Simulating the Smart Market Pricing Scheme on Differentiated-Services Architecture*

M. Yuksel[1] and S. Kalyanaraman[2]
Department of Computer Science[1]
Department of Electrical, Computer, and Systems Engineering[2]
Rensselaer Polytechnic Institute
Troy, NY, 12180, USA
yuksem@cs.rpi.edu, shivkuma@ecse.rpi.edu

Keywords: Internet Pricing, Network Efficiency, Economic Efficiency, Differentiated-Services

Abstract

Dynamic pricing techniques attracted significant attention as tools to implement better resource sharing, and address the issue of congestion in the Internet. Among the pricing schemes, the "smart market" is a purely congestion-sensitive scheme, which makes it a basis for comparing new pricing schemes. However, there are significant implementation challenges in this scheme. We investigate these issues and come up with an implementation strategy in the differentiated-services architecture. This strategy involves certain changes to the scheme and limitations on the traffic types. This paper reports our implementation strategy, scheme changes, and simulation results illustrating the performance of the smart market scheme.

1. INTRODUCTION

Pricing has recently attracted significant attention for the purpose of achieving both network and economic efficiencies in the Internet. Many researchers (MacKie-Mason, Murphy & Murphy[10]; MacKie-Mason & Varian[9] [10]; Gupta, Stahl & Whinston[6]; Clark[5]; Kelly et al.[7]) have proposed responsive pricing schemes as a option for managing *both* resource allocation and congestion levels of the network. However, these responsive pricing schemes have not been deployed because IP provides the minimal "best-effort" service model and lacks standardization in packet-forwarding behavior.

This implementation deadlock was recently broken as the Internet Engineering Task Force (IETF) standardized two approaches to support service differentiation: Integrated Services ("int-serv") [4] and Differentiated Services ("diff-serv") [2]. Though the two approaches can coexist and inter-operate, it is expected that the latter (diff-serv) or its variants will be the choice of service providers. Given this backdrop, it is important to map pricing schemes to this implementation architecture, and study performance and deployment issues.

Flat-rate pricing [3] is the most common mode of payment today for bandwidth services today. It has minimal accounting overhead and encourages usage, but it cannot differentiate between users during congestion periods. *Usage-based pricing* is an alternative, which charges users according to the amount of data they sent or received. It may have an optional flat-price component. Observe that both flat-rate and usage-based pricing do not directly address the issue of congestion management.

On the other hand, the *smart market* mechanism (proposed by MacKie-Mason & Varian [9]) is specifically designed to address both resource allocation and congestion management issues [7,8] 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 functions. In other words, it theoretically achieves both economic efficiency (optimal distribution of rates between users [12]) and network efficiency (e.g. high utilization, low queue length) goals.

In this paper, we focus on developing a baseline implementation strategy and simulation of the smart market on the diff-serv architecture. The paper is organized as follows: first, we describe details of the smart market briefly in Section 2. Next in Section 3, we outline our implementation strategy and deployment limitations. Section 4 describes simulation issues; and Sections 5 and 6 discuss simulation results followed by conclusions/future work.

2. 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 (cutoff) value and only pass those packets with bids greater than the cutoff value. This threshold value depends on the level of congestion at the particular router, and is adjusted by that router. Finally, user pays the highest threshold value among all routers that it passed through, called the "market-clearing price". Moreover, the successfully sent packets are sorted according to their bid values at each bottleneck. This behavior simulates an auction where the capacity is divided among the bidding packets on a packet-by-packet basis.

Though the smart market scheme is theoretically attractive, we can observe some implementation and deployment issues. 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 [9]. Within these limitations, we now design an implementation strategy for the smart market scheme.

3. IMPLEMENTATION STRATEGY

There are three key implementation challenges of the smart market scheme: a) How to communicate the customer's bidding parameters to the potential bottlenecks? b) How to balance the implementation requirements on core and edge nodes in a diff-serv framework? c) How to deal with parameter sensitivity, non-linearity and stability issues on a packet-based simulation?

*This work has been supported by NSF contract number ANI9819112.

The diff-serv architecture has two types of nodes: edge and core. It constrains complex data and control plane functionality to be implemented at network edges to simplify the core (Figure 1). 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 edge node and series of Interior Routers (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

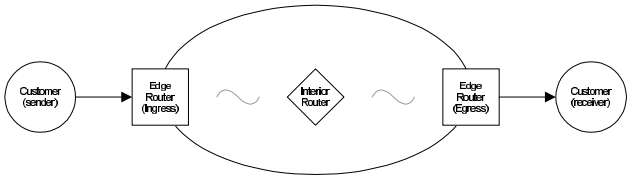

Figure 1: Representation of Differentiated-Services architecture.

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 interactions with TCP (end-to-end transport) congestion control [1]. Therefore, we are forced to limit our workloads to non-TCP flows, i.e. a significant limitation of the scheme since over 90% of current traffic use TCP.

For the communication of the bid and clearing price, we would need 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 edge node acts a measurement proxy and accounts for the clearing price of the packet. In other words, the source should pay the clearing price determined by the egress node. The egress node can accumulate each packet's clearing price and send periodic indications to each source. Assuming UDP flows, so far everything in the smart market can be adapted to the diff-serv architecture, albeit with a new field in the packet header. However, the information required by the customers in order to define their utility functions 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 real wide-area network. So, an approximation is needed.

We propose the use of fixed time intervals at edge routers (ERs) and IRs set to be larger than round-trip time (RTT) as a way to handle this feedback problem. 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 node 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, 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 also 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* [11] according to the ideas presented above. In the next section, we present the details of the simulation and our assumptions.

4. SIMULATION OF THE SMART MARKET

4.1. Customer Model

Implementation of the customer model was the most problematic part of this scheme, because the smart market scheme does not define a demand model. In other words, what should a customer do when she is fed back the current status of the network? We propose an approximation to MacKie-Mason and Varian's utility function in [9]. MacKie-Mason and Varian propose that each customer should maximize $u(x) - D(Y) - px$ by selecting the best *x*, where *u(x)* is the utility of the customer, *Y* is the utilization of the network, *D(Y)* is the delay experienced by the customer, *p* is the current price charged for a packet in the network, and *x* is the customer's demand in terms of number of packets to send.

We first modeled indifference curves [12] for delay, *D*, with respect to the number of packets to send, *x*. To be satisfied at the same level, the customer must send more packets while having less marginal delay. This can be represented with a concave function, which implies that delay is a "bad" [12] and not a "good" (see Figure 2). The idea is that the number of packets to send must be inversely proportional to the price-per-packet currently advertised by the network. Several indifference curves can be found to represent the case, but all of them must be concave as in Figure 2. We used the following equation for representing the indifference curves in the simulation:

$$x = (aD + b)^2 \qquad (4.1)$$

where *a* and *b* are some constants. Note that this equation for indifference curves does not represent the most accurate relationship between *D* and *x* for the users of the Internet. Rather, we just selected a relationship, which yields to concave indifference curves as in Figure 2.

Utility function for these indifference curves can be defined in different ways. We selected the constant *b* in Equation (4.1) as the utility function because it perfectly orders the indifference curves from left to right, i.e. the very left indifference curve gets the lowest utility value and the curves at right get higher and higher

utility values. By using this idea, we can get the utility function in terms of x and D as

$$u(x, D) = \sqrt{x} - aD \qquad (4.2)$$

which leads to marginal utility of

$$u'(x, D) = \partial u(x, D) / \partial x = 1 / 2\sqrt{x}. \qquad (4.3)$$

Since customers have to find the best value of x to maximize $u(x, D) - px$, they must equate their demand (number of packets to send, x) to the marginal utility. [9] This results in

$$p = u'(x, D) = 1 / 2\sqrt{x}. \qquad (4.4)$$

Notice that customers know the value of the current clearing price when they are fed back by the egress nodes. So, they want to

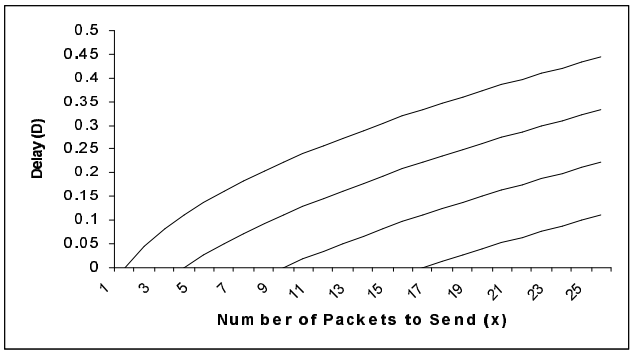

Figure 2: Sample indifference curves for delay and number of packets to send.

find the best value of x to use in the next interval. In other words, customers should adjust the number of packets to send (i.e. their demand) for the next time interval as specified in Equation (4.5).

$$x = 1 / 4p^2 \qquad (4.5)$$

This implies that customers should adjust their demand inversely proportional to the clearing price. Additionally, in our implementation, we introduce a constant factor to Equation (4.5) to compensate the incompatibility of the units (i.e. price vs. number of packets to send).

We have defined the procedure of finding the best value of x for the customers. However, it might be the case that x is not *affordable* with the current budget of the customer. So, the customer has to take the budget constraint into account. Once the customer finds the best value of x, she can now see whether she can afford sending that amount of packets with the current clearing price. If her budget is not enough for that, then she just decreases the value of x to the appropriate value, i.e. $x = Budget / p$. Finally, after finding the affordable value of x, the customer bids randomly between the clearing price and the maximum bid value that she can afford. Actually, this bidding procedure is an issue, but we propose the use of random bidding within the affordable range of the customer for simplicity.

4.2. The Cutoff Value, *T*

One big question was how to calculate the threshold value, T, at IRs. The authors of [11] determine the congestion price of a packet as $T = p = \frac{n}{K} D'(Y)$, where n is the total number of customers in the network, K is the capacity of the network, Y is the utilization of the network, and $D(Y)$ is the delay experienced by customer. Although n/K is not constant for a real network, we assumed it to be a constant for simplicity. The IRs also maintain fixed time intervals at the end of which they calculate the rate of change in the delay, $D'(Y)$, and update the threshold value, T, by mapping it to the interval [0,1].

This mapping is important especially for defining actual price per unit bandwidth. One would expect a linear relationship between the total budgets of the customers and the amount of bandwidth. In other words, if the total amount of traffic that can be transmitted through the network is 100 packets (assuming that packet size is fixed) and the sum of all customers' budgets for the network service is \$100, the network should charge \$1 per packet in steady state. If the total budget is \$200, it should charge \$2 and so on. Notice that our simulator defines T in the interval [0,1]. This means T cannot be used as the actual price per packet. Thus, there is also a need for mapping T values to actual prices, which is out of scope of this paper.

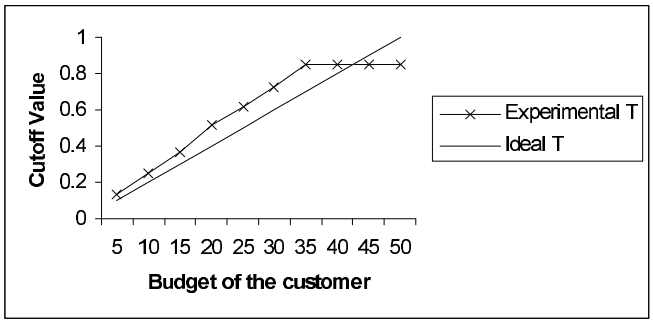

Figure 3: Steady state cutoff value, T, for different customer budgets.

In steady state, our simulator results in T values almost linearly proportional to the total budget of the customers. To show that proportionality, we ran 10 experiments on our simulator of the smart market. The experiments included single customer (sender) sending traffic to a single bottleneck, which can transmit 50 packets in one time interval. In each experiment the customer had a different budget of going from 5 to 50 with increment 5. Figure 3 represents the steady state T values that are converged by the simulator in the experiments versus the customer's budget in each experiment. It shows that the experimental steady state T values are very close to the ideal case of linearity. The experimental steady state T values are not exactly linear because of the parameter sensitivity of the theoretical formulas, which is also mentioned in the next section in more detail.

4.3. Parameter Sensitivity

Since the definition of the smart market [9] is made solely in a theoretical perspective, we had to deal with sensitivity of the theoretically defined parameters during our simulation experiments. We found that the experimental network was not stable. Instead, it was making large oscillations basically because of the sharp oscillations of T values at IRs. Observe that the variability of T is linked to the variability of queuing delays. In other words, the smart market scheme is highly sensitive to the

queuing delays. To handle that we introduced two new operations: First, we smoothed the changes in the T value over successive intervals by a smoothing factor as in the following line of code:

```
T += smoothing_factor_*(newT-T);
```

By using a smoothing factor of 0.3, we got smaller oscillations but not good enough for steady state. We realized that the calculation of T always resulted in some change in the value of T, even though the current value of T was good. This is basically because the smart market cannot be implemented in a real network completely as it was defined. The simulation simply cannot find the correct value for T and also cannot feedback the congestion information to the customer immediately. So, we added one more condition to the calculation of T as in the following line of code:

```
if (util>thresh_util_ && AvgQLength<100) return T;
```

Basically, the value of T is kept the same if utilization of the bottleneck is high enough and the queue length is low enough. We got good results after adding this above condition. One important problem is to set the best value for `thresh_util_`.

Unfortunately, due to the sensitivity of T on queuing delays, there is a trade-off between setting the `thresh_util_` high and having a high probability of reaching to steady state. When we set `thresh_util_` to 0.9 for example, we saw that the system was not able to reach steady state for very long time. In some cases, it could not reach steady state at all. This is because the optimum value of T cannot be found. Actually, if we decrease the smoothing factor to a very small value (e.g. 0.001), we can guarantee that the system will reach to steady state. However, it will require very long time to find the optimum value of T. We found that the best value for `thresh_util_` is 0.7. In other words, we may have to lose 30% of utilization for the sake of stable steady state. The problem of breaking this steady state versus transient tradeoff in the smart market is still open.

5. SIMULATION RESULTS

The smart market scheme was originally designed for congestion management (network efficiency) and resource allocation (economic efficiency) goals. However, we have had to make several compromises on the scheme for implementation reasons. Therefore an interesting question is whether the scheme still achieves its stated goals, which is the goal of this simulation section.

Figure 4 shows the basic configuration we used in the experiments. Customers at the left (senders) send constant bit rate UDP traffic (each packet 1000 bytes) to the customers at the right (receivers). Each customer makes its connection through ERs specific to her. There are only two IRs in the network. They stand at the ends of the single bottleneck in the network. The bottleneck has 1Mbps of bandwidth and 10ms of propagation delay, and all the other links have 100Mbps of bandwidth and 1ms of propagation delay. Notice that RTT of a customer's packet is 24ms in this network configuration. The two parameters, which affect economic efficiency, are the total budget of the senders and T, the threshold value at IRs, in steady state.

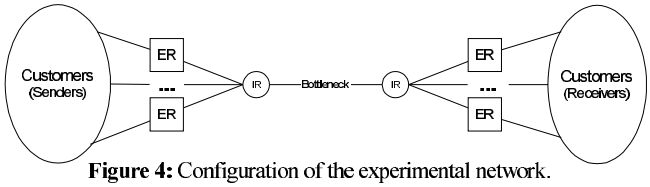

Figure 4: Configuration of the experimental network.

We use the following three metrics to characterize "network efficiency": bottleneck queue length, bottleneck utilization and packet drop rate. "Economic efficiency" is evident from the graphs of customer rate allocations, which measure rate distribution over time.

We assume that each customer has a fixed budget for each interval. The budget values defined for the experiments in Table 1 express the fixed budget values that the customers intend to spend for network service. In Experiment 1, for example, Customer 1 has 10 units of budget available for each individual interval regardless of whether she has spent the budget for the previous intervals or not. We make this assumption for simplicity.

Table 1 lists the description of each experiment in terms of the number of customers (senders) and their available budget for each interval. We make two experiments with two customers having equal and unequal budgets. We also make two more experiments with 4 and 5 customers having respectively increasing budgets, i.e. Customer 1 has 1 unit of budget, Customer 2 has 3 units of budget, and so on. For all of the experiments, the length of the time interval (to make feedback) is 0.40sec and the initial value of the cutoff value T is 0.1. Total simulation time is 40secs. Note that the maximum number of packets that can be sent through the bottleneck in a time interval is (1Mbps*0.40s)/8000b = 50 packets. In all of the experiments the drop rate was approximately 0.001%, which is very low, representing that the smart market is able to control congestion.

Figures 5, 7, 9, and 11 plot the utilization of the bottleneck, and the cutoff value, T of the experiments. Once steady state is reached the bottleneck is utilized more than 70%, which is not a very good performance. Also, we see a transient phase of approximately 10secs. Notice that this transient phase can be made even lower by changing parameters of the simulator.

Figures 6, 8, 10, and 12 represent the queue length at the bottleneck during the experiments. After the transient phase we see a very low queue length at the bottleneck, which also contributes to the fact that the smart market can control the congestion.

Experiment Number	Number of Customers	Budgets of the Customers	Total Budgets
1	2	10,10	20
2	2	10,20	30
3	4	1,3,5,7	16
4	5	1,3,5,7,9	25

Table 1: List of the experiments.

Although the smart market is able to control congestion with a low bottleneck queue length and low drop rate, it has the problem of resulting in low utilization in steady state, which violates network efficiency goals.

When we look at T values, which represent how accurately the bandwidth is being priced, we can also see that T values are almost proportional to the total budgets of the customers as we also stated at the end of Section 4.2. Total budget of the customers in Experiment 1 is 20 units, which means that the optimum value of T (price per packet) must be 20/50packets = 0.4. The simulator converges to the value of T to approximately 0.5 in steady state, which is very close to the optimum value. Similar arguments can be made for the T values of the other three experiments.

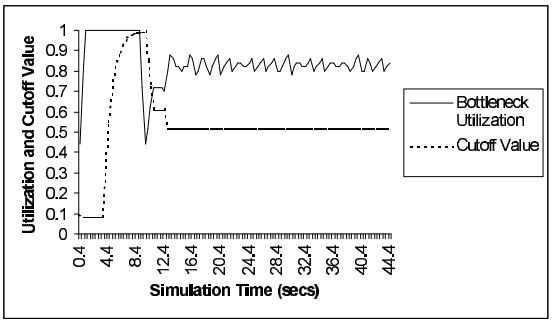

Figure 5: Bottleneck utilization and cutoff in Experiment 1.

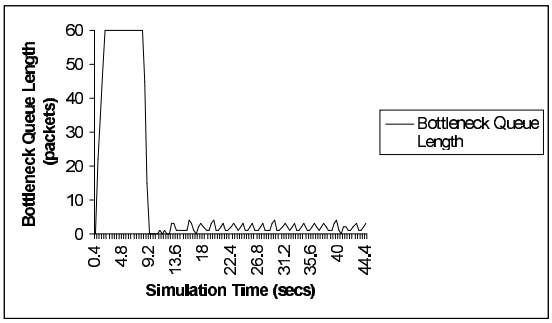

Figure 6: Queue length at the bottleneck in Experiment 1.

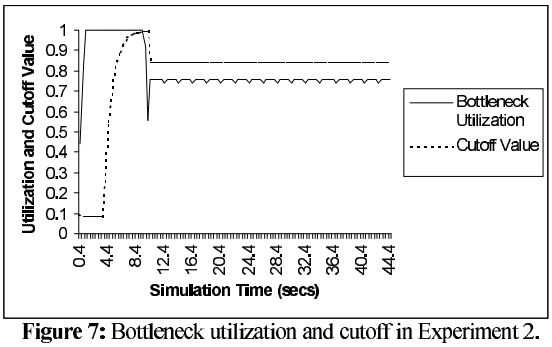

Figure 7: Bottleneck utilization and cutoff in Experiment 2.

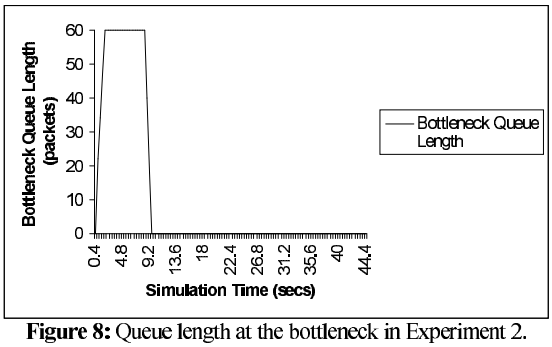

Figure 8: Queue length at the bottleneck in Experiment 2.

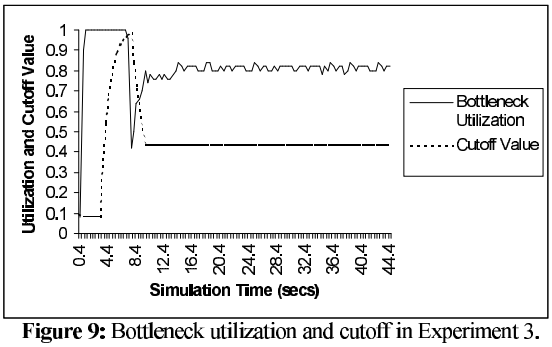

Figure 9: Bottleneck utilization and cutoff in Experiment 3.

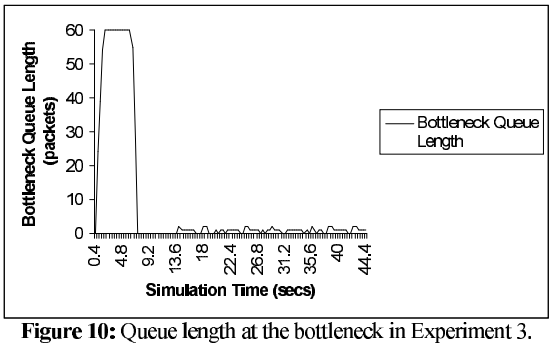

Figure 10: Queue length at the bottleneck in Experiment 3.

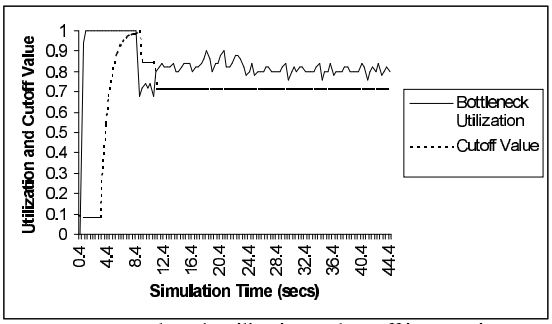

Figure 11: Bottleneck utilization and cutoff in Experiment 4.

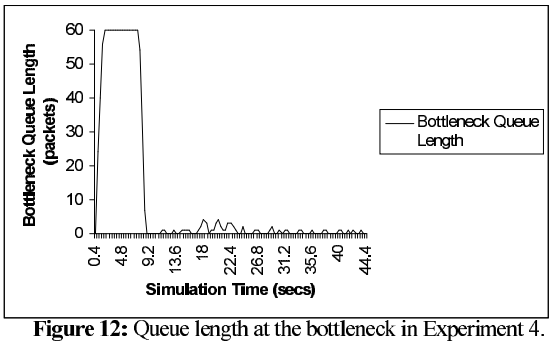

Figure 12: Queue length at the bottleneck in Experiment 4.

Figures from 13 to 16 plot the allocation of the bottleneck volume to the customers at each time interval during the experiments. On these graphs, we can also see the steady state behavior approximately after 10secs. More importantly, we can see the volume (bandwidth) allocation proportional to the customers' budgets. The two equally budgeted customers in Experiment 1 are allocated the same volume in steady state. In Experiment 2, Customer 1 is allocated with half of the volume given to Customer 2 as their budgets have the same proportionality. Also in Experiments 4 and 5, we can see the proportional increase in the allocated volume according to the customers' budgets.

In summary, not only the proportionality between the value of T and the total budget of the customers, but also the volume allocations proportional to customers' budgets represents that the

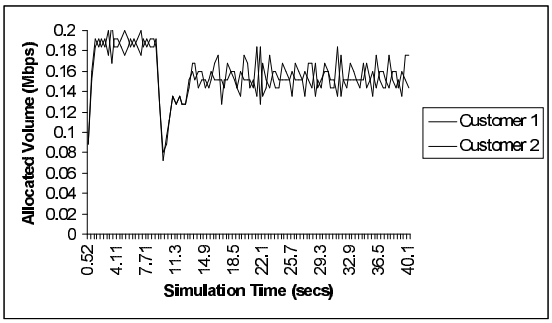

Figure 13: Volume allocations to customers Experiment 1.

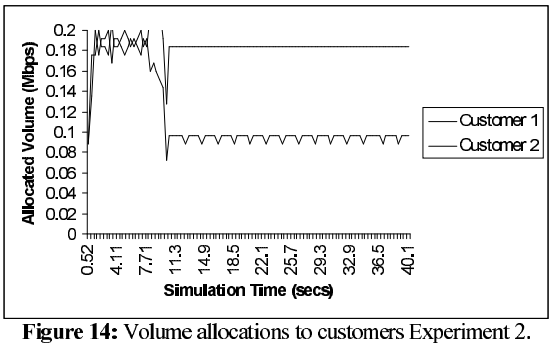

Figure 14: Volume allocations to customers Experiment 2.

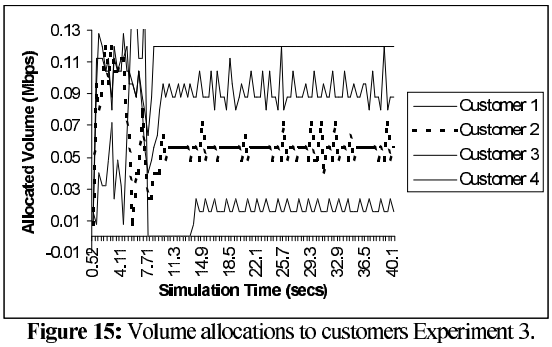

Figure 15: Volume allocations to customers Experiment 3.

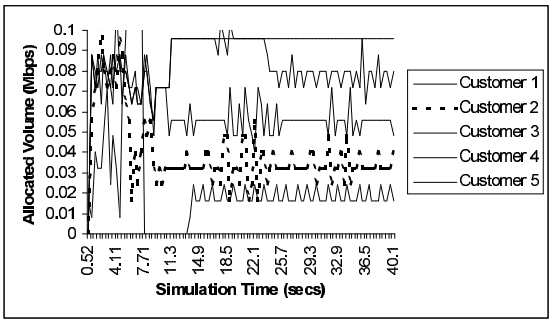

Figure 16: Volume allocations to customers Experiment 4.

smart market satisfies economic efficiency goals. On the other hand, due to the implementation compromises described earlier in this paper it fails to fully satisfy network efficiency goals. This is because the bottleneck utilization, a key metric of network efficiency, is low.

6. SUMMARY

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 (eg: modeling, packet format, architectural issues), limitations on deployment (eg: requires upgrades in both hosts and routers) and offered workload (eg: no TCP flows). We proposed the following major changes to implement the smart market on UDP flows:

- delay in feeding back the congestion information of the network to the customers
- mapping the cutoff value of the interior routers to an interval such as [0,1]
- updating the interior routers such that they maintain a queue ordered according to the "bid" values of the packets, which were assigned by the owner of the packet
- 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 simulator for the smart market comparable to simulators of possible new pricing schemes for the Internet, and presented simulation results along with their analysis. We observed that the smart market meets all *economic efficiency* [12] goals by pricing the bandwidth accurately and allocating the bottleneck volume to the customers proportional to their budgets. On the other hand, we also observed that the smart market fails to fully meet network efficiency goals, because it cannot utilize the bottleneck very well, although it is able to control congestion with low bottleneck queue length and drop rate.

Future work should include a thorough investigation of difficulties in implementing the smart market on TCP flows with consideration of multiple diff-serv domain case, and the smart market's behavior on bursty traffic patterns.

REFERENCES

[1] Allman, M.; Paxson, V.; Stevens, W. 1999. "TCP Congestion Control." IETF Internet RFC 2581, April.

[2] Blake, S. et al. 1998. "An Architecture for Differentiated Services." IETF Internet RFC 2475, December.

[3] Boyle, J. et al. 1998. "The COPS (Common Open Policy Service) Protocol." IETF Internet draft, <draft-ietf-rap-cops-02.txt>, August.

[4] Braden, R.; Clark, D.; Shenker, S. 1994. "Integrated Services in the Internet Architecture: An Overview." Internet RFC 1633, June.

[5] Clark, D. 1997. "Internet cost allocation and pricing." *Internet Economics, Eds McKnight and Bailey, MIT press.*

[6] Gupta, A.; Stahl, D. O.; Whinston, A. B. 1997. "Priority Pricing of Integrated Services Networks." *Internet Economics, Eds McKnight and Bailey, MIT press.*

[7] Kelly, F. P.; Maulloo, A. K.; Tan, D. K. H. 1998. "Rate control in communication networks: shadow prices, proportional fairness and stability." *Journal of the Operational Research Society*, 49, 237-252.

[8] MacKie-Mason, J. K.; Varian, H. R. 1995. "Pricing the congestible network resources." *IEEE J. Selected Areas Comm.*, 13, 1141-1149.

[9] MacKie-Mason, J. K.; Varian, H. R. 1993. "Pricing the Internet." *Public Access to the Internet, Kahin, Brian and Keller, James.*

[10] MacKie-Mason, J. K.; Murphy, L.; Murphy, J. 1997. "Responsive Pricing in the Internet." *Internet Economics, Eds McKnight and Bailey, MIT Press.*

[11] *ns*, `ucb/lbln/vint` network simulator – *ns* (version 2), 1997. `http://www-mash.cs.berkeley.edu/ns`.

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