Pricing Network Resources for Adaptive Applications in a Differentiated Services Network

Xin Wang and Henning Schulzrinne
Columbia University
{xinwang, schulzrinne}@cs.columbia.edu

Abstract— **The Differentiated Services framework (DiffServ) has been proposed to provide multiple Quality of Service (QoS) classes over IP networks. A network supporting multiple classes of service also requires a differentiated pricing structure. We propose a pricing scheme in a DiffServ environment based on the cost of providing different levels of quality of service to different classes, and on long-term demand. Pricing of network services dynamically based on the level of service, usage, and congestion allows a more competitive price to be offered, allows the network to be used more efficiently, and provides a natural and equitable incentive for applications to adapt their service contract according to network conditions. We develop a DiffServ simulation framework to compare the performance of a network supporting congestion-sensitive pricing and adaptive service negotiation to that of a network with a static pricing policy. Adaptive users adapt to price changes by adjusting their sending rate or selecting a different service class. We also develop the demand behavior of adaptive users based on a perceptually reasonable user utility function. Simulation results show that a congestion-sensitive pricing policy coupled with user rate adaptation is able to control congestion and allow a service class to meet its performance assurances under large or bursty offered loads, even without explicit admission control. Users are able to maintain a stable expenditure. Allowing users to migrate between service classes in response to price increases further stabilizes the individual service prices. When admission control is enforced, congestion-sensitive pricing still provides an advantage in terms of a much lower connection blocking rate at high loads.**

I. Introduction

The Differentiated Services framework (DiffServ) [1] has been proposed to provide multiple Quality of Service (QoS) classes over IP networks. Two types of Per-Hop-Behavior (PHB) are proposed: Expedited Forwarding (EF) [2] and Assured Forwarding (AF) [3]. The EF PHB is defined as a forwarding treatment where the departure rate of an aggregate's packets from any DiffServ node must equal or exceed a configurable rate. For AF service, four scheduling classes with three levels of drop precedence in each class are defined for general use.

A network supporting multiple classes of service also requires a differentiated pricing structure rather than the flat-fee pricing model adopted by virtually all current Internet services. While network tariff structures are often determined by business and marketing arguments rather than costs, we believe it is worthwhile to understand and develop a cost-based pricing structure as a guide for actual pricing. In economically viable models, the difference in the charge between different service classes would presumably depend on the difference in performance between the classes, and should take into account the average (long-term) demand for each class. In general, the level of forwarding assurance of an IP packet in DiffServ depends on the amount of resources allocated to a class the packet belongs to, the current load of the class, and in case of congestion within the class, the drop precedence of the packet. Also, when multiple services are available at different prices, users should be able to demand particular services, signal the network to provision according to the requested quality, and generate accounting and billing records. The first main goal of our work is to develop a pricing scheme in a DiffServ environment based on the cost of providing different levels of quality of service to different classes, and on long-term demand.

DiffServ supports services which involve a traffic contract or service level agreement (SLA) between the user and the network. If the agreement, including price negotiation and resource allocation are set statically (before transmission), pricing, resource allocation and admission control policies (if any) have to be conservative to be able to meet QoS assurances in the presence of network traffic dynamics. Pricing of network services dynamically based on the level of service, usage, and congestion allows a more competitive price to be offered, and allows the network to be used more efficiently. Differentiated and congestion-sensitive pricing also provides a natural and equitable incentive for applications to adapt their service contract according to network conditions. A number of adaptation schemes have been proposed for multimedia applications to dynamically regulate the source bandwidth according to the existing network conditions (a survey of this work is given in [4]).

The second main goal of our work is to integrate our pricing scheme with a dynamic pricing and service negotiation environment. In this environment, service prices have a congestion-sensitive component in addition to the long-term, relatively static price. Some or all users are capable of adapting, and adapt to price changes by adjusting their sending rate or selecting a different service class. Users with stringent bandwidth and QoS requirements maintain a high quality by paying more; adaptation-incapable applications use services offering a static price. We develop the demand behavior of adaptive users based on a perceptually reasonable user utility function.

In our simulations, prices and services are negotiated through a Resource Negotiation and Pricing (RNAP) protocol and architecture, presented in earlier work [5]. RNAP enables the user to select from available network services with different QoS properties and re-negotiate contracted services, and enables the network to dynamically formulate service prices and communicate current prices to the user. In RNAP, resource commitments are typically made for short negotiation intervals, instead of indefinitely, and prices may vary for each interval.

Using RNAP and an extended version of an existing DiffServ implementation, we develop a simulation framework to compare the performance of a network supporting congestion-sensitive pricing and adaptive service negotiation to that of a network with a static pricing policy. We study the stability of the dynamic pricing and service negotiation mechanisms. We evalu-

This work was supported by NSF CAREER grant.

ate the system performance and perceived benefit (or value-for-money) under the dynamic and static systems. We also study the relative effects on system performance of rate adaptation, dynamic load balancing between service classes and admission control. Although the simulation framework is based on the RNAP model, we try to derive results and conclusions applicable to static and congestion-driven, dynamic pricing schemes in general.

This paper is organized as follows. Section II develops a physically realistic user utility function to represent user demand behavior in response to price changes. Section III discusses our proposed pricing model in detail. Section IV summarizes our earlier work on RNAP and particularly how it supports network pricing. In section V we describe our simulation model, and in section VI we discuss simulation results. We describe some related work in section VII, and summarize our work in section VIII.

II. User Adaptation

In a network with congestion-dependent pricing and dynamic resource negotiation (through RNAP or some other signaling protocol), *adaptive* applications with a budget constraint will adjust their service requests in response to price variations. In this section, we discuss how a set of user applications performing a given task (for example, a video conference) adapt their sending rate and quality of service requests to the network in response to changes in service prices, so as to maximize the benefit or *utility* to the user, subject to the constraint of the user's budget.

Although we focus on adaptive applications as the ones best suited to a dynamic pricing environment, the RNAP framework does not require adaptation capability. Applications may choose services that provide a fixed price and fixed service parameters during the duration of service. Generally, the long-term average cost for a fixed-price service will be higher, since it uses network resources less efficiently. Alternatively, applications may use a service with usage-sensitive pricing, and maintain a high QoS level, paying a higher charge during congestion.

We consider a set of user applications, required to perform a task or *mission*. The user would like to determine a set of transmission parameters (sending rate and QoS parameters) from which it can derive the maximum benefit, subject to his budget. We assume that the user defines quantitatively, through a *utility function*, the perceived monetary value (say, 15 cents/minute) provided by the set of transmission parameters towards completing the mission.

Consumers in the real world generally try to obtain the best possible "value" for the money they pay, subject to their budget and minimum quality requirements; in other words, consumers may prefer lower quality at a lower price if they perceive this as meeting their requirements and offering better value. Intuitively, this seems to be a reasonable model in a network with QoS support, where the user pays for the level of QoS he receives. In our case, the "value for money" obtained by the user corresponds to the surplus between the utility $U(\cdot)$ with a particular set of transmission parameters (since this is the perceived value), and the cost of obtaining that service. The goal of the adaptation is to maximize this surplus, subject to the budget and the minimum and maximum QoS requirements.

We now consider the simultaneous adaptation of transmission parameters of a set of n applications performing a single task. The transmission bandwidth and QoS parameters for each application are selected and adapted so as to maximize the mission-wide "value" perceived by the user, as represented by the surplus of the *total utility*, $\hat{U}$, over the total cost C. We can think of the adaptation process as the allocation and dynamic re-allocation of a finite amount of resources between the applications.

In this paper, we make the simplifying assumption that for each application, a utility function can be defined as a function only of the transmission parameters of that application, independent of the transmission parameters of other applications. Since we consider utility to be equivalent to a certain monetary value, we can write the total utility as the sum of individual application utilities :

$$\hat{U} = \sum_i [U^i(x^i(T_{spec}, R_{spec})] \tag{1}$$

where x^i is the transmission (T_{spec}) and quality of service parameter (R_{spec}) tuple for the i_{th} application. The optimization of surplus can be written as

$$\max \sum_i [U^i(x^i) - C^i(x^i)]$$
$$\text{s. t.} \sum_i C^i(x^i) \leq b, \qquad x^i_{min} \leq x^i \leq x^i_{max} \tag{2}$$

where x^i_{min} and x^i_{max} represent the minimum and maximum transmission requirements for stream i, C^i is the cost of the type of service selected for stream i at requested transmission parameter x^i, and b is the budget of the user.

In practice, the application utility is likely to be measured by user experiments and known at discrete bandwidths, at one or a few levels of loss and delay, possibly corresponding to a subset of the available services; at the current stage of research, some possible services are guaranteed [6] and controlled-load service [7] under the int-serv model, Expedited Forwarding (EF) [2] and Assured Forwarding (AF) [3] under diff-serv. In this case, it is convenient to represent the utility as a piecewise linear function of bandwidth (or a set of such functions). A simplified algorithm is proposed in [8] to search for the optimal service requests in such a framework.

We can make some general assumptions about the utility function as a function of the bandwidth (can be equivalent bandwidth [9]), at a fixed value of loss and delay. A user application generally has a minimum bandwidth requirement. It also associates a certain minimum value with a task, which may be regarded as an "opportunity" value, and this is the perceived utility when the application receives just the minimum required bandwidth. The user terminates the application if its minimum bandwidth requirement cannot be fulfilled, or when the price charged is higher than the opportunity value derived from keeping the connection alive. Also, user experiments reported in the literature [10][11] suggest that utility functions typically follow a model of diminishing returns to scale, that is, the marginal utility as a function of bandwidth diminishes with increasing bandwidth. Hence, a utility function can be represented in a general form as a function of bandwidth as:

$$U(x) = U_0 + w \log \frac{x}{x_m} \tag{3}$$

where x_m represents the minimum bandwidth the application requires, w represents the sensitivity of the utility to bandwidth,

and U_0 is the monetary "opportunity" that the user perceives at the lowest QoS level.

The utility function is also sensitive to network transmission parameters such as loss and delay. In our work, we rely on the experimental results in [12] which show that users' perceived quality for interactive audio decreases almost linearly with either delay or loss, with a minimum acceptable quality requirement. More subjective tests are needed for other application types. Currently, we assume a similar linear dependence for all applications. Accordingly, we represent the utility function as:

$$U(x) = U_0 + w \log \frac{x}{x_m} - k_d d - k_l l, \quad \text{for} \quad x \geq x_m, \tag{4}$$

where k_d and k_l represent respectively the user's sensitivity to delay and loss. In some cases, the user's perceived sensitivity may depend on the bandwidth used. For example, tolerance to delay and loss will be different for different speech codecs. Since we are not assuming any particular application model, we assume users' delay and loss sensitivity are bandwidth independent in our simulations. A user with a higher sensitivity to delay or loss will tend to select a higher service class rather than request more bandwidth. If the utilities of all the applications are represented in the format of equation 4, the optimization process for a system with multiple applications can be represented as:

$$\max \sum_i [U_0^i + w^i \log \frac{x^i}{x_m^i} - k_d^i d - k_l^i l - p^i x^i]$$

$$\text{s. t.} \sum_i p^i x^i \leq b, \quad x^i \geq x_m^i, \forall i, \quad d \leq D, \quad l \leq L \tag{5}$$

where p_i is the price of the service class selected by the application i, D and L are respectively the loss and delay bound of an application above which it no longer functions usefully.

It is possible to represent the above optimization problem as a Lagrangian and solve it. However, we assume the availability of only a few different loss and delay levels corresponding to different service classes, and accordingly use a more heuristic method.

The optimization involves assigning a service class and a bandwidth to each application i. For a particular assignment of service classes to applications, if the user can obtain the optimal bandwidth distribution according to equation 5 at a cost below his budget, then the bandwidth allocation that maximizes the perceived surplus for an application can be shown to be:

$$x^i = \frac{w^i}{p^i} \tag{6}$$

Hence, w^i represents the money a user would spend based on its perceived value for an application.The above bandwidth distribution is considered for all possible service class assignments (constrained by application requirements and budget), and the one giving the highest total surplus is used.

If there is no set of service class assignments for which the optimal distribution of equation 6 can be obtained at a cost below the budget, the total budget is first distributed to the component applications according to their relative bandwidth sensitivity w^i. That is, each application receives a budget share b^i such that

$$b^i = b \frac{w^i}{\sum_k w^k} \tag{7}$$

Each application is then allocated a service i and bandwidth $x^i = \frac{b^i}{p^i}$ which maximizes its individual surplus according to equation 4.

The discussion so far assumes that each price p^i is per unit average bandwidth. A price based on unit equivalent bandwidth [13] may be fairer since it takes into account the burstiness of user traffic. In this case, the user adaptation of the source rate is more complicated. If effective bandwidth is used, a user could calculate a new average bandwidth when the price increases. Alternatively, it could introduce additional buffering at the source to reduce its burstiness, at the cost of a higher delay, thus reducing the effective bandwidth.

III. Pricing Strategies

A few pricing schemes are widely used in the Internet today [14]: access-rate-dependent charge (AC), volume-dependent charge (V), or the combination of both (AC-V). An AC charging scheme is usually one of two types: allowing unlimited use, or allowing limited duration of connection, and charging a per-hour fee for additional connection time. Similarly, AC-V charging schemes normally allow some amount of traffic to be transmitted for a fixed access fee, and then impose a per-unit charge. Although time-of-day-dependent charging is commonly used in telephone networks, it is not generally used in the current Internet. User experiments [15] indicate that usage-based pricing is a fair way to charge people and allocate network resources. Both connection time and the transmitted volume reflect the usage of the network. Charging based on connect-time only works when resource demands per time unit are roughly uniform. Since this is not the case for Internet applications and across the range of access speeds, we only consider volume-based charging.

In this paper, we study two kinds of volume-based pricing: a fixed-price (FP) policy with a fixed unit volume price, and a congestion-price-based adaptive service (CPA) in which the unit volume price has a congestion-sensitive component. In the fixed price model, the network charges the user per volume of data transmitted, independent of the congestion state of the network. The per-byte charge can be the same for all service classes ("flat", FP-FL), depend on the service class (FP-PR), depend on the time of day (FP-T) or a combination of time-of-day and service class (FP-PR-T).

If the price does not depend on the congestion conditions in the network, customers with less bandwidth-sensitive applications have no motivation to reduce their traffic as network congestion increases. As a result, either the service request blocking rate will increase at the call admission control level, or the packet delay and dropping rate will increase at the queue management level. Having a congestion-dependent component in the service price provides a monetary incentive for adaptive applications to adapt their service class and/or sending rates according to network conditions. In periods of resource scarcity, quality sensitive applications can maintain their resource levels by paying more, and relatively quality-insensitive applications will reduce their sending rates or change to a lower class of service. The total price consists of a congestion-dependent component and a fixed volume-based charge. The fixed volume-based charge has the same 4 charging modes as in FP, giving the pricing models CP-FL, CP-PR, CP-T, CP-PR-T.

A. Proposed Pricing Scheme

We assume that routers support multiple service classes and that each router is partitioned to provide a separate link bandwidth and buffer space for each service, at each port. We use the

framework of the competitive market model [16]. The competitive market model defines two kinds of agents: consumers and producers. Consumers seek resources from producers, and producers create or own the resources. The exchange rate of a resource is called its price. The routers are considered the producers and own the link bandwidth and buffer space for each output port. The flows (individual flows or aggregate of flows) are considered consumers who consume resources. The congestion-dependent component of the service price is computed periodically, with a price computation interval τ. The total demand for link bandwidth is based on the aggregate bandwidth reserved on the link for a price computation interval, and the total demand for the buffer space at an output port is the average buffer occupancy during the interval. The supply bandwidth and buffer space need not be equal to the installed capacity; instead, they are the targeted bandwidth and buffer space utilization. The congestion price will be levied once demands exceeds a provider-set fraction of the available bandwidth or buffer space. We now discuss the formulation of the fixed charge, which we decompose into *holding charge* and *usage charge*, and the formulation of the *congestion charge*.

A.1 Holding Charge

A service may enforce admission control to ensure some level of performance. In this case, the applications admitted into the network will impose some potential cost by depriving other applications the opportunity to be admitted even if no data is being sent. Hence, it is fair to charge the admitted applications a holding price. The holding charge can be calculated based on the following consideration. If a particular flow or flow-aggregate does not utilize the resources (buffer space or bandwidth) set aside for it, we assume that the scheduler allows the resources to be used by excess traffic from a lower level of service. The holding charge reflects revenue lost by the provider because instead of selling the allotted resources at the usage charge of the given service level (if all of the reserved resources were consumed) it sells the reserved resources at the usage charge of a lower service level. The holding price (p_h^j) of a service class j is therefore set to be proportional to the difference between the usage price for that class and the usage price for the next lower service class. The holding price can be represented as:

$$p_h^j = \alpha^j (p_u^j - p_u^{j-1}),$$

where α^j is a scaling factor related to service class j. The holding charge $c_h^{ij}(n)$ when the customer i reserves a bandwidth $r^{ij}(n)$ from class j is given by:

$$c_h^{ij}(n) = p_h^j r_{ij}(n) \tau^j.$$

where τ^j is the negotiation period for class j. $r_{ij}(n)$ can be a bandwidth requirement specified explicitly by the customer i, or estimated from the traffic specification and service request of the customer.

A.2 Usage Charge

The usage charge depends on the amount of resources consumed, the average overall user demand, the level of service guaranteed to the user, and the elasticity of the traffic. The usage price (p_u) will be set such that it allows a retail network to recover the cost of the purchase from the wholesale market, and various fixed costs associated with the service. In a network supporting multiple classes of service, the difference in the charge between different service classes would likely depend on the difference in performance between the classes. The model we consider is a network supporting J classes of services, the service price for class j is p_u^j, the long time user bandwidth demand is known (e.g., through statistics) and can be represented as $x^j(p_u^1, p_u^2, \ldots, p_u^J)$, and the cost of having capacity C during one unit of time is $f(C)$. The provider's decision problem is to choose the optimal prices for each class that optimize its profit:

$$\max_{p_u^j} [\sum_j^J x^j(p_u^1, p_u^2, ..., p_u^J) p_u^j - f(C)],$$
$$\text{subject to:} \quad r(x^j(p_u^1, p_u^2, ..., p_u^J)) \leq R, j \in J \tag{8}$$

where r represents the bandwidth requirement for all classes, and R is the total bandwidth available in the network. Assuming users choose service classes independently, the total demand for a class over a sufficiently long time period depends only on the price for that class. If we assume the users have the utility functions of Section II, the total demand of service class j can be represented as a constant elasticity model, $x^j(p_u^j) = A^j / p_u^j$, where demand varies inversely with the price of the service class. A^j reflects the total willingness-to-pay of users belonging to service class j.

Service pricing for differentiated service

DiffServ supports SLA negotiation between the user and the network. An SLA generally includes traffic parameters, which describe the user's traffic profile, and performance parameters, which characterize the level of performance that the network promises to provide to the conforming part of the user's traffic. A widely used descriptor for a user's traffic profile consists of a peak rate, a sustainable rate, and a maximum burst tolerance. Common QoS parameters are delay and loss. Mechanisms such as weighted fair queuing (WFQ) and class based queuing (CBQ) can be used to provision resources for different service classes. In general, a class with lower load leads to lower expected delays. A higher level of service class is expected to have a lower average load, and hence lower average delay. If we do not consider the difference in element costs for different classes, charging services proportional to their individual expected load seems to reasonably reflect the cost of providing the services and the differences between their performance. Assume that a unit of bandwidth of a service class would cost a basic price p_{basic} if all its bandwidth were used. If the expected load ratio of service class j is ρ^j, the unit bandwidth price for service class j can then be estimated as $p_u^j = p_{basic}/\rho^j$. The effective bandwidth consumption of an application with rate x_{ij} can be represented as x_{ij}/ρ_j. For constant elasticity demand, $x_j(p_u^j) = A_j/p_u^j$, and the effective bandwidth consumption is $A_j/(p_u^j \rho_j)$. The price optimization problem of equation 8 can then be written as:

$$\max_{p_u^j} [\sum_j^J \frac{A_j}{p_u^j} p_u^j - f(C)], \quad p_u^j = \frac{p_{\text{basic}}}{\rho_j}, \text{subject to } \sum_j^J \frac{A_j}{p_u^j \rho_j} \leq C$$

The Lagrangian for the problem can be represented as:

$$\max_{p_{basic}}[\sum_{j}^{J} A_j + \lambda(C - \frac{\sum_j^J A_j}{p_{\text{basic}}}) - f(C)]$$

The optimal solution is:

$$p_{\text{basic}} = \frac{\sum_j^J A_j}{C}, p_u^j = \frac{p_{\text{basic}}}{\rho_j} = \frac{\sum_j^J A_j}{C\rho_j}$$

The bandwidth provisioned for each service class will be given by A^j/p_{basic}, and is hence proportional to total user willingness to pay for that class. The usage charge $c_u^{ij}(n)$ for class j over a period n in which $v_{ij}(n)$ bytes were transmitted is given by

$$c_u^{ij}(n) = p_u^j v_{ij}(n).$$

A.3 Congestion Charge

A simple usage-based charging scheme monitors the data volume transmitted and charges users based on their average rate. Charging according to the mean rate encourages users to consume network bandwidth more efficiently, but does not discourage users from selecting large traffic contracts and sending the worst-case traffic allowed by their contract. An appropriate pricing scheme should provide users the incentives to select traffic contracts that reflect their actual needs. Effective bandwidth [9][17] and pricing based on effective bandwidth [13] have been proposed in a multiple-service-class environment. However, effective bandwidth normally accounts for the worst-case traffic subject to the traffic profile of the SLA. The contract for typical users has an effective bandwidth much larger than the mean rate. Provisioning based on equivalent bandwidth is not economically efficient in a DiffServ environment. Performance guarantees in DiffServ are qualitative and can be very loose. This may make it difficult to evaluate the equivalent bandwidth. Also, DiffServ does not allocate resources to applications based on their effective bandwidth. Therefore, it appears unfair to charge users based on their profile declaration only, though the charge should take the profile into account. To encourage users to reduce their resource requirements under network resource contention, we propose an additional congestion-sensitive price component under these conditions. The general network resources considered are bandwidth and buffer space. Two kinds of congestion pricing can be considered: pricing when the expected load bound is exceeded, or pricing when buffer occupancy reaches certain level. In the first case, when the average demand for a certain class exceeds a threshold, an additional congestion price is charged all users of that class.

In the case of priority dropping for AF class, the dropping precedence is only considered when the buffer occupancy reaches a certain thresholds. The same thresholds can be associated with different congestion or buffer prices. When each threshold is reached, user packets with the corresponding precedence level begin to be dropped with a certain probability, and users with higher precedence levels are charged the additional buffer price. Therefore, the higher precedence users pay the sum of buffer prices corresponding to all the exceeded thresholds. During congestion, lower precedence users will suffer lost packets, or reduce their rate, or smoothen their traffic at the source (at the cost of higher delay due to buffering), or change to a higher precedence and pay a higher price.

Both kinds of congestion price for a service class can be calculated as an iterative tâtonnement process [16]:

$$p_c^j(n) = \min[\{p_c^j(n-1) + \sigma_j(D_j, S_j)(D_j - S_j)/S_j, 0\}^+, p_{\max}^j] \quad (9)$$

where D_j and S_j represent the current total demand and supply respectively, and σ_j is a factor used to adjust the convergence rate. σ_j may be a function of D_j and S_j; in that case, it would be higher when congestion is severe. D_j and S_j will be different for bandwidth and buffer space congestion. The router begins to apply the congestion charge only when the total demand exceeds the supply. Even after the congestion is removed, a non-zero, but gradually decreasing congestion charge is applied until it falls to zero to protect against further congestion. In our simulations, we also used a price adjustment threshold parameter θ^j to limit the frequency with which the price is updated. The congestion price is updated if the calculated price increment exceeds $\theta_j p_c^j(n-1)$. The maximum congestion price is bounded by $p_{\max}^j$. When a service class needs admission control, all new arrivals are rejected when the price reaches $p_{\max}^j$. If p_c^j reaches $p_{\max}^j$ frequently, it indicates that more resources are needed for the corresponding service, or usage price for a class needs to be adjusted to reflect the new demand statistics. For a period n, the total congestion charge is given by

$$c_c^{ij}(n) = p_c^j(n) v_{ij}(n).$$

Based on the price formulation strategy described above, a router arrives at a cost structure for a particular RNAP flow or flow-aggregate at the end of each price update interval. The total charge for a session is given by

$$c_s^{ij} = \sum_{n=1}^{N} [p_h^j r^{ij}(n)\tau^j + (p_u^j + p_c^j(n)) v_{ij}(n)]$$

where N is the total number of intervals spanned by a session.

Networks may set the usage charge to zero, imposing a holding charge for reserving resources only, or apply a congestion charge during resource contention. Also, the holding charge would be set to zero for services without explicit resource reservation or admission control, for example, best effort service. Since the re-negotiation of network services will generally be driven by price changes, the stability of the negotiation process is discussed in related work with a greater focus on pricing [18][19].

IV. Resource Negotiation through RNAP

The pricing algorithms and adaptation framework presented in this paper do not depend on any particular network architecture or protocol. However in this paper, we simulated our results in an environment supporting dynamic service negotiation through the Resource Negotiation and Pricing protocol (RNAP) [5][8], using a centralized (RNAP-C) network management architecture. We first briefly review the RNAP framework, and then describe the pricing and charge formulation process used.

In the RNAP framework, we assume that the network makes services with certain QoS characteristics available to user applications, and charges prices for these services that, in general,

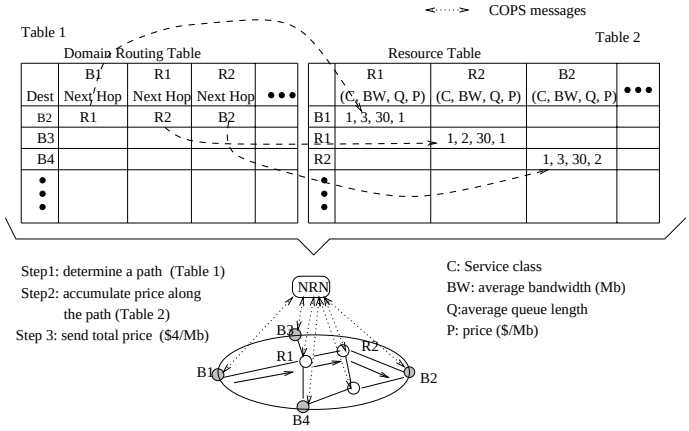

Fig. 1. Price formulation in RNAP-C

vary with the availability of network resources. Network resources are obtained by user applications through negotiation between the Host Resource Negotiator (HRN) on the user side, and a Network Resource Negotiator (NRN) acting on behalf of the network. The HRN negotiates on behalf of one or multiple applications belonging to a multimedia system. In an RNAP session, the NRN periodically provides the HRN updated prices for a set of services. Based on this information and current application requirements, the HRN determines the current optimal transmission bandwidth and service parameters for each application. It re-negotiates the contracted services by sending a *Reserve* message to the NRN and receives a *Commit* message to confirm or deny the request.

The HRN only interacts with the local NRN. If its application flows traverse multiple domains, resource negotiations are extended from end to end by passing RNAP messages hop-by-hop from the first-hop NRN until the destination network NRN, and vice versa. End-to-end prices and charges are computed by accumulating local prices and charges as *Quotation* and *Commit* messages travel hop-by-hop upstream towards the HRN.

The NRN maintains local state information for a domain for charging and other purposes. It makes the admission decision and decides the price for a service, based on the service specifications alone, or by also taking into account routing and configuration policies, and network load. In the latter case, the NRN sits at a router that belongs to a link-state routing domain (for example an OSPF area) and has an identical link state database as other routers in the domain. This allows it to calculate all the routing tables of all other routers in the domain using Dijkstra's algorithm.

The NRN maintains a domain routing table which finds any flow route that either ends in its own domain, or uses its domain as a transit domain (Fig. 1). The domain routing table will be updated whenever the link state database is changed. A NRN also maintains a resource table which allows it to keep track of the availability and dynamic usage of the resources (bandwidth, buffer space). In general, the resource table stores resource information for each service provided at a router. The resource table allows the NRN to compute a local price at each router (for instance, using the usage-based pricing strategy described in Section III). For a particular service request, the NRN first looks up the path on which resources are requested using the domain routing table, and then uses the per-router prices to compute the accumulated price along this path. The resource table also facilitates monitoring and provisioning of resources at the routers. To enable the NRN to collect resource information, routers in the domain periodically report local state information (for instance, average buffer occupancy and bandwidth utilization) to the NRN. In this paper, we extend COPS [20] for this purpose.

To compute the charge for a flow, ingress routers maintain per-flow (or aggregated flow from neighboring domain) state information about the data volume transmitted during a negotiation period. This information is periodically transmitted to the NRN, allowing the NRN to compute the charge for the period. The NRN uses the computed price and charge to maintain charging state information for each RNAP session.

A network domain manages its own pricing scheme (which may be congestion sensitive or static) independent of other domains, and will have its own per unit resource costs for each class. When an user flow traverses multiple domains, RNAP messaging collates pricing and billing information from each domain and determine the total price/charge for the user.

To reduce the overhead of per-flow RNAP message processing and storage, we consider a sink-tree based aggregation scheme in [8]. RNAP messages and state information are aggregated in the core networks, allowing data measurement and charging to be at much larger granularity.

V. Simulation Model

In this section, we describe our simulation model for the CPA and FP policies. We simulate a single DiffServ service domain, where resources are not explicitly reserved for each flow, but admission control may be used to control overall load. User resource requirements are declared explicitly through RNAP, allowing admission control to be enforced if required in an experiment. The individual and total user resource demands are also obtained through measurement. Price and network statistics are signaled to users through RNAP.

We used the *network simulator* (ns) [21] environment to simulate two network topologies, shown in Fig. 2 and Fig. 3. Topology 1 contains two backbone nodes, six access nodes, and 24 end nodes. Topology 2 contains five backbone nodes, 15 access nodes, and 60 end nodes. Topology 2 was also used in [22]. All links are full duplex and point-to-point. The links connecting the backbone nodes are 3 Mb/s, the links connecting the access nodes to the backbone nodes are 2 Mb/s, and the links connecting the end nodes to the access nodes are 1 Mb/s. At each end node, there is a fixed number N_s of sending users. We use topology 1 in most of our simulations to allow congestion to be simulated at a single bottleneck node, and use topology 2 to illustrate the CPA performance under a more general network topology [19].

We modified the DiffServ module developed by Sean Murphy to support dynamic SLA negotiation, and monitor the user traffic at ingress point. A Weighted-Round-Robin scheduler is modeled at each node, with weights distributed equally among EF, AF, and Best Effort (BE) classes. Although the DiffServ proposals mention 4 AF classes with three levels of drop precedence in each, we only simulated one AF class to make the simulations less resource-intensive, since this does not affect the general results in any way. Three different buffer management algorithms are used for different DiffServ classes - tail-dropping for EF, RED-with-In-Out [23] for AF, and Random Early Detection [24] for the BE traffic. The default queue length for EF, AF

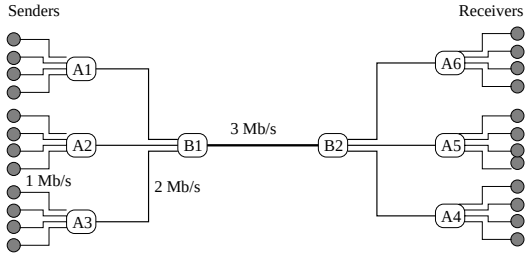

Fig. 2. Simulation network topology 1

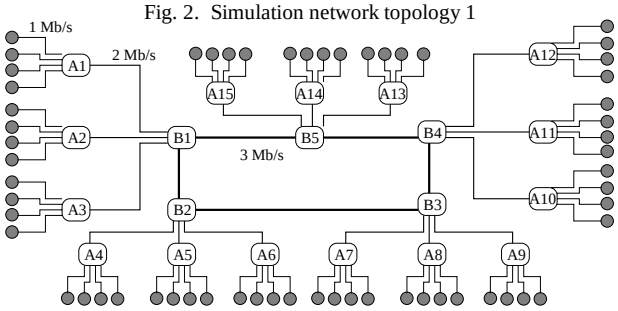

Fig. 3. Simulation network topology 2

and BE are set to 50, 100, 200 packets, respectively. Other parameters are set to the default values in ns.

A combination of exponential on-off and Pareto on-off traffic sources are used in the simulation. Unless otherwise specified, the traffic consists of 50% of each for all the service classes, and the on time and off time are both set to 0.5 seconds. The shape parameter for Pareto sources is set to 1.5. The mean packet size is set to 200 bytes. The traffic conditioners are configured with one profile for each traffic source, with peak rate and bucket size set to the 0n-off source peak rate and maximum amount of traffic sent during an on period respectively for both EF and AF classes.

We also characterize the system load by *burst index* and *offered load*. The burst index is defined as *OffTime/(OnTime + OffTime)* for both types of on-off sources. The offered load for a service class is defined as the ratio between the total user resource requirement for a service type, and the configured class capacity at the bottleneck. Under the FP policy, the total user resource requirement is also the actual resource demand from all the users. Under the CPA policy, the total user resource requirement is what the total resource demand would be if there were no resource contention at the bottleneck and the network did not impose an additional congestion-dependent price.

User requests are generated according to a Poisson arrival process and the lifetime of each flow is exponentially distributed with an average length of 10 minutes. In topology 1, users from the sender side independently initialize unidirectional flows towards randomly selected receiver side end nodes. N_s flows will be initialized at one node. At most $12N_s$ flows (60 sessions with N_s set to 5) can run simultaneously in the whole network. In topology 2, all the users initialize unidirectional flows towards randomly selected end nodes. At most $60\,N_s$ users (360 sessions with N_s set to 6) are allowed to run simultaneously in the whole network.

For ease of understanding, all prices in this section are given in terms of price per minute of a 64 kb/s transmission, currently equivalent to a telephone call. The basic price charged by the FP policy, and the basic usage price charged by CPA (p_{basic}), are both set to \$0.08/min. We set the target average load of the EF class at 40%, the AF class at 60%, and the BE class at 90%. Therefore, based on the pricing strategy proposed in Section III, the usage price for EF, AF and BE classes are set respectively as \$0.20/min, \$0.13/min, and \$0.089/min. When admission control is enforced, the holding price for the CPA policy is correspondingly set to \$0.067/min for EF class, and \$0.044/min for AF class.

Congestion pricing is applied when instantaneous usage exceeds the target load threshold of each class or when the loss or delay exceeds $1/3$ of the bounds at a node associated with the class (delay bound of 2 ms, 5 ms, and 100 ms respectively for EF, AF, and BE, and loss bounds of 10^{-6}, 10^{-4} and 10^{-2}, respectively). The price adjustment procedure is also controlled by a pair of parameters, the price adjustment step σ from equation 9 and the price adjustment threshold parameter θ, defined in Section III. Unless otherwise specified, values of $\sigma = 0.06$ and $\theta = 0.05$ are used.

The users are assumed to have the general form of the utility function shown in Section II. At the beginning of each experiment, the user population is divided into users of the EF, AF and BE classes, although in some experiments they are allowed to adapt to price changes by switching to a different class.

For EF users, the elasticity factor factor w (which is also the user's willingness to pay), is uniformly distributed between \$0.13/min and \$0.40/min for a 64 kb/s bandwidth. For AF and BE users, it is uniformly distributed between \$0.09/min and \$0.26/min, and \$0.06/min and \$0.18/min respectively. The minimum delay and loss requirements for each type of users are set to be the same as the expected performance bound of the corresponding service class. The opportunity cost parameter U_0 is set to the amount a user is willing to pay for its minimum bandwidth requirement, and is hence given by $U_0 = p_{high} \cdot x_{min}$, where p_{high} is the maximum price the user will pay before terminating his connection altogether. Users re-negotiate their resource requirements with a period of 30 seconds in all the experiments. The total simulation time for each experiment is 20,000 seconds.

We use a number of engineering and economic metrics to evaluate our experiments. The engineering metrics include the average traffic arrival rate at the bottleneck, the average packet delay, the average packet loss rate, and the user request blocking probability. The averages are computed as exponentially weighted moving averages. The economic performance metrics include the average user benefit (the perceived value obtained by users based on their utility functions) and the end-to-end price for each service class.

VI. Results and Discussion

In this section, we simulate the FP policy and CPA policy under identical traffic conditions, and compare their performance. For ease of presentation, a single traffic parameter for the AF class was varied in each experiment, and its effect on CPA and FP performance was studied. We conducted four groups of experiments. In the first and second groups, we vary the load burstiness and average load of the AF class, and evaluate CPA and FP. In the third experiment, incentive-driven traffic migration between classes is shown to improve the overall system performance. In the last experiment, we show that access control to a service class is critical in maintaining expected performance levels. Combining access control with user service adaptation reduces the request blocking rate significantly.

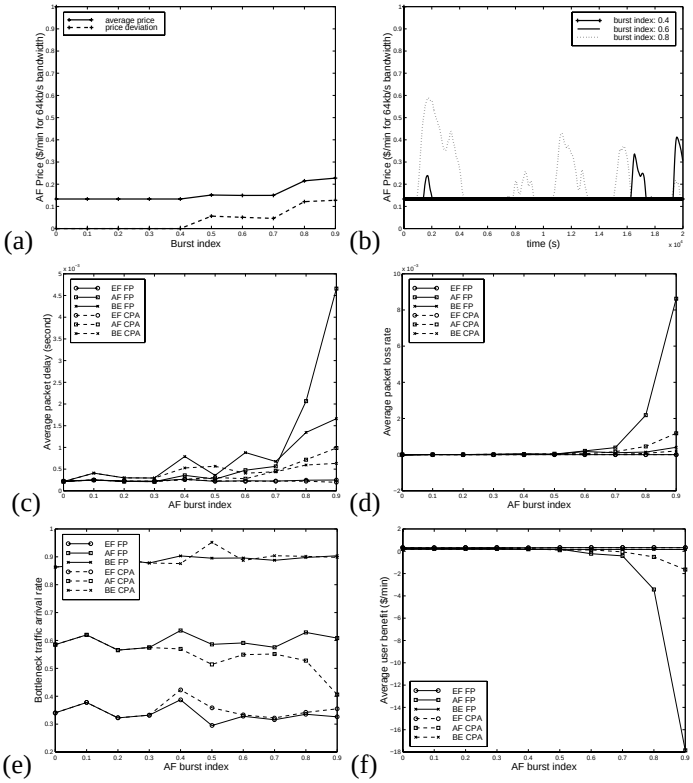

Fig. 4. System dynamics under CPA with increase in AF traffic burst index: (a) price average and standard deviation of AF class; (b) variation over time of AF. Performance metrics of CPA and FP policies as a function of burst index of AF class: (c) average packet delay; (d) average packet loss; (e) average traffic arrival rate; (f) average user benefit.

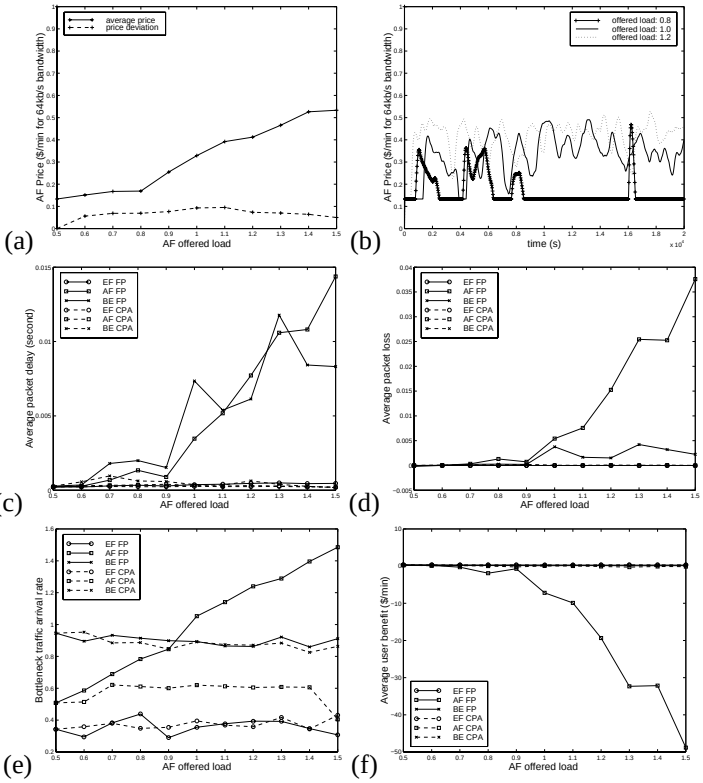

Fig. 5. System dynamics under CPA with increase in AF offered load: (a) average and standard deviation of AF class price; (b) variation over time of AF class price. Performance metrics of CPA and FP policies as a function of AF offered load: (c) average packet delay; (d) average packet loss; (e) average bottleneck traffic arrival rate; (f) average user benefit.

A. Effect of Traffic Burstiness

We first compare the performance of FP and CPA policies as the burst index of AF class increases, at a constant average offered load of 60%.

Fig. 4 (a) shows that the average AF price increases under CPA due to the increasing congestion price as the burst index exceeds 0.4. In response, the AF traffic backs off. Fig. 4 (a) also shows that the standard deviation in the AF price increases with the burst index, indicating greater fluctuations in the price. Fig. 4 (b) shows the dynamic variation of the AF class price at three different levels of burstiness, confirming this trend.

Fig. 4 (c) and (d) show that under FP policy the average packet delay and loss of the AF class increase sharply as the burst index exceeds 0.4. As a result of the user traffic back-off under CPA the delay and loss of AF class are well controlled below the respective performance bounds of 5 ms and 10^{-4} up to a burst index of 0.8. The average user benefit for CPA (Fig. 4 f) decreases due to the reduction of bandwidth, but remains higher than that of the FP policy. There is also a smaller degradation in the performance of the BE class at high burst indices. This appears to be because the BE class operates under a relatively high load, and therefore borrows bandwidth from the AF class when the AF class is lightly loaded. It can no longer do so when the AF traffic burstiness increases.

The results in this section indicate that the CPA policy takes advantage of application adaptivity for significant gains in network performance, and perceived user benefit, relative to the fixed-price policy. The congestion-based pricing is stable and effective.

B. Effect of Traffic Load

In this simulation, we keep the load and burstiness of the EF class and BE class and the burst index of the AF class at their default values, and vary the offered load of the AF class. The average AF price under CPA is seen to increase with offered load (Fig. 5 (a)). The standard deviation of the price changes only slightly and reaches a maximum around full load. Initially, the price deviation increases due to the more aggressive congestion control. At heavy loads, the increased multiplexing of user demand smooths the total demand, and therefore reduces fluctuations in the price. Fig. 5 (e) shows that the actual arrival rate of AF under CPA backs off as users adapt to the higher price.

Figs. 5 (c) and (d) show that the delay and loss of AF class under FP quickly increases after the offered load increases above 0.6 and approaches the provisioned capacity. As a result, the performance bounds for AF class can no longer be met. The high AF load also degrades BE performance. This is apparently because BE operates at a high load (0.9) and tends to borrow bandwidth from AF and EF when the latter classes are lightly loaded.

Figs. 5 (c), (d), and (e) show that CPA coupled with user adaptation is able to control congestion and maintain the total traffic load of a service class at the targeted level, and hence allows the service class to meet the expected performance bounds. Similar to our observation in Section VI-A, if the nominal price of the system correctly reflect long-term user demand, dynamic pricing driven service re-negotiation can effectively limits short-term fluctuations in load. The usage price of a class should be adjusted if persistent high user demand exist for a service.

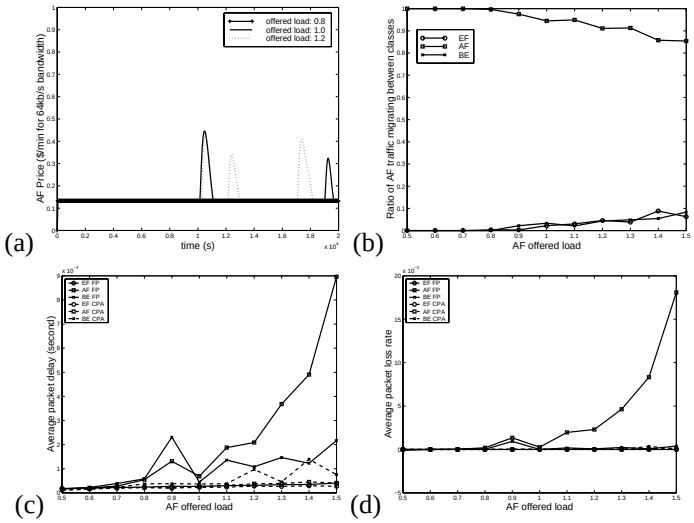

Fig. 6. Performance metrics of CPA and FP policies with traffic migration between classes: (a) variation over time of AF class price; (b) ratio of AF class traffic migrating through class re-selection; (c) average packet delay of all classes; (d) average packet loss of all classes;

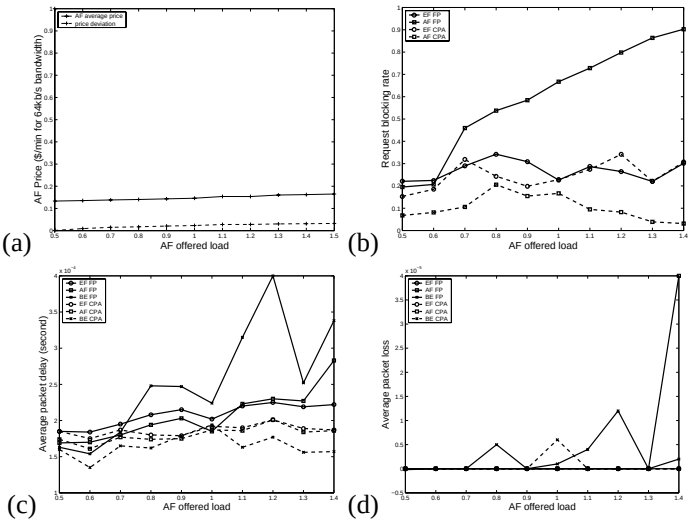

Fig. 7. System dynamics under CPA with access control CPA as AF offered load increases: (a) average and standard deviation of AF class price. Performance metrics of CPA and FP policies with access control as a function of AF offered load: (b) user requests blocking rate; (c) average packet delay; (d) average packet loss.

C. Load Balance between Classes

We saw in the previous section that the performance of a class will suffer if the load for that class is too high. In general, a user under the CPA policy will select a service class which provides it the highest benefit based on the price and performance parameters of a class as announced by the providers. The performance parameters are generally based on long-term statistics. In this section, we assume that a user can learn from network performance data received over a short period, and select the class that would provide the highest benefit based on the user utility function, network performance statistics and service price, as discussed in Section II.

In this simulation, the EF and BE classes are loaded at 30% and 80%, respectively. When the load of the AF class increases, the performance of AF class degrades and the congestion price is incurred. In response, some applications switch from the AF class to the EF class, which provides better performance guarantees, or the BE class, which allows it more bandwidth at a cheaper price. As the result of this re-selection, the load is better balanced across classes, and overall performance of the system improves (Fig. 6 (c) and (d)). Fig. 6 (a) shows that with load balancing in combination with adaptation within a single class, the congestion price needs to be invoked much less often than with adaptation within a class only, as in Fig. 5 (b). The proportion of migrating traffic is shown in Fig. 6 (b). We see even when a small portion of users select other service classes, the performance of the over-loaded class is greatly improved.

D. Effect of Admission Control

We have seen that the performance of a class cannot be predicted without access control. In this section, we compare the performance of FP and CPA for a network with admission control for the EF and AF classes. The admission threshold for each class is set to 1.5 times the target load to increase the efficiency of the network.

With admission control, the performance of EF and AF classes are well controlled (Fig. 7 c and d). However, due to the burstiness of the traffic, the blocking rate under FP is high even at a very small offered load (Fig. 7 b), and increases almost linearly as the offered load increases beyond 0.6. With congestion control and service contract re-negotiation, the blocking rate of CPA is seen to be up to 30 times smaller than that under the FP policy, and actually starts to decrease after reaching a maximum at offered load 0.8. This is because the price adjustment step is proportional to the excess bandwidth above the targeted utilization and increases progressively faster with offered load at higher loads, and the user bandwidth request decreases proportionally with the price according to the general utility function of Section II. Compared to Section VI-B, the average price under CPA (Fig. 7 a) is bounded to a smaller value at high offered loads, and and has a smaller fluctuation. The average price at the highest load is only 0.16 $/min, 0.03 $/min higher than that of the FP policy.

The results indicate that access control is important in maintaining the expected performance of a class. However, admission control by itself may lead to a high blocking rate due to the network dynamics. By combining admission control with user traffic adaptation, the network is more efficiently used. With admission control, the dynamics of the network price can also be better controlled, so that users have a more reliable expectation of the price.

VII. Related Work

Microeconomic principles have been applied to various network traffic management problems. The studies in [25][26][27] are based on a maximization process to determine the optimal resource allocation such that the utility (a function that maps a resource amount to a satisfaction level) of a group of users is maximized.

In [28][29][27][30], the resources are priced to reflect demand and supply. Some of these methods are limited by their reliance on a well-defined statistical model of source traffic, and are generally not intended to adapt to changing traffic demands. The study in [28] shows that compared to traditional flat pricing, service-class sensitive pricing results in higher network performance. Pricing for DiffServ has also been studied in [13]

through equivalent bandwidth. As has been pointed out earlier, equivalent bandwidth may be too conservative for resource provisioning in a DiffServ environment, and hence pricing based on equivalent bandwidth may not be fair to the users. Also, it is not trivial for users to adapt their requirements dynamically to meet their equivalent bandwidth constraints.

Although there is some overlap between the cited work and ours, our work is directed to studying and solving somewhat different problems – developing a pricing model for DiffServ, and studying DiffServ performance in a dynamic service and price negotiation environment.

VIII. Summary

We have developed several aspects of a DiffServ pricing model, proposing a price structure for different service classes in DiffServ based on their relative performance, long-term demand, and short-term fluctuations in demand. We have integrated this pricing model into a dynamic service negotiation environment in which service prices increase in response to congestion, and users adapt to price increases by adapting their sending rate and/or choice of service. We have also modeled the demand behavior of adaptive users based on a perceptually reasonable user utility function.

Our simulation results show that the different DiffServ classes provide different levels of service only when they operate at different target utilization. In the absence of explicit admission control, a service class loaded beyond its target utilization (under either sustained or bursty loads) no longer meets its expected performance levels. Under these conditions, a congestion-sensitive pricing policy (CPA) coupled with user rate adaptation is able to control congestion and allow a service class to meet its performance assurances under large or bursty offered loads. Users see a reasonably stable service price and are able to maintain a very stable expenditure. Allowing users to migrate between service classes in response to price increase and network performance further stabilizes the individual service prices while maintaining system performance.

When admission control is enforced beyond a threshold load for each class, performance bounds can be met with a fixed service price. However, in this case, the CPA policy provides a greatly reduced connection blocking rate at high loads by driving down individual bandwidth requests, resulting in a higher overall user satisfaction. Compared to the CPA policy without admission control, the service price is further stabilized in this case.

In this paper, we assume that users do not have the option of choosing a different path or provider, reflecting current network reality. However, pricing in the presence of competition or alternative paths remains an interesting open issue.

References

[1] K. Nichols and S. Blake, "Differentiated services operational model and definitions," Internet Draft, Internet Engineering Task Force, Feb 1998. Work in progress.

[2] V. Jacobson, K. Nichols, and K. Poduri, "An expedited forwarding PHB," Request for Comments 2598, Internet Engineering Task Force, Jun 1999.

[3] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski, "Assured forwarding PHB group," Request for Comments 2597, Internet Engineering Task Force, Jun 1999.

[4] X. Wang and H. Schulzrinne, "Comparison of adaptive internet multimedia applications," *IEICE Transactions on Communications*, Jun 1999.

[5] X. Wang and H. Schulzrinne, "RNAP: A resource negotiation and pricing protocol," in *International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'99)*, (Basking Ridge, New Jersey), pp. 77–93, Jun 1999.

[6] S. Shenker, C. Partridge, and R. Guerin, "Specification of guaranteed quality of service," Request for Comments (Proposed Standard) 2212, Internet Engineering Task Force, Sept. 1997.

[7] J. Wroclawski, "Specification of the controlled-load network element service," Request for Comments (Proposed Standard) 2211, Internet Engineering Task Force, Sept. 1997.

[8] X. Wang and H. Schulzrinne, "An integrated resource negotiation, pricing, and qos adaptation framework for multimedia applications," in *IEEE JSAC*, vol. 18, 2000.

[9] R. Guérin, H. Ahmadi, and M. Naghshineh, "Equivalent capacity and its application to bandwidth allocation in high-speed networks," *IEEE JSAC*, vol. 9, pp. 968–981, Sept. 1991.

[10] C. Lambrecht and O. Verscheure, "Perceptual quality measure using a spatio-temporal model of human visual system," in *Proc. of IS&T/SPIE*, Feb 1996.

[11] W. A. and M. A. Sasse, "Evaluating audio and video quality in low-cost multimedia conferencing systems," in *Interacting with Computers*, vol. 8, pp. 255–275, 1996.

[12] J. Janssen, D. D. Vleeschauwer, and G. H. Petit, "Delay and distortion bounds for packetized voice calls of traditional PSTN quality," in *Proceedings of the 1st IP Telephony Workshop (IPTel 2000)*, (Berlin, Germany), pp. 105–110, Apr 2000. GMD Report 95.

[13] C. Courcoubetis and V. Siris, "Managing and pricing service level agreements for differentiated services," in *Proc. of 7th IEEE/IFIP International Workshop on Quality of Service (IWQoS'99)*, (London, UK), Jun 1999. GMD Report 95.

[14] P. Reichl, S. Leinen, and B. Stiller, "A practical review of pricing and cost recovery for internet services," in *Proc. of the 2nd Internet Economics Workshop Berlin (IEW '99)*, (Berlin, Germany), May 1999.

[15] J. Altmann, B. Rupp, and P. Varaiya, "Internet user reactions to usage-based pricing," in *Proceedings of the 2nd Berlin Internet Economics Workshop (IEW '99)*, (Berlin. Germany), May 1999.

[16] H. Varian, *Microeconomic Analysis*. W.W. Norton & Co, 1993.

[17] F. P. Kelly, S. Zachary, and I. Zeidins, "Notes on effective bandwidths," in *Stochastic Networks: Theory and Applications*, pp. 141–168, 1996.

[18] X. Wang and H. Schulzrinne, "Incentive-compatible adaptation of internet real-time multimedia," technical report, Columbia University, Apr 2000.

[19] X. Wang and H. Schulzrinne, "Performance study of congestion price based adaptive service," in *International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'00)*, (Chapel Hill, North Carolina), pp. 1–10, Jun 2000.

[20] J. Boyle, R. Cohen, D. Durham, S. Herzog, R. Rajan, and A. Sastry, "The COPS (common open policy service) protocol," Request for Comments 2748, Internet Engineering Task Force, Jan 2000.

[21] "Network simulator - ns (version 2)
," http://www-mash.CS.Berkeley.EDU/ns/.

[22] M. Creis, "Rsvp/ns: An implementation of rsvp for the network simulator ns-2," Document, Computer Science Department, University of Bonn.

[23] D. D. Clark and W. Fang, "Explicit allocation of best-effort packet delivery service," *IEEE/ACM Trans. Networking*, vol. 6, pp. 362–373, Aug 1998.

[24] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Networking*, vol. 1, pp. 397–413, Aug 1993.

[25] J. F. MacKie-Mason and H. Varian, "Pricing congestible network resources," Sept. 1995.

[26] H. Jiang and S. Jordan, "A pricing model for high speed networks with guaranteed quality of service," in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, Mar 1996.

[27] D. F. Ferguson, C. Nikolaou, and Y. Yemini, "An economy for flow control in computer networks," in *Conference on Computer Communications (IEEE Infocom), (Ottawa, Canada)*, Apr 1989.

[28] R. Cocchi, S. Shenker, D. Estrin, and L. Zhang, "Pricing in computer networks: Motivation, formulation, and example," in *IEEE/ACM Transactions on Networking*, Dec 1993.

[29] N. Anerousis and A. A. Lazar, "A framework for pricing virtual circuit and virtual path services in atm networks," in *ITC-15*, Dec 1997.

[30] E. W. Fulp and D. S. Reeves, "Distributed network flow control based on dynamic competive markets," in *Proceedings International Conference on Network Protocol (ICNP'98)*, Oct 1998.