Quality of Service Provision in Noncooperative Network Environments

Kihong Park
Dept. of Computer Sciences
Purdue University
West Lafayette, IN 47907
park@cs.purdue.edu

Meera Sitharam
Dept. of Math. and Computer Science
Kent State University
Kent, OH 44242
sitharam@mcs.kent.edu

Shaogang Chen
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907
shaogang@ecn.purdue.edu

CSD-TR 97-039
Dept. of Computer Sciences
Purdue University

Abstract

This paper presents a study of the quality of service provision problem in noncooperative network environments where applications or users are assumed to behave in a selfish way. Our contributions are threefold.

First, we formulate a model of QoS provision in noncooperative network environments and present a comprehensive game-theoretic analysis of its properties. We give a complete characterization of Nash equilibria and show under what conditions they are system optimal. We show that, in general, Nash equilibria need not be system optimal nor Pareto, and gaps exist to separate the three classes in nontrivial ways. However, for "resource-plentiful" systems, we show that Nash equilibria, Pareto optima, and system optima all coincide, collapsing into a single class.

Second, we investigate the problem of facilitating effective quality of service provision in systems with multi-dimensional QoS vectors containing both mean- and variance-related QoS measures. We extend the game-theoretic analysis to multi-dimensional QoS vector games and show under what conditions the aforementioned results transfer over. We also investigate the effect of multiple QoS measures on the properties of the induced QoS levels rendered by service classes. We show that under bursty traffic conditions, it is impossible for a service class to deliver superior QoS in both mean delay and jitter if weighted fair queueing or other GPS-related packet scheduling discipline is used.

Third, whereas parts one and two dealt with questions concerning properties of system states as determined by the interaction of selfish applications or users, the third part investigates what the system can do to enhance network QoS provision performance. We present an adaptive service weight control algorithm for GPS-based switches and show its effectiveness at equilibriating imbalances between service class utilizations. We show that total utility can be improved without adversely affecting individual applications—a form of Pareto optimization.

Keywords: *Multiclass QoS Provision, Noncooperative Network Game, Distributed Network Algorithms, Integrated Control of Networks.*

1 Introduction

1.1 Background

With the increased deployment of high-speed local- and wide-area networks carrying a multitude of information from e-mail to bulk data to voice and video, provisioning adequate quality of service (QoS) to the diverse application base has become an important problem [2, 10, 24, 29]. This paper describes a QoS provision architecture suited for best-effort environments, based on ideas from microeconomics and noncooperative game theory. We construct a noncooperative QoS provision model where users are assumed to be selfish, and packets are routed over switches where, as a function of their enscribed priority, differentiated service is delivered. Quality of service is an induced phenomenon, achieved as a result of interaction between selfish applications.

The traditional approach to QoS provision uses resource reservations along a route to be followed by a traffic stream so that the stream's burstiness can be suitably accommodated. Although research abounds [5, 6, 9, 10, 15, 25, 29, 30, 31, 7], analytic tools for computing QoS guarantees rely on shaping input traffic, to preserve well-behavedness across switches which implement some form of packet scheduling discipline [8, 30] such as GPS[1] However, real-time constraints of multi-media traffic and the scale-invariant burstiness associated with self-similar network traffic [26, 36, 46, 32] limit the shapability of input traffic while at the same time reserving bandwidth that is significantly smaller than the peak transmission rate. Thus QoS and utilization stand in a trade-off relationship with each other [33, 32] and transporting application traffic over reserved channels, in general, incurs a high cost. This makes it important to organize today's best-effort bandwidth into stratified services with graded QoS properties such that the QoS requirements of a a compendium of applications can be effectively met; particularly applications with diverse but—to varying degrees—tolerant QoS requirements. This, additionally, helps amortize the cost of inefficiencies stemming from overprovisioned resources for guaranteed traffic through the filling-in effect [22].

Recently, microeconomic approaches to resource allocation have received significant interest with applications spanning a number of diverse contexts [12, 40, 19, 18, 21, 23, 13, 43, 44, 27, 4, 39, 16, 35]. The overall goal of this area is to formulate a resource allocation problem in the framework of microeconomics and game theory, and show that under certain conditions, the system achieves "desirable" allocations both from a stabilibity and optimality point-of-view. The latter are important in making stratified best-effort bandwidth usable by multi-media applications. This is because a neccessary system property in this case is a predictable level of service both in terms of stability as well as in meeting the QoS requirements of the application. The models and approaches differ along several dimensions, some of the important ones being whether applications or users are assumed to be cooperative or selfish, whether pricing is used or not, and how much computing responsibility is delegated to the user. Several papers have addressed the issue of multi-class QoS provision in high-speed networks [4, 19, 28, 40, 39, 35]. Some of the works employ a cooperative framework or place significant computing responsibilities on the part of the user [28, 39], some investigate the effect of pricing incentives [4], and others represent flow or congestion control models that only partially address

[1] Also known as weighted fair queueing.

the quality of service problem problem [19, 40].

The present approach differs from previous works in two significant ways—first, it is one of the most comprehensive noncooperative resource allocation models specifically formulated to model QoS provision, and second, we address both theoretical and practical aspects, with emphasis on bridging the gap from theory to realizability. Not only do we study the noncooperative QoS provision model from a game-theoretic perspective showing the existence of a rich structure relating Nash equilibria to Pareto and system optima, we also investigate the problem of facilitating adequate services when the model is extended to incorporate multi-dimensional QoS vectors with possibly conflicting requirements, the effect of burstiness on rendered QoS, and the adaptive relocation of network resources to hot spots using programmable GPS switches. The specific contributions are described next.

1.2 Summary of Results

Our contributions are threefold:

Game-theoretic From a game-theoretic perspective, we formulate a model of multiclass QoS provision in noncooperative network environments and analyze the structure of the system with respect to its equilibria and optima. Before we state the results, three notions are of import for their understanding (defined formally in Section 2.3): Nash equilibrium, Pareto optimum, and system optimum. Roughly speaking, a configuration is a *Nash equilibrium* if each player cannot improve its individual lot through unilateral actions affecting its traffic allocations. Thus if every player finds herself in such a "local optimum," then from the noncooperative perspective, the system is at an impasse—i.e., stable rest point. A configuration is a Pareto optimum if in order to improve the lot of any player(s), the lot of others must be sacrificed. A configuration is system optimal if the sum of the individual lots is maximized.

First, we give a complete characterization of Nash equilibria and show under what conditions they are system optimal (Theorems 3.4 and 3.6). The latter is shown to be related to the Pareto optimality of a certain normal form (Lemma 3.5) derived from Nash equilibria which points to an interesting relationship between Nash equilibria, Pareto optima, and system optima for the QoS provision game. Figure 1.1 depicts a typical picture of the structure of the network QoS provision game, annotated by the roles of the two main results.

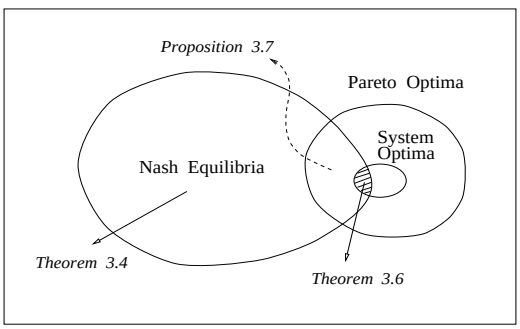

Figure 1.1: Structure of network QoS provision game.

Second, we show that there are Nash equilibria that are Pareto but not system optimal (Proposi-

tion 3.7)—this shows that Theorem 3.6 is "tight." We also show that there exist Nash equilibria that are not Pareto optimal and vice versa, and for some games in this framework, no Nash equilibria exist at all (Proposition 3.8).

Third, we show that for certain "resource-plentiful" systems, Nash equilibria, Pareto optima, and system optimal all conincide collapsing into a single class (Theorem 3.13, part (a)). This item is interesting from the perspective that it gives a sufficient condition under which Nash equilibria are guaranteed to be also desirable in the optimality sense. In part (b) of the same theorem, we show that a certain self-optimization procedure leads to quick, robust convergence to globally optimal Nash equilibria.

Multi-dimensional QoS Vectors We investigate the problem of effectively facilitating quality of service provision in systems with multi-dimensional QoS vectors containing both mean- and variance-related measures (e.g., bounds on packet loss rate, delay, and jitter).

First, we extend the game-theoretic analysis to multi-dimensional QoS vector games and show that the main results carry over if a uniformity assumption is placed either on application preference or on QoS vector functions (Propositions 4.4 and 4.5).

Second, from a performance perspective, we investigate the impact of multiple QoS measures—sometimes with contradictory requirements imposed by user requirements—on the characteristics of QoS levels rendered by the service classes. In particular, we show the surpising result that under bursty traffic conditions, it is impossible for a service class to deliver superior QoS in both mean- and variance-related QoS measures (e.g., mean delay and jitter) vis-à-vis some other service class, if weighted fair queueing or other GPS-related packet scheduling discipline is used. Under weakly bursty traffic conditions, a total ordering among service classes is possible, however, via the degenerate situation where the "superior" service classes attain zero packet loss. For mean-related QoS measures such as mean packet loss rate and delay, we show that total orderings among service classes are feasibly achievable.

Structural Adaptation Whereas the results from the previous two parts dealt with questions concerning properties of system states as determined by the interaction of selfish applications or users, the third part investigates the question of what the system—if anything—can do to enhance network QoS provision performance. We show that in systems where resources are taxed, instabilities can arise which can adversely affect the predictability property required of service classes. If utilization across service classes is uniformly high, then the problem is intrinsic and nothing can be done by the system short of shedding load through admission control or the induced departure of applications through price control.

However, we show that this problem can arise even in situations where there exist imbalances between utilizations across service classes which opens the way for the system to positively affect QoS provision performance. We formulate an adaptive service weight control algorithm in the context of systems implementing weighted fair queueing at routers where imbalances between service class utilizations are reduced iteratively. We show that the control algorithm is effective at reducing imbalances and improving total system performance by shifting idle resources from applications that don't need

them to those that do—a form of Pareto optimization. The algorithm possesses an interesting twist due to the fact that burstiness causes variance-related QoS measures to be affected in the opposite direction from mean-related QoS measures (Section 4.2.2).

1.3 Related Work

Microeconomic Approaches to Resource Allocation In recent years, there has been a surge of work in "microeconomic approaches to resource allocation" where ideas and tools from microeconomics and game theory have been applied in the formulation and solution of problems arising in flow control, routing, file allocation, load balancing, multi-commodity flow, and quality of service provision, among others [12, 40, 19, 18, 21, 23, 13, 43, 44, 27, 4, 39, 35]. A collection of papers covering a broad range of topics can be found in [3]. A brief survey of some of the literature is provided in [11]. Some standard references to game theory and microeconomics include [14, 38, 41, 42].

Many of the earlier papers including some recent ones [13, 12, 23, 28, 39] have espounded a cooperative game theory framework to model user interactions and derive results based on Pareto optimality. Although fruitful to investigate due to the powerful tools available in cooperative game theory, a potential drawback of this line of approach is the assumption that users or applications behave *cooperatively* in networking contexts. For the long-term establishment of virtual circuits or the leasing of telephone lines, the cooperative user model may indeed be viable[2]. However, for best-effort applications that comprise much of today's Internet traffic, users are largely anonymous with respect to thousands of other users who concurrently share network resources at any given time, and a noncooperative framework where each user is assumed to optimize individual performance based on his or her limited available information about network state seems better suited.

The *noncooperative* framework can be traced as far back as '81 to a paper by Yemini [47] who has since been more strongly associated with the cooperative approach where his work has played a seminal role. The noncooperative network resource allocation approach has been actively pursued by Lazar and his co-workers beginning in the late '80s [17, 1] with more recent work carried out jointly with Korilis and Orda [18, 19, 20, 21]. Their main work has revolved around an optimal flow control problem, and the development of techniques needed to show the existence of Nash equilibria using the notion of best reply correspondence [19]. Korilis et al. [20, 21] have also looked at the problem of using interventions by an impartial external entity—the network manager—to steer the system toward Nash equilibria that are also system optimal using the framework of Stackelberg games.

Another significant thrust in noncooperative network games is due to recent work by Shenker [40] where it is shown how choosing a packet scheduling discipline can influence the nature of the Nash equilibria attained. In the context of a congestion or flow control model, it is shown that for a large class of packet scheduling disciplines, a configuration being Nash need not imply that it is Pareto optimal. A GPS-related service discipline called Fair Share is defined and it is shown to lead to Nash equilibria with desirable properties including uniqueness and reachability by a class of self-optimization procedures.

[2] It is also possible that intermediaries perform long-term leasing of network resources which are then packaged and made available as high-level services to the user. Aspects of such activities may be modeled as coalition behavior.

On the implementation side, the work of Waldspurger et al. [43] deserves attention since it is one of the few works that have built a nontrivial working system—CPU allocation and load balancing in workstation networks—and demonstrated that a system based on microeconomic principles can indeed work in practice. Other implementations worth noting include Wellman's work on multicommodity flow problems [44, 45].

QoS-Related Network Games Several papers have addressed the more specific issue of multi-class QoS provision in high-speed networks using microeconomic models [4, 28, 39, 16]. In [28, 39], utility functions are defined with link bandwidth and switch buffers acting as substitutable resources, and Pareto-optimal allocation of resources among service classes is affected either by the network exercising admission control [39] or by users performing purchasing decisions [28]. In both approaches, it is assumed that QoS guarantees are computable, given specific resource reservations.

As stated earlier, an important goal of our approach is to use a noncooperative framework and shield the user from having to make complex computations to estimate service quality. The models and approaches are thus clearly dissimilar.

In [4], a general framework for investigating pricing in networks is proposed, with service discipline and pricing policy acting as design variables. Simulation results are shown that depict the existence of "desirable" price ranges related to system optimality. The simulations were carried out using a 2-service class packet scheduling algorithm where a shared FIFO queue was partitioned into two segments with high priority packets being queued at the front and low priority packets being queued at the back. Four types of applications with different QoS requirements were tested with priority settings either set to 1 or 2.

Our model is an n-application, m-service class, s-dimensional QoS vector quality service provision model, and emphasizes a different set of questions from [4] where the effect of pricing incentives are paramount. We apply noncooperative game-theoretic analysis to the *multi-dimensional* QoS vector model to understand under what conditions Nash equilibria exist and are system or Pareto optimal. We also investigate the problem of facilitating service classes with induced QoS levels that match application requirements when multiple QoS indicators are present.

Comparison with Models by Korilis et al. and Shenker The flow or congestion control models of Korilis et al. [19] and Shenker [40] represent a form of quality of service provision and it is important to explicate the differences between our model and theirs, given that all three follow the noncooperative framework. The main difference between the models by Korilis et al. and Shenker, and Park et al. is that, indeed, theirs *is* a flow or congestion control model. Phrased in the language of the QoS provision model defined in Section 2.3, both [19] and [40] correspond to the situation where $n = m$, each player i is permanently assigned to service class i, and either $\lambda_{ii} \geq 0$ [40] or $0 \leq \lambda_{ii} \leq \lambda_i$ [19], but in both cases, $\lambda_{ij} = 0$ for $i \neq j$. That is, a player, being tied to a fixed service class, has the option of controlling how much traffic [40]—or using what time schedule [19]—to send his traffic but not *where*. Since delay or any other performance measure will deteriorate with increased volume but volume itself, keeping other things fixed, will generally increase utility, there is an optimum volume assignment—i.e., optimal flow or congestion control—that maximizes player i's utility.

In our model, there is no 1-1 correspondence of players to service classes. Indeed, the very *crux* of the QoS provision problem is to give each player $i \in [1, n]$ the freedom to choose *where* she wants to send her traffic volume, from service class 1 all the way to service class m. Hence, our QoS provision model is fundamentally different from their flow control models, being more complex and producing equilibria structures that are different from [19], [40]. Secondly, our model incorporates multi-dimensional QoS vectors whose consequences are then analyzed in both game-theoretic and network performance terms.

The rest of the paper is organized as follows. In Section 2, we describe the overall set-up and formulate the network QoS provision model. This is followed by Section 3 which gives a game-theoretic analysis of the QoS provision game structure. Section 4.1 extends the game-theoretic analysis to multi-dimensional QoS vectors and Section 4.2 studies the effect of burstiness on the characteristics of rendered QoS. Lastly, Section 5 investigates the problem of structural adaptation using a service weight control algorithm. We conclude with a discussion of our results and future work.

2 Noncooperative Network Game

2.1 Network Model

The network model is depicted in Figure 2.1. A switch or router is shared by two traffic classes—*reserved* and *nonreserved* (or best-effort)—where the former constitutes background or cross traffic and the latter is the aggregate application traffic. That is, $\lambda^{NR} = \sum_{i=1}^{n} \lambda_i$ where $\lambda_1, \lambda_2, \ldots, \lambda_n$ are the mean arrival rate of n application traffic sources. The service rate of the system is given by μ and we will assume that the switch implements a form of weighted fair queueing (WFQ) with service weights $\alpha_1, \alpha_2, \ldots, \alpha_m$ where $\alpha_j \geq 0$, $j \in [1, m]$, and $\sum_{j=1}^{m} \alpha_j = 1$. Here, m denotes the number of service classes. The total service rate μ is split between the two traffic classes $\mu = \mu^R + \mu^{NR}$. Service class j of the nonreserved traffic class thus receives a service rate of $\alpha_j \mu^{NR}$.

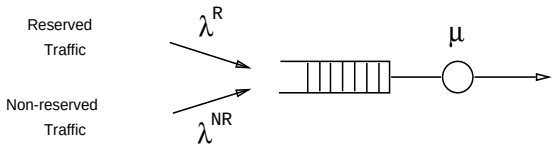

Figure 2.1: Dual traffic classification at output-buffered switch with shared priority queue implementing weighted fair queueing.

In keeping with the ATM framework, we assume fixed-size packets (i.e., *cells*) and we employ output-buffered switches. We implement a generic form of weighted fair queueing achieving perfect isolatedness and conservation of work. We ignore effient implementation considerations of WFQ, treating the processing cost at switches as fixed. The assumption of fixed-size packets simplifies the faithful rendering of service rates commensurate with the weights $\alpha_1, \ldots, \alpha_m$.

2.2 Application Model

Utility Function Given a generic network model where packets are tagged by priority labels receiving differentiated service at switches, we need a framework and control mechanism which is able to exploit this feature to provide service to applications with diverse QoS needs such that the collective good of the whole system is maximized. A *utility function* is a map $U : \mathbb{R}^s \to \mathbb{R}_+$, $s \geq 1$, from QoS vectors to the nonnegative reals indicating the level of satisfaction or utility a certain quality of service brings to an application or user. It is a purely theoretical tool to reason about application behavior assuming certain qualitative shapes about its preferences. Figure 2.2 shows two candidate utility functions, on the left, for "nonurgent" e-mail, and on the right, for a real-time video application. The packet loss rates have been exaggerated for illustrative purposes.

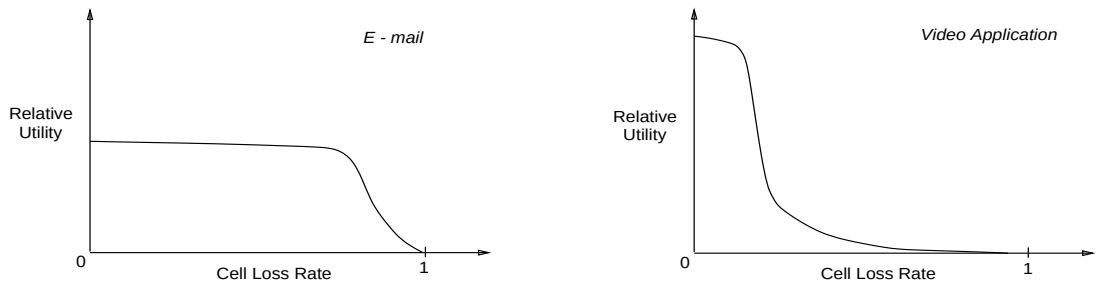

Figure 2.2: Utility functions. E-mail application (left) and video application (right).

The shapes of the utility functions indicate that non-urgent e-mail is much more tolerant to high packet loss, and unless the loss rate is "exceedingly" high, the e-mail application is almost equally satisfied whether the loss rate is 0 or somewhat higher. The video application, on the other hand, can only tolerate much smaller loss rates, and its utility is concentrated toward 0.

Selfishness Selfishness, in our context, will mean that each application $i \in [1, n]$ will try to take actions so as to maximize its individual utility U_i. The forms for U_i as well as user i's decision variables for the QoS provision problem are defined in the next section.

2.3 Definition of Network QoS Provision Game

QoS Provision Problem Assume we are given m service classes and n applications or players represented by their mean arrival rates $\lambda_1, \ldots, \lambda_n$ and utility functions $U_1, \ldots, U_n$. We arrive at a resource allocation problem in the following way. Let $\lambda_{ij} \geq 0$, $i \in [1, n]$, $j \in [1, m]$, denote the traffic volume of the i'th application assigned to service class j. Thus, $\lambda_i = \sum_{j=1}^{m} \lambda_{ij}$. That is, application i is given the freedom to choose which service classes to assign her traffic to. We also consider the special case when traffic assignments are restricted to be "all in one bag," i.e., $\lambda_{ij} \in \{\lambda_i, 0\}$, for all $j \in [1, m]$.

Let $\Lambda = (\lambda_{ij} : \ i, j)$ denote the resource assignment matrix, and let $c_1, c_2, \ldots, c_m$ be the packet loss rates of the m service classes. Each packet loss rate is a function of Λ,

$$c_j = c_j(\Lambda), \qquad j \in [1, m].$$

Assuming isolatedness[3], we have $c_j = c_j(q_j)$ where $q_j = \sum_{i=1}^{n} \lambda_{ij}$ is the total traffic volume assigned to class j. We will also make the assumption that c_j is monotone in q_j, i.e., $dc_j/dq_j \geq 0$, a property satisfied by virtually all service disciplines of interest. Isolatedness and monotonicity will be the only two properties needed of a packet scheduling discipline. We will also make the assumption that $dU_i/dc \leq 0$. That is, making the packet loss rate smaller[4] can never decrease the utility experience by player i.

The model can be extended to the case when application QoS requirements are represented by multi-dimensional QoS vectors $\mathbf{x} \in \mathbb{R}^s$, $s \geq 1$. For example, in addition to packet loss rate, $\mathbf{x}$ may specify delay requirements as well as restrictions on their fluctuations such as jitter. It turns out that the analysis of the multi-dimensional case reduces to the scalar case under certain conditions, and we will proceed with packet loss rate c as the sole QoS indicator.

The *weighted utility* of application i, given assignment Λ, is defined as

$$\bar{U}_i(\Lambda) = \sum_{j=1}^{m} \lambda_{ij} U_i(c_j).$$

Subject to the above constraints, the static optimization problem can be formulated as

$$\max_{\Lambda} \ \bar{U}(\Lambda) \ = \ \sum_{i=1}^{n} \ \bar{U}_i(\Lambda). \tag{2.1}$$

This is a nonlinear programming problem with equality constraints.

Nash Equilibria, Pareto Optima, and System Optima Any Λ^* that satisfies (2.1) is called *system optimal*. Thus system optimality corresponds to optimizing the usual resource allocation objective function. An assignment Λ^* is *Pareto optimal* if for all Λ,

$$\forall i: \ \bar{U}_i(\Lambda^*) \leq \bar{U}_i(\Lambda) \ \ \Longrightarrow \ \ \forall i: \ \bar{U}_i(\Lambda^*) = \bar{U}_i(\Lambda).$$

That is, Pareto optimality states that total utility $\bar{U}$ can only be improved at the expense of one or more individual utility $\bar{U}_i$. In general, Pareto optimality does not imply system optimality. But, clearly, Λ is system optimal implies Λ is Pareto optimal.

The formulation of Nash equilibrium needs a further definition. Given Λ, let $\Lambda_i = (\lambda_{i1}, \lambda_{i2}, \ldots, \lambda_{im})$ denote the i'th player's assignment vector. Λ_i is also called the *strategy* of player i. Let

$$\mathcal{L}_i(\Lambda) = \{\, \Lambda' : \Lambda'_k = \Lambda_k, k \neq i, \text{ and } \|\Lambda'_i\|_1 = \lambda_i \,\}$$

where $\|x\|_1 = \sum_{j=1}^{m} |x_j|$. That is, $\mathcal{L}_i(\Lambda)$ is the set of all *unilateral* strategies for player i.

An assignment Λ^* is a *Nash equilibrium* if $\forall i \in [1, n]$, $\forall \Lambda \in \mathcal{L}_i(\Lambda^*)$,

$$\bar{U}_i(\Lambda) \leq \bar{U}_i(\Lambda^*).$$

[3] Also called insularity or firewall property.

[4] An analogous assumption is made in the multi-dimensional QoS vector case.

That is, in a Nash equilibrium, player i cannot improve its individual utility $\bar{U}_i$ by unilaterally changing its strategy.

In general, a system optimal assignment need not be a Nash equilibrium and little can be said about the relation between system optimality, Pareto optimality, and Nash equilibria. In the context of the noncooperative network environment where every player acts selfishly, we are interested in characterizing assignments that are Nash since they represent stable fixed points of the system—i.e., equilibria. From a resource allocation perspective, we would also like to know under what conditions Nash equilibria are system optimal.

Simplifying Assumption To make the analysis tractable, we will work with (unit) *step utility functions* where for each player $i \in [1, n]$,

$$U_i(c) = \begin{cases} 1, & \text{if } c \leq \theta_i, \\ 0, & \text{otherwise,} \end{cases}$$

where $\theta_i \geq 0$ is a threshold that represents the i'th application's preference. Since $c_j = c_j(q_j)$, $j \in [1, m]$, there exist $b_{ij} \geq 0$ such that

$$U_i(c_j(q_j)) = \begin{cases} 1, & \text{if } q_j \leq b_{ij}, \\ 0, & \text{otherwise.} \end{cases}$$

With a slight abuse of notation, we will sometimes write $U_i(q_j)$ for the composite function when the distinction is clear from the context.

The simplification is reasonable from two perspectives. One, from the technical side, we do not lose very much by sacrificing continuity of the utility functions, since it turns out that our game produces total utility functions which are not necessarily *quasi-concave* in player i'th decision variable even though the compactness and convexity conditions for the i'th player's strategy set are satisfied. That is, even though we start out with quasi-concave utility functions[5], the individual utility function $\bar{U}_i$ need not be so. Hence, Rosen's theorem [37], which is a standard tool for showing the existence of Nash equilibria, would not be usable *even* if the utility funtions were continuous in the framework of our QoS provision game.

We should qualify that the above statements are referred to in the context of a simplified game where $m = 1$ and $0 \leq \lambda_{i1} \leq \lambda_i$. Notice that, then,

$$\bar{U}_i(\Lambda) = \lambda_{i1} U_i(c_1(q_1)).$$

Since U_i, c_1 are both monotone (with different signs), it follows that U_i is monotone in q_1 with $dU_i/dq_1 \leq 0$. It is not difficult to show that $\bar{U}_i(\Lambda)$ can be made to be bimodal, hence *not* quasi-concave, if U_i—still monotone and continuous—is chosen to have a "terrace-like" shape. Indeed, going back to our QoS provision game, we can show that for some instances of the game, no Nash equilibria whatsoever exist (Proposition 3.8).

Two, we find that threshold functions, in many instances, adequately model the way best-effort applications might interact with the network in making their QoS requirements known—i.e., through

[5]Note that monotonicity trivially implies quasi-concavity.

the specification of bounds on mean packet loss rate, delay, and their variances (i.e., jitter). The analysis of the simplified QoS provision game remains highly nontrivial, and as we shall see, the game retains a rich structure.

3 Properties of Noncooperative QoS Provision Game

3.1 Equilibria and Their Properties

This section will investigate the structure of the QoS provision game leading to the relationship depicted in Figure 1.1. Let us impose a total order on the n players given by

$$i \leq i' \Longleftrightarrow \theta_i \leq \theta_{i'}.$$

Unless otherwise stated, we will assume such a fixed order in the following development. First, a simple fact on the induced ordering of the traffic volume thresholds b_{ij}.

Proposition 3.1 $\forall i \in [1, n-1]$, $\forall j \in [1, m]$, $b_{ij} \leq b_{i+1j}$.

Proof. Since $\theta_i \leq \theta_{i+1}$, $i \in [1, n-1]$, by monotonicity of c_j, $j \in [1, m]$,

$$c_j^{-1}(\theta_i) \leq c_j^{-1}(\theta_{i+1}).$$

Noting that $b_{ij} = c_j^{-1}(\theta_i)$ completes the proof. ■

Let $I_i^+ = \{\, j \in [1, m] : q_j > b_{ij},\, \lambda_{ij} > 0 \,\}$, $I_i^- = \{\, j \in [1, m] : q_j < b_{ij} \,\}$, and $I_i^0 = \{\, j \in [1, m] : q_j = b_{ij} \,\}$. Given player i, I_i^+ denotes the set of relevant service class indices where the traffic volume assigned exceeds player i's threshold, thus yielding 0 individual utility. Conversely for I_i^- and I_i^0. Let $q_j^i = \sum_{k \neq i} \lambda_{kj}$. Hence $q_j = \lambda_{ij} + q_j^i$. That is, q_j^i is the traffic volume assigned to service class j not counting player i's contribution (if any).

Let $J_i^+ = \{\, j \in [1, m] : q_j^i \geq b_{ij} \,\}$ and $J_i^- = \{\, j \in [1, m] : q_j^i < b_{ij} \,\}$. Hence J_i^+ is the set of service class indices where, irrespective of player i's actions, player i cannot garner any utility. Let $J_i^* = \{\, j \in [1, m] : b_{ij} - q_j^i = \min_{k \in J_i^-} b_{ik} - q_k^i \,\}$. J_i^* takes on a role similar to J_i^+ when the latter is empty.

The next two propositions give uniform upper bounds on the individual utility of a fixed player where uniformity is with respect to all unilateral strategy changes by the player.

Proposition 3.2 *Given* Λ, $i \in [1, n]$, *let* $v_i = \sum_{j \in J_i^-} b_{ij} - q_j^i$. *Then*

$$\forall \Lambda' \in \mathcal{L}_i(\Lambda), \quad \bar{U}_i(\Lambda') \leq v_i.$$

Proof. Since for all $j \in J_i^+$, $U_i(c_j(\Lambda')) = 0$, the upper bound v_i follows immediately. ■

Proposition 3.3 *Given* Λ, $i \in [1, n]$, *let* $\lambda_i > v_i$ *and* $J_i^+ = \emptyset$. *Then* $\exists j^* \in J_i^*$ *such that*

$$\forall \Lambda' \in \mathcal{L}_i(\Lambda), \quad \bar{U}_i(\Lambda') \leq v_i - (b_{ij^*} - q_{j^*}^i).$$

Proof. First, $J_i^* \neq \emptyset$ since $J_i^- \neq \emptyset$. Since $\lambda_i > v_i$ and $J_i^+ = \emptyset$, for at least one $j \in J_i^-$, $q_j > b_{ij}$. This implies that $U_i(c_j(\Lambda')) = 0$. It is easily checked that

$$\max_{\Lambda' \in \mathcal{L}_i(\Lambda)} \bar{U}_i(\Lambda')$$

is achieved by Λ' such that $\lambda'_{i\ell} = b_{i\ell} - q_\ell^i$ if $\ell \neq j^*$, and $\lambda'_{ij^*} = \lambda_i - \sum_{\ell \neq j^*} \lambda'_{i\ell}$, where j^* is some element in J_i^*. Hence, $\bar{U}_i(\Lambda') = \sum_{\ell \neq j^*} b_{i\ell} - q_\ell^i = v_i - (b_{ij^*} - q_{j^*}^i)$. ∎

The next result is the first of the two main theorems giving a complete characterization of when an allocation is Nash. For each player, one of three conditions must hold: a player either achieves full individual utility λ_i, or partial utility $v_i = \sum_{j \in J_i^-} b_{ij} - q_j^i$ dumping the excess traffic $\lambda_i - v_i$ into one or more service classes belonging to J_i^+, or partial utility $v_i - (b_{ij^*} - q_{j^*}^i)$ with excess traffic being assigned to one of the service classes in J_i^*.

Theorem 3.4 (Nash Characterization) *Λ is a Nash equilibrium iff $\forall i \in [1, n]$ either*

(a) $I_i^+ = \emptyset$, *or*

(b) $I_i^- = \emptyset$, $J_i^+ \neq \emptyset$, $J_i^- \subseteq I_i^0$, *or*

(c) $I_i^- = \emptyset$, $J_i^+ = \emptyset$, $\exists\, j^* \in J_i^*$ *such that* $J_i^- \setminus \{j^*\} \subseteq I_i^0$.

Proof. ($\Leftarrow$). Assume $I_i^+ = \emptyset$ (part (a)). Since $\forall j$, $\lambda_{ij} > 0 \Longrightarrow q_j \leq b_{ij}$, we have $\bar{U}_i(\Lambda) = \lambda_i$, the trivial upper bound on $\bar{U}_i$.

Assume (b) holds. By Proposition 3.2, $\bar{U}_i(\Lambda) \leq v_i$ where $v_i = \sum_{j \in J_i^-} b_{ij} - q_j^i$. $I_i^- = \emptyset$ and $J_i^- \subseteq I_i^0$ imply $\bar{U}_i(\Lambda) = v_i$, thus achieving the upper bound which holds for any $\Lambda' \in \mathcal{L}_i(\Lambda)$. Notice that J_i^+, J_i^- do not depend on the actions of player i.

Assume part (c). $I_i^- = \emptyset$ and $\exists\, j^* \in J_i^*$ such that $J_i^- \setminus \{j^*\} \subseteq I_i^0$ imply that $\bar{U}_i(\Lambda) \geq v_i - (b_{ij^*} - q_{j^*}^i)$. If $J_i^* = \emptyset$, which holds iff $J_i^- = \emptyset$, then we are done. Assume $J_i^* \neq \emptyset$. Notice that the case $J_i^- \subseteq I_i^0$ is covered by part (b) or (a). Hence, we can assume $j^* \in I_i^+$. $I_i^- = \emptyset$ and $j^* \in I_i^+$ imply that $v_i < \lambda_i$. Thus, we can apply Proposition 3.3 which in conjunction with the lower bound on $\bar{U}_i(\Lambda)$ yields $\bar{U}_i(\Lambda) = v_i - (b_{ij^*} - q_{j^*}^i)$.

($\Rightarrow$). We will prove the contrapositive. That is, assuming $\exists\, i \in [1, n]$, given Λ, such that

$$I_i^+ \neq \emptyset \quad \wedge \quad (\, I_i^- \neq \emptyset \;\vee\; J_i^+ = \emptyset \;\vee\; J_i^- \not\subseteq I_i^0\,) \\ \wedge \quad (\, I_i^- \neq \emptyset \;\vee\; J_i^+ \neq \emptyset \;\vee\; \forall j^* \in J_i^* : J_i^- \setminus \{j^*\} \not\subseteq I_i^0\,),$$

we will show that Λ is not Nash. There are nine clauses to be considered which are grouped into five cases (i)–(v).

(i) $(I_i^+ \neq \emptyset \wedge I_i^- \neq \emptyset)$, $(I_i^+ \neq \emptyset \wedge I_i^- \neq \emptyset \wedge J_i^+ = \emptyset)$, $(I_i^+ \neq \emptyset \wedge I_i^- \neq \emptyset \wedge J_i^- \not\subseteq I_i^0)$, $(I_i^+ \neq \emptyset \wedge I_i^- \neq \emptyset \wedge J_i^+ \neq \emptyset)$, $(I_i^+ \neq \emptyset \wedge I_i^- \neq \emptyset \wedge \forall j^* \in J_i^* : J_i^- \setminus \{j^*\} \not\subseteq I_i^0)$. They all have in common the conjunction $I_i^+ \neq \emptyset \wedge I_i^- \neq \emptyset$. The latter implies $\exists\, j, j', j \neq j'$, such that $\lambda_{ij} > 0$, $q_j > b_{ij}$, and $q_{j'} < b_{ij'}$.

We can construct an assignment $\Lambda' \in \mathcal{L}_i(\Lambda)$ such that $\lambda'_{i\ell} = \lambda_{i\ell}$, $\ell \in [1, m] \setminus \{j, j'\}$, and $\lambda'_{ij} = \lambda_{ij} - \epsilon$, $\lambda'_{ij'} = \lambda_{ij'} + \epsilon$, where $\epsilon = \min\{\lambda_{ij}, b_{ij'} - q_{j'}\}$. This yields

$$\bar{U}_i(\Lambda') - \bar{U}_i(\Lambda) \geq \epsilon$$

from which it follows that Λ is not a Nash equilibrium. It can be easily checked that the argument applies to the other four clauses.

(ii) $(I_i^+ \neq \emptyset \wedge J_i^+ = \emptyset \wedge J_i^+ \neq \emptyset) = \mathsf{F}$. The implication reduces to a tautology.

(iii) $(I_i^+ \neq \emptyset \wedge J_i^- \not\subseteq I_i^0 \wedge J_i^+ \neq \emptyset)$. $J_i^- \not\subseteq I_i^0$ implies that $J_i^- \neq \emptyset$. For $j \in J_i^- \setminus I_i^0$, either $q_j < b_{ij}$ or $q_j > b_{ij}$. If $q_j < b_{ij}$, then the argument from (i) can be applied. Assume $q_j > b_{ij}$. This implies that $U_i(c_j(\Lambda)) = 0$. Since $J_i^+ \neq \emptyset$, for all $j' \in J_i^+$, $j' \neq j$ and $U_i(c_{j'}(\Lambda)) = 0$.

We can construct $\Lambda' \in \mathcal{L}_i(\Lambda)$ such that $\lambda'_{i\ell} = \lambda_{i\ell}$, $\ell \in [1,m] \setminus \{j, j'\}$, and $\lambda'_{ij} = \lambda_{ij} - \epsilon$, $\lambda'_{ij'} = \lambda_{ij'} + \epsilon$, where $\epsilon = q_j - b_{ij}$. We still have $U_i(c_{j'}(\Lambda')) = 0$, however,

$$U_i(c_j(\Lambda')) = b_{ij} - q_j^i > 0$$

since $j \in J_i^-$ and $q'_j = b_{ij}$. Hence Λ is not Nash.

(iv) $(I_i^+ \neq \emptyset \wedge J_i^+ = \emptyset \wedge \forall j^* \in J_i^* : J_i^- \setminus \{j^*\} \not\subseteq I_i^0)$. $J_i^+ = \emptyset$ implies $J_i^- \neq \emptyset$, $J_i^* \neq \emptyset$. In fact, $|J_i^-| \geq 2$. This follows from $\forall j^* \in J_i^* : J_i^- \setminus \{j^*\} \not\subseteq I_i^0$ since $J_i^* \subseteq J_i^-$, and assuming $|J_i^-| < 2$ would imply $J_i^- \setminus \{j^*\} = \emptyset$ which would violate $J_i^- \setminus \{j^*\} \not\subseteq I_i^0$.

Let $j \in J_i^-$, $j' \in J_i^*$, with $j \neq j'$. If $q_j < b_{ij}$, then the argument from (i) applies and we are done. Similarly for j'. Let $q_j > b_{ij}$. If $|I_i^+| \geq 2$, then we can choose $j'' \in I_i^+$ with $j \neq j''$ and apply the argument in (iii) with I_i^+ in place of J_i^+. Assume $|I_i^+| = 1$, i.e., $I_i^+ = \{j\}$. We need only consider the case $q_{j'} = b_{ij'}$. Notice that $J_i^- \setminus J_i^* \neq \emptyset$ since, if $J_i^- = J_i^*$ then $J_i^- \setminus \{j\} \subseteq I_i^0$ by $|I_i^+| = 1$, which would contradict the assumption $\forall j^* \in J_i^* : J_i^- \setminus \{j^*\} \not\subseteq I_i^0$.

Construct the assignment $\Lambda' \in \mathcal{L}_i(\Lambda)$ such that $\lambda'_{i\ell} = \lambda_{i\ell}$, $\ell \in [1,m] \setminus \{j, j'\}$, and $\lambda'_{ij} = \lambda_{ij} - \epsilon$, $\lambda'_{ij'} = \lambda_{ij'} + \epsilon$, where $\epsilon = q_j - b_{ij}$. Now, $U_i(c_{j'}(\Lambda')) = 0$ but $U_i(c_j(\Lambda')) = b_{ij} - q_j^i$. Since $j \in J_i^- \setminus J_i^*$ and $j' \in J_i^*$,

$$(b_{ij} - q_j^i) - (b_{ij'} - q_{j'}^i) > 0$$

which implies $\bar{U}_i(\Lambda') - \bar{U}_i(\Lambda) > 0$.

(v) $(I_i^+ \neq \emptyset \wedge J_i^- \not\subseteq I_i^0 \wedge \forall j^* \in J_i^* : J_i^- \setminus \{j^*\} \not\subseteq I_i^0)$. In the proof of (iv), $J_i^+ = \emptyset$ was only needed to establish $J_i^- \neq \emptyset$ which we can get from $J_i^- \not\subseteq I_i^0$. Hence the argument of (iv) carries over unchanged. ■

The next lemma gives a useful property of Nash equilibria which is used in the proof of Theorem 3.6. For a Nash equilibrium Λ, an equivalent assignment Λ' (not neccessarily Nash) can be found with the same total utility so that the players can be partitioned into two sets around a unique, dividing player $i_{\Lambda'}$. The first set consists of players with indices larger than $i_{\Lambda'}$ with respect to the ordering induced by Proposition 3.1, with all players having full utility. The second set consists of players with smaller indices than $i_{\Lambda'}$, all of them having zero utility. The third set is the singleton set $\{i_{\Lambda'}\}$ consisting of the dividing player who has partial utility. We will call such an assignment Λ' a *normal form* of Λ.

Lemma 3.5 (Normal Form) *Let Λ be a Nash equilibrium with $\bar{U}(\Lambda) < \sum_{i=1}^n \lambda_i$. Let $i_\Lambda \equiv \max\{i : \bar{U}_i(\Lambda) < \lambda_i\}$. Then $\exists\, \Lambda'$ with $\bar{U}(\Lambda') = \bar{U}(\Lambda)$ such that*

(a) $\forall i < i_{\Lambda'}$, $\bar{U}_i(\Lambda') = 0$, *and*

(b) $\forall i > i_{\Lambda'}$, $\bar{U}_i(\Lambda') = \lambda_i$.

Proof. Let $S_{i_\Lambda} = \{\, i \in [1,n] : i < i_\Lambda,\ \bar{U}_i(\Lambda) \neq 0 \,\}$. By the definition of i_Λ, for all $i > i_\Lambda$, $\bar{U}_i(\Lambda) = \lambda_i$, which gives (b). If $S_{i_\Lambda} = \emptyset$, then we are done.

Assume $S_{i_\Lambda} \neq \emptyset$. We will construct as assignment Λ' from Λ such that it satisfies property (a) while preserving (b). Notice that by Theorem 3.4 and $\lambda_{i_\Lambda} > \bar{U}_{i_\Lambda}(\Lambda)$, $q_j \geq b_{i_\Lambda j}$ for all $j \in [1,m]$. Also, by Proposition 3.1, $b_{i_\Lambda j} \geq b_{ij}$ for all $i \in S_{i_\Lambda}$. Let

$$\nu = \lambda_{i_\Lambda} - \bar{U}_{i_\Lambda}(\Lambda), \qquad \pi = \sum_{i<i_\Lambda} \bar{U}_i(\Lambda).$$

To achieve (a), we will distribute the excess utility π into service classes j with $q_j > b_{i_\Lambda j}$ thus nullifying their contribution. To avoid otherwise disturbing the utility assignment, we will move a commensurate amount from ν, *exactly* filling the gap left by π. That is, $q'_j = q_j$, $j \in [1,m]$, in the modified assignment Λ'. If $\nu > \pi$, the reassignment can be achieved in one round. If $\nu \leq \pi$, a refined construction is used that iteratively shrinks the violating player set S_{i_Λ} until it becomes empty. Following is a formal description of the construction.

Case (i). Assume $\nu > \pi$. Let $K^- = \{\, j \in [1,m] : q_j = b_{ij},\ \lambda_{ij} > 0,\ i \in S_{i_\Lambda} \,\}$, $K^+ = \{\, j \in [1,m] : q_j > b_{i_\Lambda j},\ \lambda_{i_\Lambda j} > 0 \,\}$. We construct Λ' as follows. For $i \in S_{i_\Lambda}$, $j \in K^-$,

$$\lambda'_{ij} = 0, \qquad \lambda'_{i_\Lambda j} = \lambda_{i_\Lambda j} + \sum_{k \in S_{i_\Lambda}} \lambda_{kj}.$$

For $i \in S_{i_\Lambda}$, $j \in K^+$,

$$\lambda'_{ij} = \lambda_{ij} + \epsilon_{ij}, \qquad \lambda'_{i_\Lambda j} = \lambda_{i_\Lambda j} - \sum_{k \in S_{i_\Lambda}} \epsilon_{kj},$$

where $\epsilon_{ij} \geq 0$, $\sum_{k \in S_{i_\Lambda}} \epsilon_{kj} \leq \lambda_{i_\Lambda j}$, and $\sum_{i \in S_{i_\Lambda}, j \in K^+} \epsilon_{ij} = \pi$. For all other i and j, $\lambda'_{ij} = \lambda_{ij}$.

By construction, $q'_j = q_j$ for $j \in [1,m]$, and since the excess utility π has been transferred into service classes belonging to K^+, we have $\bar{U}_i(\Lambda') = 0$ for $i \in S_{i_\Lambda}$. Hence, $i_{\Lambda'} = i_\Lambda$. Also, notice that

$$\bar{U}_{i_\Lambda}(\Lambda') = \bar{U}_{i_\Lambda}(\Lambda) + \pi$$

since player i_Λ's unutilized traffic volume has been tranferred to service classes in K^- where, by Proposition 3.1, they now count.

Case (ii). Assume $\nu \leq \pi$. We will perform a similar switch as in case (i), however, over (possibly) several rounds each time monotonically shrinking S_{i_Λ} and obtaining a new estimate for $i_{\Lambda'}$ by decrementing the previous estimate.

In the first round, we transfer a traffic volume of ν from players $i \in S_{i_\Lambda}$ with assignments in K^- to service classes belonging to K^+. To preserve, $q'_j = q_j$, $j \in [1,m]$, we transfer an equal amount from player i_Λ's assignments in K^+ to K^-. This is possible since $\nu \leq \pi$. This yields

$$\bar{U}_{i_\Lambda}(\Lambda') = \bar{U}_{i_\Lambda}(\Lambda) + \nu = \lambda_{i_\Lambda}.$$

Thus, $i_{\Lambda'} \leq i_\Lambda - 1$.

If $S_{i_{\Lambda'}} = \emptyset$ then we are done. If $S_{i_{\Lambda'}} \neq \emptyset$, we recursively repeat the switching process with $i_{\Lambda'}$ in place of i_Λ until $S_{i_{\Lambda'}} = \emptyset$. Since the dividing player's index monotonically decreases by at least one in

each round, the process terminates in at most $i_\Lambda - 1$ rounds. ■

Before stating and proving the second main theorem, we remark that the usefulness of the normal form of a Nash equilibrium (itself not necessarily Nash) comes into play when checking for system optimality of a Nash assignment. This is so since, as we shall see, it is sufficient to check Pareto optimality of the normal form to establish system optimality of the original Nash assignment. Moreover, a normal form is easy to obtain from the original Nash configuration (construction in the proof of Lemma 3.5) and checking for Pareto optimality is generally easier than checking for system optimality. The latter is especially true in the case of normal forms where we can construct efficient decision procedures for checking Pareto optimality.

Theorem 3.6 (Nash & System Optimal) *Let Λ be a Nash equilibrium and let Λ' be its normal form. Then Λ is system optimal iff Λ' is Pareto optimal.*

Proof. Let Λ' be the normal form constructed in the proof of Lemma 3.5. We will prove the following statement from which the theorem follows immediately: Λ' is not system optimal iff there is a Λ^* with $\bar{U}(\Lambda^*) > \bar{U}(\Lambda')$ such that

(a) $\forall i \in [1, n]$, $\bar{U}_i(\Lambda^*) \geq \bar{U}_i(\Lambda')$, and

(b) $\exists\, i \leq i_{\Lambda'}$ such that $\bar{U}_i(\Lambda^*) > \bar{U}_i(\Lambda')$.

That is, Λ' is not Pareto optimal. Note that $\bar{U}(\Lambda') = \bar{U}(\Lambda)$ by the definition of Λ'.

The '$\Leftarrow$' direction of the statement above is trivial. To show the '$\Rightarrow$' direction, we start with a $\tilde{\Lambda}$ with $\bar{U}(\tilde{\Lambda}) > \bar{U}(\Lambda')$, which exists since Λ' is not system optimal. For all $i > i_{\Lambda'}$, $\bar{U}_i(\Lambda') = \lambda_i$, hence any increase in the utility $\bar{U}(\tilde{\Lambda})$ over $\bar{U}(\Lambda')$ must come from one or more $i \leq i_{\Lambda'}$ for which $\bar{U}_i(\tilde{\Lambda}) > \bar{U}_i(\Lambda')$. Indeed, $\bar{U}_i(\Lambda') = 0$ for $i < i_{\Lambda'}$, hence (b) and part of condition (a), i.e., $\forall i < i_{\Lambda'}$, $\bar{U}_i(\tilde{\Lambda}) \geq \bar{U}_i(\Lambda')$, are already satisfied. We will construct Λ^* from $\tilde{\Lambda}$ such that the remaining part of (a), i.e., $\forall i \geq i_{\Lambda'}$, $\bar{U}_i(\tilde{\Lambda}) \geq \bar{U}_i(\Lambda')$, is satisfied as well. Let

$$L^- = \{\, i \leq i_{\Lambda'} : \bar{U}_i(\tilde{\Lambda}) > \bar{U}_i(\Lambda') \,\}, \qquad L^+ = \{\, i \geq i_{\Lambda'} : \bar{U}_i(\tilde{\Lambda}) < \bar{U}_i(\Lambda') \,\}.$$

Clearly, $L^- \cap L^+ = \emptyset$. Moreover, $i_{\Lambda'}$ need not be an element of either L^- or L^+. Let

$$\pi = \sum_{i \in L^-} \bar{U}_i(\tilde{\Lambda}) - \bar{U}_i(\Lambda'),$$
$$\nu = \sum_{i \in L^+} \bar{U}_i(\Lambda') - \bar{U}_i(\tilde{\Lambda}).$$

By $\bar{U}(\tilde{\Lambda}) > \bar{U}(\Lambda')$, we have $\pi - \nu > 0$. We can perform a switch in assignments between players in L^- and L^+, similar to the proof of Lemma 3.5, obtaining an assignment Λ^* which preserves $q_j^* = \tilde{q_j}$, $j \in [1, m]$, and which satisfies $\forall i \in L^+$, $\bar{U}_i(\Lambda^*) = \bar{U}_i(\Lambda')$, $\forall i \in L^-$, $\bar{U}_i(\Lambda^*) \geq \bar{U}_i(\Lambda')$, and for at least one element $i \in L^-$, $\bar{U}_i(\Lambda^*) > \bar{U}_i(\Lambda')$.

Pick any two players $i_- \in L^-$, $i_+ \in L^+$. Then, $\exists j_-, j_+ \in [1, m]$, $j_- \neq j_+$, such that

$$\lambda_{i_- j_-} > 0, \quad b_{i_- j_-} \geq q_{j_-} \quad \text{and} \quad \lambda_{i_+ j_+} > 0, \quad b_{i_+ j_+} < q_{j_+}.$$

The inequalities follow from Lemma 3.5, and $j_- \neq j_+$ follows from the inequalities and $b_{i_-j_-} \leq b_{i_+j_+}$.

Let $\epsilon = \min\{\lambda_{i_-j_-}, \lambda_{i_+j_+}\}$. We can move an ϵ amount of i_-'s assignment from j_- to j_+, and an equal amount of i_+'s assignment from j_+ to j_-. Since $b_{i_-j_-} \leq b_{i_+j_+}$, player i_+'s utility strictly increases by ϵ whereas player i_-'s utility strictly decreases by the same amount. The other players' utilities remain undisturbed since the total volume assignment to each service class was held invariant.

Since $\pi - \nu > 0$, this reassignment process can be repeated until a total traffic volume of ν has been shifted from players in L^- to players in L^+ and vice versa. Since $\forall i > i_{\Lambda'}$, $\bar{U}_i(\Lambda') = \lambda_i$, by the definition of ν, we have that $\forall i > i_{\Lambda'}$, $\bar{U}_i(\Lambda^*) = \lambda_i$, and thus $\forall i > i_{\Lambda'}$, $\bar{U}_i(\Lambda^*) \geq \bar{U}_i(\Lambda')$. For players $i < i_{\Lambda'}$, $\bar{U}_i(\Lambda^*) \geq \bar{U}_i(\Lambda')$ remains satisfied since $\bar{U}_i(\Lambda') = 0$.

The only consideration left is player $i_{\Lambda'}$. If $i_{\Lambda'} \notin L_- \cup L_+$, then we are done. If $i_{\Lambda'} \in L_-$, then after the switch operation, either $\bar{U}_{i_{\Lambda'}}(\Lambda^*) \geq \bar{U}_{i_{\Lambda'}}(\Lambda')$—in which case we are done—or $\bar{U}_{i_{\Lambda'}}(\Lambda^*) < \bar{U}_{i_{\Lambda'}}(\Lambda')$. In the latter, we may perform a further switch between player $i_{\Lambda'}$ and players $i < i_{\Lambda'}$ until $i_{\Lambda'}$'s utility has been suffiently increased vis-à-vis $\bar{U}_{i_{\Lambda'}}(\Lambda')$. This is possible since $\pi - \nu > 0$. If $i_{\Lambda'} \in L_+$, and after the switch we still have $\bar{U}_{i_{\Lambda'}}(\Lambda^*) < \bar{U}_{i_{\Lambda'}}(\Lambda')$, then the same process as with $i_{\Lambda'} \in L_-$ can be done yielding the desired ordering result. ■

Given the form of Theorem 3.6, one may wonder whether all assignments that are Nash and Pareto optimal are also system optimal. The next result gives a counterexample which shows that Theorem 3.6 is "tight" in the sense that there are assignments that are both Nash and Pareto optimal but not system optimal.

Proposition 3.7 *There exist Nash equilibria that are Pareto optimal but not system optimal.*

Proof. The following describes a counter example consisting of a system of 3 players and 3 service classes and an assignment Λ which is Nash and Pareto but not system optimal. As usual, using Proposition 3.1, for each service class j, we can assume that $b_{1j} \leq b_{2j} \leq b_{3j}$.

For service class 1, take $b_{11} = b_{21}$, and $b_{31} = b_{11} + 1$. For service class 2, take $b_{12} = b_{22} = b_{32} = e$ where e is a very small positive quantity. For service class 3, take $b_{23} = b_{33}$ and $b_{13} = s$. Also, let $b_{32} < b_{31} < b_{33}$.

The assignment Λ is defined as follows. The assignments to service class 1 are: $q_1 = \lambda_{11} = \lambda_1 = b_{11}$, and $\lambda_{21} = \lambda_{31} = 0$. The assignments to service class 2 are: $q_2 = \lambda_{22} = \lambda_2 = b_{22} + E$, where E is a very large quantity and $\lambda_{12} = \lambda_{32} = 0$. The assignments to service class 3 are: $q_3 = \lambda_{33} = b_{33}$ and $\lambda_{13} = \lambda_{23} = 0$. This assignment Λ is clearly a Nash equilibrium: $\lambda_{22} = \lambda_2$ is unutilized, but player 2 cannot unilaterally reassign its share to improve its utility. Players 1 and 3 have full utility. Hence the total utility for assignment Λ is $\lambda_3 + \lambda_1$.

This assignment Λ, however, is not system optimal. The total utility can be increased using the following changes to the assignment: the quantity λ_1 can be moved to service class 2 from service class 1 so that the new λ_{11} is now 0, but the new λ_{21} is now equal to λ_1. A part of λ_2 equivalent to the quantity $\lambda_1 + 1$ is moved into service class 3 so that service class 2 now has total volume q_2 that is one less than its previous value. Therefore λ_2 is now partitioned into $\lambda_{23} = \lambda_1 + 1$, with the remainder of λ_2 assigned to λ_{23} while λ_{21} remains 0. Finally, a part of λ_3 equivalent to the quantity $\lambda_1 + 1$ is moved to service class 1 so the volume of service class 1 increases overall by 1 unit, and service class

3 retains the same volume as before. Now λ_3 is partitioned into $\lambda_{31} = \lambda_1 + 1$, with the remainder of λ_3 assigned to λ_{33} while λ_{32} remains 0.

The utility of player 3 remains the same as before, i.e., it has full utility λ_3. The utility of player 1 has decreased from λ_1 to 0 and the utility of player 2 has increased from 0 to $\lambda_1 + 1$. Hence the total utility after completion of the above reassignment is $\lambda_1 + \lambda_3 + 1$ and hence it has increased by 1 overall which shows that the assignment Λ is not system optimal. It is not hard to see that Λ is, in fact, Pareto optimal; i.e., for any assignment Λ' that has higher total utility, there must be at least one player, in particular, player 1, whose individual utility in Λ' is less than that in Λ. ■

We remark that it is also not difficult to construct Nash equilibria which are not Pareto optimal. In fact, the normal form of a Nash assignment Λ obtained from the construction in the proof of Lemma 3.5 is typically itself Nash, and can be used to exhibit assignments that are Nash but not Pareto. Thus, in general, gaps are present in all the important relations between assignments that are Nash equilibria, Pareto optimal, or system optimal.

3.2 Dynamical Behavior of QoS Provision Game

Section 3.1 provides an equilibrium or static analysis of the network game, showing under what conditions an assignment is Nash and when it is system optimal. In this section, we look at the dynamical side, our interest being in the fate of time evolutions of the noncooperative network game when starting from arbitrary initial configurations. Does every instance of the network game have stable fixed points—i.e., Nash equilibria—to which to converge to? If so, are Nash equilibria reachable from every initial configuration and when they do, what is the speed of convergence?

The following proposition answers the first of the questions in the negative. That is, we construct a simple example which shows that for certain choices of λ_i, U_i, and c_j, (and, therefore, of $b_{i,j}$), no Nash equilibria exist.

Proposition 3.8 *Consider the family of 2-player/2-service class systems such that the thresholds b_{ij} on the total traffic volume of the service classes satisfy $b_{1j} < b_{2j}$, $j = 1, 2$ (i.e., the ordering of Proposition 3.1 is actually strict). Furthermore, assume the following inequalities hold:*

(a) $\lambda_2 < b_{11} + b_{12}$,

(b) $\lambda_2 + \lambda_1 > b_{21} + b_{22} > b_{11} + b_{12}$,

(c) $\lambda_2 > \max\{b_{11}, b_{12}\}$.

Then, for such choices of λ_i, b_{ij}, no Nash equilibrium exists.

Proof. To the contrary, assume Λ is a Nash equilibrium for the example described in the proposition. Due to the first inequality satisfied by the λ_i's and the b_{ij}'s, it follows that there is a service class j_1 for which $\lambda_{2j_1} = q^1_{j_1} < b_{1j_1}$. Using this observation and applying the Nash characterization from Theorem 3.4 to the player 1, we obtain (without loss of generality, by the choice of j_1),

$$q_{j_1} \leq b_{1j_1}. \tag{3.9}$$

Now, due to the second inequality (b) in the proposition, it follows that service class $j_2 \neq j_1$ has assigned traffic volume

$$q_{j_2} > b_{2j_2}. \tag{3.10}$$

Furthermore, using (3.9) and the third inequality in the proposition,

$$\lambda_{2j_2} \neq 0. \tag{3.11}$$

Moreover, since $b_{1j} < b_{2j}$, for all j, we know from (3.9) that $\lambda_{1j_1} \leq q_{j_1} \leq b_{1j_1} < b_{2j_1}$. Thus we get

$$\lambda_{1j_1} = q_{j_1}^2 < b_{2j_1}. \tag{3.12}$$

Using (3.10), (3.11), and (3.12), and applying the Nash characterization from Theorem 3.4 to player 2, we get $q_{j_1} \geq b_{2j_1}$ which contradicts (3.9) since $b_{1j} < b_{2j}$, for all j. ■

In contrast to the previous example, we now give a class of network games for which not only do Nash equilibria always exist, Nash equilibria, Pareto optima, and system optima conincide. Moreover, a "natural" dynamic process converges to Nash equilibria starting from arbitrary initial configurations. First, the definition of the dynamic update process.

For present purposes, we define a *dynamic update process* $\mathcal{P}$ as follows. We assume that the players move asynchronously, and at each step t, a single player i_t unilaterally and selfishly reassigns its λ_{i_t} so that the new assignment Λ_t maximizes its individual utility $\bar{U}_{i_t}(\Lambda)$. We further assume that no player moves *unneccessarily*—i.e., a player only makes changes to its assignment if it thereby strictly increases its individual utility.

Theorem 3.13 (Resource-Plentiful System) *If for every player $i \in [1, n]$, it holds that*

$$\sum_{j=1}^{m} q_j \leq \sum_{j=1}^{m} b_{ij}, \tag{3.14}$$

then the following statements hold.

(a) *Λ is a Nash equilibrium if and only if Λ is a system optimum if and only if Λ is a Pareto optimum. Moreover the optimum value achieved is $\bar{U}(\Lambda) = \sum_j q_i = \sum_i \lambda_i$.*

(b) *Starting from any initial assignment Λ_0, the dynamic update process $\mathcal{P}$ converges to a Nash equilibrium Λ. Moreover, Λ is attained as soon as the sequence of players (i.e., moves) in the process $\mathcal{P}$ includes the subsequence $n, n-1, \ldots, 1$.*

Proof. To show the first item above, it is sufficient to show that every Nash equilibrium Λ is system optimal with utility $\bar{U}(\Lambda) = \sum_i \lambda_i$. The equivalence of Nash, Pareto, and system optima follows immediately.

Due to the inequality in (3.14), for an assignment Λ, each player can always unilaterally reassign its λ_{ij}'s and strictly increase its own utility unless the following holds:

$$\forall i \, \forall j : \ \lambda_{ij} \neq 0 \implies q_j \leq b_{ij}. \tag{3.15}$$

Thus Λ is a Nash equilibrium (i.e., such a reassignment is impossible) only if (3.15) holds. But (3.15) is equivalent to

$$\forall i \, \forall j : \; q_j > b_{ij} \implies \lambda_{ij} = 0,$$

which, in turn, implies that Λ is system optimal.

Note that if (3.15) holds for Λ, then clearly no player contributes to any service class where the contribution would be unutilized—i.e., every player has complete utility and thus $\bar{U}(\Lambda) = \sum_i \lambda_i$. Hence Nash, Pareto, and system optima are all equivalent.

To show that the process $\mathcal{P}$ converges to a Nash equilibrium starting from any initial configuration, notice that

(1) When it is player i's turn to move, if $\bar{U}_i < \lambda_i$—the player has less than full utility—then it can always unilaterally reassign its λ_{ij}'s and achieve full utility. In other words, it can achieve the status described in (3.15). Otherwise, if player i has full utility, it does not move at all, i.e., it keeps its current assignment.

(2) Once player i has moved, the subsequent moves of players with indices $k < i$ will not affect i's (full) utility. This is due to the inequality in Proposition 3.1, and because of the observation in (1): the move of such a player k does not *newly* cause the traffic volume q_j of any service class to cross the threshold $b_{kj} \leq b_{ij}$.

Thus, once player n has moved, it achieves full utility, and subsequent moves of the other players does not affect its utility; hence player n never moves again. In general, once players $n, n-1, \ldots, n-k$ have moved, in that order, the subsequent moves of the lower players $1, \ldots, n-k-1$ do not affect the (full) utility of the higher players $n, n-1, \ldots, n-k$, and hence they never move again. It follows that a Nash equilibrium Λ is attained by the process $\mathcal{P}$, starting from any initial assignment, as soon as the sequence of players (i.e., moves) includes the subsequence $n, n-1, \ldots, 1$. ■

4 Multi-dimensional QoS Vectors

This section generalizes the QoS provision model to non-scalar QoS vectors. For example, a typical QoS vector reflecting an application's quality of service requirement may include bounds on packet loss rate, end-to-end delay, and their variances (e.g., jitter). We seek to answer two questions which arise as a result of the extension.

First, do the game theoretic results of Section 2 carry over in the multi-dimensional QoS vector case? Second, what is the effect of system variability—e.g., caused by fluctuating background or cross traffic as well as overall system noise—on the rendered QoS of the service classes? In particular, assuming that mean packet loss rate and its variance are components of the QoS vectors, is it possible, under a weighted fair queueing packet scheduling policy, to induce service classes where one service class delivers strictly better service than some other?

The first question is answered in the affirmative. With respect to the second question, we show the surprising result that under "typical" operating conditions, achieving both lower mean packet loss rate and variance is impossible.

4.1 Extension of Game-theoretic Analysis

In Section 2, we formulated and analyzed a noncooperative QoS provision game based on singleton QoS vectors, $\mathbf{x} = (c)$, where c was a bound on packet loss rate. Here, we will extend the model to multi-dimensional QoS vectors $\mathbf{x} \in \mathbb{R}^s$, $s \geq 1$, and show that, as far as Theorem 3.4 and Theorem 3.6 are concerned, the singleton vector analysis carries over unchanged.

Let $\mathbf{x} = (x_1, x_2, \ldots, x_s)^T$, and let $\mathbf{x}^j = (x_1^j, x_2^j, \ldots, x_s^j)^T$ denote the quality of service rendered to service class $j \in [1, m]$. As before, we make the monotonicity assumption $dx_r^j/dq_j \geq 0$, $r \in [1, s]$, $j \in [1, m]$, which is satisfied by most packet scheduling policies of interest including weighted fair queueing. Each player's utility function $U_i(\mathbf{x})$, $i \in [1, n]$, has the form

$$U_i(\mathbf{x}) = \begin{cases} 1, & \text{if } \forall r \in [1, s],\ x_r \leq \theta_r^i, \\ 0, & \text{otherwise,} \end{cases}$$

where $\boldsymbol{\theta}^i = (\theta_1^i, \theta_2^i, \ldots, \theta_s^i)^T \geq \mathbf{0}$ is the multi-dimensional threshold vector that represents the i'th application's preference.

In order to deal with the multi-dimensional QoS vectors and thresholds uniformly, we henceforth make one of two uniformity assumptions: either assume that the thresholds θ_r^i can be ordered such that the ordering is uniform over r, i.e.,

$$\forall r \in [1, s], \forall i \in [1, n]: \ \theta_r^i \leq \theta_r^{i+1}, \tag{4.1}$$

or we we assume that the functional forms x_r^j are uniform over r for each j, i.e.,

$$\forall j \in [1, m]: \ x_1^j = x_2^j = \cdots = x_s^j. \tag{4.2}$$

By isolatedness, $x_r^j = x_r^j(q_j)$, $r \in [1, s]$, $j \in [1, m]$, and just as in Proposition 3.1, the condition $x_r^j(q_j) \leq \theta_r^i$ can now be stated as $q_j \leq b_{ij}^r$ using the definition

$$b_{ij}^r = (x_r^j)^{-1}(\theta_r^i).$$

Let b_{ij} be the minimum over r, i.e., $b_{ij} = \min_{r \in [1,s]} b_{ij}^r$.

We can now rephrase $U_i(\mathbf{x}^j)$ as

$$U_i(\mathbf{x}^j) = \begin{cases} 1, & q_j \leq b_{ij}, \\ 0, & \text{otherwise.} \end{cases}$$

Moreover, under the assumption that the functional forms x_r^j are uniform over r for each j where x_*^j satisfies $\forall j \in [1, m], \forall r \in [1, s]$, $x_r^j = x_*^j$, and using the monotonicity of x_*^j, it can be observed that the following identity holds:

$$b_{ij} = \min_{r \in [1,s]} (x_*^j)^{-1}(\theta_r^i) = (x_*^j)^{-1}(\min_{r \in [1,s]} \theta_r^i). \tag{4.3}$$

That is, the min operator commutes with $(x_*^j)^{-1}$.

Now we are ready to state a total ordering on b_{ij} for fixed j corresponding to its counterpart Proposition 3.1.

Proposition 4.4 *For the multi-dimensional QoS vector model with assumption* (4.1) *or* (4.2)*, there exists an ordering of the players* $i \in [1,n]$ *such that* $\forall i \in [1,n-1]$, $\forall j \in [1,m]$,

$$b_{ij} \leq b_{i+1j}.$$

Proof. We will consider both uniformity assumptions on the multi-dimensional QoS vectors and thresholds simultaneously.

First, we consider the uniformity assumption (4.1) which states that the thresholds θ_r^i can be ordered such that the ordering is uniform over $r \in [1,s]$. Using this ordering and the monotonicity of x_r^j for each $j \in [1,m]$ and $r \in [1,s]$, by the definition of the b_{ij}^r, we can conclude that

$$\forall r \in [1,s], \forall j \in [1,m], \forall i \in [1,n-1], \quad b_{ij}^r \leq b_{i+1j}^r.$$

Now for any fixed i, j, let r' satisfy $\min_{r\in[1,s]} b_{ij}^r$ and let r'' satisfy $\min_{r\in[1,s]} b_{i+1j}^r$. Clearly, $b_{ij}^{r''} \leq b_{i+1j}^{r''}$, Furthermore, $b_{ij}^{r'} \leq b_{ij}^{r''}$, since b_{ij}^r is minimized at $r = r'$. It therefore follows that the same ordering on i also satisfies

$$\min_{r\in[1,s]} b_{ij}^r \leq \min_{r\in[1,s]} b_{i+1j}^r$$

from which the proposition follows immediately.

Next, we consider the uniformity assumption (4.2) which states that the functional forms x_r^j in the QoS vector $\boldsymbol{x}^j$ are uniform over $r \in [1,s]$ for each $j \in [1,m]$. In this case, we can define a natural ordering on i induced by

$$\min_{r\in[1,s]} \theta_r^i \leq \min_{r\in[1,s]} \theta_r^{i+1}.$$

Since the x_*^j's are all monotone, and as observed previously, $b_{ij} = (x_*^j)^{-1}(\min_{r\in[1,s]} \theta_r^i)$, this ordering yields the required

$$b_{ij} \leq b_{i+1j}$$

which holds for all $j \in [1,m]$ and $i \in [1,n-1]$. ∎

Proposition 4.5 *Theorem* 3.4 *and Theorem* 3.6 *hold for the the multi-dimensional QoS vector model with assumption* (4.1) *or* (4.2)*.*

The proof structure of Theorem 3.4 and Theorem 3.6 only rely on Proposition 3.1 to capture application QoS preferences. The QoS vectors (i.e., scalar packet loss indicator) and their functions affect the proof only through Proposition 3.1. Thus, under either of the uniformity assumptions, and with Proposition 4.4 in hand, it is straightforward to check that the proofs carry over unchanged giving Proposition 4.5.

4.2 Effect of Burstiness on Induced QoS

4.2.1 Problem Statement

A consequence of generalizing the QoS provision model to multi-dimensional QoS vectors *without* using either of the uniformity assumptions of the previous section is that there may no longer be a total

order on the set of application QoS requirements. That is, whereas in the scalar QoS case (e.g., take packet loss rate), applications could be linearly ordered by the bounds on their packet loss rate,

$$i \leq i' \Longleftrightarrow \theta_i \leq \theta_{i'},$$

in the vector QoS case, this is no longer the case and only a partial order can be imposed on the set of QoS requirements $\{\, \boldsymbol{\theta}^i : i \in [1,n] \,\}$ where $\boldsymbol{\theta}^i = (\theta_1^i, \theta_2^i, \ldots, \theta_s^i)^T$.

Given that the QoS rendered by a service class $j \in [1,m]$ is an *induced* phenomenon depending on the total traffic influx q_j to class j, the question arises how well the induced QoS levels match the needs of the constituent application QoS requirements. This is a very broad question and part of it (related to structural adaptation) is addressed in Section 5.

Here, we are interested in answering a very basic but important variant which *makes* one of the uniformity assumptions of the previous section, and asks whether in a 2-application class/2-service class/2-dimensional QoS vector system with mean packet loss rate and its variance[6] as the two QoS indicators, an induced QoS assignment that is strictly ordered can be achieved if the application QoS requirements are strictly ordered. That is, given

$$(\theta_c^1, \theta_\sigma^1) < (\theta_c^2, \theta_\sigma^2) \tag{4.6}$$

where $(\theta_c^1, \theta_\sigma^1)$ and $(\theta_c^2, \theta_\sigma^2)$ are the QoS requirements of application classes 1 and 2, respectively, with θ_c^i being a bound on the mean packet loss rate and θ_σ^i being a bound on the standard deviation of the packet loss rate, we seek to answer whether it is possible to achieve

$$\mathbf{x}^1 = (c_1, \sigma_1) \;<\; \mathbf{x}^2 = (c_2, \sigma_2) \tag{4.7}$$

where $\mathbf{x}^1$, $\mathbf{x}^2$ are the actual QoS levels rendered by the two service classes, respectively.

4.2.2 Qualitative Analysis

Let $(\xi(t))_{t \in \mathbb{R}_+}$ denote the stochastic process corresponding to the reserved cross traffic with mean $\mathbf{E}(\xi) = \lambda^R$. Here, $\mathbf{E}$ is the expectation operator. We will model *reservedness* by assuming $\xi(t) \leq \mu$ and

$$\eta(t) = \mu - \xi(t) \tag{4.8}$$

where η is the available service rate to the nonreserved traffic class which is itself a stochastic process[7]. Since our goal is to ascertain the effect of the cross traffic $\xi(t)$ variance on the aforementioned strict ordering of QoS provision question, we will treat the total service rate and the reserved traffic as fixed.

The packet loss rates of the rendered service class QoS vectors $\mathbf{x}^j = (c_j, \sigma_j)$, $j = 1, 2$, can be expressed as

$$c_j(t) = \max\{\, 1 - \alpha_j \eta(t)/q_j,\, 0 \,\}. \tag{4.9}$$

[6] Or, equivalently, mean end-to-end delay and its variance (i.e., jitter) which can be analyzed similarly.

[7] For present purposes, we may equivalently define $\eta(t) = \max\{\, \mu - \xi(t), 0 \,\}$.

Here, we have used the isolatedness property of WFQ. Thus $c_j(t)$ itself is a stochastic process and $0 \leq c_j(t) \leq 1$.

Note that the packet loss rate rendered by service class j is determined by its traffic volume q_j and therefore its "relative goodness" vis-à-vis other service classes is determined by the *normalized weight* $\omega_j = \alpha_j/q_j$, $j = 1, 2$. Since, by assumption, q_j is fixed, we may assume without loss of generality that

$$\omega_1 \geq \omega_2.$$

That is, service class 1 is "better" than service class 2, certainly with respect to packet loss rate since $c_1(t) \leq c_2(t)$, $\forall t \in \mathbb{R}_+$, which follows from (4.9). This also trivially implies

$$\mathbf{E}(c_1) \leq \mathbf{E}(c_2).$$

The variance, however, is more tricky. Let $\mathbf{V}$ denote the variance operator. Then

$$\mathbf{V}(c_j) \;=\; \int_{\eta \leq \frac{1}{\omega_j}} p(\eta)(1-\omega_j\eta)^2 d\eta \;-\; \mathbf{E}(c_j)^2 \tag{4.10}$$

since for $\eta \leq 1/\omega_j$, $c_j = 1 - \omega_j\eta$. By $\omega_1 \geq \omega_2$, the second moment term in (4.10) satisfies

$$\begin{aligned} \int_{\eta \leq \frac{1}{\omega_1}} p(\eta)(1-\omega_1\eta)^2 d\eta &\leq \int_{\eta \leq \frac{1}{\omega_1}} p(\eta)(1-\omega_2\eta)^2 d\eta \\ &\leq \int_{\eta \leq \frac{1}{\omega_2}} p(\eta)(1-\omega_2\eta)^2 d\eta. \end{aligned}$$

Since $\mathbf{E}(c_1) \leq \mathbf{E}(c_2)$, the two terms in (4.10) contribute in opposite directions and both $\mathbf{V}(c_1) \leq \mathbf{V}(c_2)$ and $\mathbf{V}(c_1) \geq \mathbf{V}(c_2)$ are possible depending on the distribution $p(\eta)$.

If $p(\eta)$ is concentrated toward $\max\{1/\omega_1, 1/\omega_2\}$—i.e., the distribution of ξ is concentrated toward 0—then c_1 and c_2 are close to 0 with high probability. Since $c_1(t) \leq c_2(t)$, in the degenerate case when $c_1(t) = 0$, $t \in \mathbb{R}_+$, it is certainly possible to have

$$\mathbf{E}(c_1) \leq \mathbf{E}(c_2), \qquad \mathbf{V}(c_1) \leq \mathbf{V}(c_2) \tag{4.11}$$

as desired in (4.7).

Let us consider the case when $p(\eta)$ is concentrated toward 0, i.e., the distribution of ξ is concentrated toward μ. Under such conditions of high cross traffic, $c_1(t), c_2(t) > 0$ with high probability and we will make the approximation

$$c_j(t) = 1 - \omega_j\eta(t).$$

Since $1 - \omega_j\eta(t) = 1 - \omega_j(\mu - \xi(t))$, we have

$$\mathbf{V}(c_j) = \omega_j^2 \mathbf{V}(\xi). \tag{4.12}$$

That is, the variance of the packet loss rate is proportional to the variance of the cross traffic process with constant of proportionality ω_j^2.

By $\omega_1 \geq \omega_2$, we now have $\mathbf{V}(c_1) \geq \mathbf{V}(c_2)$. Assuming strict inequality $\omega_1 > \omega_2$ between the two service classes, we get

$$\mathbf{E}(c_1) < \mathbf{E}(c_2), \qquad \mathbf{V}(c_1) > \mathbf{V}(c_2). \tag{4.13}$$

That is, the apparently "superior" service class 1 has a higher variance than service class 2 although it still has a smaller mean packet loss rate. Returning back to the original question of whether (4.7) can be achieved assuming $\omega_1 > \omega_2$, we conclude that under high cross traffic conditions, $c_1 < c_2$ but $\sigma_1 > \sigma_2$, and $\mathbf{x}^1 = (c_1, \sigma_1)$ and $\mathbf{x}^2 = (c_2, \sigma_2)$ become incomparable.

4.2.3 Numerical Estimation

Although the closed forms of the mean and variance of c_j are, in general, difficult to obtain, their numerical approximations are straightforward to compute assuming the distribution of the cross traffic process ξ is well-behaved. For example, in the case of long-range dependent processes with finite mean but infinite variance, accurately approximating $\mathbf{E}(c_j)$, $\mathbf{V}(c_j)$ will not be easy, requiring a large number of terms in the expansion to adequately reflect the polynomially decaying tail of the distribution.

Here, we will show the transition behavior (4.11), (4.13) as a function of mean cross traffic when the cross traffic process is Poisson with rate λ^R. Since $\mathbf{V}(c_j) = \mathbf{E}(c_j^2) - \mathbf{E}(c_j)^2$, we compute the first moment using

$$\mathbf{E}(c_j) \quad = \quad \sum_{k=0}^{\infty} \max\{1 - \omega_j(\mu - k),\, 0\}\, \frac{e^{-\lambda^R} \lambda^{R^k}}{k!},$$

and similarly for the second moment $\mathbf{E}(c_j^2)$.

Figure 4.1 plots the estimated mean and variance values as a function of λ^R. We have used the parameter set $\alpha_1 = 0.7$, $\alpha_2 = 0.3$, $q_1 = q_2 = 450$ (thus giving $\omega_1 > \omega_2$), $\mu = 900$, with λ^R ranging from 10 to 500. Since ξ is Poisson, $\mathbf{E}(\xi) = \mathbf{V}(\xi) = \lambda^R$. We observe that up until $\lambda^R \approx 240$ when $\mathbf{E}(c_1) = 0$, we have $\mathbf{V}(c_1) < \mathbf{V}(c_2)$, mainly due to the fact that $\mathbf{V}(c_1) = 0$ for most of the interval. However, after $\lambda^R > 200$, approximately in tandem with $\mathbf{E}(c_1)$ becoming positive, $\mathbf{V}(c_1) > 0$, and after $\lambda^R > 250$, we have

$$\mathbf{V}(c_1) > \mathbf{V}(c_2)$$

as predicted by the analysis. Notice that the transition is fairly abrupt with $\mathbf{V}(c_1) < \mathbf{V}(c_2)$ holding mostly for the degenerate case when $\mathbf{E}(c_1) = 0$, i.e., $c_1(t) = 0$.

4.2.4 Simulation Results

Set-up This section provides simulation results of noncooperative QoS provision network games, confirming the transition behavior shown in Figure 4.1. We also show the dynamics of a system which has both packet loss rate and end-to-end delay as components in the application QoS requirement vectors.

We implement the network set-up described in Section 4.2.1 with n applications (grouped into several application classes each with a different QoS requirement), m service classes, and cross traffic

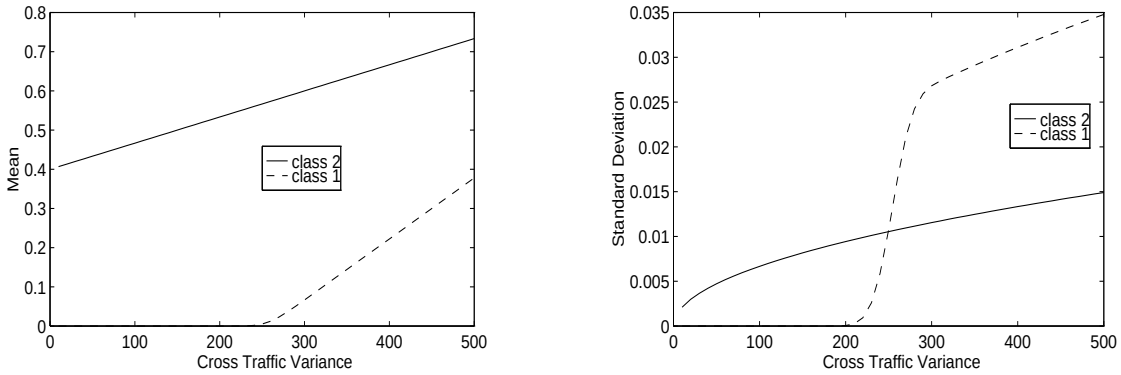

Figure 4.1: Left: Estimated mean packet loss rate of 2-service class system as a function of cross traffic variance λ^R. Right: Estimated standard deviation of packet loss rate of 2-service class system as a function of cross traffic variance.

given by a Poisson process with rate λ^R. In keeping with the ATM framework, we assume fixed-size packets (i.e., *cells*) and we employ output-buffered switches. We implement a generic form of weighted fair queueing achieving perfect isolatedness and conservation of work. We ignore effient implementation considerations of WFQ, treating the processing cost at the switches as fixed. The assumption of fixed-size packets simplifies the faithful rendering of service rates commensurate with the weights $\alpha_1, \ldots, \alpha_m$.

We associate prices $p_1, p_2, \ldots, p_m$ with the service classes, and applications incur a cost of $\lambda_{ij} p_j$ for sending a traffic volume of λ_{ij} tagged by service class identifier $j \in [1, m]$. Each application is assigned a one-time budget of B_i, $i \in [1, n]$, sufficient for the simulation duration. We also assume that assignments are of the type "all in one bag." I.e., $\lambda_{ij} \in \{0, \lambda_i\}$ and $\sum_j \lambda_{ij} = \lambda_i$. The *selfishness* behavior of applications is modeled in the following way. Given application i's QoS requirement vector $\boldsymbol{\theta}^i$, the application seeks out a *cheapest* service class j such that all its QoS requirements are satisfied. That is, $\mathbf{x}^j \leq \boldsymbol{\theta}^i$ and p_j is minimal. Thus, applications are assumed to assign a nonzero utility to "money."

If no such service class j exists—i.e., $\forall j \in [1, m]$, $\mathbf{x}^j \not\leq \boldsymbol{\theta}^i$—then application i submits its traffic to a service class j' that most closely meets its QoS requirements, however, paying a price of $p_{j'} + \delta$ where $\delta > 0$ is a bid parameter. The current price of service classes is continuously updated by the system (realized by a computational market that monitors these events), with the new price p'_j set as the maximum of the bids submitted in the previous round.

The price decrease mechanism is affected in the following way. Let A_j denote the set of application indices $i \in [1, n]$ currently assigned to service class $j \in [1, m]$. Let

$$\boldsymbol{\chi}^j = \frac{1}{|A_j|} \sum_{i \in A_j} \boldsymbol{\theta}^i. \tag{4.14}$$

That is, $\boldsymbol{\chi}^j = (\chi_1^j, \ldots, \chi_s^j)^T$ is the average application QoS requirement vector of applications currently assigned to class j. If $\boldsymbol{\chi}^j - \mathbf{x}^j > \mathbf{0}$ and $\|\boldsymbol{\chi}^j - \mathbf{x}^j\| > \Theta$ where $\Theta > 0$ is a system parameter, then

$$p'_j \leftarrow \max\{\, p_j - \delta,\, 0 \,\}.$$

In other words, the system itself exerts a downward pressure on the price of a service class j if the QoS rendered in the service class—i.e., $\mathbf{x}^j$—is significantly better than the QoS required by the constituent applications. Hence, if the system is underutilized, services are rendered at nominal prices or for free. One may use a number of different norms $\|\cdot\|$ (we have used the sup norm) depending on the QoS vector make-up and the objectives at hand.

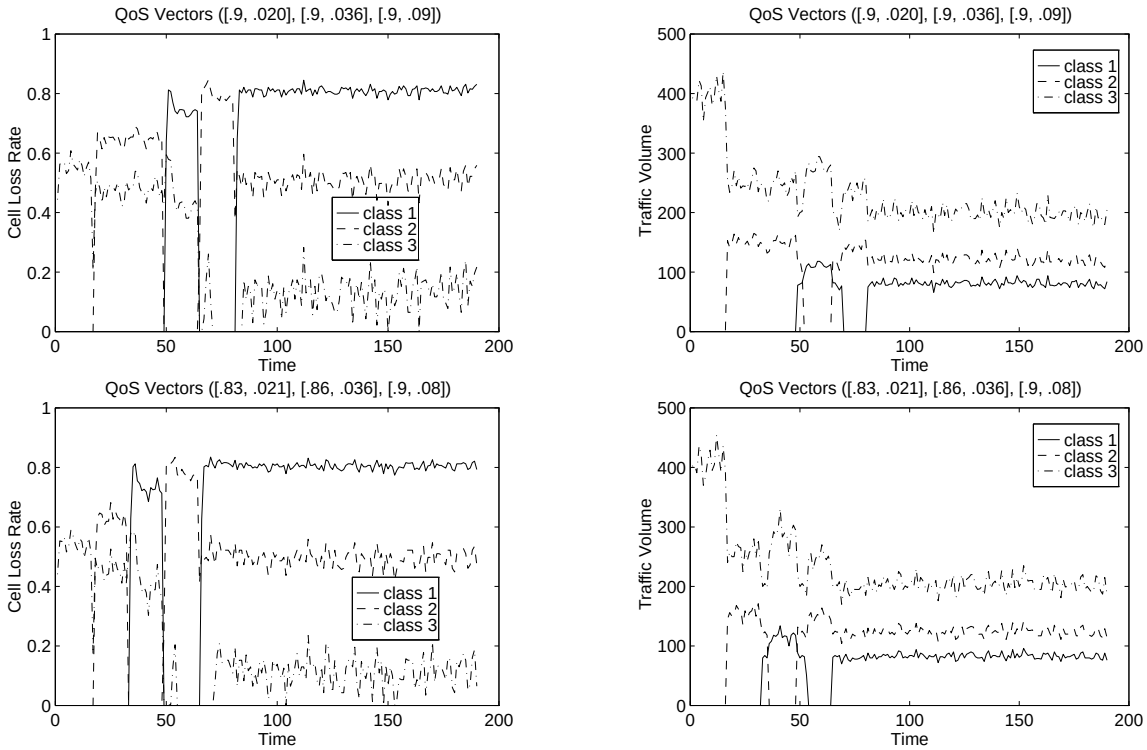

Figure 4.2: Top row. Same packet loss requirement but different variance requirements. Left: Cell loss trace shows inverted ordering where service class with least variance has highest cell loss rate. Right: Corresponding traffic volume trace q_1, q_2, q_3. Bottom row. Total order on both packet loss rate and variance. Cell loss trace with similar reverse ordering phenomenon.

Effect of Burstiness Figure 4.2 shows the trace of two 3-application class/3-service class/2-dimensional QoS vector systems where the components of the QoS vectors are packet loss rate and its standard deviation, respectively.

In the first system, there are 15 applications grouped into three application classes of 5 applications each, where the application class QoS vectors are given by $(0.9, 0.02), (0.9, 0.036)$, and $(0.9, 0.09)$. That is, they all have the same packet loss rate bound 0.9 but different bounds on the standard deviation. In the second system, the same situation holds except that the application class QoS vectors are strictly ordered with values $(0.83, 0.021) < (0.86, 0.036) < (0.9, 0.08)$. The service weights were set at $\alpha_1 = 0.2$, $\alpha_2 = 0.3$, and $\alpha_3 = 0.5$. The application traffic demands λ_i, $i = 1, 2, \ldots, 15$ were set at 85×5, 49×5, and 46×5. The service rate was $\mu = 900$ and the cross traffic rate was $\lambda^R = 500$.

The top row of Figure 4.2 shows the time evolution of packet loss rate in the three service classes for the two systems (top-left and top-right) described above. In each case, we observe that the

applications' bounds on packet loss rate are all satisfied. However, as predicted by the analysis, the application with the most stringent QoS requirement—in both cases requiring a standard deviation bound of 0.02 and 0.021, respectively—ends up receiving the worst *actual* packet loss rate rendered although they are still below the required packet loss rate thresholds.

The bottom row of Figure 4.2 shows the corresponding time evolutions of the traffic assignments to the three service classes. We observe that after a transient period, applications settle into service classes where their QoS needs are met at least cost, and stay so thereafter. That is, the applications have reached a Nash equilibrium which is also system optimal with respect to the resource allocation problem defined in Section 2.

Degenerate Assignment Figure 4.3 shows the trace of a 2-application class/2-service class/2-dimensional QoS vector system with service weights $\alpha_1 = 0.4$, $\alpha_2 = 0.6$. There were a total of 10 applications grouped into two application classes of 5 applications each, with application class QoS requirements $(0.7, 0.01), (0.7, 0.04)$. The traffic volume demands λ_i, $i = 1, 2, \ldots, 10$ were 40×5 and 140×5.

Figure 4.3 (left) shows that service class 1 has both a lower packet loss rate and a lower packet loss standard deviation than service class 2. However, this is only achieved because the packet loss rate for service class 1 is zero or near zero—the degenerate case. Note that in spite of service class 1 having a service weight of $\alpha_1 = 0.4 < \alpha_2$, due to the smaller traffic volume assigned to class 1, $q_1 < q_2$, the normalized service weight satisfies $\omega_1 > \omega_2$ thus explaining the 0 packet loss rate associated with service class 1.

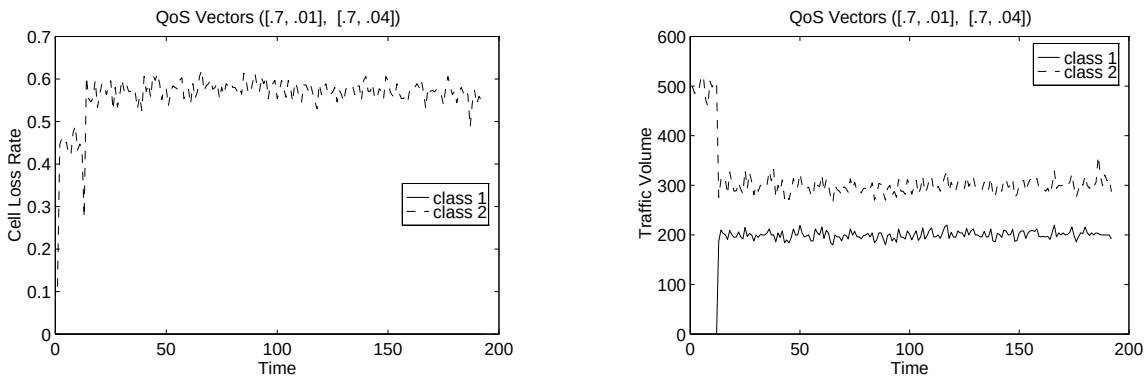

Figure 4.3: Degenerate case where QoS delivered obeys the same order as that required by constituent applications $(0.7, 0.01) < (0.7, 0.04)$. First component is packet loss rate and second component is variance. Left: Shows degenerate QoS rendered for service class 1 where packet loss rate and variance are both 0. Right: Corresponding traffic volume trace.

Packet Loss and End-to-end Delay Figure 4.4 shows the trace of a 3-application class/3-service class/2-dimensional QoS vector system where the QoS vectors now consist of bounds on packet loss rate and end-to-end delay. It is straightforward to verify that unlike in the case of packet loss variance, end-to-end delay moves in the same direction as packet loss rate with respect to the effect of cross traffic variance.

The application QoS vectors are given by $(0.2, 0.8), (0.2, 2)$, and $(0.2, 7)$ where the first component is packet loss rate. The service weights were set at $\alpha_1 = 0.6, \alpha_2 = 0.24$, and $\alpha_3 = 0.16$. The traffic volume demands of the 15 applications were 110×5, 48×5, and 42×5. The delay trace—Figure 4.4 (left)—shows that the end-to-end delay requirements of the three applications are all satisfied. Similarly for the packet loss rate requirement as seen in Figure 4.4 (right). Unlike in the packet loss variance case, however, a total ordering on the rendered QoS of the three service classes is clearly achieved as expected. That is, $\mathbf{x}^1 < \mathbf{x}^2 < \mathbf{x}^3$.

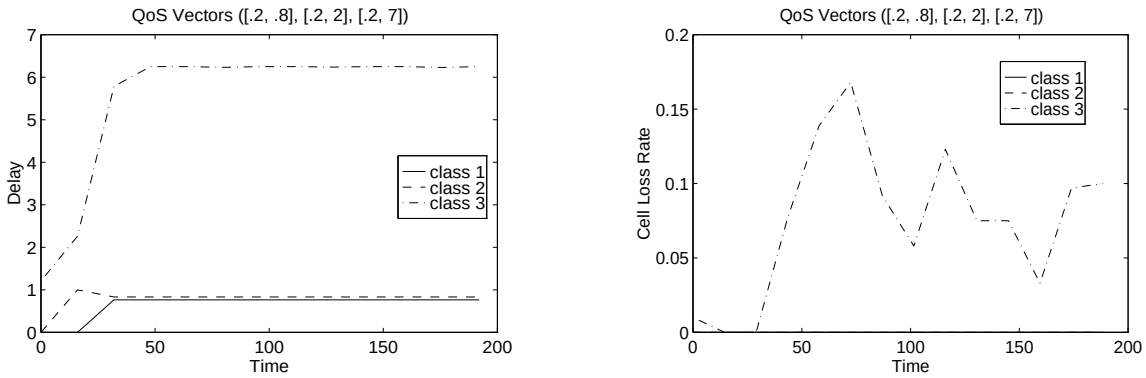

Figure 4.4: 3-service class system with packet loss rate and delay as QoS indicators. Application QoS requirements obey strict ordering $(0.2, 0.8) < (0.2, 2) < (0.2, 7)$. Left: Delay trace showing satisfactory QoS rendered by the service classes. Right: Corresponding packet loss rate trace.

5 Structural Adaptation

5.1 Problem Statement

The network QoS provision problem defined in Section 2 is fundamentally a network resource allocation problem, with noncooperativeness imparting additional dynamic structure, by imposing a constraint on what configurations are considered stable. From a resource allocation point-of-view, it is clear that the maximal total utility $\lambda = \sum_{i=1}^{n} \lambda_i$ (the trivial bound on the total utility $\bar{U}$), in general, may not be achieved by any assignment and similarly for $\bar{U}$.

In fact, for system configurations where resources are heavily taxed, Nash equilibria may not exist (see, e.g., Proposition 3.8) or be difficult to reach, and we are confronted with the potential problem of the system oscillating and erratically switching states which can adversely affect the quality of service rendered to a spectrum of service classes.

Scenario 1 On one hand, this problem may be *intrinsic* in the sense that short of adding more resources such as link capacity and buffer space to the system, the problem cannot be solved. Figure 5.1 shows the time trace of such a system which consists of 10 applications grouped into two application classes—five each—with cell loss requirements of 0.01 and 0.1, respectively. It is a 2-service class system with traffic volumes $\lambda_1, \ldots, \lambda_{10}$, service rate μ, service weights $\alpha_1 > \alpha_2$, and cross traffic rate λ^{NR} set at levels such that the assignment where the 5 applications with a 0.01-cell loss rate requirement go

to service class 1 and the remaining 5 applications with the 0.1-cell loss rate requirement go to service class 2 is not sustainable in the long run.

Figure 5.1 (left) shows the price trace which exhibits an inflationary trend spurned by the competitive bidding by the applications of the structurally scarse resources. Figure 5.1 (middle) depicts the cell loss rate trace of the two service classes which show mean cell loss rates that are below the required levels 0.01 and 0.1, respectively. However, whereas in service class 1 the entire packet loss trace stays strictly below 0.01, for service class 2, this is not the case, with the bound 0.1 being repeatedly violated due to service class 2's structural deficiency at satisfying the QoS requirement of the 0.1-cell loss rate application pool.

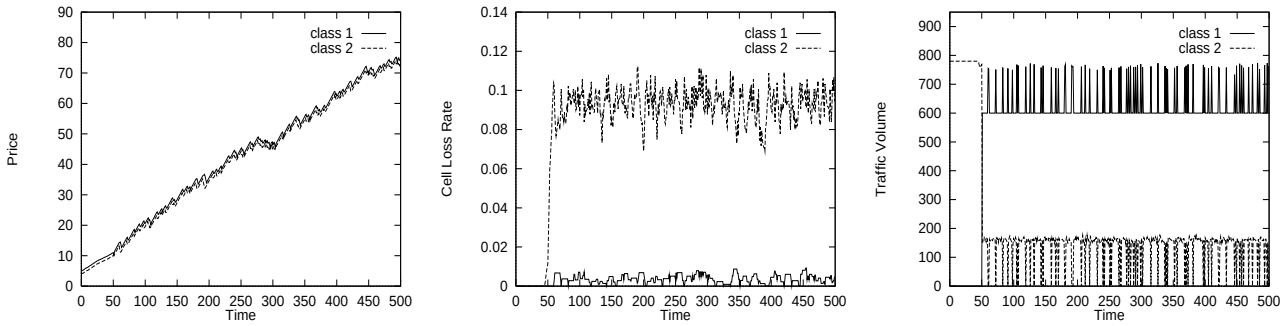

Figure 5.1: Trace of 2-service class system with taxed resources. Left: Inflationary price trend. Middle: Delivered cell loss rate given application requirements 0.01 and 0.1. Right: Traffic volume trace q_1, q_2 assigned to the two service classes.

This, in turn, prompts some of the 0.1-applications to migrate to "greener pastures" (service class 1), triggered by the selfish decision procedure described in Section 4.2.4. The departure of one or more of the five applications[8] has three side effects: one, the cell loss rate c_1 associated with service class 1 increases due to crowding; two, by the same token, the cell loss rate of service class 2 decreases due to the diminished traffic volume q_2 assigned to it; and three, the price p_2 of service class 2 decreases caused by the downward price adjustment mechanism described in Section 4.2.4.

Thus, shortly after migration, service class 2 looks attractive again since its new cell loss rate satisfies the migrated 0.1-cell loss rate applications and $p_2 < p_1$. This leads to a reverse migration back to service class 2, and when c_2 sufficiently rises again, the cycle repeats. This cyclic migration behavior is shown in Figure 5.1 (right) which depicts the traffic volumes q_1, q_2 assigned to the two service classes as a function of time.

Assuming fixed resources, this structural overutilization can only be resolved by either one or more applications departing the network—e.g., induced by budget constraints—or the background traffic changing its characteristics such as a decline in intensity[9].

Scenario 2 On the other hand, although still out of the control of applications, it may be possible that the network is in possession of sufficient resources to accommodate the QoS requirements of

[8]The applications are independent of each other, executing a distributed update mechanism which, in general, leads to asynchronous triggering of migrations, albeit highly correlated.

[9]Due to space limitations, simulation results that illustrate these effects are omitted.

constituent applications, however, is wrongly configured for the demand characteristics at hand. Such is the case, for example, when switches implement weighted fair queueing with service weights $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_m)$. What may be a structural problem for one set of weights $\boldsymbol{\alpha}$ may not be a problem for another set of weights $\boldsymbol{\alpha}'$. Many packet switching disciplines are of such a (potentially) *programmable* nature, and in the context of WFQ, the question arises how the control should be affected. This question is addressed next.

5.2 Service Weight Adaptation

5.2.1 An Example

Before we investigate adaptive strategies for changing $\boldsymbol{\alpha}$ as a function of network state, let us illustrate the potential benefit of changing $\boldsymbol{\alpha}$ using a one-time intervention when it is warranted. Figure 5.2 shows the trace of a 3-service class system with 15 applications divided into three application classes exhibiting cyclical behavior similar to the 2-service class system in Figure 5.1 during time interval $0 \leq t \leq 250$. Application packet loss requirements are 0.01, 0.05, and 0.2, respectively, for the three application classes, with service weights set at $\alpha_1 = 0.6$, $\alpha_2 = 0.3$, and $\alpha_3 = 0.1$.

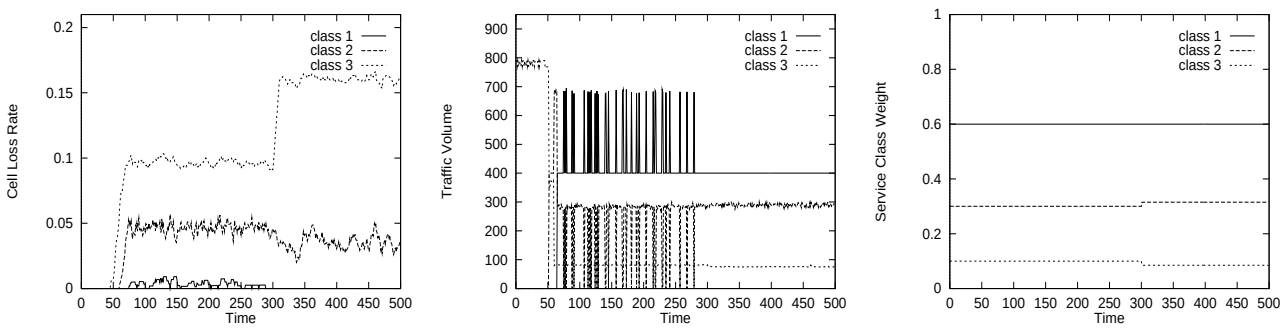

Figure 5.2: Overtaxed 3-service class system. Left: Cell loss trace shows decrease in c_2 (but increase in its variance) and increase in c_3 at time $t = 300$. Middle: Traffic volume trace shows stability after $t = 300$. Right: Service weights are changed at $t = 300$, increasing α_2 and decreasing α_3.

We observe that migration occurs between service classes 1 and 2, with the 0.05-cell loss rate applications migrating back and forth due to reasons analogous to the 2-service class system of Figure 5.1. A distinguishing factor is the presence of 0.2-cell loss rate applications who have settled into service class 3, being happily stationary receiving an actual cell loss rate of 0.1 which is significantly lower than needed.

An impartial network manager, having control over α_1, α_2, α_3, upon observing this phenomenon for a period, may conclude that service class 2 is overutilized while class 3 is underutilized and decide to shift some of the weight in α_3 to α_2, expecting to affect an improvement. That is, the updated service weights are set to

$$\alpha_1' \leftarrow \alpha_1, \qquad \alpha_2' \leftarrow \alpha_2 + \epsilon, \qquad \alpha_3' \leftarrow \alpha_3 - \epsilon,$$

where $0 < \epsilon \leq \alpha_3$ is an adjustment parameter.

Indeed, precisely this is carried out at time $t = 300$ with $\epsilon = 0.015$, yielding $\boldsymbol{\alpha} = (0.6, 0.315, 0.085)$, and its effects are captured in Figure 5.2. We observe that the mean packet loss rate of service class 2 has dropped—its variance, however, has increased as predicted by the analysis in Section 4.2.2—and it has dropped sufficiently to offset the increased variance, resulting in satisfied 0.05-applications with no further incentive to migrate. The mean cell loss rate of service class 3, as expected, has increased. However, its new of value of 0.16 is still lower than the 0.2 requirement, and thus the exchange has resulted in a net increase in total utility—in our case, even in the sense of Pareto since step utility functions are used.

5.2.2 Adaptive Weight Control

The first step in formulating an adaptive control algorithm $\mathcal{G}$ that acts on $\boldsymbol{\alpha}$, $\boldsymbol{\alpha}' \leftarrow \mathcal{G}(\boldsymbol{\alpha})$, is to quantitatively capture whether a service class is under- or over-utilized. Note that something akin to this has already been done in Section 4.2.4 in conjunction with formulating the downward price adjustment mechanism. We defined the vector $\boldsymbol{\chi}^j = (\chi_1^j, \ldots, \chi_s^j)^T$ (see (4.14) of Section 4.2.4) to be the component-wise average of the application QoS vectors currently assigned to service class j. Since $\mathcal{G}$ may act even if $\boldsymbol{\chi}^j - \mathbf{x}^j \not\succ \mathbf{0}$ where $\mathbf{x}^j$ is the vector of actual QoS rendered by service class j, we will no longer use the sup norm to measure their difference.

Let $\bar{\mathbf{x}}^j(t)$ be the local time average of $\mathbf{x}^j(t)$ at time t with memory $\tau > 0$. That is,

$$\bar{\mathbf{x}}^j(t) \;=\; \frac{1}{\tau} \sum_{t-\tau < t' \leq t} \mathbf{x}^j(t').$$

Similarly for $\bar{\boldsymbol{\chi}}^j(t)$. We will drop the time index t for notational clarity. Since varying $\boldsymbol{\alpha}$ represents structural changes above and beyond what application behavior can achieve through competitive interaction, care should be taken not to trigger such changes on the basis of short-term, spurious effects which can render the system unstable. τ allows us to control the degree of responsiveness.

Let

$$d(\bar{\boldsymbol{\chi}}^j, \bar{\mathbf{x}}^j) = \min_{r \in [1,s]} \chi_r^j - x_r^j, \tag{5.1}$$

and let $h_j \in [1, s]$ be a QoS vector component index that satisfies (5.1). It may be that $d(\bar{\boldsymbol{\chi}}^j, \bar{\mathbf{x}}^j) < 0$, and one can choose to define d in a number of different ways depending on what the particular objectives are. Our definition takes a conservative side. If $d(\bar{\boldsymbol{\chi}}^j, \bar{\mathbf{x}}^j) \geq \gamma^*$ for some threshold $\gamma^* > 0$, then we may consider service class j to be underutilized, and vice versa. Let

$$R^+ = \{\, j \in [1, m] : d(\bar{\boldsymbol{\chi}}^j, \bar{\mathbf{x}}^j) \geq \gamma^* \,\}, \qquad R^- = \{\, j \in [1, m] : d(\bar{\boldsymbol{\chi}}^j, \bar{\mathbf{x}}^j) < \gamma_* \,\},$$

where $\gamma_* \leq \gamma^*$, and let $j_- \in R^-$ satisfy

$$d(\bar{\boldsymbol{\chi}}^{j_-}, \bar{\mathbf{x}}^{j_-}) = \min_{j \in R^-} d(\bar{\boldsymbol{\chi}}^j, \bar{\mathbf{x}}^j).$$

That is, R^+ represents the set of underutilized service classes, and j_- represents the most overutilized service class. If either R^+ or R^- is empty, no action is taken by $\mathcal{G}$.

Based upon the example discussed above, we would like to increase α_{j_-} and decrease α_j for some maximal element $j \in R^+$. However, this is not necessarily correct. Since the h_{j_-}'th QoS vector component may represent a bound on a "variance-related measure" (e.g., delay jitter), from Section 4.2.2, we know that increasing α_{j_-} would actually make the variance experienced by service class j_- even worse[10]. Thus, in this case, α_{j_-} would need to be *decreased*. Since the goal is to equilibriate imbalances between service classes in R^- and R^+, this implies that we need to find a maximal element $j \in R^+$ with respect to $d(\bar{\boldsymbol{\chi}}^j, \bar{\mathbf{x}}^j)$ such that it does *not* benefit from an increase in α_j.

Let $V : [1, s] \rightarrow \{0, 1\}$ be an indicator function such that $V(r) = 1$ iff r is a variance-related QoS vector component index. We will assume that such a dual classfication is possible. Let $j_+ \in R^+$ satisfy

$$d(\bar{\boldsymbol{\chi}}^{j_+}, \bar{\mathbf{x}}^{j_+}) = \max_{j \in R^+} \{\, d(\bar{\boldsymbol{\chi}}^j, \bar{\mathbf{x}}^j) : V(h_j) = V(h_{j_-}) \,\}.$$

That is, j_+ is a maximally underutilized service class such that the dominating (with respect to the definition of $d(\cdot, \cdot)$) QoS vector component index h_{j_+} has the same property as h_{j_-}. If there is none, we will let j_+ be the minimal element in R^+ with respect to $d(\bar{\boldsymbol{\chi}}^j, \bar{\mathbf{x}}^j)$.

Now we are ready to state the control law $\mathcal{G}$. Let $\epsilon = \min\{\, \epsilon^*, \alpha_{j_+} \,\}$ where $\epsilon^* > 0$ is an adjustment parameter. Then

$$\alpha'_j \;=\; \begin{cases} \alpha_j, & \text{if } j \in [1, m] \setminus \{j_+, j_-\}, \\ \alpha_j + \epsilon\,(-1)^{V(h_j)}, & \text{if } j = j_-, \\ \alpha_j - \epsilon\,(-1)^{V(h_j)}, & \text{if } j = j_+. \end{cases}$$

Thus the indicator function V is also used in the selection of the sign of ϵ, subtracting ϵ from α_{j_-} if the dominating QoS vector component h_{j_-} is variance-related, and adding ϵ to α_{j_-} if it is not. It is straightforward to check that $\mathcal{G}$ preserves $\sum_{j \in [1,m]} \alpha'_j = 1$ and $\alpha'_j \geq 0$, $j \in [1, m]$.

At every update, a pair of maximal imbalances are reduced—i.e., brought closer—by an amount affected by ϵ. For stationary systems with $\gamma^* - \gamma_* > 0$ where *stationarity* is with respect to $\bar{\boldsymbol{\chi}}^j$ and $\bar{\mathbf{x}}^j$ (itself a function of τ), if ϵ^* is sufficiently small, then $\mathcal{G}$ will typically converge after a finite number of steps. That is, for some $T > 0$,

$$\alpha(t + T) = \alpha(T), \quad t \geq 0.$$

The dynamics of $\mathcal{G}$ may be analyzed using techniques in [34].

Figure 5.3 shows the dynamics of a system under the action of $\mathcal{G}$. It is a 3-service class system with the same system configuration as in Figure 5.2 except that the adaptive control $\mathcal{G}$ is now active. Additional parameter settings of interest are $\tau = 10$, $\gamma^* = \gamma_* = 0.005$, and $\epsilon^* = 0.001$. We observe that $\boldsymbol{\alpha}$ undergoes changes at $t = 76$ and $t = 121$, at both times increasing α_2 at the expense of α_3. The imbalance reduction at $t = 76$ has a positive effect, decreasing service class 2's cell loss rate and quenching the migration when contrasted with the trace shown in Figure 5.2. However, the adjustment proves to be insufficient in the long run registering yet another imbalance at time $t = 120$ which triggers

[10]The degenerate case need not be separately considered since if service class j_- has 0 or close to 0 variance to begin with, then it would most likely not have been chosen to be improved.

a further correction. The cause of the two triggering events can be gleamed from Figure 5.3 (bottom row, left) which depicts traces of $\bar{\mathbf{x}}^j$ (rendered cell loss rate) and $\bar{\boldsymbol{\chi}}^j$ (desired application cell loss rate) for service class $j = 2$ only. At $t = 76$ and $t = 121$, $\bar{\mathbf{x}}^2$ crosses $\bar{\boldsymbol{\chi}}^2$, exceeding the latter, and this, in turn, causes the corrective action by $\mathcal{G}$ reducing the imbalance between utilizations in service class 2 and 3 whose trade-off is clearly beneficial given that $\bar{\boldsymbol{\chi}}^3$ stays well above $\bar{\mathbf{x}}^3$ even after the weight adjustments. The system, having been configured to possess a Nash equilibrium for some service weight vector $\boldsymbol{\alpha}$, and being stationary over the simulation interval, settles down to a steady state which, in this case, is also system optimal.

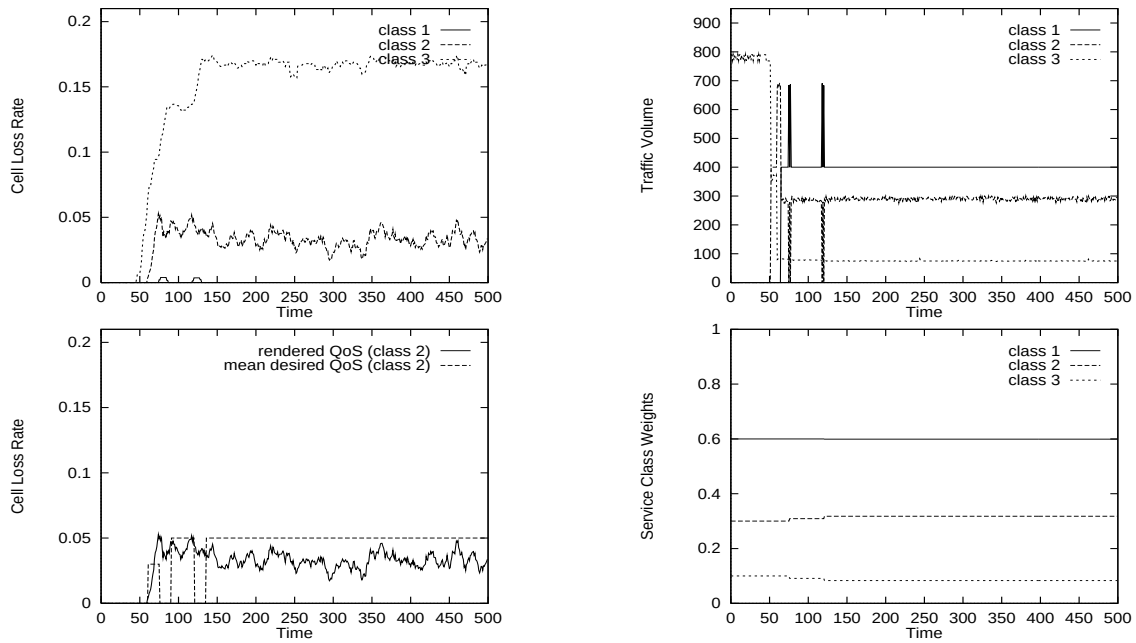

Figure 5.3: Overtaxed 3-service class system. Top row. Left: Cell loss trace. Right: Traffic volume trace exhibits stability after initial transition phase. Bottom row. Left: $\bar{\mathbf{x}}^j$ (rendered cell loss rate) vs. $\bar{\boldsymbol{\chi}}^j$ (desired application cell loss rate) for service class $j = 2$ only. Right: Service weight trace showing changes at $t = 76, 121$.

6 Conclusion

We have presented a study of the quality of service provision problem in noncooperative network environments where applications or users are assumed to behave selfishly. Our framework and its conclusions are best suited—but not exclusively so—for best-effort traffic environments where the network is not required to provide stringent QoS guarantees which can only be accomplished currently by employing conservative resource reservations. Rather, service classes with differentiated QoS levels matching the needs of constituent applications are induced by the latter's selfish interactions, providing reasonably stable QoS levels as a function of network state.

We have formulated a noncooperative QoS provision model and given a comprehensive analysis of its properties. We have shown that Nash equilibria—which correspond to stable fixed points in

noncooperative games—need not be system optimal nor Pareto, in fact, for certain QoS provision systems, Nash equilibria need not even exist. We have given a complete characterization of Nash equilibria and the subset of those that are also system optimal. We have shown that the latter is related to the Pareto optimality of a certain normal form derived from the Nash configuration. For "resource-plentiful" systems, we have shown that Nash, Pareto, and system optima all coincide, and moreover, convergence is monotone and fast if a form of asynchronous self-optimization is used.

We have extended the analysis to QoS provision systems with multi-dimensional QoS vectors containing both mean- and variance-related QoS measures. We have shown that the main results carry over if a uniformity assumption is placed either on application preference thresholds or on QoS vector functions. We have also looked at the impact of multiple QoS measures on the characteristics of induced QoS levels actually rendered by the service classes. We have shown that under bursty traffic conditions, it is impossible for a service class to deliver superior QoS in both mean- and variance-related QoS measures (e.g., mean delay and jitter) vis-à-vis some other service class if weighted fair queueing or other general processor sharing (GPS)-related packet scheduling discipline is used.

Lastly, whereas the aforementioned results dealt with questions concerning properties of system states as determined by the interaction of selfish applications or users, we have also investigated the question of what the system, if anything, can do to enhance network QoS provision performance without adversely affecting individual applications. We have formulated an adaptive service weight control algorithm for GPS-based routers and shown its effectiveness at reducing imbalances between service class utilizations thus increasing total system utility. However, the system-initiated reshuffling of resources is done in such a way so that system utility is improved without sacrificing the utility of individual applications—a form of Pareto optimization.

Current work is directed in two main avenues, one, in the extension of the game-theoretic analysis to arbitrary monotone utility functions which requires further development of analytical tools and techniques, and second, in the study of WAN-scale internetworked systems—a prime target being the realization of such QoS provision systems on the Internet—where, from a technical perspective, the interaction among routers or switches introduces new complexities and a slew of challenging problems.

References

[1] A. Bovopoulous and A. Lazar. Decentralized algorithms for optimal flow control. In *Proc. 25th Allerton Conference on Communication, Control and Computing*, 1987.

[2] A. Campbell, C. Aurrecoechea, and L. Hauw. A review of QoS architectures. To appear in *ACM Multimedia Systems Journal*, 1996.

[3] Scott Clearwater, editor. *Market-Based Control: A Paradigm for Distributed Resource Allocation, Scott Clearwater.* World Scientific, Hong Kong, 1996.

[4] R. Cocchi, S. Shenker, D. Estrin, and L. Zhang. Pricing in computer networks: motivation, formulation, and example. *IEEE/ACM Trans. Networking*, 1(6):614–627, 1993.

[5] R. L. Cruz. A calculus for network delay, part I: network elements in isolation. *IEEE Trans. Inform. Theory*, 37(1):114–131, 1991.

[6] R. L. Cruz. Quality of service guarantees in virtual circuit switched networks. *IEEE J. Select. Areas Commun.*, 13(6):1048–1056, 1995.

[7] G. de Veciana, G. Kesidis, and J. Walrand. Resource management in wide-area ATM networks using effective bandwidths. *IEEE J. Select. Areas Commun.*, 13(6):1081–1090, 1995.

[8] A. Demers, S. Keshav, and S. Shenker. Analysis and simulation of a fair queueing algorithm. *Journal of Internetworking: Res. Exper.*, 1:3–26, 1990.

[9] A. Elwalid and D. Mitra. Effective bandwidth of general Markovian traffic sources and admission control in high speed networks. *IEEE/ACM Trans. Networking*, 1(3), 1993.

[10] A. Elwalid and D. Mitra. Analysis, approximations and admission control of a multi-service multiplexing system with priorities. In *Proc. IEEE INFOCOM '95*, pages 463–472, 1995.

[11] D. Ferguson, C. Nikolaou, J. Sairamesh, and Y. Yemini. Economic models for allocating resources in computer systems. In Scott Clearwater, editor, *Market-Based Control: A Paradigm for Distributed Resource Allocation, Scott Clearwater*. World Scientific, Hong Kong, 1996.

[12] D. Ferguson, C. Nikolaou, and Y. Yemini. An economy for flow control in computer networks. In *Proc. IEEE INFOCOM '89*, pages 110–118, 1989.

[13] D. Ferguson, Y. Yemini, and C. Nikolaou. Microeconomic algorithms for load balancing in distributed computer systems. In *Proc. 8th International Conference on Distributed Computing Systems*, pages 491–499, 1988.

[14] J. W. Friedman. *Game Theory with Applications to Economics*. Oxford University Press, 1986.

[15] L. Georgiadis, R. Guérin, V. Peris, and K. Sivarajan. Efficient network QoS provisioning based on per node traffic shaping. *IEEE/ACM Transactions on Networking*, 4(4):482–501, 1996.

[16] A. Heddaya and K. Park. Parallel computing on high-speed wide-area networks: a pricing policy for its communication needs. In *Proc. 3rd IEEE Workshop on the Architecture and Implementation of High Performance Communication Subsystems*, pages 188–191, 1995.

[17] Man-Tung T. Hsiao and Aurel A. Lazar. Optimal flow control of multi-class queueing networks with decentralized information. In *Proc. IEEE INFOCOM '87*, pages 652–661, 1987.

[18] Y. Korilis and A. Lazar. Why is flow control hard: Optimality, fairness, partial and delayed information. In *Proc. 2nd ORSA Telecommunications Conference*, March 1992.

[19] Y. Korilis and A. Lazar. On the existence of equilibria in noncooperative optimal flow control. *Journal of the ACM*, 42(3):584–613, 1995.

[20] Y. Korilis, A. Lazar, and A. Orda. Architecting noncooperative networks. *IEEE J. Select. Areas Commun.*, 13(7):1241–1251, 1995.

[21] Y. Korilis, A. Lazar, and A. Orda. Achieving network optima using Stackelberg routing strategies. *IEEE/ACM Trans. Networking*, 5(1):161–173, 1997.

[22] H. T. Kung, T. Blackwell, and A. Chapman. Credit-based flow control for ATM networks: credit update protocol, adaptive credit allocation, and statistical multiplexing. In *Proc. SIGCOMM '94*, pages 101–114, 1994.

[23] J. Kurose and R. Simha. A microeconomic approach to optimal resource allocation in distributed computer systems. *IEEE Trans. on Computers*, 38(5):705–717, 1989.

[24] Jim Kurose. Open issues and challenges in providing quality of service guarantees in high-speed networks. *Computer Communication Review*, 23(1):6–15, 1993.

[25] A. Lazar and G. Pacifici. Control of resources in broadband networks with quality of service guarantees. *IEEE Network Magazine*, pages 66–73, October 1991.

[26] W.E. Leland, M.S. Taqqu, W. Willinger, and D.V. Wilson. On the self-similar nature of Ethernet traffic (extended version). *IEEE/ACM Transactions on Networking*, 2:1–15, 1994.

[27] S. Low and P. Varaiya. A new approach to service provisioning in ATM networks. *IEEE/ACM Trans. Networking*, 1(5):547–553, 1993.

[28] S. Low and P. Varaiya. An algorithm for optimal service provisioning using resource pricing. In *Proc. IEEE INFOCOM '94*, pages 368–373, 1994.

[29] R. Nagarajan and J. Kurose. On defining, computing and guaranteeing quality-of-service in high-speed networks. In *Proc. IEEE INFOCOM '90*, pages 2016–2025, 1992.

[30] A. Parekh and R. Gallager. A generalized processor sharing approach to flow control in integrated services networks: the single-node case. *IEEE/ACM Trans. Networking*, 1(3):344–357, 1993.

[31] A. Parekh and R. Gallager. A generalized processor sharing approach to flow control in integrated services networks: the multiple node case. *IEEE/ACM Trans. Networking*, 2(2):137–150, 1994.

[32] K. Park, G. Kim, and M. Crovella. On the relationship between file sizes, transport protocols, and self-similar network traffic. In *Proc. IEEE International Conference on Network Protocols*, pages 171–180, October 1996.

[33] K. Park, G. Kim, and M. Crovella. On the effect of traffic self-similarity on network performance. To appear in *Proc. SPIE International Conference on Performance and Control of Network Systems*, November 1997.

[34] Kihong Park. Warp control: a dynamically stable congestion protocol and its analysis. In *Proc. ACM SIGCOMM '93*, pages 137–147, 1993.

[35] Kihong Park. Self-organized multi-class QoS provision for ABR traffic in ATM networks. In *Proc. 15th IEEE International Phoenix Conference on Computers and Communications*, pages 446–453, 1996.

[36] V. Paxson and S. Floyd. Wide-area traffic: the failure of Poisson modeling. In *Proc. ACM SIGCOMM '94*, pages 257–268, 1994.

[37] J. B. Rosen. Existence and uniqueness of equilibrium points for concave n-person games. *Econometrica*, 33(3):520–534, 1965.

[38] J. Rosenmüller. *The Theory of Games and Markets*. North-Holland, 1981.

[39] J. Sairamesh, D. Ferguson, and Y. Yemini. An approach to pricing, optimal allocation and quality of service provisioning in high-speed networks. In *Proc. IEEE INFOCOM '95*, pages 1111–1119, 1995.

[40] Scott Shenker. Making greed work in networks: a game-theoretic analysis of switch service disciplines. In *Proc. ACM SIGCOMM '94*, pages 47–57, 1994.

[41] H. R. Varian. *Microeconomic Analysis*. Norton Press, 1993.

[42] N. N. Vorob'ev. *Game Theory*. Springer-Verlag, 1977.

[43] C. Waldspurger, T. Hogg, B. Huberman, J. Kephart, and W. Stornetta. Spawn: a distributed computational economy. *IEEE Trans. Software Engineering*, 18(2):103–117, 1992.

[44] Michael P. Wellman. A market-oriented programming environment and its application to distributed multicommodity flow problems. *Journal of Artificial Intelligence Research*, 1:1–23, 1993.

[45] Michael P. Wellman. Market-oriented programming: some early lessons. In S. H. Clearwater, editor, *Market-based control: a paradigm for distributed resource allocation*, chapter 4. World Scientific, 1995.

[46] W. Willinger, M. Taqqu, R. Sherman, and D. Wilson. Self-similarity through high-variability: statistical analysis of Ethernet LAN traffic at the source level. In *Proc. ACM SIGCOMM '95*, pages 100–113, 1995.

[47] Yechiam Yemini. Selfish optimization in computer networks. In *Proc. IEEE Conference on Decision and Control*, pages 281–285, 1981.