

A Stochastic Equilibrium Model of Internet Pricing^{*,†}

by

Alok Gupta

**Department of Management Science and Information Systems
University of Texas at Austin**

Dale O. Stahl

**Department of Economics
University of Texas at Austin**

and

Andrew B. Whinston

**Department of Management Science and Information Systems
University of Texas at Austin**

December 1995

* This Research was funded in part by National Science Foundation #IRI - 9225010, but does not necessarily reflect the views of The NSF. Partial support was also provided by Hewlett Packard Corporation.

† An earlier version of this paper was presented at the Seventh World Congress of the Econometrica Society in Tokyo, Japan, August, 1995.

Abstract

Perhaps the greatest technological innovation of the next several decades will be universal access and utilization of the Internet. Already congestion is becoming a serious impediment to efficient utilization. We introduce a stochastic equilibrium concept for a general mathematical model of the Internet, and demonstrate that the efficient social welfare maximizing stochastic allocation of Internet traffic can be supported by optimal congestion prices. We further demonstrate that approximately optimal prices can be readily computed and implemented in a decentralized system by constructing a large computer simulation model. We also address the alternative of building excess capacity to avoid congestion.

1. Introduction

The "information revolution" is currently underway and it is made possible by increasingly powerful personal computers and the rapidly developing information superhighway (The Internet). This revolution will foster a profound transformation of the economic organization of society. Already there exists a multi-billion dollar industry providing on-line services such as financial market information, news, e-mail, shopping, and interactive games. Businesses will use network services for Electronic Data Interchange (EDI), advertising, and perhaps even for selling their products through "virtual stores". Computations are being done in a networked environment where a user's program may involve several different processors.

With the advent of WWW the access to information on the Internet has become easier and more user friendly - not to mention highly data intensive. The number of people who have acquired Internet connectivity is doubling every year. The bandwidth requirements of multimedia applications such as video conferencing and movies-on-demand are orders of magnitude greater than current uses (which are predominantly text-based). The challenge facing the Internet will be to ensure a smooth transition to this new multi-service multimedia environment.

To deliver an appropriate service quality the most important factor is the bandwidth available to transmit and/or receive data. At present the Internet backbone is comprised of T1 and T3 lines with the data transmission speeds of 1.5 Mbps and 45 Mbps respectively; this is orders of magnitude greater compared to a couple of years ago when almost the entire backbone was running at 56 kbps. In the next 5 years this increase in capacity may be increased to gigabit ranges. However, the number of servers and users both have increased enormously in the last 3 years and, as mentioned earlier, the services provided on the network have become much more data intensive. Therefore, congestion will be a growing problem; in fact, congestion is the main culprit which may foil the emergence of a global, interoperable, data communication network.

A computer network can be viewed as an economy whose producers are the computer servers and whose outputs are completed services/programs. We apply economic theory to address the issue of designing an overall mechanism that coordinates the use of the Internet so that total net benefits are maximized. Our approach is more general but compatible with approaches taken in the computer science literature [Dijkstra (1973); Kleinrock (1975, 1976), and Chu and Lan (1987)], and the management science literature [Naor (1969); Mendelson (1985); Sanders (1988a,b); and Mendelson and Whang (1990)]. These single-server models were extended for distributed computing in computer networks by Stahl and Whinston (1991). The current paper develops a model of the Internet, derives theoretical results, and presents calibrated testing of our approach via simulation.¹ Some other Internet pricing approaches are taken by Mackie-Mason and Varian, 1994; and Shenker, 1995.

Several interesting modeling issues arise. The first is the definition of the commodity in this Internet economy. From the point of view of the server, the natural concept for the commodity is a "cycle" of computer processing. Then the capacity of the server determines the processing speed in cycles per second. Each cycle is a perfectly private good and carries a unique time label. However, from the point of view of a user, it is unnatural to conceive of preferences in terms of cycles; rather, user preferences are more naturally conceived in terms of services. A specific service may be obtainable from several servers using a variety of programs and imposing varying loads on the network. Further, the preferences of the user will depend on the throughput time for the service as well as the whole temporal pattern of its service needs. Taking this approach leads one to construct Arrow-Debreu contingency claims markets for each cycle available at each instant of time

¹Gupta, Stahl, and Whinston (1996) develop the priority pricing approach for network computing environments and report a variety of uncalibrated simulation results. The value-added of the current paper consists of (i) the specific modelling of the Internet, (ii) a specification of user demand tailored to Internet services, (iii) simulations calibrated to represent the Internet, and (iv) estimation of the potential social welfare benefits from implementing optimal pricing. Gupta, Stahl, and Whinston (1995) is a non-theoretical introduction to issues involved in Internet pricing and presents early Internet modeling research and a report on preliminary (uncalibrated) results.

over the infinite future, contingent on the state of the network (in terms of the length of queues and expected throughput times). The sheer complexity of such an approach renders it impractical.

Instead, our approach avoids the infinity of contingent markets and posits spot markets only. We model user preferences as depending on the service acquired and the expected throughput time of that service. A rental price and an expected waiting time are associated with each priority class at each server. Given this data, a user evaluates and selects a cost-minimizing program and priority class. The demand (flow rate) of service requests from a client depends on the net benefits to the user, and hence is sensitive to the rental prices. Optimal service demands are then translated into loads on individual servers by priority class and aggregated over all users.

With a system of queues there is a trade-off between greater throughput volume and longer waiting times. An optimal allocation of resources will maximize total welfare of all users. We prove that there is a unique welfare-maximizing allocation, and we derive the rental prices by priority class that support this optimum as a "stochastic equilibrium". That is: (i) user service requests are optimal for each user given the rental prices and anticipated waiting times; (ii) the anticipated waiting times are the correct ex ante expected waiting times given the average flow rates; and (iii) the aggregate average flow rates are equal to the welfare-maximizing rates.

Notice that our concept of a stochastic equilibrium is quite different from an Arrow-Debreu equilibrium with complete markets in which the rights to each processing cycle at each server for all time would be auctioned off, and the precise loads for each moment of time would be determined. That approach would be hopelessly unrealistic and infeasible. Our concept also differs substantially from a Nash equilibrium [e.g. Lee and Cohen (1985)] in which every user must know every other user's preferences and demands for all time and must choose a strategy for submitting requests that is a best response to all other users' strategies. That approach is also unrealistic in its informational demands.

Rather than an exact allocation that these two familiar concepts seek, we seek a stochastic allocation - a determination of average flow rates - that permits some imperfections manifested by queues with positive expected waiting times, and we adjust rental prices to optimize the tradeoff between greater throughput volume and longer waiting times. We take for granted that the first-best solutions cannot be implemented in real time because they require infeasible computational and informational resources. Our solution, while "second-best", is an optimal solution within the constraints of having only spot prices rather than complete Arrow-Debreu contracts and limited common knowledge among the users. Further, we maintain that our solution is feasible from an informational and computational point of view.

Next, we turn to the question of implementing this solution. That is, how do we find the equilibrium rental prices by priority class? One approach would be centralized computation. However, in the interest of developing a decentralized mechanism, we favor the alternative approach based on the classic tatonnement process [Hahn, 1982]. We have conducted simulation testing of our proposed price-adjustment process and have found very encouraging results. Some results are presented here, a detailed report is the focus of another paper.

The paper is organized as follows. Section 2 presents the formal model of the Internet that lends itself to economic analysis. Section 3 describes user preferences, and choice of optimal computation plans. Section 4 derives our main theorems and shows how a welfare optimum can be supported as a stochastic equilibrium. All proofs are relegated to an Appendix. Section 5 presents implementation methods, and Section 6 discusses our simulation study. Finally, Section 7 concludes with an assessment of the power and practicality of our economic approach to the design and management of large-scale computer networks such as Internet, and a brief indication of areas for future research.

2. Description of the Model of the Internet

The Internet consists of a finite number of users, servers and other hardware units with well-defined, commonly known, possibly diverse capabilities. Examples of specialized hardware units include an ethernet system, multi-server queues that route services to the next available processing unit, external disk storage devices and controllers, and output interfaces. User-clients and servers are computers with diverse capabilities, with servers generally having more computational capacity, speed and often having specialized software, including business and home oriented data bases. Let M denote the set of all user-clients, servers and hardware units, and let $m \in M$ denote a generic "machine" in the Internet.

We assume that each machine is equipped with a priority queue system. For notional simplicity, we assume that each machine offers K classes of non-interruptable priority service, $K \equiv \{1, \dots, K\}$, with $k = 1$ being the first (or highest) priority.² For analytical simplicity, we assume unlimited queue capacity. We hasten to point out, however, that in equilibrium, the expected queue lengths are finite, so for practical purposes, sufficient finite queue capacity would suffice and yield qualitatively similar results.

The machines are connected with each other through a network. We model shared communication devices (such as data bases and ethernets) as distinct machines. Given this modeling trick, the abstract network will consist of direct connections only; that is, if m and m' are directly connected in the network representation, then there exists a dedicated physical linkage between m and m' that is not shared by any other machine. Let A_m denote the set of machines from which m can receive direct input, and let B_m denote the set of machines to which m can send direct output. We can formally represent the abstract

²It is straightforward to modify our results for interruptable priority service, or a mixture of interruptable and non-interruptable priority services. We chose the non-interruptable case because it is quite common, and because we wanted to keep the notation at a minimum.

network as $N \equiv \{((A_m, m), (m, B_m)), m \in M\}$. In addition, let C denote the subset of user-client machines: the machines where users interface with the Internet.

A typical Internet will support many programming languages. Let P denote the set of all finite "programs" in the languages of the Internet system. While a program could be a specification of required computations in a multi-processor network, it could also be interpreted as a specification of a collection of high-level services desired by a user, such as different news services, leaving to the user-client machine or a server the task of specifying the detailed code and routing. We can always structure programs so there are two distinguishable components: instructions and data. The instructions specify (among other things) the user-client machine where the program starts. A machine reads the instructions, carries out whatever operations are called for on the data (and instructions), and then routes the output according to the instructions. Machines also pass through to the output, instructions which are then carried out by successor machines. Thus, inputs to machines are viewed as "batch" programs complete with all relevant data and output disposition instructions.

A machine $m \in M$ can be represented by a triplet (v_m, f_m, q_m) , where v_m is the processing speed in cycles per second, $f_m(p)$ gives the output when $p \in P$ is the input, and $q_m(p)$ gives the expected number of cycles required to process input p . Then, $q_m(p)/v_m$ is the expected execution time of program p at machine m . Note that restricted access to a particular machine (such as a user's client) can be represented by a production function that performs the desired function only if p contains a specific access password, and otherwise outputs an access-denied message. In a similar vein, a machine not capable of executing some program, outputs an unable-to-read message, and if the run time exceeds the limit (if any) specified in the program, then it outputs a time-exceeded message.

We can now formally characterize the impact of any program $p \in P$ on the Internet: the expected total load on each machine $m \in M$. Suppose program p starts at machine m , the output is $p' \in f_m(p)$ and is sent to machine m' ; then machine m' produces output $p'' \in$

$f_{m'}(p')$ which is sent to machine m'' , which produces output $p''' \in f_{m''}(p'')$, which is finally returned to the originating machine m where it terminates. The total load on machine m is $Q_m(p) = q_m(p) + q_m(p''')$, the total load on machine m' is $Q_{m'}(p) = q_{m'}(p')$, and the total load on machine m'' is $Q_{m''}(p) = q_{m''}(p'')$. For any program p , given that p is started at the machine specified by the instructions, for each machine m , we define $Q_m(p)$ to be the expected total load on machine m imposed by program p until that program is terminated and exits from the Internet.³ More generally, given a program p started at the machine specified by the instructions, let $P_m(p)$ denote the set of "intermediate programs" (e.g., p, p'' in the above example) resulting from the execution of p that are processed by machine m . Then, $Q_m(p) = \sum_{p_m \in P_m(p)} q_m(p_m)$. We assume the user's client interface software can estimate $\{q_m(p), m \in M\}$.⁴

Given a stationary stochastic arrival process for services, let $w = \{w_{mk}, m \in M, k \in K\}$ denote the vector of expected queue waiting times at the machines. As just illustrated, a program may access a given machine several times in the course of executing. Let $\mu_m(p) = \# P_m(p)$, the number of intermediate programs of p that must be processed at machine m . Then, program p of priority class k has an expected waiting time of $w_{mk} \mu_m(p)$ at machine m .⁵

³While most machines may be deterministic, given idiosyncratic differences (such as different data), it may be more realistic to view the load as a random number, albeit with a known mean and small variance. Also some machines may route a job to successor machines based on the current queue of those machines, so the "path" through the system may be probabilistic. In the event that a user mistakenly attempted to start a program p at a machine incompatible with the program instructions, the machine would issue an unable-to-read message and terminate - thus imposing a negligible load on the Internet.

⁴This may appear to be a formidable task, especially in view of the "halting problem." However, we envision that almost all users will employ standardized commercial software in which case rule-of-thumb load estimation techniques could be developed and widely available. Moreover, the client software could update the rules-of-thumb based on accumulated experience. In the case of new user-specific programs, load estimation could be added to the compiler task in a new generation of compilers.

⁵What matters are the beliefs of the clients about the waiting times. We assume that the Internet provides reliable information to the user's client about w , which form the basis for common beliefs. We further assume that beliefs about w change at a much slower rate than programs move through the Internet, so a constant w for a given originating program p is justified.

Finally, assuming that a user chooses a single priority class, say k , for all phases of the execution of a program p started at the machine specified by the program's instructions, the total expected throughput time is

$$\tau(p,k,w) = \sum_m [Q_m(p) / v_m + w_{mk} \mu_m(p)] . \quad (1)$$

3. Users, Preferences and Demands

Let I denote the set of users, and for each $i \in I$ let $C_i \subset C$ denote the set of client machines to which user i has direct access. While there is a close linkage of users and clients, we will continue to employ both terms in specific contexts. The user is a human (or group of humans in an organizational team) with preferences which form the basis for choice. In contrast, the client is a machine, and as such, has no preferences of its own. While we may endow the client machines with sophisticated software to automate many of the decision-making functions, the source of value resides with the human user. For example, in the case of home use, a family member may want a collection of services from the network (a bundled product). The client machine in the home, configured for this family, would figure out the best way to obtain the desired services, and might give options such as: "you may obtain these services now for a cost of \$75, but if you can wait until after 5 PM, it will cost you only \$25."

It is extremely useful to think of users as desiring some specific Internet service (such as a financial report) while being indifferent over alternative programs that can deliver that service satisfactorily. We can then focus on the task of identifying efficient programs for given services. Let S denote the class of services potentially provided by the Internet. Services $s \in S$ can be viewed as subroutines which operate on the specific data, provided that the characteristics of the data (such as size and format) are compatible with the subroutine. Different qualities (such as precision) are represented formally as different services.

Since programs specify the client machine at which they must start, the set of feasible programs for user i is a subset of all possible programs. We let $P_i(s) \subset P$ denote the subset of programs that will successfully deliver service s for user i ; that is, if a user i initiates execution of program $p \in P_i(s)$ at the appropriate machine $m_i \in C_i$, then the Internet will execute the program and return a satisfactory output.

We assume that the instantaneous value to customer i of service s is a random variable V_{is} , and that the delay cost per unit time is a random variable δ_{is} , i.e., at different times the same customer can have different values and delay costs for the same job. Let G_{is} denote the distribution of (V_{is}, δ_{is}) , with density function $g_{is}(V_{is}, \delta_{is})$ and compact support.

The expected throughput time of program p and priority k is $\tau(p, k, w)$, as defined in equation (1). Then, $\delta_{is}\tau(p, k, w)$ is the total expected cost of delay, of using program p .

Expected service costs are determined by the expected load imposed by the program p , the priority class k , and the rental prices of the machines. Let $r \equiv \{r_{mk}(\bullet), m \in M, k \in K\}$, where $r_{mk}(q_m)$ is the priority price paid at machine m in priority class k for processing q_m units of work. Then, the expected cost of program p and priority class k at machine m is $\sum_{p_m \in P_m(p)} [r_{mk}(q_m(p_m))]$. Thus, the total expected costs of product j using scheme $p \in P_i(s)$ and priority k are

$$c_{iskp}(\delta_{is}, r, w) = \delta_{is} \tau(p, k, w) + \sum_m \sum_{p_m \in P_m(p)} [r_{mk}(q_m(p_m))]. \quad (2)$$

The task of finding a minimum-cost scheme could be subsumed by the client software at users' machines.⁶

⁶ This might appear to be an infinite dimensional problem, however, for practical purposes, the smart agents will have knowledge of a small subset $p_{is} \subset P_i(s)$, so that a cost minimization program is finite and manageable.

Clearly, a user will prefer a program and priority class that minimizes total expected costs. Accordingly define

$$\begin{aligned} c_{is}^*(\delta_{is}, r, w) &\equiv \min \{c_{iskp}(\delta_{is}, r, w) \mid k \in K, p \in P_i(s)\}. \\ \text{and } v_{is}(V_{is}, \delta_{is}, r, w) &\equiv V_{is} - c_{is}^*(\delta_{is}, r, w). \end{aligned} \quad (3)$$

By standard arguments, $c_{is}^*(\cdot)$ is a continuous function.⁷

We can conceptualize the flow of user service requests as consisting of two components. The first component is an exogenous arrival rate of problems that arise and could be potentially addressed by some Internet service. This component is independent of service costs. The second component is the cost-sensitive decision to actually request an Internet service for each problem. Let λ_{is} denote the exogenous arrival rate of service requests from user $i \in I$ for service $s \in S$, such that the total exogenous arrival rate to the system is $\lambda = \sum_i \sum_s \lambda_{is}$. Further, let π_{iskp} be the probability that customer i 's request s is submitted to the network in priority class $k \in K$ using program $p \in P_i(s)$.

Given instantaneous value V_{is} and delay cost δ_{is} , the expected net value passed on to user i , for service s using program p in priority class k is

$$u_{is}(\pi_{iskp}; V_{is}, \delta_{is}, r, w) = (V_{is} - c_{iskp}(\delta_{is}, r, w)) \pi_{iskp} \quad (4)$$

We assume that users make the decision to submit a job to the network based on current expected waiting times w and the expected costs and are not concerned with how processing their service request might affect waiting times w and rental prices r_{mk} . The rationale for this assumption is: first, that the instantaneous service request will in fact not affect the waiting times for their own request, but only for those requests that follow

⁷ There will be a unique cost-minimizing scheme and priority class for G_{is} - almost every δ_{is} .

shortly after; second, that the Internet serves a large set of users who can be assumed to produce demands independently of each other, and a decision by a single user will not significantly affect waiting times and rental prices.

Accordingly, each user i will choose π_{iskp} to maximize $u_{is}(\pi_{iskp}; V_{is}, \delta_{is}, r, w)$, taking (r, w) as fixed; $\pi_{iskp}(V_{is}, \delta_{is} | r, w)$ is characterized by⁸

$$\pi_{iskp}(V_{is}, \delta_{is} | r, w) = \begin{cases} 0 & \text{if } V_{is} - c_{iskp}(\delta_{is}, r, w) < v_{is}(V_{is}, \delta_{is}, r, w) \text{ or } v_{is}(V_{is}, \delta_{is}, r, w) < 0 \\ [0, 1] & \text{if } V_{is} - c_{iskp}(\delta_{is}, r, w) = v_{is}(V_{is}, \delta_{is}, r, w) = 0 \\ 1 & \text{if } V_{is} - c_{iskp}(\delta_{is}, r, w) = v_{is}(V_{is}, \delta_{is}, r, w) > 0 \end{cases} \quad (5)$$

We let $\pi_{is}(V_{is}, \delta_{is} | r, w) \equiv \{\pi_{iskp}(V_{is}, \delta_{is} | r, w), k \in K, p \in P_i(s)\}$ denote the vector of user i 's optimal probabilistic choice functions for service s .

Let x_{iskp} denote the average flow rate of service s for user i in priority class k using program p , which is actually submitted to the network. where

$$x_{iskp} \equiv \lambda_{is} \iint \pi_{iskp}(V_{is}, \delta_{is} | r, w) g_{is}(V_{is}, \delta_{is}) dV_{is} d\delta_{is}. \quad (6)$$

Further, let $X(r, w) = \{x_{iskp}(r, w), i \in I, s \in S, k \in K, p \in P_i(s)\}$ denote the entire distribution of actual flows. Note that $X(r, w)$ is a complete description of demands and flow through the network, because program $p \in P_i(s)$ uniquely determines the routing and processing needs through the network for customer i 's service request s .

4. Optimal Resource Allocation and Stochastic Equilibrium.

⁸ $\pi_{iskp}(\bullet | r, w)$ will be uniquely determined for G_{is} - almost every (V_{is}, δ_{is}) .

The most natural definition of an equilibrium would be that average demand $z_m = \sum_i \sum_s \sum_k \sum_p x_{iskp} Q_m(p)$ at each m equals the "supply" v_m . However, with a Poisson arrival process, equality of the arrival rate and the service rate would yield infinite expected queue waiting times. But if queue waiting times are infinite, then there is no value to computation services, so demand would sink to zero, an inconsistency. Clearly, we want to have consistency between expected waiting times, demands, and actual waiting times.

Given our framework, the waiting times at a machine m by priority class will depend on the distribution of job arrival rates by job size and priority at that machine. Accordingly, we define $\zeta_m(p, Q) \equiv \{ p_m \in P_m(p) \mid q_m(p_m) = Q \}$ and $\mu_m(p, Q) = \# \zeta_m(p, Q)$, the set and the number of subprograms of program p which put a load of Q on machine m , respectively. Then, the job arrival rate of size Q in priority k at machine m is $\psi_{mkQ} = \sum_i \sum_s \sum_p x_{iskp} \mu_m(p, Q)$. Define $\Psi_m(X) \equiv \{ \psi_{mkQ}(X), k \in K, Q \in \mathbb{IN} \}$. We assume that the expected waiting time at m given priority class k is a function of the distribution $\Psi_m(X)$ and the capacity v_m .⁹ Thus, the waiting times can be defined as

$$w_{mk} = \Omega_k(\Psi_m(X); v_m), \quad (7)$$

where $\Omega_k(\cdot; v_m)$ is continuously differentiable, strictly increasing and convex as long as $\sum_k z_{mk} < v_m$, and $\Omega_k(0; v_m) = 0$. Further, $\Omega_k(\Psi_m(X); v_m) \rightarrow \infty$ as $\sum_k z_{mk} \rightarrow v_m$. We further assume that $\partial \Omega_j / \partial \psi_{mkq} \geq \partial \Omega_j / \partial \psi_{mk'q}$ for all $k < k'$; in other words, the incremental waiting time imposed on priority j jobs is greatest for new arrivals of the highest priority jobs. $\Omega_k(\Psi_m(X); v_m)$ gives the physical tradeoff between waiting time and throughput for priority class k , as illustrated in Figure 1, where w_{mk} is plotted in the downward direction. Users prefer points to the northeast of this convex boundary; i.e. they prefer more throughput and less waiting time.

⁹ This assumption seems to provide very good approximation of average waiting time in our simulation studies. It is a fairly general representation and does not restrict one to use a specific queueing approximation; in fact, most queueing approximations can be used under this assumption.

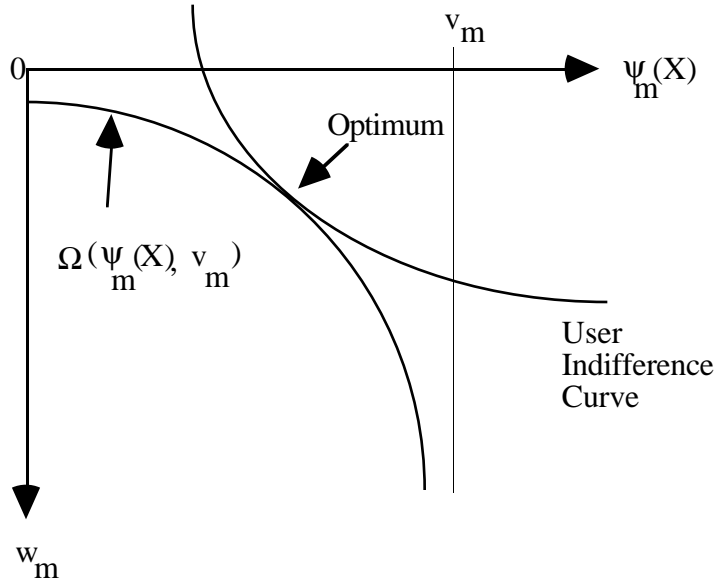


Figure 1 - Tradeoff Between Waiting Time and Throughput

To derive the optimal trade-off we need to define a system-wide welfare function.

It is natural to take the sum of non-pecuniary user benefits:

$$W(\pi, w) \equiv \iint \sum_i \sum_s \sum_k \sum_p \lambda_{is} [V_{is} - \delta_{is} \tau(p, k, w)] \pi_{iskp} g_{is}(V_{is}, \delta_{is}) dV_{is} d\delta_{is} \quad (8)$$

We assume here that there are no costs to operating the Internet and the services, so the only social costs are due to congestion. However, it would be a trivial matter to extend the analysis to include positive operating costs.

We then seek an allocation, $\pi \equiv \{\pi_{iskp}(V_{is}, \delta_{is}, i \in I, s \in S, k \in K, p \in P_i(s))\}$, and waiting times w , that maximizes $W(\pi, w)$ subjected to equation (7). The Kuhn-Tucker conditions of this benefit maximization are [see the Appendix]:

$$V_{is} - \delta_{is}\tau(p,k,w) < \sum_m \sum_h \sum_Q \mu_m(p, Q) [\partial \Omega_h / \partial \psi_{mkQ}] \gamma_{mh} \\ \Rightarrow \pi_{iskp}(V_{is}, \delta_{is}) = 0 ; \quad (9a)$$

$$V_{is} - \delta_{is}\tau(p,k,w) \geq \sum_m \sum_h \sum_Q \mu_m(p, Q) [\partial \Omega_h / \partial \psi_{mkQ}] \gamma_{mh} \\ \Rightarrow \pi_{iskp}(V_{is}, \delta_{is}) = [0, 1] ; \quad (9b)$$

$$\text{and} \quad \gamma_{mh} = \sum_i \sum_s \sum_p \delta_{is} x_{ishpm} \quad (10)$$

where γ_{mh} is the Lagrangian multiplier for equation (7) and $x_{ishpm} = x_{ishp} \mu_m(p)$ if $q_m(p) > 0$; otherwise $x_{ishpm} = 0$.

Equation (10) defines the shadow price of waiting time for priority class k at machine m . Equation (9) requires the marginal net profit for service s at priority class k to be greater than or equal to the total shadow cost of the induced incremental waiting times; if less, then the optimal $\pi_{iskp}(V_{is}, \delta_{is}) = 0$. Let $\gamma \equiv \{\gamma_{mk}, m \in M, k \in K\}$.

In the Appendix, we prove:

Theorem 1. There exists a generically unique (π^*, w^*, γ^*) that maximizes $W(\pi, w)$, subject to equation (7) and satisfying equations (9-10).

We now come to the question of whether we can support this benefit-maximizing allocation with a rental-price mechanism. That is, when will vector of probability, π , which optimizes each $u_{is}(\pi_{iskp}; V_{is}, \delta_{is}, r, w)$, also optimize $W(\pi, w)$? Recall from section 3 that, if $\pi_{iskp}(V_{is}, \delta_{is} | r, w) > 0$, then $V_{is} \geq \delta_{is}\tau(p, k, w) + \sum_m \sum_{p_m \in P_m(p)} [r_{mk}(q_m(p_m))]$ where R.H.S. of this inequality is equal to $\delta_{is}\tau(p, k, w) + \sum_m \sum_{p_m \in P_m(p)} r_{mk}(q_m(p_m))$ from equation 9(a). Then, noting that,

$$\Sigma_Q \mu_m(s, Q) [\partial \Omega_l / \partial \psi_{mk} Q] = \Sigma_{P_m \in P_m(p)} [\partial \Omega_l / \partial \psi_{mk} Q | Q=q_m(p_m)]. \quad (11)$$

$\pi_{iskp}(V_{is}, \delta_{is} | r, w)$ will satisfy equations (10) if

$$r_{mk}(Q) = \Sigma_h [\partial \Omega_h / \partial \psi_{mk} Q] \gamma_{mh} \quad (12)$$

In other words, the benefit maximizing rental price at a machine m with priority k must equal the average cost of aggregate delay, weighted by the waiting time/throughput trade-off at m . Given our assumption about $\Omega(\cdot; v_m)$, the rental prices are decreasing in priority k : $r_{mk} > r_{m(k+1)}$. We could think of r_{mK} as the base price, and $(r_{mk} - r_{mK})$ as the premium for higher priority service.

However, equation (12) is not an "explicit" formula for r_{mk} , since r_{mk} enters the right-hand side via x_{ishpm} . In the Appendix, we prove

Theorem 2: There exists an (r^*, w^*) such that (i) $\pi_{iskp}(V_{is}, \delta_{is} | r^*, w^*)$ maximizes $u_{is}(\pi_{iskp}(V_{is}, \delta_{is} | r^*, w^*); r^*, w^*)$ for all $i \in I, s \in S$; (ii) $(\pi(V_{is}, \delta_{is} | r^*, w^*), w^*)$ maximizes $W(\pi, w)$; and (iii) $w_{mk}^* = \Omega_k[\psi_m(X(r^*, w^*); v_m)]$ for all $m \in M$ and $k \in K$.

In other words, given (r^*, w^*) , individual customers choose a probability of $\pi_{iskp}(V_{is}, \delta_{is} | r^*, w^*)$ for their submissions, which results in optimal flow rates $x_{iskp}^*(r^*, w^*)$, which in turn generate expected waiting-times w^* satisfying equation (7). Furthermore, these demands and waiting times maximize the total benefits $W(\pi, w)$.

An alternative interpretation in terms of competitive equilibrium can be made. Let x^* denote the benefit-maximizing flows from Theorem 1. Then, Theorem 2 asserts the existence of rental prices r^* and expected waiting times w^* such that (i) demands $x(r^*, w^*)$ equal optimal flows x^* , and (ii) these demands via the queues, equation (7), generate

expected waiting times w^* . We call this interpretation a "stochastic equilibrium," in that expected waiting times are correct and "excess demand" in terms of flow rates $(x - x^*)$ is zero.

5. Implementation: Price Determination

The solution to the Internet resource allocation problem we propose is to charge welfare-maximizing rental prices for each machine in the Internet. The task of determining the welfare-maximizing prices requires information about (i) the flow rates, $x_{iskp}(r,w)$, (ii) the expected waiting times, w_{mk} , and (iii) the user's cost of delay parameters δ_{is} . The choice of priority classes by users reveals their underlying delay parameters. A sampling and estimation procedure could be developed to extract this information from data on those choices, thus making this approach incentive compatible.

Our recommended approach to finding equilibrium prices is motivated by the classic tatonnement process [Hahn, 1982]. Heuristically, one wants to raise the rental price when the social costs of delays at a queue are excessive, and vice versa. We first describe the method of obtaining the required information for the tatonnement process.

In lieu of estimating (or knowing) the demand functions, we suggest measuring the average flows at each machine queue. Let $\hat{x}_{iskp}(t)$ denote the current time-averaged estimate of x_{iskp} , and let $\hat{w}_{mk}(t)$ denote the current time-averaged estimate of expected waiting times. These estimates can be used to estimate the functions Ω_k and derivatives $[\partial\Omega_j / \partial\psi_{mkq}]$.

It is also useful to define a new quantity that permits an insightful rewriting of equation (12). Let $y_{iskm} = \sum_p x_{iskp} \mu_m(p)$, the flow rate of s "jobs" of priority k from user i at m - counting each time service s accesses m as a "job". Then, equation (12) becomes

$$r_{mk}(q) = \sum_j [\partial\Omega_j / \partial\psi_{mkq}] \sum_i \sum_s \delta_{is} y_{isjm} . \quad (13)$$

From equation (13), we see that it will suffice to estimate the flow rates y_{iskm} for each machine. The measurement of $\hat{w}_{mk}(t)$ and $\hat{y}_{iskm}(t)$ could be decentralized to each machine. Then, let $\hat{r}_{mk}(t)$ denote the current value of the right-hand side of equation (13) using the above estimates. To lessen the chances for instabilities due to over-responsiveness, we suggest a partial adjustment. Let $r_{mk}(t)$ denote the rental prices in "period" t , and let $\alpha \in (0,1)$; then, set

$$r_{mk}(t+1) = \alpha \hat{r}_{mk}(t) + (1-\alpha) r_{mk}(t) \quad (14)$$

It is natural to begin with $r_{mk}(1) = 0$, and let the system run for one "period" (perhaps a minute or an hour). Then, take the measurements $\hat{w}_{mk}(1)$ and $\hat{y}_{iskm}(1)$, and compute $\hat{r}_{mk}(1)$ from equation (13) and $r_{mk}(2)$ from equation (14). The continuation of these steps for every period defines a stochastic dynamic process. Will $r(t)$ tend towards a small neighborhood of r^* sufficiently rapidly and stay there?¹⁰ How do the methods of estimating $\hat{w}_{mk}(t)$ and $\hat{y}_{iskm}(t)$, the adjustment parameter α , and the period length affect dynamic stability?

To ensure stability, the time required to measure accurately the statistics of the stochastic flows and queues should be small relative to the time between rental price adjustments, which in turn should be small relative to changes in the external environment (e.g., exogenous changes in Internet services demanded).

Although stability is more likely to be attainable for stationary environments (users and services), simulation testing can be employed to experiment with gradually changing environments. There will be a trade-off between rapid adjustment to the equilibrium on the one hand, and the stability of the process on the other. In addition, simulations can provide evidence of the potential gains in Internet efficiency from optimal rental prices.

¹⁰A similar process was studied by Sanders (1988a) and shown to be stable.

We have conducted extensive simulation testing in specific environments to explore these stability issues with very encouraging results. For a model of the Internet consisting of 50 processors, 100 distinct services, a distribution of user delay costs and demand elasticities, we have had little difficulty in achieving rapid convergence of rental prices accompanied by significant reductions in expected waiting times and significant increases in net user welfare. We present a description of simulation study and some results in the next section, the complete study will be reported in a future paper.

6. Simulation Model and Results

In this section we first describe the conceptual model of the Internet along with a specific implementation in subsection 6.1. In subsection 6.2, some results from the simulation study are presented. Finally, in subsection 6.3, we deal with the question of providing over capacity to reduce congestion losses.

6.1 The Simulation Model

Figure 2 presents a conceptual model of The Internet. Essentially, we model the Internet infrastructure as a black-box, i.e., we aggregate the total delay at the server such that it appears that delay is only suffered at the server.¹¹ The users are connected to the Internet through some access providers (which we can consider as a service in itself). The access providers and the service providers, e.g., news, movies, video-conferencing, databases, etc., are "directly" connected to the Internet through a data-pipeline of a certain capacity. In this model, the capacity of the data-pipeline is essentially the bottleneck for the service providers.¹² The network service providers are able to monitor the loads at different servers, and they impose prices according to the load imposed by the servers on

¹¹This assumption is used to simplify the model, however, the results presented here can easily be extended to the cases where the delay is suffered at multiple nodes as shown in Gupta, Stahl, and Whinston (1994, 1995).

¹²From the users' perspective, in reality, the bottleneck is either the server's pipeline or the slowest data communication link in their path to the server.

the backbone due to the congestion at their gateways. Since these prices are not estimated at the equilibrium conditions, they are approximate at any given time.¹³

Figure 3 provides a flow diagram of the simulation model. The arrival rates to the system are price/cost sensitive; to explore the effect of demand scaling (from exogenous growth trends), we vary a parameter X_0 . We can also interpret X_0 as the arrival rate to the system that would occur if there were free access and zero expected waiting times (i.e., the hypothetical uncongested arrival rate or the demand for network services). Note that realized arrivals into the system, being price and delay-sensitive, are always less than X_0 .

Upon the arrival of a potential service request the type of service required is identified; a service is characterized by the amount of computational cycles required at a server. Then, the current estimates of prices and predicted waiting times are obtained for all the servers offering the particular service (and updated every T units of time). We generate user values and delay costs from normal distributions for which the mean delay costs are set to be less than 1% of the mean job value. The user then evaluates the total expected cost of this service in terms of her delay cost and the service cost against her value of the service. If the total cost of the service is higher than her value for the service the user quits the system; otherwise, she submits the request for obtaining the service.¹⁴

¹³Very little is known about the performance and/or design of feedback mechanisms to control the processes in real-time. To our knowledge, our effort is the first systematic, theoretically supported, approach to such real-time problem solving. We have gained valuable insights and have developed several mechanisms for consistent predictions in real-time. The details of these approaches are beyond the scope of this paper and will be presented elsewhere.

¹⁴Realistically, this work would be done by a smart agent or client process executing on the users machine.

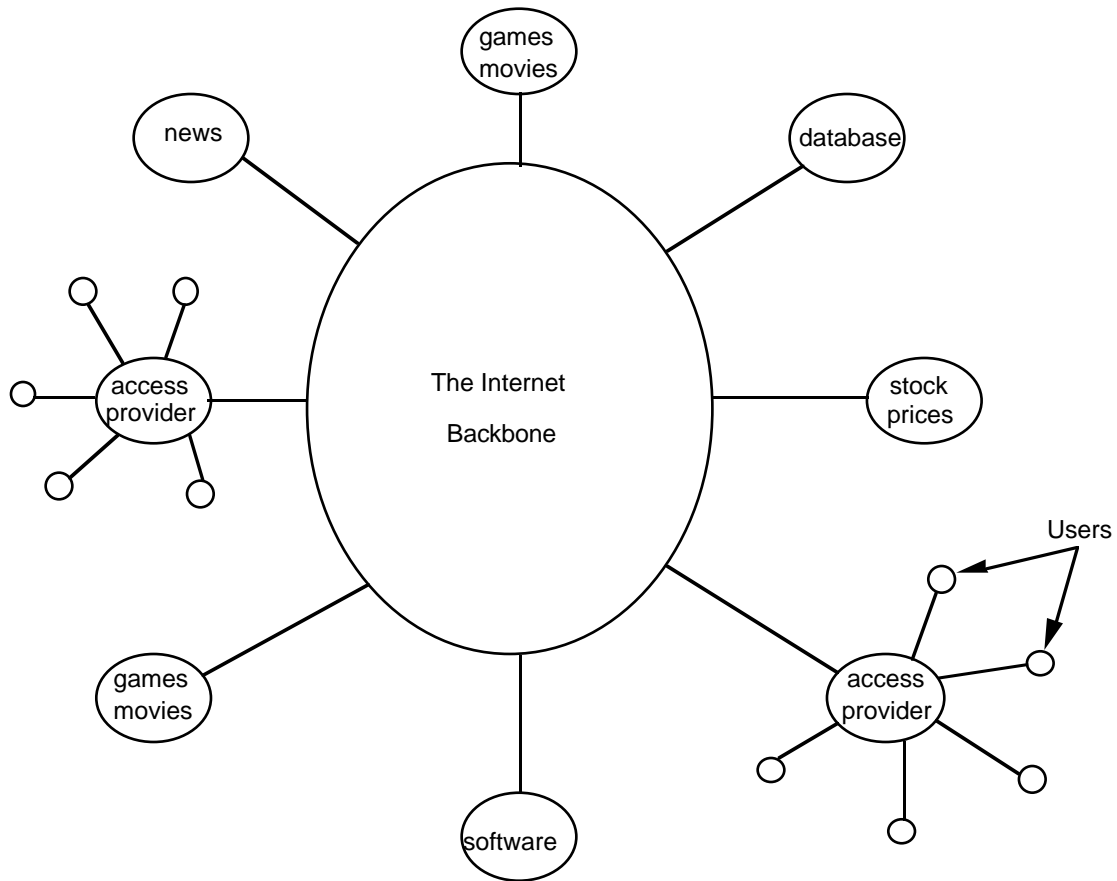


Figure 2 - A Conceptual Model of The Internet

A user's request is sent to the server which was chosen as the least cost server. If the server queue is empty, the request is immediately processed; however, if some job requests exist in the server queue, then the requests are handled in a FIFO manner.

The results presented here are based on a model which has 50 servers and 100 services. A server can provide several of the 100 services, and a service can be provided on up to 25 servers. A service "directory" was determined randomly and fixed throughout the simulation run. The capacity of the data pipelines at the servers are generated through a random process to be among the following: (i) 128 kbps (kilobits per second), (ii) 256 kbps, (iii) 384 kbps, (iv) 1.544 Mbps (megabits per second), (v) 4.0 Mbps, and (vi) 10.0 Mbps. The first three choices here represent 2, 4, or 6 multiplexed ISDN (Integrated

Services Digital Network) data channels respectively, the fourth choice is the capacity of a T1 line, and fifth and sixth choices are typical of those achieved via Framerelay or SMDS (Switched Multimegabit Data Services) connections. The size of each service is also randomly generated to be in the range of 10 Kb - 15 Mb; the distribution is chosen such that there are higher number of smaller services to simulate a more realistic service request distribution. The mean size of service is 2.4 Mb. The service directory and the network configuration, in terms or service sizes and server capacities, were kept constant for all the results reported here.

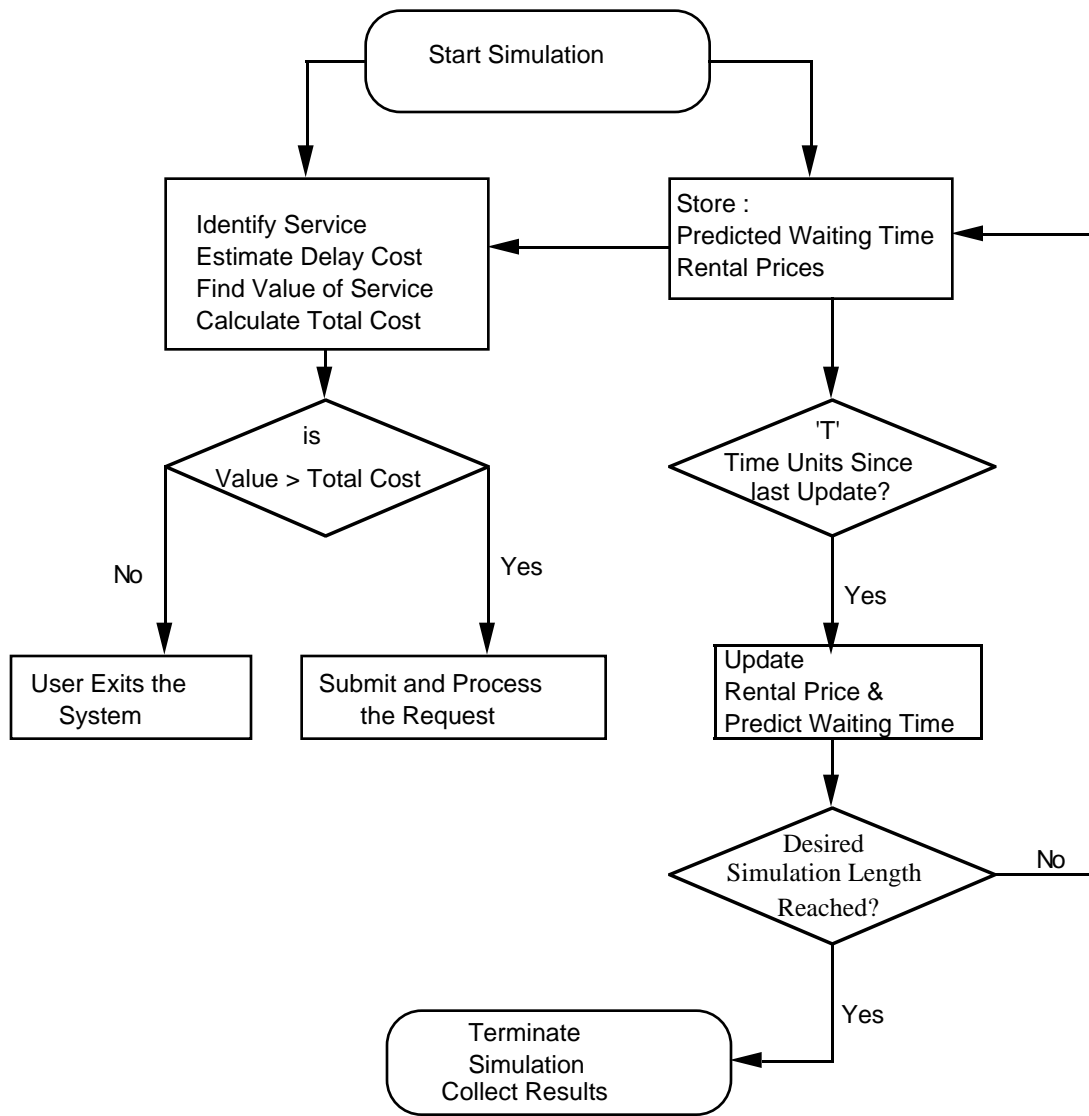


Figure 3 - Flow Chart of the Simulation Model

For the purpose of this simulation we assume that the waiting times, Ω_k , at each server can be approximated by a M/G/1 system, which in our framework could be derived to be

$$\Omega_k(\Psi_m(X); v_m) = \sum_k \sum_q \Psi_{mkq} q^2 / [2 v_m^2 (1 - \sum_{h < k} \rho_{mh}) (1 - \sum_{h \leq k} \rho_{mh})], \quad (15)$$

where $\rho_{mk} = \sum_q \Psi_{mkq} q / v_m$ is the utilization ratio of machine m in priority class k; the rest of the parameters are the same as defined in earlier sections. The partial derivatives of Ω_k are then computed and used in calculating estimates of new prices in the each period, as described in the last section.

We examine this system under a free access policy and optimal priority pricing. We compare these two pricing policies under different sets of load conditions by increasing the scaling parameter X_0 . A higher X_0 induces more load on the system and helps in understanding the behavior of a network with fixed capacity under increasing load.

6.2. Simulation Results

The results presented below involve two different information conditions. First, the results for the free-access policy is based on perfect information regarding the waiting times.¹⁵ However, providing perfect information in a realistic situation is not practical because of excessive cost involved in computing new information for every new request; furthermore, several requests can be submitted virtually at the same time making this waiting time information invalid even if it was financially possible to provide perfect information.

In the more realistic condition of imperfect condition, users would need to generate (or access) estimates of expected waiting times during the current period. Because

¹⁵Note that the perfect waiting time information scenario case is the "best-case" scenario for our implementation of the free access policy because users first check where can they get the fastest service and the information they get is exact.

stochastic queues have the property that the variance is at least as large as the mean, the task of generating accurate estimates from a finite amount of data in real-time is non-trivial. We have experimented with many statistical methods, and have had good results under the optimal pricing policy using an autoregressive process to predict the future waiting times. However, the predictions under free-access at higher loads are sufficiently poor to result in negative net benefits. Therefore, to provide a conservative estimate of the cost of congestion, the reported net benefits from a free-access policy are based on perfect information. The results for the optimal pricing policy are based on predicted waiting times instead of perfect information. In this case both prices and predicted waiting times are updated at the same time.

Figure 4 presents the benefits derived by using a 2-priority system versus a non-priority system using a periodic update scenario. The figure presents net and customer benefits per unit of time. Customer benefits are customer surplus (net benefits - rent). As the figure indicates both the customer benefits and the net benefits are higher with 2-priority system.

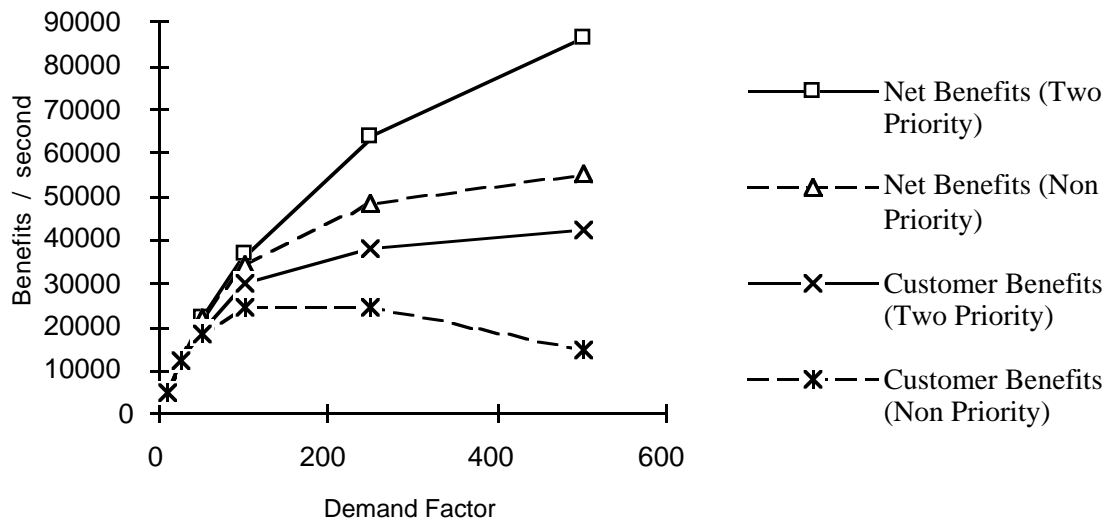


Figure 4 - Net and Customer benefits with Pricing Under Periodic Update

Table 1 displays net benefits and delay costs in dollars per month per server; these computations are based on calibrating the capacity cost according to the average cost of a T1 connection as follows. The current rental cost of a T1 line is about \$1500 per month, which implies that the cost of a 2.45 megabit/second capacity (the average capacity of servers in our simulation) is about \$2000 per month, or \$0.00077 per second. The average job size in our simulation program was 2.4 Mb, so an average server would handle one job/sec. Thus, it is reasonable to assume that the mean value of a job is at least the cost of processing: i.e. \$0.00077. In our simulation, the mean cost of delay was set to 0.008 times the mean value of a job (or only \$0.022 per hour).¹⁶ Table 1 compares the performance of a periodic update pricing case where delay information is not perfect with performance of free access with perfect information. This is an extremely conservative comparison, still pricing does significantly better than free access resulting in substantial gains. If we use the same information assumptions for free-access and optimal pricing, then the estimated Internet benefits of optimal pricing over free-access double.

Arrival Rate (X_0)	Benefits with prices (\$'s)	Benefits free access (\$'s)	Delay Cost with prices (\$'s)	Delay Cost free access (\$'s)	Internet Benefits (\$'s)
50	1749.3276	1338.9921	123.9569	639.0016	103 Million
100	2776.1812	630.6611	266.6029	1732.0273	536 Million
250	3881.9087	515.7304	1629.5195	2041.7187	842 Million
500	4401.1711	483.7473	3261.6245	2145.9051	979 Million

Table 1 - Estimated Dollar Benefits per Month
with Perfect Information for Free access and Periodic Updates for Pricing

¹⁶Note that this figure for delay cost is probably too low by an order of magnitude. However, recalibrating delay cost to 10% of the mean value of a job (or \$0.277 per hour) leaves the net benefits reported in Table 1 essentially unchanged. On the other hand, a uniform rescaling of the value of a job and the delay cost would, of course, simply rescale all the numbers in Table 1 proportionally. Thus, Table 1 is conservative.

To scale these per-server estimates to the U.S. Internet (last column), we multiplied by 250,000, as a conservative estimate of the number of servers on the Internet having an average capacity of an average server in our simulation.¹⁷ Thus, we arrive at a conservative estimate of the potential efficiency loss in the order of \$10 billion annually. Given the historical growth rate of the Internet (100%), the potential loss will exceed \$100 billion annually by the year 1999 or sooner.

The results presented here are suggestive of the benefits of applying a near-optimal pricing scheme on the Internet. Essentially, without a pricing mechanism users with zero or low delay cost have nothing to discourage them from over utilizing the services; however, with a pricing mechanism they are forced to obtain only the services for which their value is higher than the cost. In addition they choose the appropriate service class dependent upon their service requirements. Service providers, on the other hand have incentives to provide multiple service classes because they can generate higher revenues and provide better service overall.

6.3 Capacity Expansion

An alternative to optimal congestion tolls is to increase the capacity of the Internet so no one experiences congestion. We believe that the arguments in favor of simply over-providing the capacity on the Internet are in error. There are physical and cost limitation of providing capacity, whereas on the application level the desire for additional capacity seems boundless. Furthermore, the future applications on the Internet will have inherently different quality of service requirements. Thus, it is essential that appropriate resource management techniques be developed and tested for a multi-service class network.

To explore the cost of this capacity-expansion approach within our simulation model, we set $X_0 = 250$ and incrementally increased the capacity of every server in

¹⁷Conservatively, as of October 1994, there were 2.5 million servers on the Internet (source: MATRIX News); it is safe to assume that collectively 10% of those servers have a capacity equivalent to our server capacity.

proportion to the aggregate delay costs experienced at that server until the aggregate net benefits rose to the level obtainable with optimal pricing and the original capacity. The required increase in capacity was 4.274 times the original capacity. The current monthly rental cost of this capacity expansion is \$6550 compared with a benefit increase of \$3184 per server. Thus, it would be uneconomical to increase capacity enough to achieve the same net benefits that could be obtained from optimal pricing.

7. Conclusions

The coordination of users of a Internet to achieve efficient utilization of the resource is a complex and important problem. With the expected growth of Internet an inefficient coordination mechanism will cause significant economic losses. Computer scientists have considerable experience in creating coordinating mechanisms that are part of a networked operating system. While these mechanisms do coordinate the different processors, there has been no attempt to develop optimal solutions. The issue we explored is the design of a mechanism that can potentially achieve the highest level of resource allocation. To achieve this goal, we have successfully modeled a Internet with priority queues and general stochastic arrivals as an economic resource allocation problem.

The social welfare maximization problem was specified and solved to define the optimal allocation of resources. We then derived rental prices for each priority class at each machine that would support the optimal allocation as a decentralized stochastic equilibrium. Given optimal expected queue waiting times w^* and optimal rental prices r^* , users independently choose when and how many service requests to submit. The aggregation of these submittals constitutes a stochastic process that generates queue waiting times w^* . Moreover, the average flow rates and queue waiting times maximize social welfare.

We suggested a dynamic tatonnement process for implementing this pricing solution. We implement this dynamic tatonnement process using simulation, the results are

very promising. Further simulation studies will develop endogenous stabilization techniques and improve prediction mechanisms and delay cost estimation.

We underscore the need to develop user-interface software that would be installed in user machines to perform many of the crucial programming and evaluation functions. A user of any Internet must translate his service needs into Internet code, and automated user-friendly methods of accomplishing this task are being developed and improved all the time. We suggest enhancing this function by including the ability to estimate the load requirements. Given Internet provided information about current rental prices and expected waiting times, it is a simple matter to compute the expected cost of alternative programs and priority classes. We note that peak-load pricing of existing mainframe systems induces significant changes in user behavior, and we argue that users will have considerable incentives to use this cost information when designing programs and requesting services, thereby bringing about significant improvements in Internet resource utilization.

Our economic approach has the added benefit of providing a sound basis for evaluating Internet investment alternatives, using a process analogous to free entry and exit in free-enterprise economies. In other words, a by-service of optimal pricing is an economic approach to engineering design. All of these theoretical advantages motivate further study. In addition, the system impact and profitability of a new service installed on a server, such as home shopping, could be investigated via simulation in much the same way as for infrastructure investments.

In future research we will extend the theoretical model to account for organizational constraints such as the need to recover the cost of operating the Internet. In this context, notice that the optimal rental prices could generate significant revenue, perhaps adequate enough to cover operating costs. In any event, introducing additional constraints to the welfare optimization will lead to a different expression for the rental prices and will raise questions about the role of these prices and their determination. We are also beginning to explore through simulation and evolutionary game theory, the strategic interaction of price

setters on the future privatized Internet, and to study the impact of alternative public policies. Ultimately, we are interested in anticipating how the real players might behave, and what public policies should be adopted to protect the common resource aspect of the Internet. Obviously it would be desirable to be able to test alternative regulatory policies before they are implemented. Given the complexity of the Internet environment, simulation is the only practical way to pursue these issues.

APPENDIX

Proof of Theorem 1.

Since the queue waiting time function $\Omega_k(\bullet, v)$ is strictly convex $\forall z < v$ and $\Omega_k(\psi(X); v) \rightarrow \infty$ as $z \rightarrow v$, it follows that for any finite scalar $L > 0$, the set $F(L) \equiv \{(X, w) \mid X \geq 0, X = \lambda p \text{ and } \Omega_k(\psi_m(X); v_m) \leq w_{mk} \leq L \forall m \in M \text{ and } k \in K\}$ is a compact convex set. Since $W(\pi, w)$ is continuous, $W(\pi, w)$ has a maximum in $F(L)$. We claim that there is a L^* such that the global maxima of $W(\pi, w)$ are contained in $F(L^*)$. Suppose, to the contrary, that, as $L \rightarrow \infty$, the constrained maxima have $w_{mk} \rightarrow \infty$ for some (m, k) . Now, since $\pi_{iskp} > 0$ for some p with $q_m(p) > 0$, then $\tau(p, k, w) \rightarrow \infty$, and since V_{is} have compact support, eventually W is declining - a contradiction. Therefore, $W(\pi, w)$ has a global maximizer (π^*, w^*) and it lies in a compact convex set.

To solve the global maximum problem using Lagrangian method, we define the Lagrangian function:

$$L(\pi, w, \gamma) \equiv W(\pi, w) + \sum_m \sum_k \gamma_{mk} (w_{mk} - \Omega_k[\psi_m(X); v_m]) . \quad (A1)$$

Since we have just proven that a solution (π^*, w^*) exists, by the Kuhn-Tucker Theorem there exists a γ^* such that (π^*, w^*, γ^*) satisfy equations (9-10) [e.g., Intriligator, 1982]. Furthermore, these conditions are sufficient, since $L(\bullet, \bullet, \gamma)$ is concave in $(\pi, -w)$ for all γ .

Moreover, since $W(\bullet)$ is linear in (π, w) and Ω_k is strictly convex in $\psi(X)$, it follows that $\pi_{iskp}(V_{is}, \delta_{is})$ is uniquely determined for G_{is} - almost every (V_{is}, δ_{is}) .

Proof of Theorem 2.

Let $\pi_{iskp}^*(V_{is}, \delta_{is})$ denote the benefit maximizing allocation given by Theorem 1, and define $x_{ishpm}^* = \lambda_{is} \iint \pi_{ishp}^*(V_{is}, \delta_{is}) \mu_m(p) g_{is}(V_{is}, \delta_{is}) dV_{is} d\delta_{is}$. Then, using equation (12), let

$$r_{mk}^*(Q) = \sum_h [\partial \Omega_h / \partial \psi_{mk} Q] \sum_i \sum_s \sum_p \delta_{is} x_{ishpm}^* \quad (A2)$$

Now, whenever $\pi_{iskp}^*(V_{is}, \delta_{is}) > 0$, equations (9) imply that $V_{is} \geq \delta_{is} \tau(p, k, w^*) + \sum_m \sum_{p_m \in P_m(p)} r_{mk}(q_m(p_m))$, and whenever $V_{ij} < \delta_{ij} \tau(p, k, w^*) + \sum_m \sum_{p_m \in P_m(p)} r_{mk}(q_m(p_m))$, we have $\pi_{iskp}^*(V_{is}, \delta_{is}) = 0$. Hence, $\pi_{iskp}^*(V_{is}, \delta_{is})$ solves equations (4) for G_{is} - almost every (V_{is}, δ_{is}) .¹⁸

¹⁸While $x_i(r^*, w^*)$ may be a set, we always have $x_i^* \in x_i(r^*, w^*)$, and for generic δ_{is} , $x_i(r^*, w^*)$ is single-valued. With a continuum of agents, multi-valued demands have no consequence for the stochastic equilibrium or the welfare maximum. With a finite number of agents, this "decentralization" problem can be overcome with an ϵ -optimal approach [Stahl, 1986].

REFERENCES

- Chu, W., and M. Lan, "Task Allocation and Precedence Relations for Distributed Real-Time Systems," IEEE Transactions on Computers, 36, 667-679, 1987.
- Debreu, G., The Theory of Value, Yale University Press, 1959.
- Dijkstra, E. "Self-stabilizing Systems in Spite of Distributed Control," Communications of the ACM, 17, 643-644, 1973.
- Gupta, A., D. O. Stahl, and A. B. Whinston, "An Economic Approach to Network Computing with Priority Classes," Journal of Organizational Computing, 1996, forthcoming. **
- Gupta, A., D. O. Stahl, and A. B. Whinston, "Pricing of Services on The Internet," in IMPACT: How IC² Research Affects Public Policy and business Markets, Fred Phillips and W. W. Cooper (eds.), Greenwood Publishing, CT, forthcoming, 1995. **
- Hahn, F., "Stability," in Handbook of Mathematical Economics, II, Ch. 16, Arrow, K. and Intriligator, M. (ed), North-Holland, 1982.
- Intriligator, M., "Mathematical Programming with Applications to Economics," in Handbook of Mathematical Economics, II, Ch. 2, Arrow, K. and Intriligator, M. (eds), North-Holland, 1982.
- Kleinrock, L., Queueing Systems, Volumes 1 and 2, Wiley, 1975 and 1976.
- Lee, L.L., and M. A. Cohen, "Multi-Agent Customer Allocation in a Stochastic Service System," Management Science, 31, 752-763, June 1985.
- MacKie-Mason, J. K., and H. R. Varian, "Pricing the Internet," in Public Access to the Internet, B. Kahin and J. Keller (eds.), Prentice-Hall, NY, 1995.
- Mendelson, H. "Pricing Computer Services: Queuing Effects," Commun. ACM, 28, 312-321, 1985.
- Mendelson, H., and S. Whang, "Optimal Incentive-Compatible Priority Pricing for the M/M/1 Queue," Operations Research, 38, 870-83, 1990.
- Naor, P., "On the Regulation of Queue Size by Levying Tolls," Econometrica, 37, 15-24, 1969.
- Sanders, B., "An Asynchronous, Distributed Flow Control Algorithm for Rate Allocation in Computer Networks," IEEE Transactions on Computers, 37, 779-787, 1988.

These papers are available in HTML format on the World Wide Web server of Center for Information Systems Management, University of Texas at Austin. The URL for linking to this server is "http://cism.bus.utexas.edu." The papers can be accessed by clicking on the "button" marked **Papers at the top of CISM's home page.

- Sanders, B., "An Incentive Compatible Flow Control Algorithm for Rate Allocation in Computer Networks," IEEE Transactions on Computers, 37, 1067-1072, 1988.
- Shenker, S., "Service Models and Pricing Policies for an Integrated Services Internet," in Public Access to the Internet, B. Kahin and J. Keller eds., MIT Press, Cambridge, MA, 1995.
- Stahl, D., "Stochastic Decentralization of Competitive Allocations," Economics Letters, 22 (2), 111-113, 1986.
- Stahl, D., and A. B. Whinston, "A General Equilibrium Model of Distributed Computing," Center for Economic Research Working Paper 91-09, Department of Economics, University of Texas, 1991; also in New Directions in Computational Economics, W. W. Cooper and A. B. Whinston (eds.), Kluwer Academic Publishers, Netherlands, 175-189, 1994.