Note that in the model presented above, users choose at each stage an action that maximizes their utility function, depending on the actions of all other flows. This dependence appears in the loss probability p_i. In our case, a user can determine her utility function without hypothesis of full knowledge: since users only need to have *aggregate* information about other flows (like the total rate $\sum_{j\in\mathcal{I}}\lambda_j$ in the CBR-only scenario), there are in principle no scalability issues, i.e., no need of exchanging or storing per-flow information at the routers. The issue of how such aggregate values are signaled to sources is outside the scope of this paper.

Remark 3. As already mentioned, in supermodular games, one can obtain equilibrium dynamically in stages, such that during each stage, users choose an action that maximizes their utility function, depending on the actions of all other flows. This dependence appears in the loss probability p_i. In our case, a user can determine his utility function without hypothesis of full knowledge of the actions of other players. Indeed, in real-time connections, it is possible for each source to obtain the sufficient information for determining its actions by using RTP/RTCP (each source can obtain the receiver reports (RRs) that include the reception quality statistics such as the number of packets received, fraction lost, and cumulative number of packets lost). Hence the source i can obtain the loss probability $\bar{p}_i$ at each stage. For TCP connections, the ACK packets could be sufficient to acquire the loss probabilities p_i.

In case of only real-time connections, the loss probability at stage k is given by

$$p_i(t_i, [t^{k-1}]_{-i}) = t_i \frac{\sum_{j\in\mathcal{I}}\lambda_j - \mu}{\sum_{j\neq i}\lambda_j t_j^{k-1} + \lambda_i t_i}, \tag{15}$$

where t_i is the action of user i and t_j^{k-1} is the optimal action of user j at stage $k-1$. At stage $k-1$, we have

$$\bar{p}_i^{k-1} = p_i(t_i^{k-1}, [t^{k-1}]_{-i}) = t_i^{k-1}\frac{\sum_{j\in\mathcal{I}}\lambda_j - \mu}{\sum_{j\in\mathcal{I}}\lambda_j t_j^{k-1}},$$

where t_i^{k-1} is the optimal action of user i at stage $k-1$. Note that the loss probability $\bar{p}_i^{k-1}$ is estimated through RTP/RTCP protocol at end of stage $k-1$. Hence at stage k, when the action of user i is t_i, the loss probability (15) of user i becomes:

$$\begin{aligned} p_i(t_i, [t^{k-1}]_{-i}) &= t_i \frac{\sum_{j\in\mathcal{I}}\lambda_j - \mu}{\sum_{j\neq i}\lambda_j t_j^{k-1} + \lambda_i t_i}, \\ &= \frac{t_i}{t_i^{k-1}}\left(\frac{\sum_{j\in\mathcal{I}}\lambda_j t_j^{k-1} + \lambda_i(t_i - t_i^{k-1})}{t_i^{k-1}\left(\sum_{j\in\mathcal{I}}\lambda_j - \mu\right)}\right)^{-1} \\ &= \frac{t_i}{t_i^{k-1}}\left(\frac{1}{\bar{p}_i^{k-1}} + \frac{\lambda_i(t_i - t_i^{k-1})}{t_i^{k-1}\left(\sum_{j\in\mathcal{I}}\lambda_j - \mu\right)}\right)^{-1} \\ &= \hat{p}(t_i, t_i^{k-1}, \bar{p}_i^{k-1}). \end{aligned}$$

Thus, the utility function becomes:

$$\begin{aligned} U_i(t_i, [\mathbf{t}]_{-i}) &= a_i\lambda_i(1 - \hat{p}(t_i, t_i^{k-1}, \bar{p}_i^{k-1})) \\ &\quad - b_i\hat{p}(t_i, t_i^{k-1}, \bar{p}_i^{k-1}) - d(t_i) \\ &= \hat{U}_i(t_i, t_i^{k-1}, \bar{p}_i^{k-1}). \end{aligned}$$

In the above formulation, the sources need to know the total rate through the bottleneck router in order to execute the iteration. This can be achieved using bottleneck capacity estimation (e.g., pathrate, pathchar) and available bandwidth estimation tools (e.g., pathload), see [19, 20].

In the case of only TCP connections, the loss probability at stage k is given by

$$p_i(t_i,[t^{k-1}]_{-i}) = \frac{t_i\left(-\mu+\sqrt{\mu^2+4\alpha\left\{\sum_{j\neq i}\left(\frac{1}{R_j\sqrt{t_j^{k-1}}}\right)+\frac{1}{R_i\sqrt{t_i}}\right\}\left\{\sum_{j\neq i}\left(\frac{\sqrt{t_j^{k-1}}}{R_j}\right)+\frac{\sqrt{t_i}}{R_i}\right\}}\right)^2}{4\alpha\left(\sum_{j\neq i}\frac{\sqrt{t_j^{k-1}}}{R_j}+\frac{\sqrt{t_i}}{R_i}\right)^2},$$

where t_i is the action of TCP i and t_j^{k-1} is the optimal action of TCP j at stage $k-1$. Note that at stage $k-1$, we have

$$\bar{p}_i^{k-1} = p_i(t_i^{k-1},[t^{k-1}]_{-i}) = \frac{t_i^{k-1}\left(-\mu+\sqrt{\mu^2+4\alpha\sum_j\left(\frac{1}{R_j\sqrt{t_j^{k-1}}}\right)\sum_j\left(\frac{\sqrt{t_j^{k-1}}}{R_j}\right)}\right)^2}{4\alpha\left(\sum_j\frac{\sqrt{t_j^{k-1}}}{R_j}\right)^2}.$$

From the definition of loss probability at stage k, it is difficult to express p_i as a function of $\bar{p}_i^{k-1}$, t_i and t_i^{k-1}, as in real-time connections case. The source needs to estimate the value of $\left(\frac{1}{R_j\sqrt{t_j^{k-1}}}\right)$ and of $\left(\frac{\sqrt{t_j^{k-1}}}{R_j}\right)$.

From this reason, we define another algorithm which allows us to obtain the Nash equilibrium without estimating the value of $\left(\frac{1}{R_j\sqrt{t_j^{k-1}}}\right)$ and $\left(\frac{\sqrt{t_j^{k-1}}}{R_j}\right)$. In this algorithm we consider that the probability at stage k is approximated by

$$p_i = p_i(t_i,t_i^{k-1},\bar{p}_i) = \frac{t_i\left(-\mu+\sqrt{\mu^2+4\alpha\sum_j\left(\frac{1}{R_j\sqrt{t_j^{k-1}}}\right)\sum_j\left(\frac{\sqrt{t_j^{k-1}}}{R_j}\right)}\right)^2}{4\alpha\left(\sum_j\frac{\sqrt{t_j^{k-1}}}{R_j}\right)^2} = \frac{t_i}{t_i^{k-1}}\bar{p}_i^{k-1}.$$

Thus, the utility function becomes

$$U_i(t_i,[\mathbf{t}]_{-i}) = a_i\lambda_i(1-p(t_i,t_i^{k-1},\bar{p}_i^{k-1}))-d(t_i) = \hat{U}_i(t_i,t_i^{k-1},\bar{p}_i^{k-1}).$$

From the definition of the loss probability at stage k (see (16)), we can see that if this algorithm converges, it will be to a Nash equilibrium. We postpone to future work the mathematical analysis of the convergence of that algorithm. Nonetheless, using numerical simulations, we have found out that so far, the iterative algorithm always converged to a Nash equilibrium.

6. Symmetric users

In this section, we assume that all flows have the same utility function (for all i, $a_i = a$, $\lambda_i = \bar{\lambda}$ and $b_i = b$ for real-time sessions and $a_i = a$ and $R_i = R$ for TCP connections) and the same intervals for strategies ($t^i_{\min} = t_{\min}$ and $t^i_{\max} = t_{\max}$).

6.1. Algorithm for symmetric Nash equilibrium

For symmetric Nash equilibrium, we are interested in finding a symmetric equilibrium strategy $\mathbf{t}^* = (t^*, t^*, \ldots, t^*)$ such that for any flow i and any strategy t_i for that flow (real-time session or TCP connection),

$$U(\mathbf{t}^*) \geqslant U(t_i,[\mathbf{t}^*]_{-i}).$$

Next we show how to obtain an equilibrium strategy. We first note that due to symmetry, to see whether $\mathbf{t}^*$ is an equilibrium it suffices to check (6) for a single flow. We shall thus assume that there are $L+1$ flows all together, and that the first L flows use the strategy $\mathbf{t}^{\mathrm{o}} = (t^{\mathrm{o}},\ldots,t^{\mathrm{o}})$ and flow $L+1$ uses t_{L+1}. Define the set

$$\mathcal{Q}_{L+1}(\mathbf{t}^{\mathrm{o}}) = \arg\max_{t_{L+1}\in[t_{\min},t_{\max}]}(U(t_{L+1},[\mathbf{t}^{\mathrm{o}}]_{-(L+1)})),$$

where $\mathbf{t}^{\mathbf{o}}$ denotes (with some abuse of notation) the strategy where all flows use t^{o}, and where the maximization is taken with respect to t_{L+1}. Then $\mathbf{t}^*$ is a symmetric equilibrium if

$$t^* \in \mathcal{Q}_{L+1}(\mathbf{t}^*).$$

Theorem 6. *Consider real-time connections only, operating in the linear region. Assume that the functions d are convex in* $T_i := 1/t_i$*. The symmetric equilibrium* t^* *satisfies*:

$$T^* \frac{\partial \hat{d}(T)}{\partial T}\bigg|_{T=T^*} = \frac{a\lambda + b}{(I\bar{\lambda})^2},$$

where $T^* = 1/t^*$ *and* $\hat{d}(T) = d(\frac{1}{T})$.

Proof. Recall that $\lambda = I\bar{\lambda}$. Then for real-time connections, we have

$$U = a\bar{\lambda} - (a\bar{\lambda} + b)\frac{(\lambda - \mu)}{\bar{\lambda} + T_i\sum_{j\neq i}\bar{\lambda}/T_j} - \hat{d}(T_i),$$

which gives, when considering the derivative,

$$\frac{\partial U}{\partial T_i} = (a\bar{\lambda} + b)\frac{(\lambda - \mu)\sum_{j\neq i}\bar{\lambda}/T_j}{(\bar{\lambda} + T_i\sum_{j\neq i}\bar{\lambda}/T_j)^2} - \frac{\partial \hat{d}(T_i)}{\partial T_i}.$$

Equating $\frac{\partial U}{\partial T_i} = 0$ we obtain (6). □

7. Optimal pricing

The goal here is to determine a pricing strategy that maximizes the network's benefit. Typically, pricing is motivated by two different objectives: (1) it generates revenue for the system and (2) it encourages the players to use the system resources more efficiently.

Our focus here is on pricing strategies for revenue maximization, i.e., how a service provider should price resources to maximize revenue. The corresponding maximization problem is given by

$$c(\mathbf{t}^*) = \arg\max_d \sum_{i=1}^{I} d(t_i^*),$$

where $\mathbf{t}^*$ is a Nash equilibrium which can be obtained when considering special classes function of $d(\cdot)$ depending on a real parameter that we will also (with some abuse of notation) call d. We then obtain a system of equations that can be solved numerically (to get the t^* satisfying the Nash equilibrium), and a numerical optimization over the parameter d can be obtained. We use in our numerical example $d(t) = d/\exp(t)$. We also considered other families of pricing functions such as d/t, d/t^2 and so on and we observed monotonous behaviour of cost c as a function of d.

Nevertheless, an assumption of this optimization problem is that the network knows the number of flows and the parameters a_i, b_i and R_i $\forall i$. A more likely situation is when the network only knows the distribution of the number of players I (now a random variable) and the distribution of parameters a_i, b_i and R_i (assumed independent and independent between flows for convenience). A numerical investigation of optimal parameters can be realized as well.

8. Numerical examples

In the following simulations, we obtain a unique Nash equilibrium for only real-time sessions or only TCP connections. Moreover, the GA algorithm converges as it satisfies the conditions of supermodularity. All the conditions of supermodular games (Theorems 3 and 5) and uniqueness of Nash equilibrium (Theorems 2 and 3) are only sufficient but not necessary as shown in the numerical results. The pricing function that we use for player i throughout this section is $d/\exp(t_i)$. We shall investigate how the choice of the constant d will affect the revenue of the network.[5]

8.1. Symmetric real-time flows

In the following numerical evaluations, we show the variation of different metrics as function of d. Figs. 2 and 3 correspond to a unique symmetric Nash equilibrium case in which all the real-time flows have $\lambda_i = 2$ Mbps with $t_{\min} = 1$, $t_{\max} = 15$, $\mathscr{I} = 20$, $q_{\min} = 10$, $q_{\max} = 40$, $\mu =$ 30 Mbps. Here we set the values of parameters to ensure that the

[5] We note that it is desirable to have a "nontrivial" parameterized pricing function that leads to an optimal revenue for some parameter. We also tested other pricing functions that did turn out to be "trivial" in the sense that the benefit was always monotone in the parameter; an example of such a function is $\exp(-\beta t_i)$ and the network optimizes with respect to β.

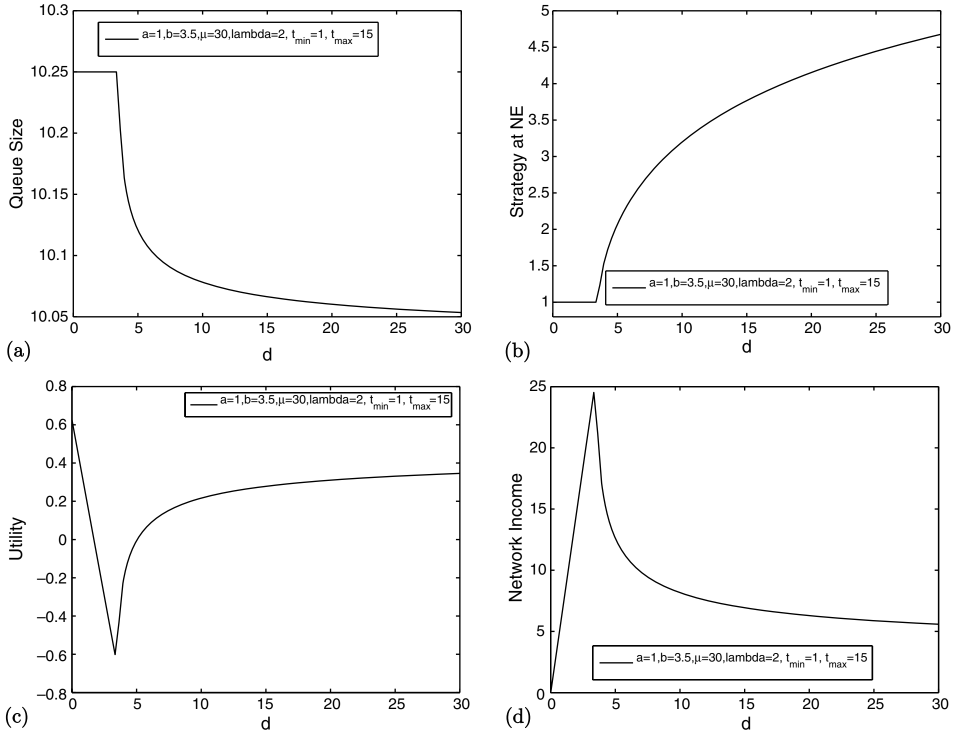


Fig. 2. Symmetric real-time flows. (a) Queue size vs. d, (b) t^* vs. d, (c) utility vs. d and (d) network income vs. d.

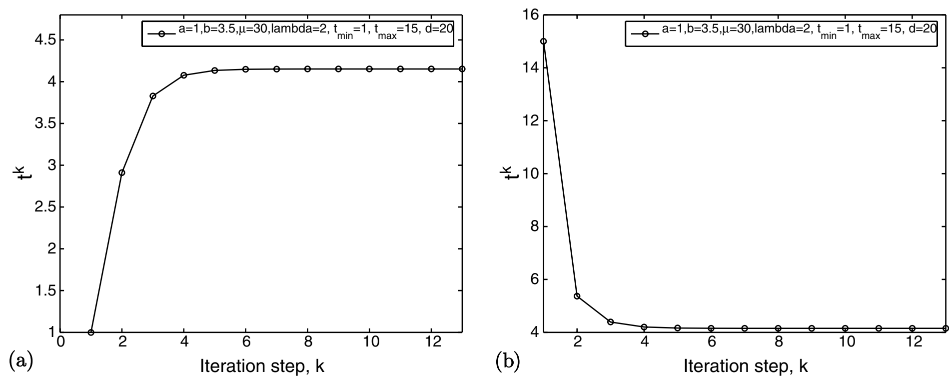


Fig. 3. Symmetric real-time flows: convergence to Nash equilibrium. (a) $t^0 = t_{\min}$ and (b) $t^0 = t_{\max}$.

system operates in the linear region such as $t_{\min} > \frac{1}{\Delta q}\left(1 - \frac{\mu}{\sum_{j\in\mathscr{I}}\lambda_j}\right) = 0.0083$. Moreover, the values above also ensure the uniqueness of the equilibrium (Theorem 3). The bound on $t_{\max}$ is needed only to limit the value of loss probability to 1. The value of d which maximizes the network revenue occurs at $d = 3.33$. All the flows attain a loss rate of 0.25. Note that for the real-time flows symmetric case, $p_i^* = (\sum_{j\in\mathscr{I}}\lambda_j - \mu)/\sum_{j\in\mathscr{I}}\lambda_j$ at the

Nash equilibrium is a constant. The average queue size, given by $q_{\min} + p_i^*/t_i^*$, is shown in Fig. 2. We observe the value of t^* at which maximum network income is achieved is close to $t_{\min}$ while the system operates in the linear region of RED throughout.

We plot in Fig. 3 sample paths of a connection that uses the GA Algorithm for symmetric users (Section 6) (the evolution for all connections is the same). The figure illustrates convergence to the same Nash equilibrium when t^0 started from $t_{\min}$ or $t_{\max}$. We plot it for $d = 20$. In Fig. 3(a), the value of t^* is 4.152208, and in Fig. 3(b), it is 4.152208.

8.2. Nonsymmetric real-time flows

In the next experiment, instead of having the symmetric case, the rates λ_i are drawn uniformly

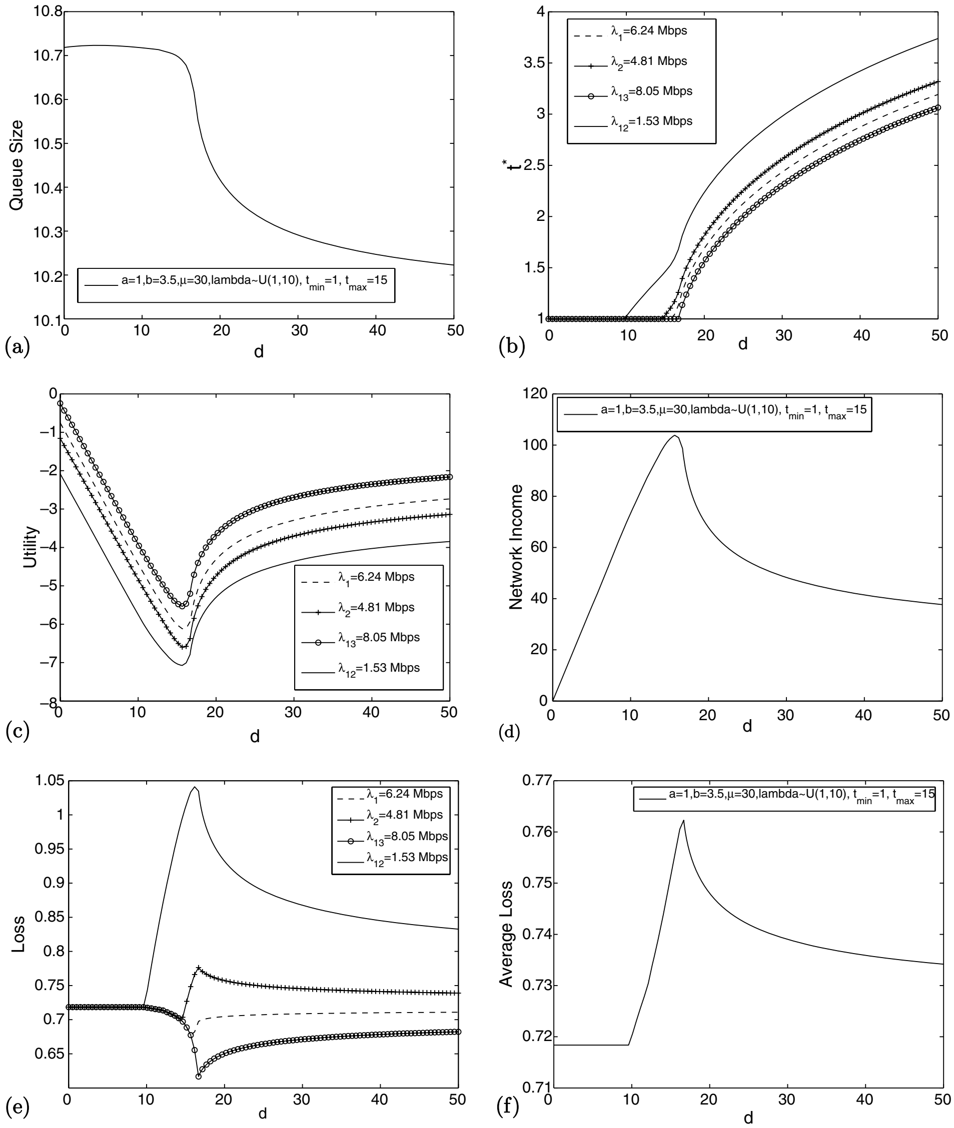


Fig. 4. Nonsymmetric real-time flows. (a) Queue size vs. d, (b) t^* vs. d, (c) utility vs. d, (d) network income vs. d, (e) loss probability vs. d and (f) average loss probability vs. d.

from $[1, 10]$ Mbps with $t_{\min} = 1$, $t_{\max} = 15$, $q_{\max} = 40$, $q_{\min} = 10$, $\mathscr{I} = 20$, $\mu = 30$ Mbps. Fig. 4 shows how different metrics vary with d at unique Nash equilibrium. To ensure that the flows operate in the linear region, we need $t_{\min} > \frac{1}{\Delta q} \geqslant \frac{1}{\Delta q}\left(1 - \frac{\mu}{\sum_j \lambda_j}\right)$. We observe that $d = 15.66$ maximizes the network revenue. Fig. 4(b) shows that values of t^* for flows having higher rates increase slower than that of flows having lower rates, i.e., higher rate flows experience less loss rates. Fig. 4(c) shows that flows having different rates gain similarly in their utility functions. We plot the individual and average loss rate in Fig. 4(e) and (f). We confirm in these experiments about uniqueness of Nash equilibrium, although the sample path of different connections will depend on the connection rates. The condition for uniqueness is that $2t_{\max}^3\lambda_{\min}^2 > t_{\min}^3\lambda_{\max}^2$ which in our case is given by, $2 \times 15^3 \times 1^2 = 6750 > 100 = 1^3 \times 10^2$.

8.3. Symmetric TCP connections

For symmetric TCP connections we have considered $R_i = R = 20$ ms for all connections with $t_{\min} = 2$, $t_{\max} = 20$, $\mu = 30$ Mbps, $N = 20$, $a = 0.1$. Fig. 5(a)–(d) show the effect of increasing d on the queue size, equilibrium strategy, utility and network income. Fig. 6(a)–(b) show the convergence to Nash equilibrium in case of symmetric TCP connections starting from $t_{\min}$ and $t_{\max}$ respectively. The maximum value of network revenue is found at $d = 0.6704$. In this symmetric case, the loss probability is given by

$$p^* = \frac{R^2}{3N^2}\left\{\mu^2 + \frac{3N^2}{R^2} - \mu\sqrt{\mu^2 + \frac{6N^2}{R^2}}\right\} = 0.0017.$$

To ensure that the symmetric TCP flows operate in the linear region, we satisfy the condition on $t_{\min}$:

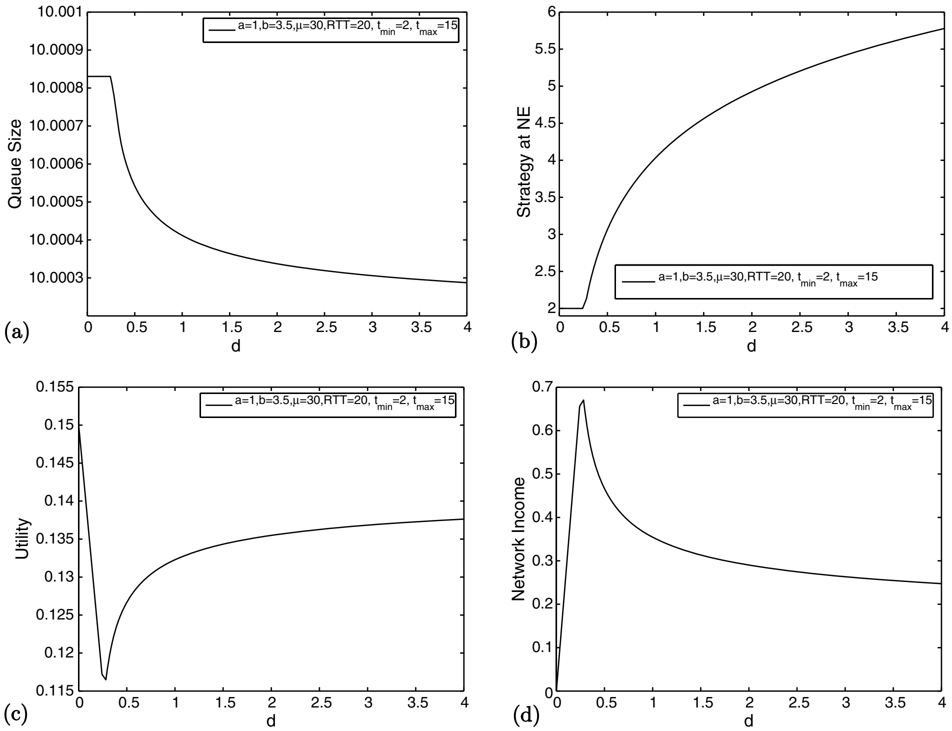


Fig. 5. Symmetric TCP Flows. (a) Queue size vs. d, (b) t^* vs. d, (c) utility vs. d and (d) network income vs. d.

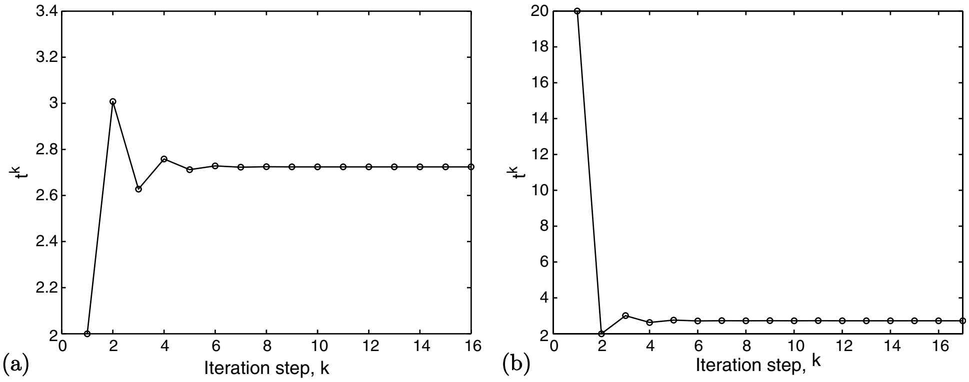


Fig. 6. Symmetric TCP flows: convergence to Nash equilibrium. (a) $t^0 = t_{\min}$ and (b) $t^0 = t_{\max}$.

$$t_{\min} > \left(\frac{-\mu + \sqrt{\mu^2 + 4(\sum_{j\in\mathcal{N}} \frac{1}{R_j})^2}}{4\sqrt{\alpha\Delta q}\sum_{j\in N} \frac{1}{R_j}}\right)^2 = 4.6271 \times 10^{-5}.$$

We plot sample paths of a connection which illustrate convergence to Nash equilibrium when t^0 started from $t_{\min}$ or $t_{\max}$. We plot it for $d = 0.1$. In Fig. 6(a), the value of t^* is 2.724076, and in Fig. 6(b), it is 2.724076.

8.4. Nonsymmetric TCP connections

We present a nonsymmetric case in Fig. 7 in which the R_is are drawn uniformly from [1, 20] ms with $t_{\min} = 2$, $t_{\max} = 20$, $\mu = 30$ Mbps, $N = 20$, $a = 0.2$, $q_{\max} = 40$, $q_{\min} = 10$. The value of d at which network revenue is highest is 0.8948. We ensure that the nonsymmetric connections operate in the linear region by setting:

$$t_{\min} > \left(\frac{-\mu + \sqrt{\mu^2 + 4(\sum_{j\in\mathcal{N}} \frac{1}{R_j})^2}}{4\sqrt{\alpha\Delta q}\sum_{j\in\mathcal{N}} \frac{1}{R_j}}\right)^2 = 0.5476.$$

8.5. Real-time flows and TCP connections

In this experiment (see Fig. 8), we combine both real-time and TCP connections. We have $I = 15$, $N = 15$, $\mu = 13$ Mbps, RTT = 10 ms, $t^{\text{real}}_{\min} = 5$, $t^{\text{real}}_{\max} = 11$, $t^{\text{TCP}}_{\min} = 5$, $t^{\text{TCP}}_{\max} = 11$, $\lambda = 1$ Mbps, $q_{\min} = 10$, $q_{\max} = 40$, a for both real-time and TCP connections are 100 and $b = 4$. We found two Nash equilibriums for each values of d. Therefore, we plot two curves in each plot corresponding to each Nash equilibrium. The highest network revenue is achieved at $d = 353.15$, $t^{\text{real}} = 11$, $t^{\text{TCP}} = 5$ and at $d = 254.35$, $t^{\text{real}} = 5$, $t^{\text{TCP}} = 11$. In the simulations, we observe the values of $q < q_{\max}$ and since there is at least one TCP flow i with throughput $\lambda_i > 0$, this implies that the flow has loss probability $p_i > 0$ and average queue length $q > q_{\min}$. We conclude that the system operates in the linear region. Our objective in this set of experiments is to show that there exists a Nash equilibrium for both real-time and TCP connections. The loss experienced by real-time flows at the first NE is 0.3676 and that by TCP flows is 0.1826. The corresponding values at the second NE is 0.2349 and 0.4461.

9. Conclusions and future work

We have studied in this paper a fluid model of the RED buffer management algorithm with different drop probabilities applied to both UDP and TCP traffic. We first computed the performance measures for fixed drop policies. We then investigated how the drop policies are determined. We modeled the decision process as a noncooperative game and obtained its equilibria. We showed the existence of the equilibria under various conditions, and provided ways for computing them (establishing also convergence properties of best-

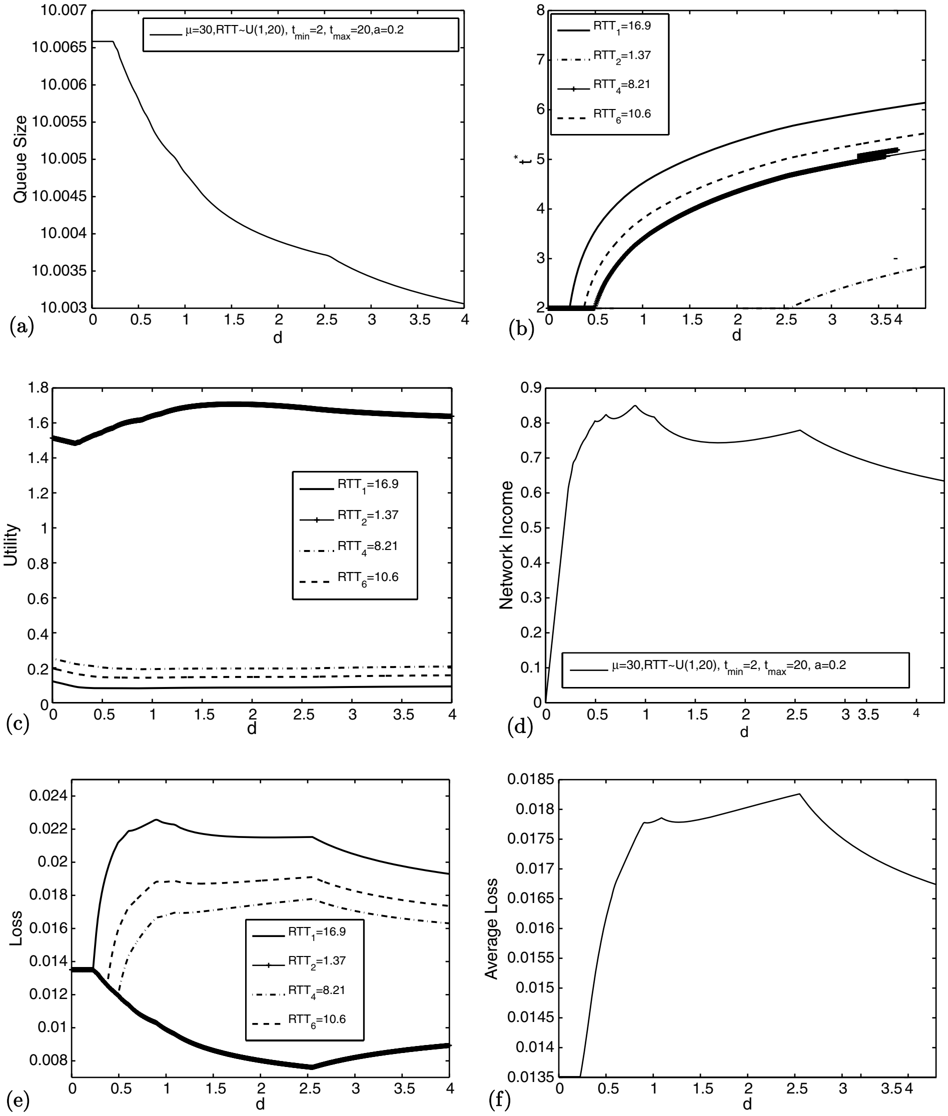


Fig. 7. Nonsymmetric TCP. (a) Queue size vs. d, (b) t^* vs. d, (c) utility vs. d, (d) network income vs. d, (e) loss probability vs. d and (f) average loss vs. d.

response dynamics). The goal of the network provider is to use a pricing function that is going to optimize its revenue. Determining an optimal function seems a difficult problem (left for future research), and we restricted ourselves to specific classes of functions where only one parameter varies. We finally addressed the problem of optimizing the revenue of the network provider.

Concerning the future work, we are working on deriving sufficient and necessary conditions for operating at the linear region when there are both real-time and TCP connections; these seem to be more involved than the conditions we have obtained already. We will further study the impact of buffer management schemes on the performance and on the revenues of the network; in particular,

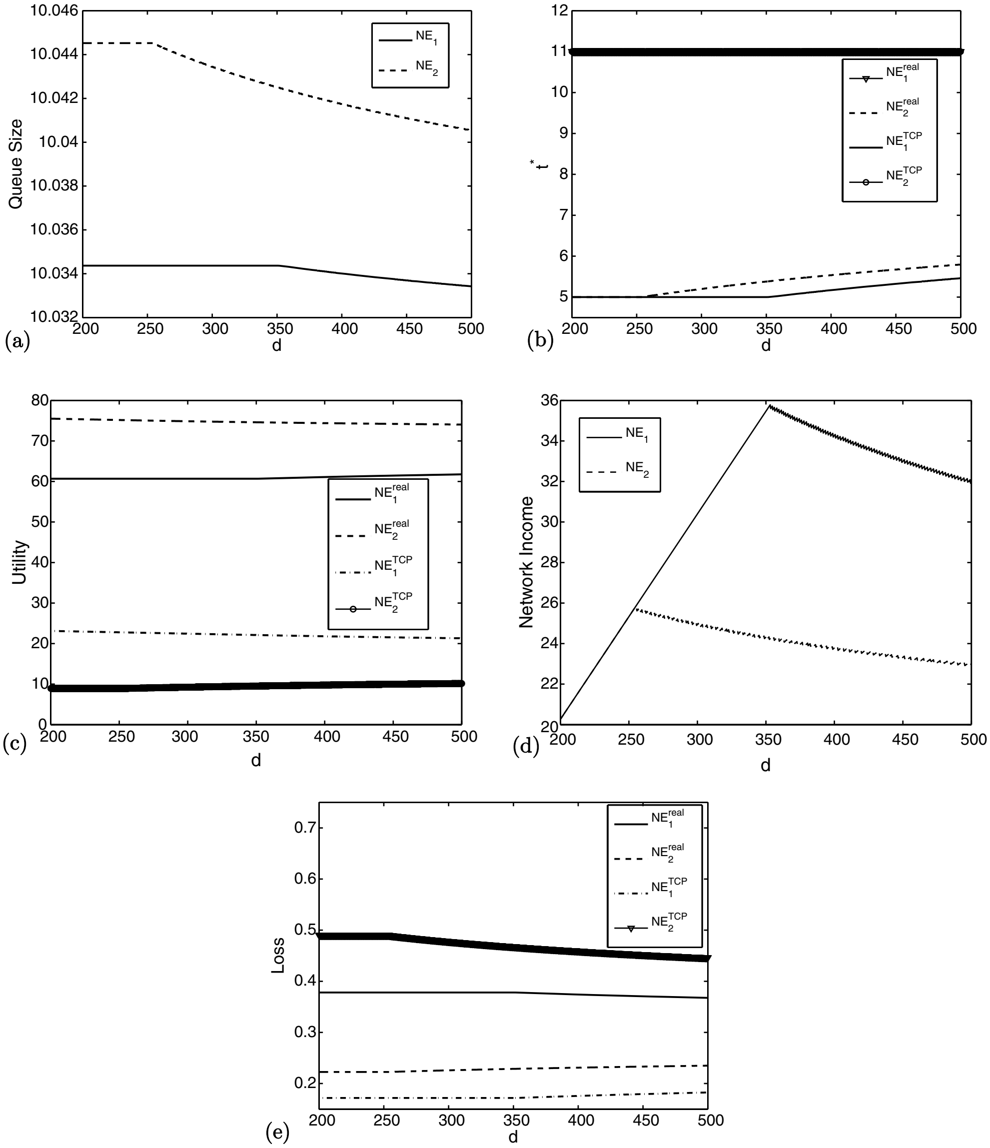


Fig. 8. Both real-time and TCP flows. (a) Queue size vs. d, (b) t^* vs. d, (c) utility vs. d, (d) network income vs. d and (e) average loss probabilities vs. d.

other versions of RED will be considered (such as the gentle-RED variant). We will also examine how well the fluid model is suitable for the packet-level model that it approximates.

We intend to consider in the future other utility functions, and in particular include delay and/or jitter terms. We plan to compare the performance of Nash equilibrium with the team problem in which the whole network efficiency is maximized. We shall then consider other pricing functions which would increase the efficiency of the Nash equilibrium.

Acknowledgements

The work of E. Altman and R. El Azouzi was supported in part by research contract 001B001 with France Telecom R&D. The work of D. Barman was performed during internship at INRIA, financed by the INRIA's PrixNet ARC collaboration project. All authors were supported in part by the EuroNGI Network of Excellence.

Appendix A

A.1. Proof of Part 2 of Theorem 1

For only TCP connections, we have,

$$\sqrt{q - q_{\min}} \leqslant \sqrt{q_{\max} - q_{\min}} = \sqrt{\Delta q}. \tag{16}$$

From (7), we get the following sufficient and necessary condition for $q \leqslant q_{\max}$:

$$\frac{-\mu + \sqrt{\mu^2 + 4\alpha\left(\sum_j \frac{\sqrt{t_j}}{R_j}\right)\left(\sum_j \frac{1}{R_j\sqrt{t_j}}\right)}}{2\sqrt{\alpha}\left(\sum_j \frac{\sqrt{t_j}}{R_j}\right)} \leqslant \sqrt{\Delta q}$$

or equivalently,

$$\sqrt{\mu^2 + 4\alpha\left(\sum_j \frac{\sqrt{t_j}}{R_j}\right)\left(\sum_j \frac{1}{R_j\sqrt{t_j}}\right)} \leqslant \mu + 2\sqrt{\alpha}\left(\sum_j \frac{\sqrt{t_j}}{R_j}\right)\sqrt{\Delta q},$$

which is equivalent to

$$\mu^2 + 4\alpha\left(\sum_j \frac{\sqrt{t_j}}{R_j}\right)\left(\sum_j \frac{1}{R_j\sqrt{t_j}}\right) \leqslant \mu^2 + 4\sqrt{\alpha}\mu\left(\sum_j \frac{\sqrt{t_j}}{R_j}\right)\sqrt{\Delta q} + 4\alpha\Delta q\left(\sum_j \frac{\sqrt{t_j}}{R_j}\right)^2$$

or

$$\alpha\left(\sum_j \frac{1}{R_j\sqrt{t_j}}\right) \leqslant \mu\sqrt{\alpha\Delta q} + \alpha\Delta q\left(\sum_j \frac{\sqrt{t_j}}{R_j}\right).$$

A sufficient condition for the latter is

$$\sum_{j\in\mathcal{N}} \frac{1}{R_j\sqrt{t_{\min}}} \leqslant \mu\sqrt{\alpha\Delta q} + \alpha\Delta q \sum_{j\in\mathcal{N}} \frac{\sqrt{t_{\min}}}{R_j}. \tag{17}$$

Solving the quadratic equation (17) for $t_{\min}$, we see that this is implied by (10).

Finally the fact that we are not below the lower extreme of the linear region (i.e., $p_i > 0$ for all i) is a direct consequence of the fact that zero loss probability would imply infinite throughput (see (1)), which is impossible since the link capacity μ is finite.

A.2. Proof of Theorem 2

We first show that the utility function is concave in the case of only real-time sessions. Replacing t_i by $1/T_i$ in Eq. (6), we obtain

$$p_i = \frac{\sum_{j\in\mathcal{I}}\lambda_j - \mu}{\lambda_i + T_i\sum_{j\neq i}\lambda_j/T_j},$$

which is convex in T_i. The convexity of p_i in T_i follows from the fact that $\sum_{j\in\mathcal{I}}\lambda_j - \mu > 0$. Hence U_i are concave in T_i and continuous in T_j. The existence then follows from [21]. For TCP connections, we have

$$\frac{\partial^2 U_i}{\partial T_i^2} = a_i\left[\frac{\partial^2\lambda_i}{\partial T_i^2}(1-p_i) - 2\frac{\partial\lambda_i}{\partial T_i}\frac{\partial p_i}{\partial T_i} - \lambda_i\frac{\partial^2 p_i}{\partial T_i^2}\right] - \frac{\partial^2\hat{d}(T_i)}{\partial T_i^2}, \tag{18}$$

where $\hat{d}(T_i) = d(1/t_i)$. On the other hand, (1) implies

$$\frac{\partial p_i}{\partial T_i} = -\frac{2\alpha}{R_i^2}\frac{\frac{\partial\lambda_i}{\partial T_i}}{\lambda_i^3}$$

and

$$\frac{\partial^2 p_i}{\partial T_i^2} = -\frac{2\alpha}{R_i^2\lambda_i^4}\left[\frac{\partial^2\lambda_i}{\partial T_i^2}\lambda_i - 3\left(\frac{\partial\lambda_i}{\partial T_i}\right)^2\right]. \tag{19}$$

Then (18) becomes

$$\frac{\partial^2 U_i}{\partial T_i^2} = a_i\left[\frac{\partial^2\lambda_i}{\partial T_i^2}\left(1 + \frac{\alpha}{R_i^2\lambda_i^2}\right) - \left(\frac{\partial\lambda_i}{\partial T_i}\right)^2\frac{2\alpha}{R_i^2\lambda^3}\right] - \frac{\partial^2\hat{d}(T_i)}{\partial T_i^2}. \tag{20}$$

Since the function $\hat{d}$ is convex in T_i, then form (20), it is sufficient to show that the second derivative of λ_i with respect to T_i is nonpositive. We have

$$\lambda_i = \frac{1}{R_i}\sqrt{\frac{\alpha}{p_i}} = \frac{\frac{2\alpha}{R_i}\left(\sum_{j\in\mathcal{N}}\frac{\sqrt{t_j}}{R_j}\right)}{\sqrt{t_i}\left(-\mu+\sqrt{\mu^2+4\alpha\sum_{j\in\mathcal{N}}\left(\frac{1}{R_j\sqrt{t_j}}\right)\sum_{j\in\mathcal{N}}\left(\frac{\sqrt{t_j}}{R_j}\right)}\right)}$$

$$= \frac{\frac{2\alpha}{R_i}\left(\frac{\sqrt{t_i}}{R_i}+C_1\right)}{\sqrt{t_i}\left(-\mu+\sqrt{\mu^2+4\alpha\left(\frac{1}{R_i\sqrt{t_i}}+C_2\right)\left(\frac{\sqrt{t_i}}{R_i}+C_1\right)}\right)}$$

$$= \frac{\frac{2\alpha}{R_i}\sqrt{T_i}\left(\frac{1}{\sqrt{T_i}R_i}+C_1\right)}{\left(-\mu+\sqrt{\mu^2+4\alpha\left(\frac{\sqrt{T_i}}{R_i}+C_2\right)\left(\frac{1}{\sqrt{T_i}R_i}+C_1\right)}\right)}$$

$$= \frac{\frac{\sqrt{T_i}}{2R_i}\left(\mu+\sqrt{\mu^2+4\alpha\left(\frac{\sqrt{T_i}}{R_i}+C_2\right)\left(\frac{1}{\sqrt{T_i}R_i}+C_1\right)}\right)}{\left(\frac{\sqrt{T_i}}{R_i}+C_2\right)}$$

$$= \frac{1}{2R_i}\left[\frac{\sqrt{T_i}\mu}{\left(\frac{\sqrt{T_i}}{R_i}+C_2\right)} + \frac{\sqrt{T_i}\sqrt{\mu^2+4\alpha\left(\frac{\sqrt{T_i}}{R_i}+C_2\right)\left(\frac{1}{\sqrt{T_i}R_i}+C_1\right)}}{\left(\frac{\sqrt{T_i}}{R_i}+C_2\right)}\right]$$

$$= \frac{1}{2R_i}[F_1(T_i)+F_2(T_i)],$$

where $C_1 = \sum_{j\neq i}\frac{\sqrt{t_j}}{R_i}$ and $C_2 = \sum_{j\neq i}\frac{1}{\sqrt{t_j}R_i}$. Now, we must prove that the second derivative of the functions F_1 and F_2 are nonpositive for all $C_1 \geqslant 0$ and $C_2 \geqslant 0$. We begin by taking the second derivative of F_1. After some simplification, we obtain

$$\frac{\partial^2 F_1(T_i)}{\partial T_i^2} = -1/4\frac{\mu R_i^2 C_2(3\sqrt{T_i}+C_2R_i)}{(T_i^{3/2}(\sqrt{T_i}+C_2R_i)^3)},$$

which is positive. For the second function F_2, since the function F_2 is positive, it suffices to show that the second derivative of function $[F^2(T_i)]^2$ is nonpositive, we have

$$\begin{aligned}\frac{\partial^2[F_2(T_i)]^2}{\partial T_i^2} = &-\frac{R_i}{2T_i^{3/2}(\sqrt{T_i}+C_2R_i)^4}\\ &\times(6T_i\alpha C_2 + 8\sqrt{T_i}\alpha C_2^2R_i + 2\alpha T_i^2C_1\\ &+ 8T_i^{3/2}\alpha C_2R_iC_1 + 6\alpha T_iR_i^2C_1C_2^2\\ &+ 2\alpha R_i^2C_2^3 + 3T_i\mu^2R_i^2C_2),\end{aligned}$$

which is nonpositive.

A.3. Proof of Theorem 4

Under supermodular condition, to show the uniqueness of Nash equilibrium, it suffices to show that [22],

$$-\frac{\partial^2 U_i}{(\partial T_i)^2} \geqslant \sum_{j\neq i}\frac{\partial^2 U_i}{\partial T_i\partial T_j} \tag{21}$$

or equivalently,

$$\frac{\partial^2 p_i}{(\partial T_i)^2} + \sum_{j\neq i}\frac{\partial^2 p_i}{\partial T_i\partial T_j} \geqslant 0. \tag{22}$$

For the case of only real-time sessions, $p_i = \frac{\sum\lambda_j-\mu}{\lambda_i+T_i\sum_{j\neq i}\lambda_j\frac{1}{T_j}}$. We have

$$\frac{\partial p_i}{\partial T_i} = -\frac{(\lambda-\mu)\sum_{k\neq i}\frac{\lambda_k}{T_k}}{\left(\lambda_i+T_i\sum_{k\neq i}\frac{\lambda_k}{T_k}\right)^2},$$

$$\frac{\partial^2 p_i}{\partial T_i^2} = \frac{2(\lambda-\mu)\left(\sum_{k\neq i}\frac{\lambda_k}{T_k}\right)^2}{(\lambda_i+T_i\sum_{k\neq i}\frac{\lambda_k}{T_k})^3},$$

$$\frac{\partial^2 p_i}{\partial T_i\partial T_j} = (\lambda-\mu)\frac{\lambda_j}{T_j^2}\frac{\lambda_i-T_i\sum_{k\neq i}\frac{\lambda_k}{T_k}}{\left(\lambda_i+T_i\sum_{k\neq i}\frac{\lambda_k}{T_k}\right)^3}.$$

Therefore, in order to get the uniqueness, we need that

$$\begin{aligned}&\frac{\partial^2 p_i}{\partial T_i^2} + \sum_{j\neq i}\frac{\partial^2 p_i}{\partial T_i\partial T_j}\\ &\quad= \frac{2(\lambda-\mu)\left(\sum_{k\neq i}\frac{\lambda_k}{T_k}\right)^2}{\left(\lambda_i+T_i\sum_{k\neq i}\frac{\lambda_k}{T_k}\right)^3}\\ &\qquad+(\lambda-\mu)\frac{\lambda_i-T_i\sum_{k\neq i}\frac{\lambda_k}{T_k}}{\left(\lambda_i+T_i\sum_{k\neq i}\frac{\lambda_k}{T_k}\right)^3}\sum_{j\neq i}\frac{\lambda_j}{T_j^2}\\ &\quad= \frac{(\lambda-\mu)}{\left(\lambda_i+T_i\sum_{k\neq i}\frac{\lambda_k}{T_k}\right)^3}\left[\lambda_i\sum_{j\neq i}\frac{\lambda_j}{T_j^2} - T_i\left(\sum_{k\neq i}\frac{\lambda_k}{T_k}\right)\right.\\ &\qquad\left.\times\left(\sum_{k\neq i}\frac{\lambda_k}{T_k^2}\right)+2\left(\sum_{k\neq i}\frac{\lambda_k}{T_k}\right)^2\right] \geqslant 0.\end{aligned}$$

This leads to the sufficient condition:

$$\frac{\lambda_{\min}^2}{T_{\min}^2} - T_{\max}\frac{\lambda_{\max}^2(I-1)^2}{T_{\min}^3} + 2\frac{\lambda_{\min}^2}{T_{\max}^2} \geqslant 0,$$
$$2\frac{\lambda_{\min}^2}{T_{\max}^2} > T_{\max}\frac{\lambda_{\max}^2}{T_{\min}^3},$$
$$2T_{\min}^3\lambda_{\min}^2 > T_{\max}^3\lambda_{\max}^2.$$

A.4. Proof of Theorem 5

For supermodularity on TCP connections, we consider the sufficient condition $\frac{\partial^2 U_i}{\partial t_i \partial t_j} \geqslant 0$. It follows that

$$U_i = \frac{a_i}{R_i}\sqrt{\frac{\alpha}{p_i}}(1-p_i) - d(t_i)$$
$$= \frac{a_i}{R_i}\sqrt{\alpha}(p_i^{-1/2} - p_i^{1/2}) - d(t_i).$$

Then, for $j \neq i$,

$$\frac{\partial U_i}{\partial t_j} = \frac{a_i\sqrt{\alpha}}{R_i}\left(\frac{-p_i^{-3/2}}{2}\frac{\partial p_i}{\partial t_j} - \frac{p_i^{-1/2}}{2}\frac{\partial p_i}{\partial t_j}\right),$$
$$\frac{\partial^2 U_i}{\partial t_i \partial t_j} = \frac{a_i\sqrt{\alpha}}{R_i}\left[\left(\frac{3p_i^{-5/2}}{4} + \frac{p_i^{-3/2}}{4}\right)\frac{\partial p_i}{\partial t_i}\frac{\partial p_i}{\partial t_j} - \left(\frac{p_i^{-3/2}}{2} + \frac{p_i^{-1/2}}{2}\right)\frac{\partial^2 p_i}{\partial t_i \partial t_j}\right].$$

Thus a sufficient condition for supermodularity $\left(\frac{\partial^2 U_i}{\partial t_i \partial t_j} \geqslant 0, \quad \forall i,j,j \neq i\right)$ is

$$(3+p_i)\frac{\partial p_i}{\partial t_i}\frac{\partial p_i}{\partial t_j} \geqslant 2p_i(p_i+1)\frac{\partial^2 p_i}{\partial t_i \partial t_j}, \quad \forall i,j,j \neq i.$$

References

[1] E. Altman, D. Barman, R. El Azouzi, D. Ros, B. Tuffin, Pricing differentiated services: a game-theoretic approach, Proceedings of NETWORKING 2004, Lecture Notes in Computer Science, vol. 3042, Springer, Athens, Greece, 2004, pp. 430–441.

[2] P. Pieda, J. Ethridge, M. Baines, F. Shallwani, A Network Simulator Differentiated Services Implementation, Technical report, Available from: <http://www.isi.edu/nsnam/ns>, July 2000.

[3] A.G.D. Dutta, J. Heidemann, Oblivious AQM and Nash equilibria, in: IEEE Infocom, 2003.

[4] J. Janssen, D. De Vleeschauwer, M. Büchli, G.H. Petit, Assessing voice quality in packet-based telephony, IEEE Internet Computing (2002) 48–56.

[5] M. Mandjes, Pricing strategies under heterogeneous service requirements, Computer Networks 42 (2003) 231–249.

[6] T. Alpcan, T. Basar, A game-theoretic framework for congestion control in a general topology networks, in: 41st IEEE Conference on Decision and Control, Las Vegas, Nevada, 2002, pp. 10–13.

[7] E. Altman, T. Boulogne, R. El-Azouzi, T. Jiménez, L. Wynter, A survey on networking games in telecommunications, Computers and Operations Research 33 (2) (2006) 286–311.

[8] Y. Jin, G. Kesidis, Nash equilibria of a generic networking game with applications to circuit-switched networks, in: IEEE INFOCOM, 2003.

[9] T. Basar, R. Srikant, A Stackelberg network game with a large number of followers, Journal of Optimization Theory and Applications 115 (3) (2002) 479–490.

[10] Y.A. Korilis, A.A. Lazar, A. Orda, Achieving network optima using Stackelberg routing strategies, IEEE ACM Transactions on Networking 5 (1) (1997) 161–173.

[11] S. Floyd, V. Jacobson, Random early detection gateways for congestion avoidance, IEEE/ACM Transactions on Networking 1 (4) (1993) 397–413.

[12] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, An architecture for differentiated services, Internet Standards Track RFC 2475, IETF, December 1998.

[13] J. Heinanen, F. Baker, W. Weiss, J. Wroclawski, Assured forwarding PHB group, Internet Standards Track RFC 2597, IETF, June 1999.

[14] S. Shakkottai, R. Srikant, How good are deterministic fluid models of Internet congestion control, in: Proceedings of IEEE INFOCOM, New York, 2002.

[15] Numerical Recipes in C, The Art of Scientific Computing, secpnd ed., Section 5.6, Tech. report, Available from: <http://www.ulib.org/webRoot/Books/Numerical_Recipes/bookc.html1>.

[16] B. Tuffin, Charging the Internet without bandwidth reservation: an overview and bibliography of mathematical approaches, Journal of Information Science and Engineering 19 (5) (2003) 765–786.

[17] D. Topkis, Equilibrium points in nonzero-sum *n*-person submodular games, SIAM Journal of Control and Optimization 17 (1979) 773–787.

[18] D.D. Yao, S-modular games with queueing applications, Queueing Systems 21 (1995) 449–475.

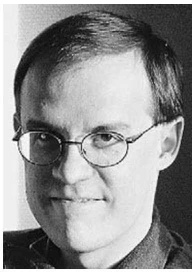
[19] C. Dovrolis, P. Ramanathan, D. Moore, Packet-dispersion techniques and a capacity-estimation methodology, IEEE/ACM Transactions on Networking 12 (6) (2004) 963–977.

[20] A.B. Downely, Using pathchar to estimate Internet link characteristics, in: Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, Cambridge, MA, United states, 1999.
[21] J.B. Rosen, Existence and uniqueness of equilibrium points for concave n-person games, Econometrica 33 (1965) 153–163.
[22] F. Bernstein, A. Federgruen, A general equilibrium model for decentralized supply chains with price- and service-competition, Technical report, Available from: <http://faculty.fuqua.duke.edu/ fernando/bio/>.

E. Altman received the B.Sc. degree in Electrical Engineering (1984), the B.A. degree in Physics (1984) and the Ph.D. degree in Electrical Engineering (1990), all from the Technion-Israel Institute, Haifa. In (1990) he further received his B.Mus. degree in Music Composition in Tel-Aviv University. Since 1990, he has been with INRIA (National Research Institute in Informatics and Control) in Sophia-Antipolis, France. His current research interests include performance evaluation and control of telecommunication networks and in particular congestion control, wireless communications and networking games. He is in the editorial board of several scientific journals: Stochastic Models, JEDC, COMNET, SIAM SICON and WINET. He has been the (co)chairman of the program committee of several international conferences and workshops (on game theory, networking games and mobile networks). More information can be found at http://www.inria.fr/mistral/personnel/Eitan.Altman/me.html.

Dhiman Barman received his B.Tech degree in Computer Science and Engineering at Indian Institute of Technology, Bombay, India in 1996, and MA in Computer Science at Boston University in 2003. He is a Ph.D. student in Computer Science at Boston University. His research interests include protocol design and performance evaluation, game-theory and Internet measurement.

Rachid El Azouzi received the Ph.D. degree in Applied Mathematics from the Mohammed V University, Rabat, Morocco (2000). He joined INRIA (National Research Institute in Informatics and Control) Sophia Antipolis for post-doctoral and Research Engineer positions. Since 2003, he has been a researcher at the University of Avignon, France. His research interests are mobile networks, performance evaluation, TCP protocol, error control in wireless networks, resource allocation, networking games and pricing.



David Ros received his B.Sc. (with honors) and M.Sc. degrees, both in Electronics Engineering, from the Simón Bolívar University, Caracas, Venezuela in 1987 and 1994, respectively, and his Ph.D. in Computer Science from the Institut National de Sciences Appliquées (INSA), Rennes, France in 2000. He is currently working as an Associate Professor (Maître de Conférences) at GET/ENST Bretagne, Rennes, in the Networking and Multimedia Department. His active research interests include transport protocols, network modelling and simulation, as well as pricing and quality of service issues in IP networks.

Bruno Tuffin (IRISA/INRIA) received his Ph.D. degree in applied mathematics from the University of Rennes 1 in 1997. Since, he has been with INRIA-Rennes, France. His research interests include developing Monte Carlo and quasi-Monte Carlo simulation techniques for the performance evaluation of computer and telecommunication systems, and more recently developing Internet active measurement techniques and new pricing schemes. On this last topic, he is the coordinator of the INRIA's cooperative research action PRIXNET (see http://www.irisa.fr/armor/Armor-Ext/RA/prixnet/ARC.htm).