A Strategy for Implementing Smart Market Pricing Scheme on Diff-Serv

Murat Yuksel, Shivkumar Kalyanaraman
Rensselaer Polytechnic Institute, Troy, NY
yuksem@cs.rpi.edu, shivkuma@ecse.rpi.edu

***Abstract*— In this paper we present a baseline implementation strategy for the well-known Smart Market pricing scheme on diff-serv. Our strategy tries to violate Smart Market's theoretically defined properties as less as possible. In order to suit diff-serv framework, we propose ways of focusing Smart Market's complex operations at the edges while keeping the interior simple. Based on the proposed implementation strategy, we develop packet-based simulation of Smart Market. By simulation, we then investigate Smart Market's performance in terms of stability, fairness, and service differentiation on UDP and TCP traffic. We also look at importance of packet sorting (i.e. sorting of packets at routers according to their bids as proposed in Smart Market) in Smart Market's performance. By several simulations, we find that packet sorting does not really improve the performance for all the three metrics (stability, fairness, and service differentiation). So, it is not necessary to implement packet sorting, which significantly reduces necessary router upgrades for Smart Market's possible deployment.**

***Index Terms*—Network Pricing, Congestion Pricing, Quality-of-Service, Congestion Control**

I. Introduction

Pricing has recently attracted significant attention for the purpose of achieving economic efficiency in the Internet. Many researchers have proposed pricing schemes [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12] as an option for managing both resource allocation and network congestion. Many of these proposals have focused on congestion pricing. However, these responsive pricing schemes have not been deployed mainly because of excessive upgrades required by the proposals.

For example, Kelly et al.'s Packet Marking [1] (also known as Proportional Fair Pricing) scheme requires all network routers to send feedback to users, which requires upgrade to all the routers and low time-scale user involvement. Low et al.'s optimization flow control [2] framework requires routers to implement a dynamic pricing algorithm and send feedbacks to users about the price, which requires similar implementation upgrades as Packet Marking. Wang and Schulzrinne's RNAP [12] again requires all network routers to be upgraded because it requires all local routers to participate in determining price of edge-to-edge capacity.

One of the earliest pricing proposals is MacKie-Mason and Varian's [5] Smart Market, which is specifically designed to address both resource allocation and congestion management issues and therefore is referred to as a congestion-sensitive pricing scheme. This scheme charges the users on a packet-by-packet basis depending upon the current level of congestion in the network, and the users in turn lower their demands according to their utility. In other words, it theoretically achieves both economic efficiency (optimal distribution of rates between users [13]) and network efficiency (e.g. high utilization, low queue length) goals. However, the way it is defined, Smart Market is not possible to implement because it requires several upgrades to the current wide area networks.

In this paper we focus on developing a baseline implementation strategy and simulation of the Smart Market on the diff-serv [14] architecture. We specifically investigate necessity of packet sorting (to be explained in Section II) at routers. We also investigate Smart Market's abilities in terms of service differentiation and fairness.

The paper is organized as follows: First, we describe details of the Smart Market briefly in Section II. Next in Section III, we outline our implementation strategy for the Smart Market and determine deployment limitations for it. Section IV develops a set of simulation experiments and presents results along with discussions. Finally, Section V implications and conclusions of the work and proposes possible future work.

II. The Smart Market Scheme

This section presents important and overall characteristics of the Smart Market scheme. The Smart Market imposes a per-packet-charge, which reflects incremental congestion costs (which could be zero). The price-per-packet varies dynamically depending on the level of congestion in the network. Users assign a *bid* value for each packet sent into the network. The network bottlenecks maintain a current threshold value and only pass those packets with bids greater than the threshold value. The packets with bids less than the threshold are simply dropped. This threshold value depends on the level of congestion at the particular router, and is adjusted by that router. For each packet, the user pays the highest threshold value among all routers that the packet passed through, called the *market-clearing price*. As another requirement for the routers, Smart Market requires the successful (i.e. the ones that are not dropped) packets are sorted according to their bid values at each router before they are served. This behavior imitates an auction where the capacity is divided among the bidding users on a packet-by-packet basis.

Though the Smart Market scheme is theoretically attractive, we can observe some implementation and deployment issues.

This work is sponsored by NSF under contract number ANI9819112 and co-sponsored by Intel Corporation.

A preliminary version of this work was published in the Proceedings of Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS) part of SCS Western Multi-Conference (WMC), January 2001.

For example, the Smart Market scheme does not provide a guaranteed service to users and can lead to packet re-ordering. Moreover, the "bidding" procedure requires support at end-systems (or proxy agents) and the "clearing" procedure is required at all potential bottlenecks in the network [5]. Within these limitations, we now design an implementation strategy for the Smart Market scheme.

III. Implementation Strategy

There are two key implementation issues of the Smart Market scheme:

- How to communicate the necessary information (customer's bids, network's charges to the customer) between customers and the network?
- How to calculate threshold value at a local router?

We now address these issues within diff-serv framework.

A. Communication Between Customers and The Network

The diff-serv architecture classifies network routers (or nodes) into two types: edge router (ER) and interior (core) router (IR). It constrains complex data and control plane functionality to be implemented at network edges to simplify the core. For a traffic flow passing through a diff-serv domain, the ER at the entry point is called as *ingress*-ER and the one at the exit point is called as *egress*-ER.

To implement Smart Market on diff-serv, we propose that the sender (or proxy) sets the bid value, b, in the packet and sends it to the network. The packet passes through an ingress-ER and series of IRs, each of which has a threshold value T. IRs simply drop the packet if it does not satisfy the condition $b \geq T$. If the packet satisfies the condition, it is placed into a priority queue and sorted according to its bid value. The priority queue may potentially reorder packets, which leads to negative effects on TCP congestion control [15], [16]. We will investigate importance of the sorting of packets later by experiments in Section IV.

For the communication of the bid and clearing price, there is a need for two fields in packet headers. The *bid field* is written only by the customer and is read by IRs. The *clearing price field* (initialized to 0 by the customer) is updated at each IR to the maximum of the prior value of the field and the current threshold value, T, at that particular IR bottleneck. In other words, if the value of the threshold, T, is greater than the value of the clearing price field of the packet, the value of T is copied into that field. Else, the field is left unchanged. When the packet reaches the egress edge router, it contains the maximum of the threshold at all the IRs it passed through.

The egress-ER acts a measurement proxy and accounts for the clearing price of the packet. In other words, the source pays the clearing price determined by the egress-ER. Egress-ERs accumulate each packet's clearing price and send periodic indications to each corresponding source. So far, everything in the Smart Market can be adapted to the diff-serv architecture, albeit with two new fields in the packet header. However, the information required by the customers in order to adjust their sending rates and bids require new feedback mechanisms. Theoretically, the Smart Market assumes that the customers are fed back such information immediately without any delay, which is not possible to implement on a wide-area network. So, an approximation is needed.

We propose to use a simple probing procedure which happens at fixed-time *probing intervals* set to be larger than round-trip time (RTT) as a way to handle this feedback problem. The customer (or the ingress-ER on behalf of the customer) sends a *probe packet* (in addition to data packets) to investigate the current status of the network at these fixed time intervals. This probe packet goes through IRs and finds the current clearing price. The egress-ER receives this probe packet and sends the customer a *feedback packet* containing the current clearing price of the network. The feedback packet also includes the total of clearing prices (bill) for that customer in the latest interval. The customer uses the feedback information to adjust her packet's bid values, capacity demand (number of packets to send) and available budget.

Note that if we want the IRs to treat the probe packets and the feedback packets just like data packets, they must have bid values as high as possible to ensure that they will not be lost and will encounter minimum delay. That means there has to be a maximum value for the bids of the data packets, which is a deviation from the Smart Market because it does not impose any limiting value for the bids. The fixed length bid field also implicitly constrains the size of bids. Alternatively, the IRs of the network have to behave differently for these probe and feedback packets. However, this will increase the processing time of a packet at the IRs, i.e. each packet will be checked whether it is a data, probe or feedback packet. We choose to normalize the bid values into a range (e.g. 0 to 1) and hence define a maximum value for the bids, i.e. 1. Once normalization (mapping to [0,1]) is done, there must also be a way of reversing this mapping back. What is going to be the actual money in dollars to charge for a clearing price of, for example 0.75? We currently leave this question, which is important for the service providers, unanswered. We simulated the Smart Market in ns [17] according to the ideas presented above. In the next section, we present the details of the simulation and our assumptions.

B. The Threshold Value T

MacKie Mason and Varian [5] determine the congestion price of a packet as

$$T = p = \frac{n}{K} D'(X/K) \tag{1}$$

where n is the total number of customers in the network, K is the capacity of the network, X is customer's capacity demand, and $D(X/K)$ is the delay experienced by customer. So, we propose that the IRs maintain fixed time *update intervals* at the end of which they calculate the rate of change in the delay, $D'(X/K)$, and update the threshold value, T. Specifically, threshold value for update interval i is calculated as follows:

$$T[i] = \begin{cases} T[i-1] & , X[i-1] - X[i-2] = 0 \\ \frac{n}{K} \frac{D[i-1]-D[i-2]}{X[i-1]-X[i-2]} & , otherwise \end{cases} \tag{2}$$

where $D[i]$ is the delay experienced by the customer in interval i. Notice that the term $\frac{D[i-1]-D[i-2]}{X[i-1]-X[i-2]}$ corresponds to the term $D'(X/K)$ in (1).

IV. Simulation Experiments

A. User Model

We model the user's utility function with the well-known logarithmic utility function $u(x) = wlog(x)$ [1], [2], [18], where x is the capacity given to the user, and w is user's willingness-to-pay. User will maximize her surplus by making sure that her capacity demand is $x = w/p$ where p is the current clearing price. Notice that the parameter w is equivalent to user's budget for expenditures to network service. So, in our simulations, user i is initially assigned a budget value W_i, and at the end of every probing interval the user i adjusts her bids such that her capacity demand is w_i/p for the next interval.

B. Experimental Configuration

We perform our experiments on single-bottleneck and multi-bottleneck topology. The single-bottleneck topology has a bottleneck link, which is connected to n edge nodes at each side where n is the number of users. The multi-bottleneck topology has $n-1$ bottleneck links, that are connected to each other serially. There are again n ingress and n egress edge nodes. Each ingress edge node is mutually connected to the beginning of a bottleneck link, and each egress node is mutually connected to the end of a bottleneck link. All bottleneck links have a capacity of 10Mb/s and all other links have 15Mb/s. Propagation delay is 0.1ms on bottleneck link(s) and 10ms on all other links. Figure 1-a shows a single-bottleneck topology with $n = 3$. Figure 1-b shows multi-bottleneck topology with $n = 4$. The white nodes are edge nodes and the gray nodes are interior nodes. These figures also show the traffic flow of users on the topology. To ease understanding the experiments, each user sends its traffic to a separate egress. For the multi-bottleneck topology, one user sends through all the bottlenecks (i.e. long flow) while the others cross that user's long flow.

In order to investigate importance of packet sorting (please refer to Section III-A) on performance, we simulate two versions of Smart Market: Smart Market with Sorted Queues (SM-SORTED), Smart Market with FIFO Queues (SM-FIFO). We run experiments both on UDP and TCP traffic. The UDP traffic will have an average packet size of 1000B.

The initial value of T at all IRs is set to 0.1. The probing interval at ERs and the update interval at IRs are set to 1sec, unless otherwise said so.

C. Experiments

1) System Dynamics and Stability: To see Smart Market's dynamics, we first run an experiment of SM-SORTED on the single-bottleneck topology with three flows as represented in Figure 1-a. The flows generate UDP traffic. Budgets of flows 0, 1, 2 are $w_0 = 100$, $w_1 = 75$, $w_2 = 25$ respectively. Total simulation time is 3000 seconds. Initially, only flow 0 is active, and the other flows get active with 1000 seconds intervals. Figure 2-a shows the instantaneous queue at the bottleneck, and Figure 2-c shows bottleneck utilization. Controlled queue and high utilization show that Smart Market provides stable operation. Figure 2-b shows instantaneous rates of the flows. We can observe that flows share the bottleneck capacity in proportion to their budgets. Figure 2-d shows the instantaneous threshold value at the bottleneck. As new flows join in, Smart Market adapts its threshold value accordingly.

In order to compare SM-SORTED and SM-FIFO in terms of system dynamics, we run a series of experiments on the single-bottleneck topology for various values of the update and probing intervals from 1 to 30 seconds. Moreover, to see traffic effects we run the experiments for both UDP and TCP traffic. Figures 2-e, 2-f, and 2-g show average bottleneck queue length, maximum bottleneck queue length, and average bottleneck utilization respectively for various values of update and probing intervals. We observe that SM-SORTED performs significantly worse (approximately 30%) in utilizing the bottleneck, which causes almost zero average and maximum queue length. This is because packet sorting causes extra timeouts and hence causes TCP to back off unnecessarily.

In general, we observe that packet sorting affects system performance negatively. We can see this by comparing results for SM-SORTED and SM-FIFO on TCP traffic, and also by comparing results for SM-SORTED and SM-FIFO on UDP traffic. For example, SM-SORTED on TCP traffic utilizes bottleneck a lot less than SM-FIFO on TCP traffic. Also, from Figure 2-g, SM-SORTED on UDP traffic utilizes less than SM-FIFO on UDP traffic when update interval exceeds 15 seconds.

Overall, both versions of Smart Market (i.e. SM-SORTED and SM-FIFO) perform better on UDP traffic than TCP traffic. This is mainly due to burstiness of TCP traffic.

Also, we can oserve that as the update and probing intervals get larger performance metrics get worse for all the cases. This is because Smart Market's fidelity of control lowers as update/probing intervals get larger.

2) Service Differentiation: We run a series of experiments for SM-SORTED and SM-FIFO on the single-bottleneck topology with UDP and TCP traffic. There are two flows in the experiments and for each experiment we vary the ratio of their budgets from 1 to 200, i.e. $w_0/w_1 = 1..200$. Given the budget ratio w_0/w_1, we then observe ratio of the two flows's rates during the simulation. Figure 2-h shows results of these experiments. The horizontal axis shows the ratio of the flows's budgets, which is set at the beginning of simulation, and the vertical axis shows the ratio of the two flows's average rates observed during the simulation. Observe that only SM-SORTED on TCP traffic cannot differentiate the two flows, while Smart Market is able to differentiate them pretty well in all other cases. The reason why SM-SORTED on TCP does not perform well in service differentiation is again due to the negative effects of packet sorting on TCP performance.

3) Fairness: Our last series of experiments are on the multi-bottleneck topology with $n = 14$ bottlenecks (the case of $n = 3$ is shown in Figure 1-b). All the flows have equal budget of 10 \$/Mb, and they generate UDP traffic. We simulate SM-SORTED. Our aim is to observe behavior of the long flow (i.e. flow 0)'s rate as number of bottlenecks on its way increase.

At time 0, only the long flow is active. The other flows (i.e. cross flows) join in one after another with an interval of 1000 seconds. Total simulation time is 15,000 seconds. So, as new flows join in the number of bottlenecks in the system increases. Figure 2-i shows the long flow's rate as the number of bottle-

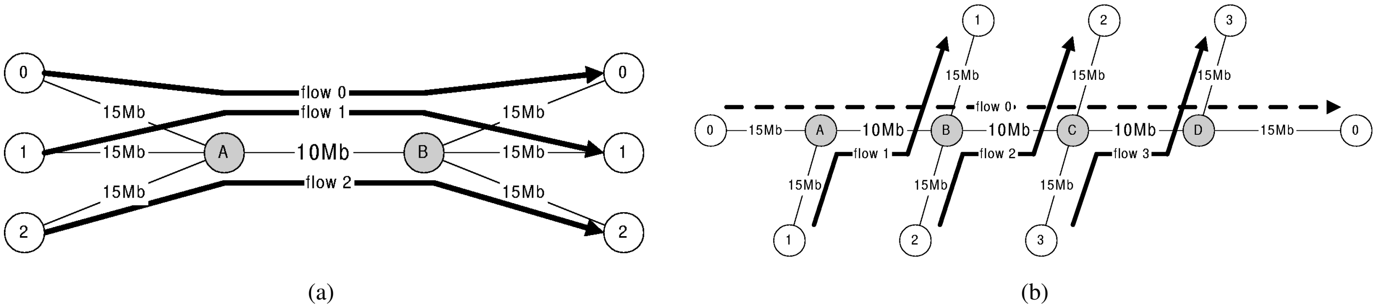

Fig. 1. (a) Single-bottleneck (b) Multi-bottleneck network for Smart Market experiments.

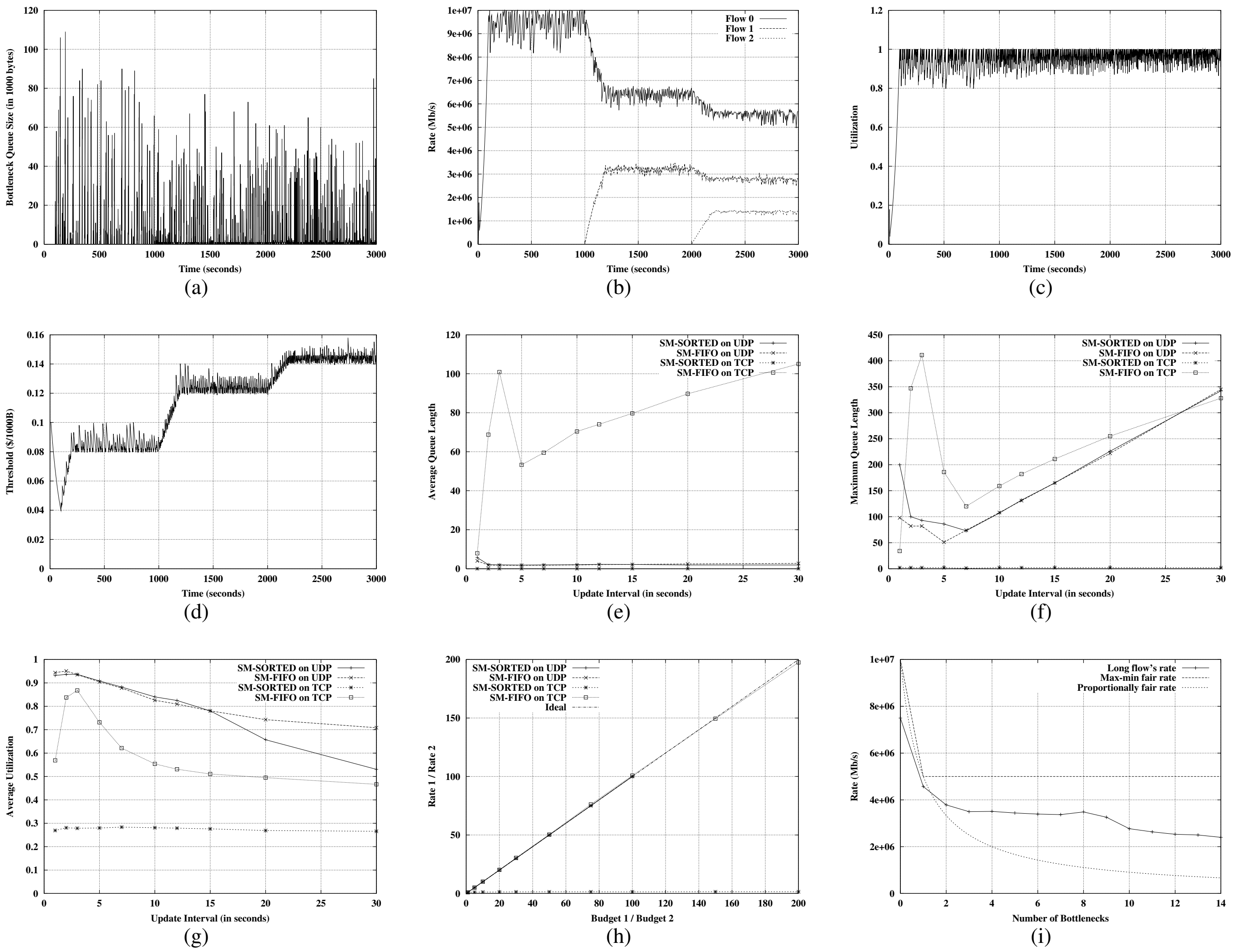

Fig. 2. Simulation results of SM-SORTED with UDP traffic on single-bottleneck topology: (a) Bottleneck queue length (b) Rate given to flows (c) Threshold and utilization at the bottleneck (d) Threshold at the bottleneck. Comparison of SM-SORTED and SM-FIFO on UDP and TCP traffic: (e) Average queue length (f) Maximum queue length (g) Average utilization (h) Service differentiation. (i) Simulation results of SM-SORTED with UDP traffic on multi-bottleneck topology. Long flow(flow 0)'s rate relative to max-min fair and proportionally fair rates versus number of bottlenecks.

necks increases on its way. The figure also plots theoretical rates of the long flow for max-min and proportional fair cases. In the max-min fair case, the long flow and the cross flows share the bottleneck capacity equally, i.e. 10/2 Mbps in our experimental topology. In the proportional fair case, the long flow gets less than the cross flows in proportion to the number of bottlenecks on its way, i.e. the long flow gets $10/(r+1)$ Mbps and the cross flows get $10r/(r+1)$ Mbps in our experimental topology. We observe from Figure 2-i that Smart Market allocates the bottleneck capacity in such a way that it is between max-min and proportional fair rate allocations.

In the definition of Smart Market, each flow pays the clearing-price for its route, which is the maximum of the bottleneck thresholds in the route. For our experimental topology, the long flow should be paying approximately the same price as the cross flows since the bottleneck capacities are equivalent. So, one may expect that the capacity allocation our experiment should be max-min fair. The result shows that it is not. The reason behind this is that the long flow is experiencing more delay (both propagation delay and queuing delay) than the cross flows. This makes effective capacity for the long flow less than it is supposed to be, which in turn causes the long flow to get less of the capacity than the cross flows.

V. Conclusions and Discussions

We investigated the difficulties in implementing the Smart Market, a well-known congestion-sensitive pricing scheme for the Internet, on a network with diff-serv architecture. We found that the Smart Market cannot be implemented to a real network without important changes (e.g. modeling, packet format, architectural issues), limitations on deployment (e.g. requires upgrades in both hosts and routers) and offered workload (e.g. effect on TCP flows). We proposed the following major changes to implement the Smart Market on diff-serv architecture:

- delay in feeding back the congestion information of the network to the customers
- mapping the threshold value of the interior routers to an interval such as [0,1]
- concentrating more functionality at the ERs versus less functionality at the IRs to suit a diff-serv implementation

By applying the above changes, we developed a packet-based simulation for the Smart Market and presented simulation results along with their analysis. We observed that Smart Market is able to control congestion with low bottleneck queue length and high bottleneck utilization. Also, we observed that packet sorting at IRs on TCP traffic negatively affects system performance significantly. To see the real importance of packet sorting in Smart Market's performance we simulated the Smart Market with and without packet sorting, i.e. SM-SORTED and SM-FIFO. Simulation results showed that packet sorting does not really improve system performance. In fact, we found that it degrades the system performance especially on TCP traffic, which is currently the dominant traffic type in the Internet. So, it makes more sense to implement SM-FIFO since it is a lot easier to implement.

Also, we have shown by simulation that Smart Market provides fairness in between max-min and proportional. Future work should consider multiple diff-serv domains case, and the Smart Market's behavior on bursty traffic patterns.

References

[1] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate control in communication networks: Shadow prices, proportional fairness and stability," *Journal of Operations Research Society*, vol. 49, pp. 237–252, 1998.

[2] S. H. Low and D. E. Lapsley, "Optimization flow control – I: Basic algorithm and convergence," *IEEE/ACM Transactions on Networking*, vol. 7, no. 6, pp. 861–875, 1999.

[3] J. K. MacKie-Mason, L. Murphy, and J. Murphy, *Responsive pricing in the Internet*, Eds McKnight and Bailey, MIT Press, 1997.

[4] J. K. MacKie-Mason and H. R. Varian, "Pricing the congestible network resources," *IEEE Journal on Selected Areas of Communications*, vol. 13, pp. 1141–1149, 1995.

[5] J. K. MacKie-Mason and H. R. Varian, *Pricing the Internet*, Kahin, Brian and Keller, James, 1993.

[6] A. Gupta, D. O. Stahl, and A. B. Whinston, *Priority pricing of Integrated Services networks*, Eds McKnight and Bailey, MIT Press, 1997.

[7] D. Clark, *Internet cost allocation and pricing*, Eds McKnight and Bailey, MIT Press, 1997.

[8] R. Cocchi, S. Shenker, D. Estrin, and L. Zhang, "Pricing in computer networks: Motivation, formulation and example," *IEEE/ACM Transactions on Networking*, vol. 1, December 1993.

[9] A. M. Odlyzko, "A modest proposal for preventing Internet congestion," Tech. Rep., AT & T Research Lab, 1997.

[10] N. Semret, R. R.-F. Liao, A. T. Campbell, and A. A. Lazar, "Pricing, provisioning and peering: Dynamic markets for differentiated Internet services and implications for network interconnections," *IEEE Journal on Selected Areas of Communications – to be published*, 2001.

[11] N. Semret, R. R.-F. Liao, A. T. Campbell, and A. A. Lazar, "Market pricing of differentiated Internet services," in *Proceedings of IEEE/IFIP International Workshop on Qualityof Service (IWQoS)*, 1999, pp. 184–193.

[12] 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)*, 1999, pp. 77–93.

[13] H. R. Varian, *Intermediate Microeconomics: A Modern Approach*, W. W. Norton and Company, 1999.

[14] S. Blake et. al, "An architecture for Differentiated Services," *IETF RFC 2475*, December 1998.

[15] M. Allman, V. Paxson, and W. Stevens, "TCP congestion control," *IETF RFC 2581*, April 1999.

[16] E. Blanton and M. Allman, "On making TCP more robust to packet reordering," *ACM Computer Communication Review*, vol. 32, no. 1, January 2002.

[17] "UCB/LBLN/VINT network simulator - ns (version 2)," http://www-mash.cs.berkeley.edu/ns, 1997.

[18] S. Kunniyur and R. Srikant, "End-to-end congestion control: Utility functions, random losses and ecn marks," in *Proceedings of Conference on Computer Communications (INFOCOM)*, 2000.