Toward a Theory of Differentiated Services

Huan Ren   Kihong Park
Network Systems Lab
Department of Computer Sciences
Purdue University
West Lafayette, IN 47907
{renh,park}@cs.purdue.edu

Abstract— **Architecting networks capable of providing scalable, efficient, and fair services to users with diverse QoS requirements is a challenging problem. The differentiated services framework has advanced a set of building blocks comprised of per-hop and access point behaviors with the aim of facilitating scalable services through aggregate-flow control inside the network and per-flow traffic control at the edge. In spite of recent efforts, little is known about how to select "good" per-hop and edge controls, in part, due to a lack of cohesive criteria with respect to which the choices can be effectively reasoned, evaluated, and justified.**

In this paper, we provide a theoretical framework for reasoning about differentiated services networks, constrained to be implementable in IP networks. The control framework incorporates assumptions, albeit weak, about selfish user behavior and service provider behavior. This is necessitated by the essential role they play in influencing end-to-end QoS, without which an effective evaluation of Diff-Serv architectures remains incomplete. We show that there is an intimate relationship between the properties exported by per-hop and edge control, and the "goodness" of the resource allocation and QoS attained in a noncooperative network environment.

Our control framework—Scalar QoS Control—generalizes per-hop and edge control achievable by setting a scalar value in packet headers, e.g., the TOS field of IP. We develop a theory of optimal classifiers and the properties they exhibit which facilitate end-to-end QoS via the joint action of aggregate-flow control per-hop and per-flow control at the edge. We show the stability and efficiency properties of the overall network system when users are allowed to influence the choice of scalar values in the DS field at the edge, and service providers export costs to users commensurate with the QoS received.

I. INTRODUCTION

A. Motivation

Architecting networks capable of providing scalable, efficient, and fair services to users with diverse QoS requirements is a challenging problem. The traditional approach uses resource reservation and admission control to provide both *guarantees* and *graded* services to application traffic flows. Analytical tools for computing and provisioning QoS guarantees [1], [2], [3], [4] rely on overprovisioning coupled with traffic shaping/policing to preserve well-behavedness properties across switches that implement a form of generalized processor sharing packet scheduling. For applications needing guaranteed services, the unconditional protection afforded by per-flow resource reservation and admission control is a necessity. For the population of elastic applications that require QoS-sensitive services but not guarantees, it would be overkill to provision QoS using the mechanisms of per-flow reservation and admission control. In addition to the service mismatch, overhead associated with administering resource reservation and admission control which require per-flow state at routers impedes scalability. On the other hand, relying on homogenous best-effort service, characteristic of today's Internet, would be equally unsatisfactory.

This research is supported by NSF grants ANI-9875789 (CAREER) and EIA-9972883.

K.P.: Contact author; tel.: (765) 494-7821, fax.: (765) 494-0739. Additionally supported by NSF grants ANI-9714707 and ESS-9806741, and grants from PRF, Santa Fe Institute, and Sprint.

Recently, efforts have been directed at designing network architectures with the aim of delivering QoS-sensitive services by introducing weaker forms of protection or assurance to achieve scalability [5], [6], [7], [8], [9]. The differentiated services framework [10], [6], [11], [9] has advanced a set of building blocks comprised of per-hop and access point behaviors with the aim of facilitating scalable services through aggregate-flow resource control inside the network and per-flow traffic control at the edge. By performing a many-to-one mapping, as flows enter the network, from the large space of individual flows to the much smaller space of aggregate flow labels, scalability of per-hop control is achieved while at the same time introducing uncertainty and volatility by flow-aggregation and aggregate-flow packet switching per-hop.

B. Key Issues

A number of works have studied the behavioral characteristics of specific instances of differentiated services networks. In previous work [5], [12], [13], we introduced aggregate-flow per-hop control mechanisms motivated by game theoretic considerations—a router performs class-based label switching which emulates user optimal service class selection with respect to selfish users—without considering the space of all aggregate-flow per-hop controls which is carried out in this paper. In [14] simplified models of Assured Service [11] and Premium (or Expedited) Service [15] are presented and analyzed with respect to their performance when compared with simulations. In [16], an adaptive 1-bit marking scheme is described, and the resulting bandwidth sharing behavior demonstrated via simulations when the priority level is controlled end-to-end. In [7], the authors describe the proportional differentiation model which seeks to achieve robust, configurable service class separation—i.e., QoS differentiation—with the support of two candidate packet schedulers. They use simulation to study the behavioral properties. Other related works include [5], [6], [8], [17].

In spite of these efforts, a comprehensive understanding of the power and limitation of differentiated services networks is still in its infancy. Little is known about how to select "good" aggregate-flow per-hop controls—including optimal ones—per-flow end-to-end (or edge) controls, and what criteria to apply when designing these components. Following the divide-and-conquer approach to network design, we would like to reduce the scalable QoS provisioning problem to subproblems and solve them individually without worrying about the details of other subsystems except through well-defined interfaces and "black box" function definitions. Although the same approach is undertaken in this work, we find that there are intimate relationships between the selection of per-hop and end-to-end controls, on the one hand, and the dynamics of a differentiated services network when driven by selfish users and service providers, on the other. The efficiency and stability of noncooperative network systems is influenced by the properties of the per-hop and edge controls, and this dependence necessitates the joint consideration of network mechanism selection and user behavior in an expanded framework within which the relevance of per-hop and edge control properties can be evaluated. The two key focus points of this paper are: (1) formulation and solution of optimal per-hop and edge controls for differentiated services networks, first, *without* regard to user behavior issues, and (2) relating the network control properties to the dynamics of the system when engaged in a noncooperative network environment with respect to efficiency and stability.

C. New Contributions

Our contributions are twofold. First, we give a general framework of differentiated services networks where packet labels can be set from a finite label set and routers provide differentiated treatment of packets based on the labels enscribed. We define the meaning of optimal per-hop control within this context and find the optimal solution for aggregate-flow control. We show that the optimal per-hop control satisfies certain properties—denoted (A1), (A2), and (B), and defined in Section II-C—which relate how label values impact the service a flow receives at a router. We augment the general result by presenting optimal solutions when restricting the packet scheduling disciplines to variants of GPS, and the consequences on the core properties.

Second, we expand the framework by introducing selfish users who can influence QoS provisioning behavior by regulating the label values assigned to their traffic streams. Based on the properties exported by the network control—(A1), (A2), and (B)—we show how a population of selfish users with diverse QoS requirements setting their packet labels can arrive at a global allocation of resources that is *stable* (Nash equilibrium) and *efficient* (system optimal). We show that even in situations when network resources are scarce such that no resource allocation—differentiated service, per-flow reservation, or otherwise—can satisfy all users' QoS requirements, the system is stable and reaches a Nash equilibrium. We show that the optimal per-hop control is also "optimal" in the noncooperative game context in the sense that when network resources are configurable such that all users' QoS requirements can be satisfied, then there exists a Nash equilibrium that is system optimal. We augment the user control results by introducing a selfish service provider who is able to export specific costs—i.e., prices—to users commensurate with the general requirement that a superior QoS (and thus greater resource consumption) incurs a higher cost than a lower QoS (and thus smaller relative resource usage)[1].

II. Architecture and Modeling Assumptions

A. Overall System Structure

The network system is comprised of four principal components—*per-hop control*, *edge control*, *user control*, and *service provider control*—where the first two make up the network system proper, and the latter two are incorporated to evaluate the "goodness" of the first two components. Figure II.1 depicts the overall system structure. A user's traffic flow, upon entering the network, is assigned a label from a set of L values, e.g., enscribed in the TOS field of IPv4. The routers provide differentiated treatment of packets based on their enscribed labels, and end-to-end QoS is determined by the treatment of an user's flow on all hops along a given path. The label values are set at the edge on a per-flow basis—either once-and-for-all (open-loop), or dynamically as a function of network state (closed-loop)—facilitating end-to-end control as part of edge control. A second component of edge control is *access control* which prevents users from arbitrarily assigning labels to their packet flows without consequences. Access control may be achieved by policing, traffic shaping, and pricing. We assume that the network (in general, service provider) exports a cost to each user which increases with service quality, or equivalently, with the resources received. The system is completed by incorporating selfish users who can regulate the label values on their packet streams to satisfy their QoS requirements at least cost, and a selfish service provider who sets prices—which determines user cost—to maximize profit.

The job of the network system proper—per-hop control and edge-control—is to provide sufficient and efficient network mechanisms such that for a set of users or traffic flows with diverse QoS requirements, by suitable setting of the packet labels, user-specified services in the form of *target end-to-end QoS* can be provided. The setting of the label value, whether it is done by access control on behalf of a user or by a user directly, should be powerful enough so that the users' QoS requirements can be satisfied without necessitating the engagement of other traffic controls to the extent possible[2].

[1] We omit the service provider results due to space constraints. The full paper, including the proofs, is available as a technical report [18].

[2] If an end-to-end delay of 30ms is desired but the route assigned has a propagation latency of 50ms, then clearly no amount of class-based label switching

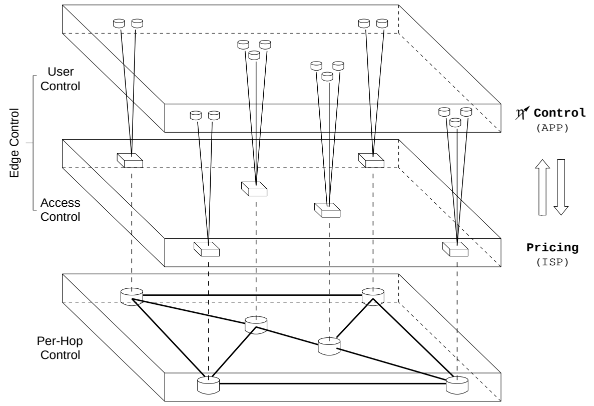

Fig. II.1. Overall QoS provisioning architecture. Network exports per-hop and edge control, user exercises scalar QoS control (η-control), and service provider exports QoS cost to user.

The network control substrate should also promote stability in a noncooperative network habited by selfish users and service providers, and facilitate efficient allocation of network resources as an outcome of selfish interactions.

B. Basic Definitions

Assume there are n flows or users. A user $i \in [1, n]$ sends a traffic stream at average rate $\lambda_i \geq 0$ (bps). In the following, we will assume λ_i is given and fixed ("fixed bandwidth demand"). The case when λ_i is variable ("variable bandwidth demand") is considered separately. Let $\mathbf{x}^i = (x^i_1, x^i_2, \ldots, x^i_s)$ denote the vector of end-to-end QoS rendered to user i. For example, x^i_1 may represent mean delay, x^i_2 packet loss rate, x^i_3 delay jitter (e.g., as measured by some second-order statistic), and so forth. We assume that all QoS measures are represented such that a smaller magnitude means better QoS. A packet belonging to user i is enscribed with a scalar

$$\eta_i \in \{1, 2, \ldots, L\}$$

taking on L distinct values. Unless otherwise specified, we will use $[a, b]$, for $a \leq b$, to denote the set of integers between a and b. Typically, the number of users is very large vis-à-vis the range of η_i, i.e., $n \gg L$, and per-flow identity—as conveyed by η_i—is lost as soon as a packet enters the network. Thus by the many-to-one mapping implied by $n > L$, *aggregate-flow QoS control* is imposed on per-hop behavior and executed per-hop at routers on an end-to-end path. In our implementation design [19], we use a number of bits in the DS field of IPv4 (and IPv6) to carry the η value (i.e., DSCP).

C. Per-hop Control

C.1 Per-hop Control Components

Per-hop control consists of a *classifier* and a *packet scheduler*. We assume a GPS packet scheduler with m service classes and service weights $\alpha_k \geq 0$, $\sum_{k=1}^m \alpha_k = 1$, for an output port whose link bandwidth μ is shared in accordance with the service weights. It is not necessary to have GPS as the underlying packet scheduling discipline—e.g., priority queues, multiple copies of RED with different thresholds are alternatives—but we will show that GPS has certain desirable properties when considering the problem of selecting an optimal aggregate-flow per-hop control for differentiated services. An important component is the classifier which is given by a map $\xi : [1, L] \to [1, m]$. That is, n flows—effectively L (or less) flows from the router's perspective since packets are scheduled by their label values only—routed to the same output port on a switch are mapped to m service classes. For *aggregate-flow* control, $n > L$ and $L \geq m$. Thus

$$n > L \geq m,$$

and if $L > m$, this leads to a further aggregation per-hop in addition to the many-to-one mapping exercised at the edge due to $n > L$. For some choice of classifier and packet scheduler, the QoS received by flow $i \in [1, n]$ at a switch is determined—explicitly or implicitly—by a performance function x^i, $\mathbf{x}^i = x^i(\boldsymbol{\eta}, \boldsymbol{\lambda})$, where $\boldsymbol{\eta} = (\eta_1, \ldots, \eta_n)$ and $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_n)$. More precisely, flow i's performance, in the aggregate-flow case, is determined by the performance function $x^k(\boldsymbol{\eta}^a, \boldsymbol{\lambda}^a)$ associated with service class $k \in [1, m]$ where

$$k = \xi(\eta_i), \quad \boldsymbol{\eta}^a = (1, 2, \ldots, L), \quad \boldsymbol{\lambda}^a = (\lambda^a_1, \lambda^a_2, \ldots, \lambda^a_L),$$
$$\text{and} \quad \lambda^a_\ell = \sum_{j : \eta_j = \ell} \lambda_j.$$

That is, the switch sees only (up to) L "super users" (or aggregate flows). With a slight abuse of notation, we will denote an aggregate flow at a switch by the index i, and $\boldsymbol{\eta}^a$, $\boldsymbol{\lambda}^a$ by $\boldsymbol{\eta}$, $\boldsymbol{\lambda}$ without the superscript. The distinction will be clear from the context.

C.2 Per-hop Control Properties

There are three properties of the per-hop control, listed below, which are of interest and deemed desirable from a QoS control perspective. Let $\mathbf{e}_i = (0, \ldots, 0, 1, 0, \ldots, 0)$ denote the unit vector whose i'th ($i \in [1, n]$) component is 1, and 0, otherwise. In the following, $i \in [1, n]$ refers to the end user, and $x^i(\cdot)$ denotes the individual user's performance function induced by the performance function of the service class that the user is mapped to by ξ. The properties are:

- (A1) for each flow i and configuration $\boldsymbol{\eta}$, $x^i(\boldsymbol{\eta} + \mathbf{e}_i) \leq x^i(\boldsymbol{\eta})$ and $x^i(\boldsymbol{\eta} - \mathbf{e}_i) \geq x^i(\boldsymbol{\eta})$;
- (A2) for any two flows $i \neq j$ and configuration $\boldsymbol{\eta}$, $x^j(\boldsymbol{\eta} + \mathbf{e}_i) \geq x^j(\boldsymbol{\eta})$ and $x^j(\boldsymbol{\eta} - \mathbf{e}_i) \leq x^j(\boldsymbol{\eta})$;
- (B) for two flows $i \neq j$ and configuration $\boldsymbol{\eta}$, $\eta_i \geq \eta_j$ implies $x^i(\boldsymbol{\eta}) \leq x^j(\boldsymbol{\eta})$.

In the definitions, the range of $\boldsymbol{\eta}$ is such that the perturbations remain in the n-dimensional lattice, i.e., $\boldsymbol{\eta} + \mathbf{e}_i, \boldsymbol{\eta} - \mathbf{e}_i \in$

can achieve the target QoS.

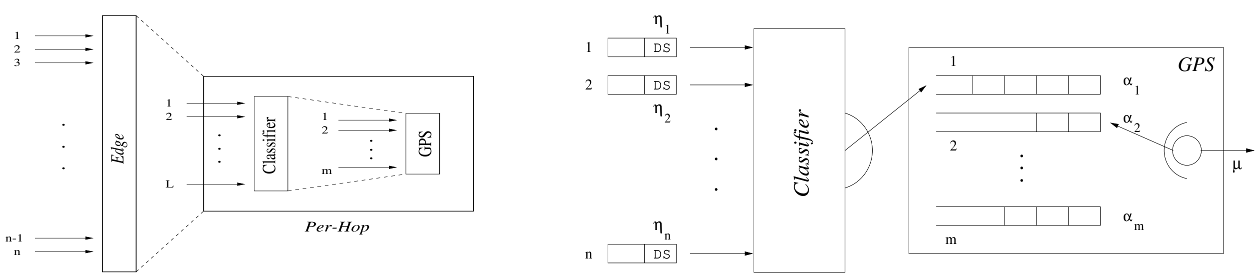

Fig. II.2. Left: Aggregate-flow QoS control affected by two stages of "information loss" via many-to-one coarsification—at edge and per-hop. Right: η value in DS field of IP datagram is used by the classifier to select service class in GPS packet scheduler.

$[1, L]^n$. Property (A1) states that, other things being equal, increasing the label value of flow i improves the QoS received by flow i (recall that "small" means "better" QoS in our representation). Property (A2) states that increasing η_i will not increase the QoS received by any other flow j. Property (B) states that if flow i has a higher η value than flow j, then the QoS it receives is superior to that of flow j. We call property (B) the *differentiated service* property. Note that (B) has the immediate consequence $x^i(\boldsymbol{\eta}) = x^j(\boldsymbol{\eta}) \Leftrightarrow \eta_i = \eta_j$. Thus there is no absolute, a priori QoS level attached to the η_i values. It is the magnitude of η_i—*relative* to other flows' label values—that will determine the QoS received by a flow i. We will show that the three properties, collectively, facilitate effective QoS differentiation and control via η control—i.e., scalar QoS control—and furthermore, allow selfish users to share resources efficiently when setting their η values commensurate with their QoS requirements.

D. Edge Control

D.1 Access Control

The properties exported by per-hop control—if satisfied—are not sufficient by themselves to render end-to-end QoS commensurate with user requirements. End-to-end (or edge) control complements per-hop control by setting the value of η per-flow in accordance with user needs. We assume that the network exercises *access control* at the edge such that users are not permitted to assign η values to their packets at will—if every user assigns the maximum η value L to their flows, then QoS control via η loses its meaning (degenerates to FIFO-based best-effort service by property (B)). This can be done by performing per-flow policing, traffic shaping, or assigning costs via pricing. Open-loop control is used in the Assured Service and Expedited Service instantiations of differentiated services—also called *absolute* differentiated services [7]—and is generally suited for short-lived flows for which feedback control, when subject to long round-trip times (RTT), is ineffective. Figure II.3 depicts the overall structure of the end-to-end control framework.

D.2 End-to-end Control

Our framework (also referred to as *relative* differentiated services in [7]) allows end-to-end control to dynamically adjust the η value in accordance with a user's QoS needs. Properties (A1), (A2), and (B) admit to composability in a WAN environment where a user's traffic flow goes through several hops along an end-to-end path. That is, if a property holds for any single per-hop control, it also holds for a sequence of per-hop controls in a network of switches when viewed as implementing a composite performance function[3]. An end-to-end control of the form

$$\eta_i(t+\tau) = \begin{cases} \eta_i(t)+1, & \text{if } \mathbf{x}^i > \boldsymbol{\theta}^i, \\ \eta_i(t)-1, & \text{if } \mathbf{x}^i < \boldsymbol{\theta}^i, \\ \eta_i(t), & \text{otherwise}, \end{cases} \tag{II.1}$$

where $\boldsymbol{\theta}^i$ represents user i's QoS requirement vector—i.e., expressed as a threshold with delay less than θ_1^i, packet loss rate less than θ_2^i—and $\tau > 0$ represents the next update, is asymptotically stable with respect to a *single* user[4]. Properties (A2) and (B) reflect the *resource-boundedness* property of a router, and come into play when considering a collection of selfish users engaged in end-to-end scalar QoS control, and the dynamics this induces as a result of interaction.

E. User Control

E.1 User Utility and Selfishness

User i's QoS requirement can be represented by a *utility function* U_i which has the form $U_i(\lambda_i, \mathbf{x}^i, p_i)$ where λ_i is the traffic rate, $\mathbf{x}^i$ the end-to-end QoS received, and p_i the unit price charged by the service provider. The total cost to user i is given by $p_i \lambda_i$. We assume that U_i satisfies the *monotonicity* properties[5]

$$\partial U_i/\partial \lambda_i \geq 0, \; \partial U_i/\partial \mathbf{x}^i \leq 0, \text{ and } \partial U_i/\partial p_i \leq 0. \tag{II.2}$$

Other things being equal, an increase in the traffic rate is favourably received by a user, so is an improvement in QoS, but an increase in the price charged by the service provider has a detrimental effect on user satisfaction. These are minimal,

[3] In general, under flow conservation for (A1) and (A2), or certain packet loss dominance conditions.

[4] This assumes a total order on the union of reachable and required QoS vectors. See [20] for a discussion of QoS ordering.

[5] U_i need not be differentiable, nor even be continuous. We use continuous notation here for notational clarity; monotonicity is the only property required.

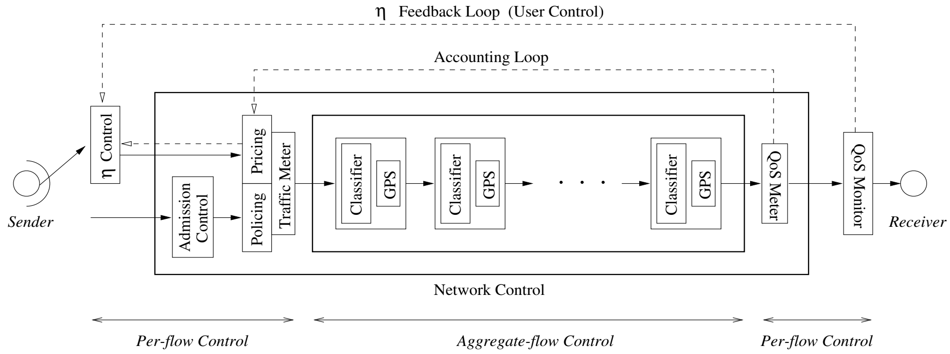

Fig. II.3. Structure of forward QoS control path. "Lower" path comprised of admission control, policing/shaping, per-hop control—open-loop control. "Upper" control path comprised of dynamic η control, pricing, receiver QoS monitoring, QoS feedback—closed-loop control.

weak requirements on the qualitative form of user utility. If η control is allowed to be exercised by the user, then a *selfish* user i can be defined as performing the self-optimization

$$\max_{\eta_i \in [1,L]} U_i(\lambda_i, \mathbf{x}^i, p_i) \tag{II.3}$$

where η_i influences user i's utility U_i via its effect on the QoS received $\mathbf{x}^i$. We assume $p_i(\mathbf{x}^i)$ is a monotone (nonincreasing) function of $\mathbf{x}^i$ which corresponds to the price function exported by the service provider. A slightly different formulation of selfish, "cost-conscious" user behavior is obtained by the constrained optimization formulation

$$\min_{\eta_i} \ \lambda_i p_i(\mathbf{x}^i) \tag{II.4}$$
$$\text{subject to} \quad \mathbf{x}^i \le \boldsymbol{\theta}^i$$

where $\boldsymbol{\theta}^i$ is user i's QoS requirement vector. Thus the user wants to minimize cost—i.e., achieve efficient resource allocation—while satisfying his QoS requirements. Threshold utilities expressed as bounds on the QoS received is a useful means of representing and conveying a user's QoS requirement—delay less than 33ms, packet loss rate less 10^{-4}, jitter less than 3ms, and so forth. The user is asked to convey her QoS preference as a quantifiable threshold when interacting with the network system (e.g., through a Web browser interface) which is employed in some practical systems [21].

E.2 Noncooperative Game

User i's QoS is influenced by the actions (η_j values) of other users $j \neq i$ via $\mathbf{x}^i = x^i(\boldsymbol{\eta})$ as captured by properties (A2) and (B). If all users engage in self-optimization, this leads to a *noncooperative game*. The first point-of-interest is *stability*. In a noncooperative game, a configuration $\boldsymbol{\eta} = (\eta_1, \ldots, \eta_n)$ which determines the global QoS allocation is stable if no user, under (unilateral) selfish actions, can improve her utility from that achieved at $\boldsymbol{\eta}$. More precisely, $\boldsymbol{\eta}$ is a stable configuration or *Nash equilibrium* if for all users $i \in [1, n]$,

$$U_i(\lambda_i, x^i(\boldsymbol{\eta} + c\,\mathbf{e}_i), p_i(\boldsymbol{\eta} + c\,\mathbf{e}_i)) \le U_i(\lambda_i, x^i(\boldsymbol{\eta}), p_i(\boldsymbol{\eta})) \tag{II.5}$$

for all $c \in \mathbb{Z}$ such that $\eta_i + c\,\mathbf{e}_i \in [1, L]$. Since all users are stuck at $\boldsymbol{\eta}$ with respect to selfish moves, the system finds itself at an impasse, i.e., rest point. A similar characterization holds for (II.4). Existence of Nash equilibria and their efficiency properties are of import since they characterize the behavioral aspect of a differentiated services network when put into action in a noncooperative environment such as the Internet. We will show that the global resource allocation properties in a noncooperative network environment are intimately tied with the properties exported by the per-hop control.

F. Service Provider Control

For a single router *shared* by flows i and j, the only pricing constraint we impose is

$$\mathbf{x}^i \le \mathbf{x}^j \Rightarrow p_i \ge p_j. \tag{II.6}$$

That is, the better the QoS received at a shared resource (i.e., router), the higher the per unit flow cost charged to the user receiving superior QoS. Since $\mathbf{x}^i \le \mathbf{x}^j$ if, and only if, the relative resources (in the present framework, bandwidth) allocated to flow i is greater than that of flow j, relation (II.6) just says that the more resources a flow consumes—thus receiving superior QoS—the higher the cost it incurs vis-à-vis a flow that consumes comparatively less resources. Relation (II.6), due to its generality, leaves open the degree of freedom of setting the *magnitude* of the prices which we assume is under the control of a service provider. The service provider can be treated as yet another player in the game—assigned the index zero—and, if selfish, will try to maximize his individual utility U_0. U_0 is assumed to have the form of revenue minus cost (i.e., profit) given by $U_0(\boldsymbol{\eta}, \boldsymbol{\lambda}) = \sum_{i=1}^n \lambda_i p_i(\mathbf{x}^i) - Cost_0$ where $Cost_0$ is the total cost incurred by the service provider in delivering the services. The service provider exports a *price function* $p = p(\mathbf{x})$ where $p(\cdot)$ is monotone decreasing in $\mathbf{x}$. Thus a selfish service provider performs the self-optimization

$$\max_{p(\cdot)} \sum_{i=1}^n \lambda_i p_i(\mathbf{x}^i) \tag{II.7}$$

assuming fixed $Cost_0$. "Closing" the system by incorporating the actions of a selfish ISP leads to a $(n+1)$-player noncooperative game.

III. Optimal Classifiers and Per-hop Control

We take a reductionist approach to optimal aggregate-flow per-hop control by first defining what optimal *per-flow* control is when packets are enscribed with a value from L possible choices. Aggregate-flow control can then be viewed as an *approximation* to the QoS achieved by per-flow control in a well-defined sense. Comparability between aggregate-flow and per-flow control is facilitated by the fact that, even in aggregate-flow control, an end user's QoS remains well-defined, and the loss in power due to coarsification affected by flow aggregation can be exactly quantified.

A. Optimal Per-flow Classification

Consider the per-flow control or classifier problem for n users who choose packet labels from $[1, L]$. Technically, per-flow classification means $n = m$ (each flow's service can be individually configured), and L is either greater or smaller than n. The range L may be finite or unbounded, and the variable $\eta_i \in [1, L]$ discrete or continuous. The influence of *boundedness* and *discreteness* can be subtle, and its effect is shown in Section IV with respect to system optimality of Nash equilibria where we quantify the negative performance impact of boundedness and discreteness affected by loss of resolution. When n users mark their flows with a value $\eta_i \in [1, L]$ drawn from the metric space $[1, L]$ with property (A1) satisfied—larger η_i values, other things being equal, result in a greater apportionment of resources and thus better QoS—η_i can be viewed as codifying a user's QoS or resource demand with respect to some measurement unit. For example, η_i may represent bandwidth demand in units of Mbps. If network resources are *infinite*, then a flow's request can be satisfied based on the η_i value specified, without consideration of the needs specified by other flows (except, possibly, for pricing issues). That is, independence or decoupling holds. If, on the other hand, resources are *finite*—an OC-12 link is shared among bandwidth intensive users—then, in general, the users' collective resource demand may exceed the available bandwidth. In the presence of such *resource contention*, a conflict resolution scheme is needed, including the criteria by which resource allocation is decided.

Assume available bandwidth is normalized such that total available bandwidth is $\mu = 1$. First, assume $\eta_i \in \mathbb{R}_+$ is a *continuous* variable over the real unit interval $[0, 1]$, expressing user i's normalized bandwidth demand *per unit flow*. Let $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$ with $\alpha_i \geq 0$, $\sum_{k=1}^{n} \alpha_k = 1$, represent the fraction of resources apportioned by the per-flow classifier to $i \in [1, n]$, and let $\omega_i = \alpha_i / \lambda_i$ denote the fraction of resources allocated to i per unit flow. Under the above *semantics*, given $\boldsymbol{\eta}$ (and $\boldsymbol{\lambda}$), the optimization

$$\min_{\boldsymbol{\alpha}} \sum_{i=1}^{n} (\eta_i - \omega_i)^2 \tag{III.1}$$

measures the "goodness" of a resource allocation $\boldsymbol{\omega}$ with respect to users' codified needs $\boldsymbol{\eta}$ in the mean-square sense[6]. Since (III.1) penalizes by the *difference* error, the relative importance of higher η_i values is preserved, and resources are apportioned accordingly. For general $\eta_i \in \mathbb{R}_+$, including the discrete and bounded case $\eta_i \in \{1, \ldots, L\}$ which is of special interest, define the normalization

$$\hat{\eta}_i = \begin{cases} \frac{\eta_i - \eta_{\min}}{\eta_{\max} - \eta_{\min}}, & \text{if } \eta_{\max} \neq \eta_{\min}, \\ 1, & \text{otherwise,} \end{cases} \tag{III.2}$$

where $\eta_{\min}$, $\eta_{\max}$ are the minimum and maximum values of $\{\eta_1, \eta_2, \ldots, \eta_n\}$, respectively. Note that $\hat{\eta}_i \in [0, 1]$, and unless all η_i values are equal, $\eta_{\min} = 0$ and $\eta_{\max} = 1$. Let $\hat{\omega}_i$ denote the normalization of ω_i via (III.2). Given $\boldsymbol{\eta}$, the optimization corresponding to (III.1) is

$$\min_{\boldsymbol{\alpha}} \sum_{i=1}^{n} (\hat{\eta}_i - \hat{\omega}_i)^2. \tag{III.3}$$

(III.3) realizes the same semantics as (III.1), however, generalized by the function or "code" (it is not 1-1) given by (III.2) to η_i values not restricted to the real unit interval $[0, 1]$. If L is bounded, then the 1-1 function $\hat{\eta}_i = \eta_i / L$ achieves a similar purpose. (III.3) possesses the same desirable properties as (III.1), which are characterized by the following two results.

Proposition III.4 (Optimal Per-flow Classifier) *Given $\boldsymbol{\eta}$, $\boldsymbol{\lambda} \in \mathbb{R}_+^{\ltimes}$, the solution to* (III.3) *is*

$$\alpha_i = (1-\nu) \frac{\lambda_i \hat{\eta}_i}{\sum_{j=1}^{n} \lambda_j \hat{\eta}_j} + \nu \frac{\lambda_i}{\sum_{j=1}^{n} \lambda_j} \tag{III.5}$$

for all $i \in [1, n]$ where $0 \leq \nu \leq 1$ is a parameter which defines a continuous family of solutions.

The parameter ν, which stems from the dimension reduction associated with (III.2), has an appealing interpretation. The second term in (III.5) corresponds to the proportional share achieved by FIFO scheduling, whereas the first term corresponds to proportional share of the corresponding *virtual* flows $\lambda_i \hat{\eta}_i$, which are the original flow rates weighted by their relevancy variable $\hat{\eta}_i$ derived from η_i. Thus, if $\nu = 1$, then the per-hop control effectively ignores the label values and behaves as a FIFO queue. If $\nu = 0$, then the router acts like a GPS scheduler with service weights given by the first term. For any other value of ν, (III.5) represents a convex combination of the two behavioral modes.

Proposition III.6 (Per-flow Classifier Properties) *The optimal per-flow classifier given in* (III.5) *satisfies properties* (A1), (A2), *and* (B).

[6] The generalization to other norms is treated separately.

B. Optimal Aggregate-flow Classification

With the semantic set-up of optimal per-flow classification, let us consider the aggregate-flow classifier problem where $n > m$. The original aggregate-flow classifier problem, $n > L = m$, is subsumed by the more general set-up where L can take on any value. From a QoS provisioning perspective, the ultimate goal of a differentiated services network comprised of aggregate-flow per-hop controls is the provisioning of *end-to-end QoS* commensurate with each user's needs. Aggregate-flow control, whether it has many or few labels, must service n flows using $m < n$ service classes which results in a reduced ability to effectively shape end-to-end QoS with respect to the performance criterion (III.3) when compared to per-flow control. That is, the minimum value of (III.3) achieved by optimal per-hop control is smaller than that of optimal aggregate-flow control. This is a consequence of a more general result given by Proposition III.9.

We give a formal definition of aggregate-flow per-hop control. An *aggregate-flow per-hop control with parameter* (m, L) is a function

$$\Phi_{m,L} : (\boldsymbol{\eta}, \boldsymbol{\lambda}) \mapsto (\xi, \boldsymbol{\alpha}) \tag{III.7}$$

where $\xi : [1, L] \to [1, m]$ is the *classifier* and $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_m)$ is the vector of service weights assigned to the m service classes. With respect to end users, $\Phi_{m,L}$ induces—explicitly or implicitly—a performance function $\varphi^i_{m,L}$ for each user $i \in [1, n]$

$$\varphi^i_{m,L} : (\boldsymbol{\eta}, \boldsymbol{\lambda}) \mapsto \alpha_i, \tag{III.8}$$

where $\alpha_i = \varphi^i_{m,L}(\boldsymbol{\eta}, \boldsymbol{\lambda}) \geq 0$ is user i's share of the bandwidth allocated by $\Phi_{m,L}$. With a slight abuse of notation, we use α_i to denote both user i's ($i \in [1, n]$) apportioned resource, as well as the service weight allocated by $\Phi_{m,L}$ to service class i ($i \in [1, m]$). In the per-flow case, they coincide. Since the traffic rate $\boldsymbol{\lambda}$ is fixed, we will omit it from the argument list. The two-stage interpretation of aggregate-flow per-hop control is depicted in Figure II.2.

Proposition III.9 (Service Class Monotonicity) *Let $\Phi_{m,L}$ be an aggregate-flow per-hop control, and let $S_m = \{\boldsymbol{\alpha} : \varphi_{m,L}(\boldsymbol{\eta}) = \boldsymbol{\alpha}$ for some $\boldsymbol{\eta}\}$. Then* (III.3) *achieves a smaller value with more service classes, i.e., $m' \geq m$ implies*

$$\Big\{ \min_{\boldsymbol{\alpha} \in S_{m'}} \sum_{i=1}^{n} (\hat{\eta}_i - \hat{\omega}_i)^2 \Big\} \leq \Big\{ \min_{\boldsymbol{\alpha} \in S_m} \sum_{i=1}^{n} (\hat{\eta}_i - \hat{\omega}_i)^2 \Big\}.$$

Consider a special type of aggregate-flow per-hop control $\Phi_{m,L}$—called *Reduction Classifier*—whose behavior is completely determined by its classifier $\xi : [1, L] \to [1, m]$, in the following sense. Let

$$U_k = \{ i \in [1, L] : \ \xi(i) = k \}, \qquad k \in [1, m],$$

be the partition of $[1, L]$ induced by ξ. On input $(\boldsymbol{\eta}, \boldsymbol{\lambda})$, $\Phi_{m,L}$ behaves as

$\Phi_{m,L}(\boldsymbol{\eta}, \boldsymbol{\lambda})$:

1. Compute $\lambda^k = \sum_{i \in U_k} \lambda_i$ for each $k \in [1, m]$.

2. Compute $\hat{\eta}^k$ for $k \in [1, m]$ as follows,

$$\hat{\eta}^k = \begin{cases} 0, & \text{if } \exists i \in U_k, \hat{\eta}_i = 0; \\ 1, & \text{if } \exists i \in U_k, \hat{\eta}_i = 1; \\ \sum_{i \in U_k} \hat{\eta}_i / |U_k|, & \text{otherwise.} \end{cases}$$

3. Use per-flow optimal solution (Proposition III.4) with new input $\tilde{\boldsymbol{\eta}} = (\hat{\eta}^1, \dots, \hat{\eta}^m)$, $\tilde{\boldsymbol{\lambda}} = (\lambda^1, \dots, \lambda^m)$, to solve the reduced per-flow classifier problem consisting of m superusers.

A reduction classifier reduces the L label (or n user) problem to an m user per-flow classification problem by aggregation of component flows and centroid computation, then solves the reduced problem by applying the optimal per-flow classification solution. The resource share received by individual flows can be computed as follows. Let α^k, $k \in [1, m]$, be the solution returned by Step 3. For $i \in U_k$, set α_i such that $\sum_{i \in U_k} \alpha_i = \alpha^k$, and $\alpha_i / \lambda_i =$ constant. This is the share received by user $i \in [1, n]$.

Theorem III.10 (Reduction Classifier) *Let $\Phi_{m,L}$ be a reduction classifier represented by its classifier ξ. Then $\Phi_{m,L}$ is an* optimal aggregate-flow per-hop control, *i.e., satisfies* (III.3) *if, and only if, ξ is a solution to*

$$\min_{\xi'} \sum_{k \in [1,m]} \sum_{i \in U_k} (\hat{\eta}_i - \hat{\eta}^k)^2 \tag{III.11}$$

where the minimum ranges over all reduction classifiers ξ'.

Theorem III.10 shows that an optimal aggregate-flow classifier must be a reduction classifier, and furthermore, it must efficiently cover—in the mean-square sense—the set of label values $\{\hat{\eta}_1, \hat{\eta}_2, \dots, \hat{\eta}_n\}$ using m centroids $\{\hat{\eta}^1, \dots, \hat{\eta}^m\}$. Thus optimal aggregate-flow per-hop control is a clustering or classification problem in the statistical classification sense. This is made more precise by the next result.

A classifier ξ is *well-formed* (also called a *grouping*) if the three conditions $\eta_i < \eta_j$, $\xi(i) = \xi(j)$, and $\eta_i \leq \eta_k \leq \eta_j$ jointly imply $\xi(k) = \xi(i)$. Thus if two different label values are mapped to the same service class, then all η values "sandwiched" in-between must be mapped to the same service class. ξ can be represented by well-formed parentheses on the totally ordered set $\eta_1 \leq \eta_2 \leq \dots \leq \eta_n$, where adjacent values are grouped into the same partition except, possibly, at boundaries.

Theorem III.12 (Grouping) *An optimal aggregate-flow classifier is well-formed.*

Thus aggregate-flow per-hop control is, mathematically, an optimal clustering problem. Unlike its many brethren in higher dimensions that are, with few exceptions, NP-complete [22], the clustering problem given by (III.11) in Theorem III.10 has a poly-time algorithm; e.g., it can be solved by dynamic programming. When $L = m$—the practically relevant case where there are as many labels as service classes—optimal aggregate-flow classification has a linear time algorithm.

C. Properties of Optimal Aggregate-flow Classifiers

Although optimal per-flow classifiers satisfy properties (A1), (A2), and (B), the same is not necessarily true of optimal aggregate-flow classifiers.

Theorem III.13 (Aggregate-flow Classifier Properties) *An optimal aggregate-flow per-hop control satisfies property* (B)*, but need not satisfy properties* (A1) *and* (A2)*.*

Property (A2) is more subtle than (A1) and (B), but of import in influencing the stability and dynamical structure of noncooperative networks built on top of a differentiated services network substrate.

Theorem III.14 (Classifier Properties with $L = m$) *An optimal aggregate-flow per-hop control with parameters $L = m$ satisfies properties* (A1)*,* (A2)*, and* (B)*.*

The $L = m$ constraint advanced by Theorem III.14 coincides with practical considerations that derive from an implementation perspective. For example, assuming four bits from the TOS field in IPv4 are used to encode the label set $\{a, a+1, \ldots, a+15\}$ for some $a \geq 0$, then we may configure 16 service classes at routers, one for each of the 16 possible label values. The classifier results and properties for fixed service weights are treated separately.

IV. Game Theoretic Structure

The roadmap of the game theoretic results is as follows. First, we derive stability properties—existence of Nash equilibria and their structure—and dynamics of the noncooperative QoS provision game when users are allowed to set their η values end-to-end. Second, we show efficiency properties with respect to system optimality, in particular, when Nash equilibria are system optimal.

A. Basic Definitions

To satisfy user i's QoS requirement $\boldsymbol{\theta}^i$, the per-hop control—whatever its specific form—must apportion a fraction $\alpha_i^* \geq 0$ of the available bandwidth. Let α_i^* denote the minimal such bandwidth. We will find it more convenient to work in the service weight space $\{\boldsymbol{\alpha} : \boldsymbol{\alpha} \geq 0 \text{ and } \sum_{i=1}^n \alpha_i \leq 1\}$. We will use $\varphi^i(\cdot)$ to denote the performance function correponding to $x^i(\cdot)$ which allocates—explicitly or implicitly—a service weight to user i for a given input $\boldsymbol{\eta}$.

We will call the pair $(\boldsymbol{\eta}, \boldsymbol{\eta}')$ of control vectors a *selfish move* of user $i \in [1, n]$ with respect to α_i^* if $\boldsymbol{\eta}' = \boldsymbol{\eta} \pm \mathbf{e}_i$, and the following two conditions are satisfied:
(i) $\varphi^i(\boldsymbol{\eta}) < \alpha_i^*$ implies $\boldsymbol{\eta}' = \boldsymbol{\eta} + \mathbf{e}_i$ and $\varphi^i(\boldsymbol{\eta}') > \varphi^i(\boldsymbol{\eta})$;
(ii) $\varphi^i(\boldsymbol{\eta}) > \alpha_i^*$ implies $\boldsymbol{\eta}' = \boldsymbol{\eta} - \mathbf{e}_i$ and $\alpha_i^* \leq \varphi^i(\boldsymbol{\eta}') < \varphi^i(\boldsymbol{\eta})$.
Thus an "unhappy" user tries to improve his happiness by increasing η_i, while an "overly" satisfied user tries to reduce the satisfaction level to match his actual needs. We will call a pair of control vectors $(\boldsymbol{\eta}, \boldsymbol{\eta}')$ a *concurrent selfish move* (in the negative direction) if for some $J \subseteq [1, n]$, $\boldsymbol{\eta} = \boldsymbol{\eta} - \sum_{i \in J} \mathbf{e}_i$, and $(\boldsymbol{\eta}, \boldsymbol{\eta} - \mathbf{e}_i)$ is a selfish move for all $i \in J$. An analogous definitions holds for concurrent selfish moves in the *positive direction*. We will sometimes refer to selfish moves as *sequential* selfish moves to distinguish from concurrent ones. The definition of selfish move describes an efficient or cost conscious user who only consumes just enough resources to satisfy her QoS needs.

For user i, let $\mathcal{A}_i = \{\boldsymbol{\eta} : \varphi^i(\boldsymbol{\eta}) \geq \alpha_i^*\}$. Thus $\mathcal{A}_i$ represents the set of configuration where user i's QoS requirement is satisfied. Let

$$\mathcal{A}^* = \bigcap_{i=1}^n \mathcal{A}_i.$$

Thus all users' QoS requirements are satisfied at $\boldsymbol{\eta} \in \mathcal{A}^*$. A configuration $\boldsymbol{\eta}$ is *system optimal* if $\boldsymbol{\eta} \in \mathcal{A}^*$, and for all $\boldsymbol{\eta}' \neq \boldsymbol{\eta}$, $\boldsymbol{\varphi}(\boldsymbol{\eta}') > \boldsymbol{\varphi}(\boldsymbol{\eta})$ does not hold. In a system optimal configuration, the users' QoS requirements are met while expending the minimal amount of resources. In an *overloaded* system, i.e., $\sum_{i=1}^n \alpha_i^* > 1$, by definition, there cannot exist a way of allocating network resources such that all users' QoS requirements are satisfied. $\boldsymbol{\eta} \in \mathcal{A}^*$ is a *corner point* of $\mathcal{A}^*$ if the set of selfish moves from $\boldsymbol{\eta}$ is empty.

B. Nash Equilibria and Stability Properties

B.1 Dynamics inside $\mathcal{A}^*$

First, we will present the dynamical properties of the noncooperative QoS provision game when $\mathcal{A}^*$ exists (i.e., is nonempty) and $\boldsymbol{\eta} \in \mathcal{A}^*$.

Proposition IV.1 (Projection) *For user i and configuration $\boldsymbol{\eta} \in \mathcal{A}_i$, let $\mathcal{M}_i(\boldsymbol{\eta}) = \{\boldsymbol{\eta}' : \eta_i' = \eta_i, \text{ and } \eta_j' \leq \eta_j \text{ for } j \neq i\}$. Then $\mathcal{M}_i(\boldsymbol{\eta}) \subseteq \mathcal{A}_i$.*

Proposition IV.1 is a consequence of property (A2) of the per-hop control. We can use Proposition IV.1 and property (A1) to show a closure property of $\mathcal{A}^*$.

Lemma IV.2 (Closure) *$\mathcal{A}^*$ is closed under selfish moves, sequential and concurrent. That is, for $\boldsymbol{\eta} \in \mathcal{A}^*$ and any subset of users $J \subseteq [1, n]$ such that $(\boldsymbol{\eta}, \boldsymbol{\eta} - \mathbf{e}_i)$ is a selfish move for all $i \in J$,*

$$\boldsymbol{\eta} - \sum_{i \in J} \mathbf{e}_i \in \mathcal{A}^*.$$

Thus selfish users, even when making simutaneous selfish changes to their η values, cannot escape from the set $\mathcal{A}^*$ where their QoS requirements are all satisfied, some more than necessary. A concurrent selfish move, with respect to users in $J \subseteq [1, n]$ and intersection set $\bigcap_{i \in J} \mathcal{A}_i$, can be represented by a subset of $J' \subseteq J$ that shows the users making a move since selfish moves within $\bigcap_{i \in J} \mathcal{A}_i$ can only occur in the downward direction (a consequence of the more general result Lemma IV.6).

Theorem IV.3 (Monotone Convergence) *Any initial configuration $\boldsymbol{\eta} \in \mathcal{A}^*$ converges to a corner point of $\mathcal{A}^*$ under selfish moves, sequential or concurrent.*

Thus a corner point of $\mathcal{A}^*$ is a fixed point under the dynamics of selfish moves within $\mathcal{A}^*$, from which users cannot escape by selfish actions due to closure. Theorem IV.3 also shows that $\mathcal{A}^*$ always possesses a corner point, not necessarily unique. A corner point $\boldsymbol{\eta}$ represents an *efficient* allocation of resources for all users in the sense that each user i's QoS requirement is satisfied by $\boldsymbol{\eta}$, i.e., $\alpha_i^* = \varphi^i(\boldsymbol{\eta}) \geq \alpha_i^*$. Furthermore, any incremental action by i will either violate his QoS requirement or increase the apportioned resources beyond what is needed to satisfy the user's QoS requirement. We will show that a non-incremental action by user i will have the same consequences (Theorem IV.4). If $\varphi^i(\boldsymbol{\eta}) = \alpha_i^*$ then $\boldsymbol{\eta}$ is efficient in an absolute sense.

Theorem IV.4 (Corner Point and Nash) *Let $\boldsymbol{\eta}$ be a corner point of $\mathcal{A}^*$. Then $\boldsymbol{\eta}$ is a Nash equilibrium.*

We remark that a corner point of $\mathcal{A}^*$ must be Nash equilibrium, but the converse need not be true. Indeed, there are Nash equilibria that need not be in $\mathcal{A}^*$, even when it is nonempty.

Theorem IV.5 (Nash and System Optimality) *A configuration $\boldsymbol{\eta}$ is Nash and system optimal if, and only if, $\boldsymbol{\eta}$ is a corner point of $\mathcal{A}^*$.*

B.2 Dynamics outside $\mathcal{A}^*$

When proving Lemma IV.2, it turns out to be inessential that the intersection set be $\mathcal{A}^*$. For $J \subseteq [1, n]$, the same argument goes through when selfish moves are restricted to users in J. In fact, Lemma IV.2 is a special case of the following more general result.

Lemma IV.6 (Closure with User Restriction) *For $J \subseteq [1, n]$, $\bigcap_{i \in J} \mathcal{A}_i$ is closed under sequential and concurrent selfish moves when restricted to users in J.*

Thus keeping the η values of some users fixed, there are subspaces in lower dimensions where closure with respect to the remaining users' selfish moves can hold for a more relaxed intersection set. For any configuration $\boldsymbol{\eta}$, define

$$J(\boldsymbol{\eta}) = J^+(\boldsymbol{\eta}) \cup J^-(\boldsymbol{\eta})$$

as the set of all selfish moves where $J^+(\boldsymbol{\eta})$ is the set of moves in the positive direction and $J^-(\boldsymbol{\eta})$ represents the set of selfish moves in the negative direction. By the definition of selfish move, it follows that $J^+(\boldsymbol{\eta})$, $J^-(\boldsymbol{\eta})$ form a partition, and $i \in J^+(\boldsymbol{\eta})$ implies $\boldsymbol{\eta} \in \overline{\mathcal{A}_i}$, and $i \in J^-(\boldsymbol{\eta})$ implies $\boldsymbol{\eta} \in \mathcal{A}_i$.

Theorem IV.7 (Cycles) *There exist network systems with $\mathcal{A}^* \neq \emptyset$ such that for some $\boldsymbol{\eta} \in \overline{\mathcal{A}^*}$ and finite sequence $J_1, J_2, \ldots, J_r$ of concurrent selfish moves, $J_r(J_{r-1}(\cdots J_1(\boldsymbol{\eta}) \cdots)) = \boldsymbol{\eta}$. That is, configurations outside $\mathcal{A}^*$ can exist from which concurrent selfish moves lead to a cycle.*

Cycles turn out to have limited impact with respect to instability in that they cannot arise under sequential selfish moves, and they are transient as shown by the next result.

Theorem IV.8 (Transience of Cycles) *Cycles, when they exist, are* transient *in the sense that from any configuration $\boldsymbol{\eta}$ on the cycle, there exist sequential or concurrent selfish moves that lead to a Nash equilibrium.*

Corollary IV.9 (Nash Existence) *There always exist Nash equilibria.*

We have presented the results such that existence of Nash is an immediate consequence of Theorem IV.4 and Theorem IV.8. A Nash equilibrium $\boldsymbol{\eta} \notin \mathcal{A}^*$ has a specific monotonic form; we omit the detailed characterization due to space constraints.

C. System Optimality and Structural Properties

We turn our focus to characterizing when $\mathcal{A}^*$ is nonempty. The next result is the only general result that holds from (A1), (A2), and (B) without exploiting further properties of the optimal aggregate-flow classifier solution for $L = m$.

Proposition IV.10 (Diagonal Inclusion) *Let $\mathcal{D} = \{\boldsymbol{\eta} : \eta_i = \eta_j$ for all $i, j \in [1, n]\}$. If $\alpha_i^* \leq \lambda_i / \sum_{j=1}^n \lambda_j$ for all users $i \in [1, n]$, then $\mathcal{D} \subseteq \mathcal{A}^*$.*

Note that $\alpha_i^* \leq \lambda_i / \sum_{j=1}^n \lambda_j$ for all $i \in [1, n]$ implies that $\sum_{i \in [1,n]} \alpha_i^* \leq 1$. Next, we find weaker conditions for $\mathcal{A}^* \neq \emptyset$, and characterize the loss of power resulting from having a bounded, discrete label set $\{1, 2, \ldots, L\}$. To achieve this, we utilize the properties of the optimal aggregate-flow classifier solution for $L = m$. First, consider the case when $\eta_i \in \mathbb{R}_+$ for all $i \in [1, n]$, and $n = m$. The case of interest, $\boldsymbol{\eta} \in [1, L]^n$ in the aggregate-flow case can be analyzed by relating it to the unrestricted case.

Theorem IV.11 (Unrestricted Intersection) *Assume $\eta_i \in \mathbb{R}_+$ for all $i \in [1, n]$. Let $n = m$, and let ξ be the optimal per-flow classifier. Then $\mathcal{A}^* \neq \emptyset$ if, and only if,*
(a) $\exists\, i \in [1, n]$ *such that* $\alpha_i^* \leq \nu \lambda_i / \sum_{j=1}^n \lambda_j$, *and*
(b) $\sum_{j=1}^n \max\{\alpha_i^*, \nu \lambda_i / \sum_{j=1}^n \lambda_j\} \leq 1$.

Here $\nu \geq 0$ is the solution parameter of the optimal per-flow classifier which determines how much proportional sharing to inject in the service weight allocation ($\nu = 1$ degenerates per-hop control to FIFO). Theorem IV.11 is a tight characterization of $\mathcal{A}^*$'s nonemptiness in the unrestricted case where properties

(a) and (b) stem from the particular form of the optimal per-flow classifier solution given by Proposition III.4. Note that as $\nu \to 0$, (b) becomes $\sum_{j=1}^{n} \alpha_i^* \leq 1$ which is the weakest possible condition for nonemptiness of $\mathcal{A}^*$. The next result is an immediate consequence of Theorem IV.11.

Corollary IV.12 (Empty Restricted Intersection) *If $\mathcal{A}^* = \emptyset$ in the unrestricted case, then $\mathcal{A}^* = \emptyset$ in the restricted case where $\eta_i \in \{1, 2, \ldots, L\}$ for all $i \in [1, n]$, and $L < \infty$.*

The aggregate-flow and per-flow cases with respect to nonemptiness of $\mathcal{A}^*$ can be related by the next result which is a consequence of Theorem III.14.

Proposition IV.13 (Per-flow and Aggregate-flow Relation) *Let $\eta_i \in \{1, 2, \ldots, L\}$ for all $i \in [1, n]$, and $L < \infty$. $\mathcal{A}^* \neq \emptyset$ in the per-flow case (i.e., $n = m$) if, and only if, $\mathcal{A}^* \neq \emptyset$ in the aggregate-flow case with $m = L$.*

Given the relationship of nonemptiness of $\mathcal{A}^*$ between the per-flow and aggregate-flow case under $\eta_i \in \{1, 2, \ldots, L\}$, what remains is a quantitative characterization of the loss of power due to discreteness and boundedness of the label set $[1, L]$ in the aggregate-flow case.

Theorem IV.14 (Loss of Power due to Restriction) *Let $L = m < n$. If there exists $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ with $\alpha_{min} = 0$, $\alpha_{max} = 1$, $0 \leq \alpha_i \leq 1$, such that*

$$(1-\nu)\frac{\lambda_i \alpha_i}{\sum_{j=1}^{n} \lambda_j \alpha_j} + \nu \frac{\lambda_i}{\sum_{j=1}^{n} \lambda_j} \geq \alpha_i^* + \frac{1-\nu}{L-1} \frac{\lambda_i}{\sum_{j=1}^{n} \lambda_j \alpha_j} \qquad \text{(IV.15)}$$

for all $i \in [1, n]$, then $\mathcal{A}^* \neq \emptyset$.

The left-hand-side of inequality (IV.15) just denotes a valid service weight vector with respect to the optimal aggregate-flow classifier. The second term in the right-hand-side of (IV.15) of Theorem IV.14 quantifies the loss of power due to coarsification. If $L \to \infty$, then the loss-of-power term drops out. In practice, L is a small finite value (e.g., using 4 bits in the precedence field of IP, $L = 16$). The next result shows that $n \gg L$—the raison d'etre of aggregate-flow control—facilitates tightness of the bound.

Corollary IV.16 (Nonempty Discrete Intersection) *Under the same conditions as* Theorem IV.14, *let $d_i = \lfloor (L-1)\alpha_i \rfloor$, $i \in [1, n]$. Then, $\mathcal{A}^* \neq \emptyset$ if for all $i \in [1, n]$*

$$(1-\nu)\frac{\lambda_i \alpha_i}{\sum_{j=1}^{n} \lambda_j \alpha_j} + \nu \frac{\lambda_i}{\sum_{j=1}^{n} \lambda_j} \geq \alpha_i^* + (1-\nu)\frac{\lambda_i}{\sum_{k=1}^{L-1} k \sum_{j:d_j=k} \lambda_j} \qquad \text{(IV.17)}$$

For $n \gg L$, we can expect $\frac{\lambda_i}{\sum_{k=1}^{L-1} k \sum_{j:d_j=k} \lambda_j} \ll 1$, and (IV.15) gives a tight bound on the existence condition of system optimal Nash equilibria.

References

[1] R. L. Cruz, "Quality of service guarantees in virtual circuit switched networks," *IEEE J. Select. Areas Commun.*, vol. 13, no. 6, pp. 1048–1056, 1995.

[2] G. de Veciana, G. Kesidis, and J. Walrand, "Resource management in wide-area ATM networks using effective bandwidths," *IEEE J. Select. Areas Commun.*, vol. 13, no. 6, pp. 1081–1090, 1995.

[3] 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*, vol. 1, no. 3, pp. 344–357, 1993.

[4] 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*, vol. 2, no. 2, pp. 137–150, 1994.

[5] S. Chen and K. Park, "A distributed protocol for multi-class QoS provision in noncooperative many-switch systems," in *Proc. IEEE International Conference on Network Protocols*, 1998, pp. 98–107.

[6] D. Clark and W. Fang, "Explicit allocation of best-effort packet delivery service," *IEEE/ACM Trans. Networking*, vol. 6, no. 4, pp. 362–373, 1998.

[7] C. Dovrolis, D. Stiliadis, and P. Ramanathan, "Proportional differentiated services: Delay differentiation and packet scheduling," in *Proc. ACM SIGCOMM '99*, 1999.

[8] J. MacKie-Mason and H. Varian, "Economic FAQs about the Internet," in *Internet Economics*, L. McKnight and J. Bailey, Eds., pp. 27–63. MIT Press, 1996.

[9] K. Nichols, V. Jacobson, and L. Zhang, "A two-bit differentiated services architecture for the Internet," Internet Draft, 1997.

[10] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated service," RFC 2475, 1998.

[11] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski, "Assured Forwarding PHB Group," RFC 2597, 1999.

[12] S. Chen and K. Park, "An architecture for noncooperative QoS provision in many-switch systems," in *Proc. IEEE INFOCOM '99*, 1999, pp. 864–872.

[13] K. Park, M. Sitharam, and S. Chen, "Quality of service provision in noncooperative networks: heterogeneous preferences, multi-dimensional QoS vectors, and burstiness," in *Proc. 1st International Conference on Information and Computation Economies*, 1998, pp. 111–127.

[14] M. May, J. Bolot, A. Jean-Marie, and C. Diot, "Simple performance models of differentiated services schemes for the Internet," in *Proc. IEEE INFOCOM '99*, 1999, pp. 1385–1394.

[15] V. Jacobson, K. Nichols, and K. Poduri, "An Expedited Forwarding PHB," RFC 2598, 1999.

[16] W. Feng, D. Kandlur, D. Saha, and K. Shin, "Adaptive packet marking for providing differentiated services in the Internet," in *Proc. IEEE International Conference on Network Protocols*, 1998, pp. 108–117.

[17] Andrew Odlyzko, "Paris Metro Pricing: The minimalist differentiated services solution," in *Proc. IEEE/IFIP International Workshop on Quality of Service*, 1999.

[18] H. Ren and K. Park, "Toward a theory of differentiated services," Tech. Rep. CSD-TR-00-020, Department of Computer Sciences, Purdue University, 2000.

[19] H. Ren and K. Park, "Efficient shaping of user-specified QoS using aggregate-flow control," Tech. Rep. CSD-TR-00-021, Department of Computer Sciences, Purdue University, 2000.

[20] S. Chen, K. Park, and M. Sitharam, "On the ordering properties of GPS routers for multi-class QoS provision," in *Proc. SPIE International Conference on Performance and Control of Network Systems*, 1998, pp. 252–265.

[21] John-Francis Mergen, ," Personal communication.

[22] M. Garey and D. Johnson, *Computers and Intractability*, W. H. Freeman and Company, 1979.