DEC-TR-510

Congestion Avoidance in Computer Networks with a Connectionless Network Layer Part IV: A Selective Binary Feedback Scheme for General Topologies

K. K. Ramakrishnan, Raj Jain, Dah-Ming Chiu

Digital Equipment Corporation
550 King St. (LKG1-2/A19)
Littleton, MA 01460

Network Address: Jain%Erlang.DEC@DECWRL.DEC.COM

August 1987
This report has been released for external distribution.

Congestion Avoidance in Computer Networks with a Connectionless Network Layer Part IV: A Selective Binary Feedback Scheme for General Topologies

K. K. Ramakrishnan
Dah-Ming Chiu
Raj Jain
Distributed Systems Architecture and Performance
Digital Equipment Corporation
Littleton, Massachusetts

Abstract

With increasingly widespread use of computer networks, and the use of varied technology for the interconnection of computers, congestion is a significant problem. In this report, we refine a scheme for *congestion avoidance* in networks using a connectionless protocol at the network layer.

In the part II of this report series, we proposed a scheme for congestion avoidance using feedback from the network. The users of the network respond to the feedback by reducing the amount of traffic injected into the network. The assumption made in part II was that all the users shared the same set of network resources. As a result, the simplified fairness criterion that we used was that all the users have the same window size.

In this report, we relax the assumption of having the same set of resources shared by the different users of the network. Users may share arbitrary sets of resources of the network. We define a more general fairness goal to achieve in the light of the relaxation of the assumption. We present a solution alternative to achieve this goal, by having the routers in the network selectively feedback the congestion information only to those users that are using more than their fair share of each individual router's resources.

We continue to maintain the need for the scheme to be distributed and adapt to the dynamic state of the network. The scheme converges to the 'optimal' operating point and is simple to implement. We also address the performance of the scheme under transient changes in the network, and for a variety of other network conditions.

This is part IV of a series of reports on our work on congestion avoidance for connectionless networks.

Contents

1 Introduction

Congestion in computer networks is becoming a significant problem with the increasing use of the networks for distributed processing. The technological advances that result in greater link speeds in effect causes more mismatches in the link speeds than we have seen in the past. Congestion control has been the subject of several studies, for example in [Kle78], [Nag84]. In this report series[1] we have proposed a new approach to solve the problem of congestion in computer networks. In Part I of this series of reports,[JR87], we have introduced the concept of 'congestion avoidance' and the various components that are part of a scheme attempting to perform congestion avoidance. In Part II, [RJ87], we described a specific scheme for 'congestion avoidance' in connectionless networks. The theoretical reasoning behind the selection of certain aspects of the 'congestion avoidance' scheme were described in Part III, [CJ87].

In [JR87], the characteristics to be achieved by an optimal congestion avoidance policy were outlined as being:
1) operate the network at the maximum possible effeciency
2) Offer Fair service to the users of the network. The scheme described in [RJ87] acheived these criteria for operating the network at the maximum effeciency and maximum achievable fairness. But, the design was based on the assumption that all the users utilize the same set of resources in the computer network. This simplified the definition of the correct operating point of the routers in the network, to achieve maximum effeciency. The fairness criterion translated quite easily to that of achieving equal window sizes for all the users using this common set of resources in the network. We relax this assumption in this paper. The topologies in the network may be arbitrary, and the users in the network may share different sets of these resources in the network. We generalize the optimality criteria that was dèscribed in [JR87] to allow us to define an optimal (efficient and fair) operating point. This is primarily in the definition of fairness of the service offered to the users of the network.

In the next section we describe the reasons why the original scheme provides sub-optimal performance, when the users share different sets of resources. Subsequently, in Section 3, we define the performance goals, in terms of efficiency and fairness that are of interest. We also constructively define the optimal operating point that satisfy our criterion of maximally fair and efficient allocation. In Section 4, we present alternative solution approaches. We then study one of the solution approaches in detail and present extensive results using the solution of having routers in the network selectively set the 'congestion indication' bit to the different users sharing that router. In Section 5, we present areas that are currently under study and directions for future work. Finally, we

[1]All four parts of this report series are summarized in: Raj Jain, K. K. Ramakrishnan, and D. M. Chiu, "Congestion Avoidance in Computer Networks with Connectionless Network Layer," DEC Technical Report, DEC-TR-506, Digital Equipment Corporation, August 1987, 17pp. Also published in C. Partridge, Ed., *Innovations in Internetworking*, Artech House, 1988.

present the conclusions of this study.

2 Problem Definition

The goals that the congestion avoidance policy attempts to achieve were to provide for maximum efficiency of the use of the resources of the network and to provide for fairness in the services offered to the users of the network. To achieve the first goal of efficiency of the resources, we attempt to provide the best tradeoff between throughput and delay for users of the network. This is attained by operating the router at the 'knee' of the response time curve. We discuss this in detail in [JR87] and [RJ87]. We found that when viewing each router by itself, the best tradeoff between throughput and delay was to minimize the queueing delay at the router and also minimize the idle time of the router. In the simple case, the operating point achieving maximum efficiency translates to maximizing the *power* at the router. We showed that the maximum efficiency was obtained when the average number in the system was maintained at 1. This consideration of maintaining the number in the system at 1 was used to set the 'congestion indication' bit by the router. We also showed that by setting the 'congestion indication' bit using this policy, we were able to operate a network of multiple nodes and multiple users at the knee of the network response time. This policy of setting the bit at the router is independent of the topolgy of the network and the users that are sharing individual router resources.

2.1 Disadvantages of Achieving Identical Window Sizes

The goal we had strived for in the case where the paths that were used by the various users were identical, was to have the same throughput for all users [RJ87]. When all the users share the same path, having the same window size for all the users achieved the goal of providing fair service to the requesting users. Maintaining fairness in the window size, which is the characteristic used by the end-node to control the amount of traffic on the network was sufficient. When the users using the network go through different paths, as is generally the case, then maintaining the same window size for all the users is not a sufficient goal, both from the view of fairness as well as efficient operation of the network. We find that the overall throughput from the network is smaller, since users are indiscriminately limited by a router even if that user does not receive a fair share of that router's resources.

Consider for example two users which use two different paths, but share a common bottleneck, which is congested. Let us assume that the service time at each hop is the same. As a consequence, the optimal aggregate window size would be a function of the number of hops, H [Jai86]. User 1 uses a short path, and therefore it's optimal window size is small, when it is the only user on the path. User 2 uses a longer path, with a optimal window size which is much larger, if it was the only user on that path. Let us now consider the situation when both the users are active, and the only resource that is congested is the bottleneck which is shared by both users. Because of the common shared bottleneck, the signals received by the two users from that bottleneck, when they are both active, would be identical. As a result of the algorithms implemented by the users, the window sizes used by them would be driven in such a way that the are ultimately equal. Since the overall aggregate window size is now limited by the common bottleneck, the window size of both the users would be equal - and would be limited to the smaller window size that user 1 uses.

Equal window sizes would imply unequal throughputs for the users which go through different paths. In this example:
Let W = window size used by each user.
Let Rd_1 and Rd_2 = round trip delay for user 1 and 2 respectively.
Let Th_1 *and* Th_2 = throughput of user 1 and 2 respectively.
A user that uses a much longer path would encounter a much larger round trip delay, and therefore lower throughput. In Rd_1 time units, user 1 is allowed to transmit W packets and in Rd_2 time units, user 2 is allowed to transmit W packets. The throughputs of user 1 and 2 are therefore:
$Th_1 = W/Rd_1$ and $Th_2 = W/Rd_2$ and thus $Th_1 \neq Th_2$.
As a consequence of using a smaller window size the path that is utilized by user 1 is not utilized optimally. The resources used by user 1 experience considerable idle time. Figure 1 demonstrates the effect on the throughput of the two users when the bottleneck generates the same signal and the users use the same increase and decrease algorithms (as described in [RJ87]), with non-identical paths.

In the next section, we define the maximally fair and efficient operating point in the general case of multiple users sharing different sets of network resources.

3 Performance Metrics

A congestion avoidance scheme may be viewed as a resource allocation mechanism in which the subnet (set of intermediate nodes, or routers) is a set of m resources which has to be allocated to n users (source-destination pairs). There are two parties involved in any resource allocation mechanism: the resource manager and the user. The resource manager's goal is to use the resource as efficiently as possible. Users, on the other hand, are more interested in getting a fair share of the resource. We therefore need to define efficiency and fairness. We discussed the case of a single resource in [JR87] and [RJ87]. In this paper, we are more interested in the issue of efficiency and fairness with multiple users accessing , in general, one or more resources. The concepts introduced here, therefore, are general and apply to other distributed resource allocation problems as well.

For our current problem of congestion avoidance, the routers are our resources and therefore we use the terms *routers* and *resources* interchangeably. The demands and allocations are measured by packets/second (throughput) but the concepts apply to other ways of quantifying demands and allocations.

The fairness goal that we desire to maintain with the congestion avoidance policy is to provide an equal share of a resource's capacity to each of the users placing a demand on it. When the demand of all the users utilizing a resource is less than the resource's capacity, we do not need to limit any of the users, even though the individual utilizations are not equal. When the demand of all the users is greater than the resource's capacity, then we must limit the users such that the capacity is fairly allocated over the users. Informally, any user that is placing a smaller demand than its fair share of the resource's capacity would not be limited by this resource. The residual capacity of the resource may then be allocated to the other users that are placing more demand on the resource.

The efficiency goal is achieved by maximizing *power* [Kle79], which is the ratio of throughput to delay. We must emphasize here that we are concerned with resource power and not with user power. The two are different. Resource power is the throughput to delay ratio at the resource, whereas the user'power is the throughput (same as the throughput at the bottleneck) to the roundtrip delay (sum of the average delay at each resource on its path) experienced by the user.

To develop the concept of fairness with multiple resources and multiple users, we start by considering the case of a single resource, (which may be an individual router), and a set of users with unequal demands. The primary goal we redefine here is the definition of a maximally fair allocation of the resource to the users making a demand.

Let us start with some definitions.
Let C_j^{knee} = Knee capacity of a resource j.
Let A_{fair} = Fair allocation of a resource to each user.

3.1 One Resource, Multiple Users with Unequal Demands

Given a resource with knee-capacity of C^{knee}, each of the n users deserves a fair share of C^{knee}/n. However, part of the resource would be wasted by allocating C^{knee}/n to a user who is demanding less than C^{knee}/n. It would be better to give the excess to another user who needs more of the resource. This argument leads us to the concept of *maximally fair allocation* which requires dividing the users into two groups: those whose demands are less than the fair share, and those whose demands are more than the fair share. Let their numbers be $n - h$ and h, respectively, and the sum of their demands be d_{low} and d_{high}, respectively. Then,

The Fair Share $\quad A_{fair} = (C^{knee} - d_{low})/h$

and the maximally fair allocation consists of

$$A_i = min\{d_i, A_{fair}\},$$

where $A_i \quad = \quad$ allocation to the i^{th} user.

Notice that finding the fair share and the maximally fair allocation is an iterative procedure starting with $h = n$. We illustrate the procedure with an example.

Example : For a resource with knee-capacity of 100 packets/second being shared by 5 users demanding 10, 30, 40, 20, and 60 packets/second.

Here, $n = 5$ and $C^{knee} = 100$.

Iteration 1: $h = n$, Fair share $A_{fair} = C^{knee}/h = 100/5 = 20$, $A_i = 20$, $i = 1, \ldots, 5$.

Iteration 2: h= Number of users demanding more than the fair share 20=3

$d_{low} = 10 + 20 = 30$, Fair share $A_{fair} = (100 - 30)/3 = 23.33$

Iteration 3: h=Number of users demanding more than the fair share 23.33=3.

The new value of h is same as that in the previous iteration. Therefore, we stop.

The maximally fair allocation is {10, 23.33, 23.33, 20, 23.33}.

[]

Given the knee capacity of a resource and individual user demands, the above procedure allows us to determine the maximally fair allocation $\{A_1^*, A_2^*, \ldots, A_n^*\}$. If the actual allocation $\{A_1, \ldots, A_n\}$ is different from this, we need a distance function to quantify the fairness. We do this by using the *fairness function* [JCH84]:

$$Fairness = \frac{(\sum_{i=1}^{n} x_i)^2}{n \sum_{i=1}^{n} x_i^2} \tag{1}$$

where $x_i = A_i / A_i^*$.

This function has the property that its value always lies between 0 and 1 and that 1 (or 100%) represents a maximally fair allocation.

The efficiency of the resource usage can be computed as before [JR87], by computing the resource throughput, which is given as the sum of user throughputs in this case.

$$\text{Resource Throughput} = \sum_{i=1}^{n} A_i$$

$$\text{Resource Power} = \frac{\text{Resource Throughput}}{\text{Resource Response Time}}$$

$$\text{Efficiency} = \frac{\text{Resource Power}}{\text{Resource Power at Knee}}$$

The allocation which is 100% efficient and 100% fair is the **optimal allocation**. The above discussion applies to the case of an individual router accessed by multiple sources as well as the situation when there are multiple (m) routers but all routers are shared by all n users. In the latter case, the set of m routers can be combined and considered as one resource. A user using many resources will have an average response time which is the sum of the response times at these resources. Although the concept of power is well known (see, for example, [Kle79]), its application to resources rather than users is our unique contribution. This discovery helped us avoid many of the problems encountered earlier in using power [Jaf81].

3.2 Multiple Resources, One User

We now extend the above concepts to a distributed system with multiple resources. Let us first consider a case of a single user so that fairness is not an issue. The user is characterized by the set of resources it uses, which we call its **path** P. If

$$R = \{1, 2, \ldots, M\}$$

is the set of all resources (routers and links etc) in the network, then P is a subset of R. The resource in P with the lowest service rate determines the user's throughput and is called the **bottleneck** resource. The bottleneck resource has the highest utilization (ratio of throughput to service rate) and contributes the most to user's response time. The maximally efficient operating point for the system is defined as the same as that for the bottleneck router. Thus, given a path of m resources, we determine the bottleneck, and define its efficiency as the **global efficiency** and its knee as the maximally efficient operating point for the path.

Global Efficiency = Efficiency of the Bottleneck Resource

Note that the global efficiency as defined here depends upon the response time at the bottleneck resource and not on the user response time (which is a sum of response time at m resources). Also, we must point out that if a user visits a resource more than once, the service time to be used to determine the bottleneck is the total time used in all visits. Thus, for example, if every packet is followed by a returning acknowledgment passing through the same router, the service time is the sum of the times required to forward the packet and the acknowledgment.

3.3 Multiple Resources, Multiple Users

Now let us examine the case for multiple users. The global efficiency is still defined by the bottleneck resource which is identified by the resource with the highest utilization. The problem of finding the maximally efficient and maximally fair allocation is now a constrained optimization problem as it has to take differing user paths into account. The definition of *optimal* criteria for the operating point of such a network has attracted a lot of attention, as seen in [Kle79], [Jaf81], [GB82]. We have developed an algorithm which gives the globally optimal (fair and efficient) allocation for any given set of resources and users (paths). This algorithm is described next.

Intuitively our fairness goal is to divide each resource equally among all users using it. Since users demand different sets of resources, this equality allocation only applies to users sharing the same resource as their common bottleneck. For instance consider two users A and B sharing some resource K. Let A's bottleneck be K whereas B is limited by some other bottleneck resource not shared by A (the determination of the bottleneck resource for each user is part of the algorithm to be detailed below). In this case, making A and B share the same amount of K could underutilize K thus lead to unnecessary *inefficiency*.

This fairness and efficiency criterion for the multiple resource, multiple user case can be defined by a constructive algorithm as below.

The set of resources is denoted

$$R = \{1, 2, \ldots, m\}$$

as earlier where resource j has a knee capacity C_j^{knee}. The set of users is

$$U = \{1, 2, \ldots, n\}$$

where user i has a path P_i, which is a subset of R. We will be using the terms path and user interchangeably.

The problem is to allocate the *flow* (or *demand*) for each user according to a vector A^* where

$$A^* = \{A_1^*, A_2^*, \ldots, A_n^*\}$$

so that the total allocation at each resource does not exceed the capacity of that resource, and this assignment represents the "optimal" tradeoff between efficiency and fairness.

Let S denote the set of users whose flows have not been determined yet. Initially $S = U$. Let M denote the set of resources whose utilization level has not been finalized yet. Initially $M = R$. Also we use C_j to denote *remaining* capacity of resource j to be allocated.

1. Initialize $S = U, M = R$, and $C_j = C_j^{knee}$ for each j.
2. For all resources in M, calculate N_j as the number of users in S contending for resource j;
3. For each resource j in M, calculate

$$B_j = \frac{C_j}{N_j}$$

4. Compute

$$k = \arg \min_{\forall j \in M} B_j$$

$$B_k = \min_{\forall j \in M} B_j$$

Remove resource k from M.

5. For each user i in U, if k is in P_i, then assign

$$A_i^* = B_k$$

and for each resource j in P_i and still in M, assign

$$C_j = C_j - B_k$$

; and remove i from S.

6. if S is empty then stop; else repeat steps 2 to 5.

The efficiency-fairness tradeoff criterion above is widely referred to as the *max-min fairness* criterion. The most important property is that all paths sharing and limited by the same resource will be assigned the same flow, and that no resource can have more utilization without causing some other resource to exceed its capacity. It should be clear that the above centralized algorithm terminates since in each iteration at least one resource is removed. Further, since all paths are constrained by some resource they must all be removed from S before we finish the loop when M is empty.

Example: Consider the example of 3 users sharing a network of 5 resources (routers or links) of different speeds. This configuration is shown in Figure 3. Resource 1 and 2 have a knee capacity $C^{knee} = 50$ and resources 3, 4 and 5 have a knee capacity of $C^{knee} = 100$. User 1 shares all the resources, user 2 shares resources 1 and 2 and user 3 shares resources 4 and 5. Thus, $m = 5$ and $n = 3$. Describing the steps followed by the algorithm, we have:

1. Initialize : $C_1^{knee} = C_2^{knee} = 50; C_i^{knee} = 100, \; i = 3,4,5;$

2. Iteration 1:
   Step 2: $N_1 = N_2 = 2; \; N_3 = 1; \; N_4 = N_5 = 2$
   Step 3: $B_1 = 25 = B_2; \; B_3 = 100; \; B_4 = B_5 = 50$
   Step 4: The first bottleneck $k = 1$ (or 2 - doesn't matter).
   Step 5: $A_1^* = A_2^* = 25;$ Assign $C_3 = 100 - 25 = 75; \; C_4 = C_5 = 75.$

3. Iteration 2:
   Step 2: The number of users whose flows have not been determined yet is 1.
   Step 3: $B_3 = B_4 = B_5 = 75$.
   Step 4: The next bottleneck is $k = 3$.
   Step 5: $A_3^* = 75$.
   Step 6: Since all users have their allocations determined, set S is empty. Stop.

As a result of the execution of algorithm, we have determined the globally optimal allocations of each of the resources to the 3 users to achieve a maximally fair allocation.

Once the globally optimal allocation $\{A_1^*, A_2^*, \ldots, A_n^*\}$ has been determined, it is easy to quantify fairness of any other allocation $\{A_1, A_2, \ldots, A_n\}$ by using the same fairness function as in the single resource case (equation 1) with $x_i = A_i/A_i^*$. This fairness is called **global fairness** and the efficiency of the bottleneck resources is called the global efficiency. An allocation which is 100% globally efficient and 100% globally fair is said to be **globally optimal**.

Notice that we have a multi-criteria optimization problem since we are trying to maximize efficiency as well as fairness. One way to solve such problems is to combine the multiple criteria into one, for instance by taking a weighted sum or by taking a product. We chose instead to put a strict priority on the two criteria. Efficiency has a higher priority than fairness. Given two alternatives, we prefer the more efficient alternative. Given two alternatives with equal efficiency, we choose the fairer alternative. This is because networks tend to be used in bursts. Most routers are generally idle. Users come on, use the network for a very short interval, and go off for a long time. The overlap between users is small and the single user case is predominant. Therefore, the need for efficiency is predominant. The fairness is required only when there are multiple users. It is therefore considered to be secondary to efficiency.

4 Solution

In the previous section we described the metrics to determine the optimality (in fairness and efficiency) of an allocation of the resources of the network, and also outlined an algorithm for computing the allocations to users in the case when there are multiple resources that these users are sharing. While we have outlined a centralized algorithm for calculating the *maxmin* flows, it is not a reasonable way to solve the problem because it requires too much information at a single resource manager. Distributed algorithms to solve this problem have been studied in [Jaf81], [GB82], [Mos84] and recently in [Hah85]. What we propose in the following differ from the previous studies in that we strive to make the resource managers as simple as possible, and we constrain the feedback to be of the binary form.

In a connectionless network, the mechanism we propose using to communicate the information from the router to the sources is a single 'congestion indication' bit that exists in the network layer header. The setting of the congestion indication bit by the routers when their average queue size is greater than a threshold value (which is a value of 1) ensures that we achieve our criterion of maximum efficiency. To achieve our goal of maximally fair allocation we considered two alternatives. The first scheme attempts to keep the end system policies the same as suggested in [RJ87] and requires the active participation of the routers to achieve fairness. We call this the *router based* approach. The second alternative keeps the policies within the subnetwork the same, with the routers setting the congestion indication bit for all users, but the end systems use different policies to achieve fairness. We call this the *transport based* approach. We describe the router based approach in detail here, as the scheme we are recommending. We also discuss the transport based alternative in Section 6.2.

4.1 Router Based Alternative

In this approach, routers participate in achieving the goal of fair allocation of the resources across the users. Each source-destination pair is considered to be a user of the router and hence of the network. This alternative requires routers to identify the users sending packets through the router. The router also performs monitoring to determine the share of the router's resource being used by each of the users. The router computes, over a period of time, the number of packets that each user (source-destination pair) has sent in unit time. The number of packets sent may then be used to determine those users that are using more than their fair share. The 'congestion detection' mechanism, which determines that the average queue size at the router is greater than a threshold, would indicate if the overall demand on the router is greater than its capacity. The router algorithm selectively determines those users that are using more than their fair share only if we determine the router is congested and that the 'congestion indication' bit has to be set on packets flowing through the router.

Ideally, the router algorithm should provide to each user the exact capacity that is allocated by the routers. This may be, for example, in terms of a rate of number of packets second that the user may send. If each router were to independently transmit the rate to the users, then the user may pick the smallest of the rates to determine the rate at which it may send packets to that path. But, the transport protocol uses a window, rather than a rate to transmit packets into the network. Further, a criterion followed in the design is to not inject additional traffic into the network when congested. With the restriction that we have only one bit of information to send to the user, the mechanism that we adopt is to ***selectively*** set the bit on packets from users that use more than their fair share of a ***congested router's*** resources. The users that have their bits set slow down, and those that do not, speed up (i.e., decrease or increase their window size). Consequently, the window sizes of the users are adjusted to the point that the throughputs of the various users are equal.

The next question that arises is to determine the particular users that must have the congestion bit set. By allocating an average of the capacity to each of the users of the router, we do not achieve the desired maximally fair allocation. The example in Section 3.1 showed this when a resource has multiple users with unequal demands. We use an iterative algorithm (outlined briefly in Section 3.1) to achieve the maximally fair allocation in detail.

The approach for determining the users for which the bit must be selectively set is based on the principle that any left over capacity should be used by the other users that may be able to utlilize it. The router maintains a count of the number of packets processed by it from each user. Let T = the time interval over which the counts are maintained. From these counts, we may derive the demand that is placed on the router by each user (in packets/sec.). The demand at the router for any user is based on the allocation at the previous router. The selective feedback algorithm is invoked only when the congestion detection mechanism indicates that the router is congested. This algorithm is similar to that descibed in Section 3.3, except that it is now a distributed algorithm instead of being performed by an omniscient observer. Each resource that is managing its usage executes the algorithm independently. By using the basic congestion avoidance algorithms followed by the transport entities the routers achieve the correct level of traffic flowing through them to enable operation at the maximally efficient operating point. Given a different fairness definition (to only prevent starvation, for example), we can translate it to an equivalent selective algorithm executed by the routers.

Selective Feedback Algorithm

Let C^{knee} = Knee capacity of this router (in packets/sec.).
Let D_i = demand of user i on router (packets/sec.), $i = 1, \ldots, n$.
Let n = number of users placing a demand on the router

The problem is to allocate a share of the capacity C to each of the n users.
Let A_i = allocation to user i by the router
Let C= Remaining capacity of router that may be allocated.
Let S = set of remaining users to have their allocation determined.
Let A_{fair} = Fair share of the resource's knee capacity.

1. Compute the allocations A_i only if $\sum_{i=1}^{n} D_i > C^{knee}$.
   Otherwise, the allocation to all users equal their demands. i.e., $A_i = D_i$.
2. Initially, $card(S) = n$. ($card(S)$ is the cardinality of the set S)
   Initially, $C = C^{knee}$
3. $C = C - \sum_{i \in S} A_i$
4. Compute $A_{fair} = C \, / \, card(S)$
5. **For each user $j \in S$, whose $D_j < A_{fair}$, assign $A_j = D_j$**
   Subtract the allocation A_j from C - i.e., $C = C - A_j$
   Remove each of these users j from the set S.
5. If no new allocations were made in step 5, stop.
   Otherwise, repeat steps 3 through 5 until no new allocations are made in step 5.
   For each user j with $D_j \geq A_{fair}$, set bit on packets of user j.

This algorithm allocates the resources of each individual router to the users (source-destination pairs) that place a demand on it. All those users whose $D_i \geq A_{fair}$ will be candidates for having their 'congestion indication' bit selectively set by this router. The other users do not have their congestion indication bit set even though the router is congested. This policy is implemented at every resource in the network to determine who are the users that utilize a greater share than is allowed by the fairness criterion. The 'congestion indication' bit is actually set only when the router is congested, as determined by the 'congestion detection' mechanism, thus achieving maximum efficiency as well.

We have assumed that the user demands and the knee capacity of the router are known. We discuss these in the next subsections.

4.2 Interval over which Demands are Estimated

In the algorithm described above, we had assumed a time interval T, for determining the demand D_i from each of the users, in terms of the number of packets that each individual router processes. The 'congestion detection' mechanism, to detect that a router is congested uses an adaptive averaging scheme at each individual router [RJ87]. The averaging is also needed to generate a consistent signal to all the users placing a demand on the router. The interval over which the averaging takes place at the router is based on the time between regeneration points at the router. The regeneration point we have chosen is the instant at which the router becomes busy after an idle time. Thus, the averaging interval is the length of a (busy+idle) period at the router. The router may go through busy and idle cycles at an arbitrary rate, as a function of the number of users placing a demand on it and their workload characteristics. We use this same interval as the time interval T to determine the demand by users on the router.

However, a router may spend a considerable amount of time being busy before becoming idle. In that period, it may go from an 'uncongested' state to being 'congested'. If the averaging were performed strictly over (busy+idle) cycles alone, we observed the likelihood of the situation that a busy period persists, and the correction of the window sizes of the users takes place too late [RJ87]. This same phenomenon is also observed with the determination of the demands on the router. When the allocation is determined only at the end of a (busy+idle) cycle interval, those 'candidate' users that are selectively fedback to slow down may receive unfair treatment at the routers. Consider the case when there are two users A and B placing a demand on router R. At time T, user A may be placing a greater demand than it's fair share allows. As a result of 'selective feedback', user A would get it's 'congestion indication' bit set, while user B will not have them set. User A would reduce the window while user B would increase the window size. Let us assume that router R is busy for a 'long' period of time. If the demands are not re-evaluated, then user A would continue to see the bits set and therefore reduce it's window size while user B would increase the window size. This continues till we find that B receives a much higher share of R's capacity than is fair, while A slows down and therefore receives a much lower share of R's capacity. The policy would thus result in the allocation going to a fair one and then subsequently becoming unfair. Thus, we find that when the router is busy for a 'long' time, there is a need to re-evaluate the demands of the individual users.

As it is difficult to define what is a 'long' period of time, we adopt the same strategy that we had earlier for the adaptive averaging. This is to determine the demands over the 'previous' (busy−idle) cycle and then to add to this the demands over the portion of the 'current' busy cycle. The demands over the 'previous' cycle are erased at the end of the 'current' (busy+idle) cycle.

4.3 Capacity Estimation

For determining the users that would have 'congestion indication' bit set, the selective feedback algorithm used the 'capacity' of the router, which at the maximally efficient operating point is the 'knee capacity'. The 'knee capacity' is the number of packets that the router may process when it is operating at the 'knee', which however is not known to the router apriori. This is typically less than the service rate of the router (the 'maximum capacity' of the router). Instead of using a predetermined value for the knee capacity of the router, we compute this value from the processing that is done by the router itself.

Over each (busy+idle) cycle interval, we know the number of packets that have been processed by the router, as an aggregate value, summed across all the users of the router. When the router is not congested, the number processed by the router would be the number of packets that arrive at the router, since the throughput is not limited by the service rate of the router. Under this circumstance, we find that the router does not invoke the selective feedback algorithm. It is only when the router is congested that we invoke the algorithm. When the router is congested, the average queue length builds up. This means that the number of packets that the router processes during the period over which the queue length is estimated is limited by the service rate of the router, which is the 'maximum capacity' of the router. The 'knee capacity' of the router, is a value that is smaller than the maximum capacity, when the service time distribution is anything other than deterministic. The throughput of the router if modeled as an M/M/1 server (Poisson arrivals, exponential service time), is half of the 'maximum throughput' when it is operating at the knee. Since we do not know the distribution of the inter-arrival and service times apriori, we use a factor to multiply the maximum throughput estimated over a cycle to determine approximately the 'knee capacity' of the router when it is operating at the knee. This factor, which we call the 'capacity factor' is clearly between 0 and 1. We find that the performance of the network, is better with larger values of the capacity factor (close to 1). In the following, we study the sensitivity of the performance of the network with varying values of the 'capacity factor', primarily from the viewpoint of convergence of the performance of individual users to fair values.

4.3.1 Sensitivity to Capacity Factor in Capacity Estimation

The two characteristics that are relevant as far as the convergence of the network's users to a fair value is the extent of convergence and the time to reach convergence. The measure that appears most appropriate to study the convergence of the algorithm is the throughput of the different users, that share a common bottleneck.

The time delay for convergence is a complex function of the round trip delay, the extent of difference in the service times of the resources that are shared by the users, the parameters of the increase and decrease policies and the signal filtering policies that are adopted at the users.

Both characteristics of convergence are impacted by the determination of the capacity of each individual router - in particular for the routers that are congested. As part of the 'selective feedback' mechanism, the router estimates its 'knee capacity' to be used by the selective feedback algorithm. The algorithm chooses the users that have their 'congestion indication' bit set. If the router is congested (i.e., the average queue length is greater than 1), then the number that has been processed is the service rate of the router. The knee capacity is estimated using a 'capacity factor' - *cf*, to multiply the service rate (the number of packets that have been processed in that cycle). The capacity factor determines the number of users that selectively have their 'congestion indication' bit set. We studied by simulation, the sensitivity of the performance of the window and throughputs of the individual users to the value of the 'capacity factor'. Figure 6 and Figure 7 show the throughput of multiple users with the values of the 'capacity factor' (*cf*) being 0.5 and 0.9. These curves are for the configuration of user A going over 2 hops and user B going over 10 hops shown in Figure 2. Comparison of the two figures shows that a smaller *cf* improves the throughput of the user A who goes over a smaller number of hops. As we increase *cf*, the throughput of the user B increases, at the expense of the other user. One possible explanation for this is that when any one of the routers is congested, a lower *cf* results in both of the users being targeted to have their 'congestion indication' bit set. As a result, the user going through a smaller number of hops (user A) would experience a better throughput at the expense of the user going over a larger number of hops (user B). Since that user obtains a smaller amount of the router's resources, a very low estimate of the knee capacity would mean that user B would also have become a candidate, unlike the situation when the knee capacity is estimated properly. A larger capacity factor *cf* doesn't set the congestion indication bit for user B earlier than essential.

4.4 Modifications for an Overloaded Resource with Varying Individual Users

When a resource is overloaded (busy, with no idle period at all), for an extended period of time, the selective feedback algorithm deals with the situation of allocating a fair share of the resource among the users who place a load on it. However, it only knows the demands of individual users from the time the current busy period started. If the busy period is caused by a small subset of users, who continuously place a demand on the resource, the accumulated demands of these users would tend to dominate any load placed by new users that arrive a long time period after the busy period began. The new users that arrive have to place a load on the resource for a sufficiently long period before they become candidates for having their congestion avoidance bit set by a congested (and overloaded) router. Thus, new users have their individual window sizes grow to a large extent before they place a large enough demand. As a result, the new users are not selected to have their congestion indication bit set, although the router is overloaded and all of the users should in fact have their congestion indication bit set so that all their window sizes are further reduced. The router operates beyond the knee, as defined for maximal efficiency, for a significant period of time.

This limits the dynamic range of the selective feedback algorithm. There are several alternatives to solve the problem posed by such a situation. One is to have a limit on the time during which demands are retained, even though the router continues to be busy. Another is to not have selective feedback when we have the router identify that it is in such a situation. One over-riding consideration we have is to introduce as few parameters as possible, particularly when the parameter may be highly configuration sensitive. This reason made us rule out the choice of a time interval to drop information regarding user demands.

The modification we have adopted to the basic 'selective feedback' algorithm is to have an over-riding policy which turns off the selection mechanism when we are beyond the dynamic range of the selective feedback algorithm. This means that when a router is overloaded beyond a certain point (we have found that a threshold value of 2 for the average queue size is a reasonable point), the algorithm followed by the router is to set the bit on all the packets that pass through it, rather than performing it selectively.

The modification limits the capability of the network to be selective when a large number of users are consistently congesting the network resources, but still operates the network close to the maximally efficient operating point, while not being as fair as would be ideal under these extreme circumstances. We show the behavior of an overloaded network with this modification in place in Section 5.6.

5 Simulation Results

Several characteristics of the overall congestion avoidance policy were studied through simulation. We first consider the characteristics of fairness in the service offered to the users by the network when the users are using different paths. We may study this by considering the overall throughput achieved by the users. Let us first look at a configuration where two users share a common bottleneck, but have different path lengths. All the routers have the same service time of 2 units, while the users have a nominal service time of 1 unit. This configuration (we shall call this configuration 1) is shown in Figure 2. User A goes over 2 hops, while user B has a longer path of 10 hops. Let us observe the throughput of the overall system and the individual users, as shown in Figure 5. User A initially sees a higher throughput. The initial difference in throughputs is because of the delay involved in the routers (and particularly the bottleneck) to determine the fair allocation and communicate it to the users. Furthermore, there are considerable differences in the delays in feeding back the signal to the two users. This impacts the frequency of update of the window size by the users. User A (going over the shorter hops) receives the feedback earlier compared to user B and therefore, in the initial phase, increases the window size more rapidly than user B. Also, since the optimal window size for user A is smaller than for user B (it is clear in this simple configuration), user B would wait for the acknowledgement of a larger number of packets than user A, between updates. This can be seen in the figure showing the behaviour of the window sizes of user A and B in Figure 4. As time progresses, with both the users active, the throughput of user A reduces and that for user B increases, to the point where they converge.

We tested the 'selective feedback' scheme that we propose in this report for a variety of configurations and number of users of the network. The sequence of tests undertaken is described in [JR87].

5.1 Efficiency and Fairness

The first case we considered was to confirm that the scheme continued to work efficiently when the users of the network shared the same path, which was the assumption we had made in the basic scheme proposed in [RJ87]. We use a path of non-homogeneous routers in the network, including a satellite link in the path (we called it the MDI configuration in [JR87]). We find that the 'selective feedback' scheme continues to maintain the optimal window size for the network, as we had seen in the original scheme. This is because of the fact that when all the users are sharing the same path, all of the users are either candidates for setting the 'congestion indication' bit or all of them are not. Thus, we continue to operate at the efficient point. Introduction of the 'selective feedback' does not adversly impact the performance of the network with all the users sharing the same path.

The next case we studied was the situation where instead of the two users using identical paths, only some part of each individual path are shared. We use configuration 1, shown in Figure 2 to illustrate most of the features of the 'selective feedback' mechanism. The service times of all servers in the network are 2 units each. The two hops used by user A are shared between the two users. The optimal operating point is such that the throughput of each of the users through the shared servers are identical (since these are the bottlenecks in the configuration). Figure 5 shows the graph of the throughput of the two users, varying with time.

Consider the issue of efficiency first. The throughput of user A (using the short path) increases rapidly to the optimal value of 0.25, while the throughput of user B increases more gradually, and reaches its value of 0.25 at a later point. This is because user A updates the window more frequently and reaches it's optimal point earlier, while the initial increase of the window of user B is slower, since its round trip delay is larger. When the bottleneck server reaches its optimal operating point, user A is using a larger share of the server's resources compared to user B. User A reduces its window while user B increases the window, until the two users reach a point of sharing the resource equally. We therefore achieve the goal of having the throughputs reach the maximally efficient operating point.

The second issue is of fairness. Initially, the throughput for the individual users are at an unfair value, as the users going through multiple hops take longer to have the throughput build up. As we progress in time, the throughputs build up till we converge to a fair value. We therefore also see that the scheme achieves the maximally fair value as well.

5.2 Robustness of the Scheme

The service time of the packets at each server is assumed to be proportional to the packet size. We have studied the operation of the scheme for different packet size distributions including exponential and uniform distributions. Figure 8 shows the behavior of the throughput of the individual users, with an exponential packet size distribution for the same configuration described earlier with two users. We find that the throughput of the two users reach a fair value, in about the same time as we saw with deterministic packet size distribtions.

Figure 9 shows the behavior of the throughput of the individual users, with a uniform packet size distribution. The results appear to be similar to that seen with the exponential packet size distribtion.

5.3 Response to Transients

One very important and necessary characteristic of a congestion avoidance scheme is the ability to respond to transient changes in the network. Transients may be with respect to changing number of users of the network as well as changes in the configuration or service characteristics of the servers in the network.

5.3.1 Response to Service Time Transients

Let us consider the response to transients in network service characteristics first. Figure 10 shows the behavior of the throughput with time, when the service time of the bottleneck router changes from its original service time to twice its value. We believe that this transient would reflect situations such as a network reconfiguration and change to an alternate path.

In the simulation, users start up initially on an idle network. After some time has elapsed, measured in the number of packets that are transmitted relative to the total number to be sent during the period, the bottleneck service time increases to twice its value. Finally, during the third and final phase, the service time of the bottleneck goes back to the original value. The throughputs of the individual users increase in the beginning as the users enter the network and increase their window size according to the increase policy. In this example, even before the two users reach a fair value, the transient occurs, causing the decrease in the throughput, reflected more prominently for the user going over a fewer number of hops. Also shown in the curve is the overall throughput, which also reduces. During the transient, we see a consistent decrease in the throughtput, primarily because the throughput is a time average starting at T=0. The instantaneous throughput levels off for the two users after a rapid decrease limited by the service time of the bottleneck. At the end of the transient, the throughput of the two users start to increase back to their new target values. The window sizes also recover to their original values. Notice that the two users do not finish as close to each other as in the example without the transient, primarily because the users did not have the opportunity to reach a fair value before the transient occured.

The example shows that the scheme is responsive to the transients in the service time of the servers in the network.

5.3.2 Response to Transients in Number of Users of Network

We show in Figure 11 the behavior of the window size when there is initially one user who achieves steady state and subsequently another user joins the network. The transient change in the load on the network should result in the allocation of the resources of the network also changing. The new user progressively gets a greater share of the resources, until all the users have a fair share of the resources. We find that ultimately, the steady state window size of the two users reaches a fair value. This is clearer, when we look at the behavior of the throughput, shown in Figure 12, which shows that the two throughputs of the users reaches equal values, after beginning at different values. These tests show that the scheme responds to the typical transients that are seen in a computer network.

5.4 Starting at an Arbitrary Window Size

An important characteristic of any dynamic scheme is the need to be able to converge to the correct values irrespective of the starting point. We test the convergence of the congestion avoidance scheme by allowing users to start at arbitrary initial values for the window size. The Figure 13 shows the behavior of the window size of the two users for our configuration 1, when the users start at a window size of 32. We find that both the users receive congestion indication bits which are set so that they consistently reduce their window sizes until their window sizes reach the optimal value. Figure 14 shows the behavior of the throughput for this case as well. We find, as we would expect that the two users have unequal throughputs to start with. But as they reduce their window sizes, and reach a steady state, they also reach a fair value relative to each other.

5.5 User Bound Configuration

Another test that the dynamic scheme needs to satisfy is the case where the throughput and hence the window size is limited by a user bottleneck. To test this we use configuration 1 once again. The difference here from the previously chosen configuration is that the users have twice the service time of the routers. Figure 15 shows the behavior of the window size. The individual user windows increase initially, starting from the initial value of 1, as the network feeds back information indicating that it is not congested. When the throughput out of the users (and hence the corresponding window size) reaches the point where it is limited by the service rate of the individual users, the window size does not increase anymore, even though the network is not congested. This demonstrates the capability of the scheme to adjust to bottlenecks at the users as well as the network.

5.6 Behavior in an Overloaded Network

This test relates to the behavior of the scheme when the number of users is substantially larger than the aggregate optimum window size of the network. We use a different configuration here than the one used in the previous experiments. This had 4 routers with different service times of 2, 3, 4 and 5 relative the user's service times. There were 9 users which started placing a load on the network in a staggered fashion, with each user sending 1000 packets. We find that as long as the number of users were less than the optimal window size which is around 3.5, the oscillations that we typically observe are in place. As we increase the number of users on the network, the basic selective feedback algorithm fails to accomodate such a wide dynamic range of loads on the network. When the network becomes overloaded, each of the individual routers see a prolonged busy period. As new users come on to the network, the users that have already been on the network have a larger accumulated demand, compared to the new user's demands. Thus, the new users have their individual window sizes grow to a large extent before they place a large enough demand to be chosen by the selective feedback algorithm. As described in Section 4.4, we have a modification which is to have an over-riding policy which turns off the selection mechanism when a router is overloaded beyond a certain point (we have found that a threshold value of 2 for the average queue size is a reasonable point). Figure 16 shows the behavior of the window size with 9 users for this ND9 configuration, operating under selective feedback, with the modification in place. We find that we still have one user who experiences delay in receiving the feedback signal, which results in that user being allowed to increase the window size beyond the optimal point (in this case it would be 1) before having the congestion avoidance bits set. We feel that the extent to which this behavior is sub-optimal is limited, and therefore, the modification being adopted is satisfactory

6 Areas for Further Study

6.1 Router Based Approach

In this section, we discuss some of the related characteristics of the selective feedback scheme. Further work is needed to examine these properties and evaluate, quantitatively, their effect.

In a hostile environment, with non-cooperating users, selective feedback can play a useful part. When users who do not cooperate and continue to use an unfair share of their resources, the network server that is congested may selectively provide poorer service to those guilty users. For example, the server may drop packets of those users who persistently use a larger share of a congested server. Another advantage is that such poorer service need to be provided only when the user is contributing to the congestion, and even then, only as a penalty.

In a connectionless network, adhereing to the principles of layering, only the source and destination addresse are visible at the router. This implies that fairness is provided only across source-desitnation pairs and not logical links as would be ideal. As a result, if a user were to attempt to obtain more than his fair share of the network's resources, this may be achieved by setting up more logical links. Thus, it is now the responsibility of the end systems to provide the control to avoid such counter measures adopted by users. We believe that these policies may be incorporated in the flow control mechanisms at the end systems.

Another consideration to keep in mind is the additional overhead that is imposed on the router to perform the selective feedback algorithm. First, there is the cost of looking up the source and destination addresses. In the normal forwarding of the packet, the destination address is the only one that is looked up. As a result of the selective feedback, we need to look up the source address as well. The demands are maintained at each router, by having a small table indexed by the source-destination address pair. The size of the table is determined by the maximum number of source-destination pairs (users) that would be placing a demand on the router during a busy period. If the table overflows, then we would no longer be fair to the rest of the users. This means that we have a tradeoff between the number of users that we would like each router to be fair across versus the size of the table and the cost of look-ups on that table. This is one possibility for minimizing the overhead of selective feedback.

Another means to reduce the overhead of the selecive feedback algorithm is to execute the algorithm periodically. Updates to determine who are the users (source-destination pairs) that are using an unfair share of the server's resources is currently being performed on every packet arrival, in synchrony with the update to the averaging algorithm. We believe that this update need not be performed at such a frequency, as a means of mitigating the cost of this selective feedback. We may be able to perform such an update every n packets. Clearly, once every cycle is insufficient, since we see that a long busy period with selective feedback can lead to unfairness again. This is because a user who is receiving the selectively set 'congestion indication' bit would reduce the window while others would increase the window. Thus the user may be forced to continue to reduce the window resulting in providing very little (and unfair share) of the congested server's resources to this user if the busy period continues for very long. Therefore, updating the decision to select the users that have their congestion indication bit set may done once every n packets at the server, where n has to be determined.

6.2 Transport-based Alternative

Another approach to achieve the global efficiency and fairness criterion is based on modifying the increase and decrease algorithms in the transport entities; we call this approach ***transport-based approach***. For the transport-based approach, the requirement for the network to generate feedback is the same as in [JR87], [RJ87]. Therefore the network can be kept simpler in its role of providing feedback, without having to recognize addresses as in the selective feedback case. It is entirely the responsibility of the clients of the network (i.e. transport connections) to regulate their flow rates to achieve efficient and fair use of the network.

At this time, we have not been able to design an algorithm based on this approach that can be proved to converge to the desired optimal operating point and can be demonstrated to be reasonable for implementaión. Thus we have chosen the selective feedback scheme. Below, we will describe the basic approach, some algorithms we considered, and discuss some of the issues still to be addresses to make it a feasible solution.

6.2.1 Description of Basic Algorithm

The basic algorithm is similar to that outlined in [RJ87]. The difference is that here we explicitly implement an mechanism for the purpose of achieving equal throughput rates for all transport connections sharing the same bottleneck. The transport-based approach can be desccribed be concentrating on a set of transport connections sharing the same bottleneck, as follows:

pairs) that are using an unfair share of the server's resources is currently being performed on every packet arrival, in synchrony with the update to the averaging algorithm. We believe that this update need not be performed at such a frequency, as a means of mitigating the cost of this selective feedback. We may be able to perform such an update every n packets. Clearly, once every cycle is insufficient, since we see that a long busy period with selective feedback can lead to unfairness again. This is because a user who is receiving the selectively set 'congestion indication' bit would reduce the window while others would increase the window. Thus the user may be forced to continue to reduce the window resulting in providing very little (and unfair share) of the congested server's resources to this user if the busy period continues for very long. Therefore, updating the decision to select the users that have their congestion indication bit set may done once every n packets at the server, where n has to be determined.

6.2 Transport-based Alternative

Another approach to achieve the global efficiency and fairness criterion is based on modifying the increase and decrease algorithms in the transport entities; we call this approach ***transport-based approach***. For the transport-based approach, the requirement for the network to generate feedback is the same as in [JR87], [RJ87]. Therefore the network can be kept simpler in its role of providing feedback, without having to recognize addresses as in the selective feedback case. It is entirely the responsibility of the clients of the network (i.e. transport connections) to regulate their flow rates to achieve efficient and fair use of the network.

At this time, we have not been able to design an algorithm based on this approach that can be proved to converge to the desired optimal operating point and can be demonstrated to be reasonable for implementaión. Thus we have chosen the selective feedback scheme. Below, we will describe the basic approach, some algorithms we considered, and discuss some of the issues still to be addresses to make it a feasible solution.

6.2.1 Description of Basic Algorithm

The basic algorithm is similar to that outlined in [RJ87]. The difference is that here we explicitly implement an mechanism for the purpose of achieving equal throughput rates for all transport connections sharing the same bottleneck. The transport-based approach can be desccribed be concentrating on a set of transport connections sharing the same bottleneck, as follows:

1. The bottleneck network resource gives the same feedback to all transport connections sharing it;
2. Each transport connection calculates the *current* throughput rate at window adjustment epochs;
3. All transport connections applies the same increase/decrease algorithm to adjust its new throughput demand (or allocation) for the next cycle.
4. A new window size is approximately calculated according to the new throughput rate.

Guaranteeing the same feedback to the transport connections relies on the same router policies in the original feedback scheme.

The calculation of the *current* throughput rate can be accomplished in a number of different ways. The straight forward way is to measure the time interval between window adjustments, T_{wa}, and record the number of packets transmitted during this interval, N_W (with window size W). Then the throughput

$$A = \frac{N_W}{T_{wa}}$$

is the effective *current resource allocation* at the bottleneck. Since T_{wa} may have some randomness associated with it, independent of congestion conditions, it is appropriate to filter it (by using for example the exponential averaging in roundtrip time calculations). Most transport protocols make some time measurements for each connection, for example the roundtrip delay or interpacket time. It is thus possible, as an implementation optimization, to base the throughput calculation on such time measurements, saving the need for additional time measurements.

Once the throughput rate is calculated, an appropriate increase/decrease algorithm can be applied to adjust the rate A according to the feedback b.

$$A_{new} = f(A, b)$$

The choice of such an increase/decrease algorithm that leads to fairness as well as efficiency is detailed in [CJ87]. In our experimentations, we have used the simplest additive increase and multiplicative decrease algorithm for $f()$.

Finally the calculation of the new window size can be done by using the reverse mapping that we used for calculating the throughput rate from the current window size. Suppose the new allocated throughput is A_{new} and assume the new interval time is approximately the same as the current interval time. Thus the new window is

$$W_{new} = A_{new} T_{wa}$$

6.2.2 Issues Need to Be Addressed

While the transport-based approach is attractive considering its simplicity in the router policy, there are a couple of major issues that require further studies:

1. The analysis in [CJ87] assumes that all the clients (transport connections) make their adjustments synchronously. In other words, all connections sharing the same bottleneck must adjust their windows and allow some time for the result to be observed, before they all adjust again; and so on. It is not clear how this synchronization is best achieved. The simple method amounts to making the frequency of window adjustments conmensurate to the lengthiest roundtrip delay time. This would severely compromise the convergence time of the algorithm. The nonlinear algorithms described in [CJ87] are less stringent on this synchronous adjustment requirement. But they typically require careful setting of more parameters based on network speed and size, thus making the algorithm more configuration dependent.

2. In [RJ87] the increase policy uses an incremental amount of 1 for adjusting the window size. This increase increment has some concrete meaning in terms of resource consumption. In order for the transport-based algorithm to be effective, it is necessary to select an incremental rate for the increase policy that is conmensurate to the network configuration. Specificly, this incremental rate should be a small fraction of the bottleneck resource's throughput. If the incremental rate is too high, then we suffer from more burstiness and oscillation; if the incremental rate is too low, we would reduce to the orginal scheme of increase window by one. [1] This sensitivity to parameter value compromises the configuration independency goal.

These issues require further studies to make the transport-based approach attractive.

[1]This assumes that we make the minimum increase to be one in window size

7 Conclusions

In this paper, we have studied the binary feedback scheme introduced in [RJ87] in greater detail, with regard to fairness of the allocation of resources to users. The scheme proposed in [RJ87] uses a feedback mechanism to indicate to users of the network (users being transport level entities) whether the network is congested or not. An assumption made in the analysis of the scheme was that all users shared the same set of network resources, i.e., routers. When the users do not share the same set of routers, the scheme proposed in [RJ87] does not result in a fair allocation of the network resources.

In this paper, we began by defining a generalized goal for fairness in the allocation of resources, where users share arbitrary sets of resources in the network by using a constructive algorithmic definition. We then considered two alternative schemes for achieving the goal. The first requires that transport entities modify their increase algorithms to account for the differences in the paths that are shared by users. The second requires that routers selectively feedback the congestion avoidance information to the users (transport entities). Based on engineering considerations, detailed in Section 4, we have adopted the alternative in which routers selectively feedback the congestion information.

The selective feedback algorithm implemented by the routers distinguishes between the different source-destination pairs, in providing fair allocation of the resources of the network. We described the selective feedback alogrithm and the scheme to estimate the demands of the users of the network on each individual router. We also introduce another parameter, called the capacity factor, that is needed to estimate the knee-capacity of the router from the measured number of packets processed by a congested router. We studied the sensitivity of the performance of the network, both in terms of efficiency and fairness, to the capacity factor and recommend a value close to 0.9. The interval over which the demands were estimated was using the same adaptive averaging algorithm that was used by the congestion detection mechanism.

We also presented simulation results indicating that the behavior of the scheme was acceptable under a variety of network conditions, including transient changes in network or load characteristics, overloading of the network and behavior under different service time distributions for packet sizes.

8 Acknowledgements

Several other individuals participated in the refinement of the design of the binary feedback scheme to be applicable for general topologies. In particular, Tony Lauck provided considerable input into the solution. Bill Hawe, Art Harvey, Dick Flower and Joel Emer provided useful feedback to help us solve the problem. Almost all the members of the architecture group contributed to the refinement in one way or another. We would also like to thank Dave Oran and John Harper for their feedback and interest.

References

[CJ87] Dah-Ming Chiu and Raj Jain. *Congestion Avoidance in Computer Networks with a Connectionless Network Layer, Part III-Analysis of the Increase and Decrease Algorithms.* DEC Technical Report TR-509, Digital Equipment Corporation, April 1987.

[GB82] Eli Gafni and Dimitri Bertsekas. Dynamic control of session input rates in communication networks. In *Proceedings of MILCOM '82*, Boston, Massachussetts, October 1982. (MIT Technical Report LIDS-P-1228).

[Hah85] Ellen L. Hahne. *Round Robin Scheduling for Fair Flow Control in Data Communication Networks.* Proposal for Thesis Research, Massachusetts Institute of Technology, May 1985.

[Jaf81] Jeffrey M. Jaffe. Bottleneck flow control. *IEEE Transactions on Communications*, COM-29(7):954–962, July 1981.

[Jai86] Raj Jain. A timeout-based congestion control scheme for window flow-controlled networks. *IEEE Journal on Selected Areas in Communications*, October 1986.

[JCH84] R. K. Jain, Dah-Ming Chiu, and William R. Hawe. *A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Systems.* DEC Technical Report TR-301, Digital Equipment Corporation, September 1984.

[JR87] Raj Jain and K. K. Ramakrishnan. *Congestion Avoidance in Computer Networks with a Connectionless Network Layer, Part I-Concepts, Goals and Alternatives.* DEC Technical Report TR-507, Digital Equipment Corporation, April 1987.

[Kle78] Leonard Kleinrock. On flow control in computer networks. In *Proceedings of the International Conference on Communications*, June 1978.

[Kle79] L. Kleinrock. Power and deterministic rules of thumb for probabilistic problems in computer communications. *Proceedings of the International Conference on Communcations*, June 1979.

[Mos84] Jeannine Mosely. *Asynchronous Distributed Flow Control Algorithms.* Ph. D. Thesis LIDS-TH-1415, Massachusetts Institute of Technology, May 1984.

[Nag84] John Nagle. Congestion control in tcp/ip internetworks. *Computer Communication Review*, 14(4), October 1984.

[RJ87] K. K. Ramakrishnan and Raj Jain. *Congestion Avoidance in Computer Networks with a Connectionless Network Layer, Part II-An Explicit Binary Feedback Scheme.* DEC Technical Report TR-508, Digital Equipment Corporation, April 1987.

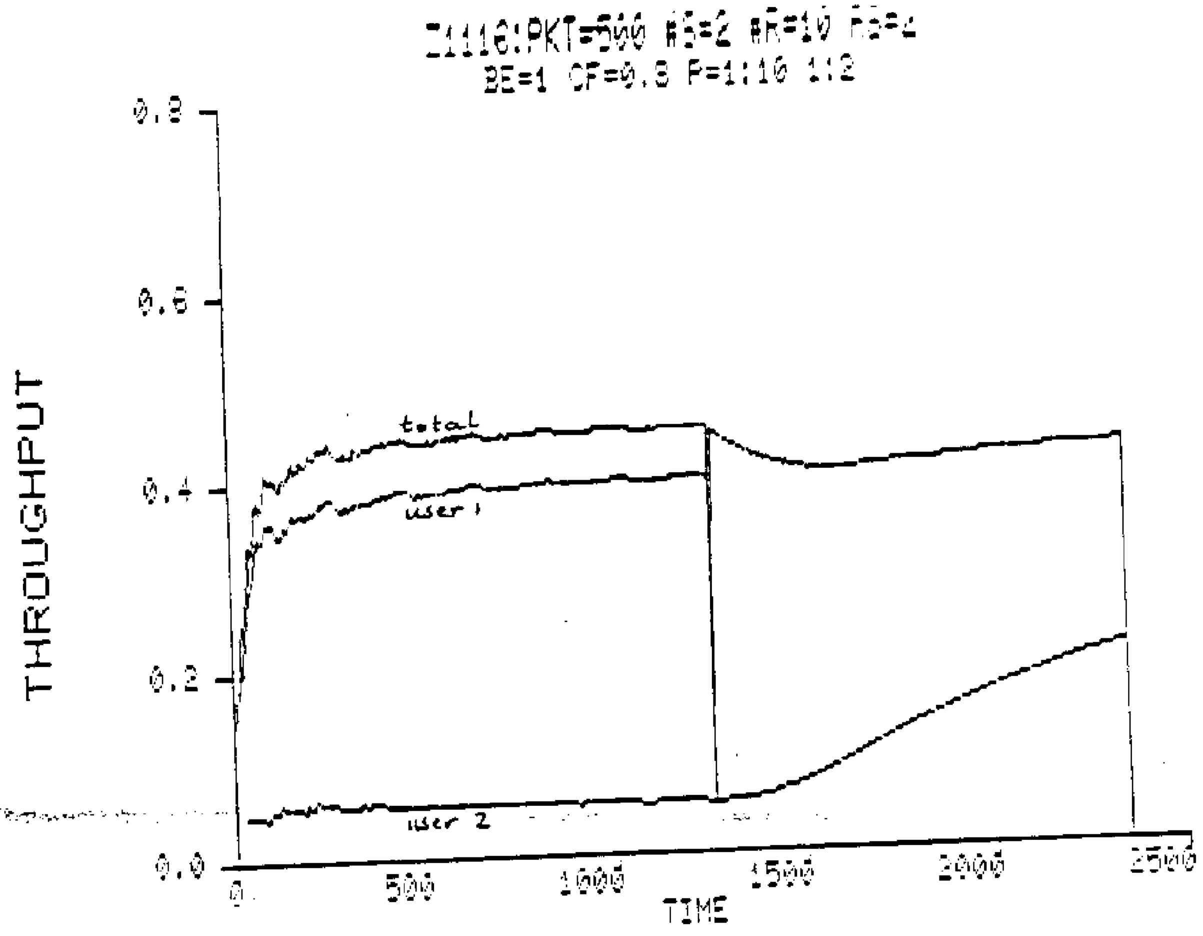

Figure 1: Non-selective bit setting - unfairness with different paths

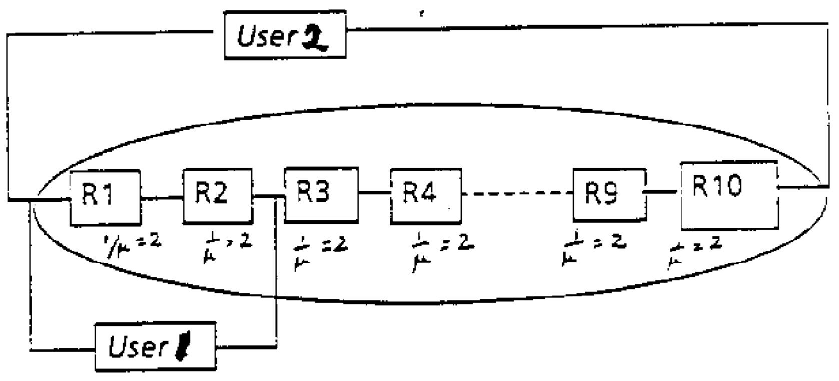

Figure 2: Example General Topology with 2 source-destination pairs: Configuration 1.

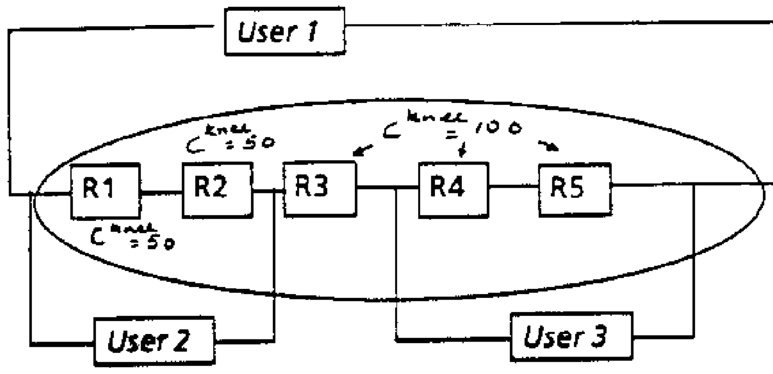

Figure 3: Example Topology with 3 source-destination pairs

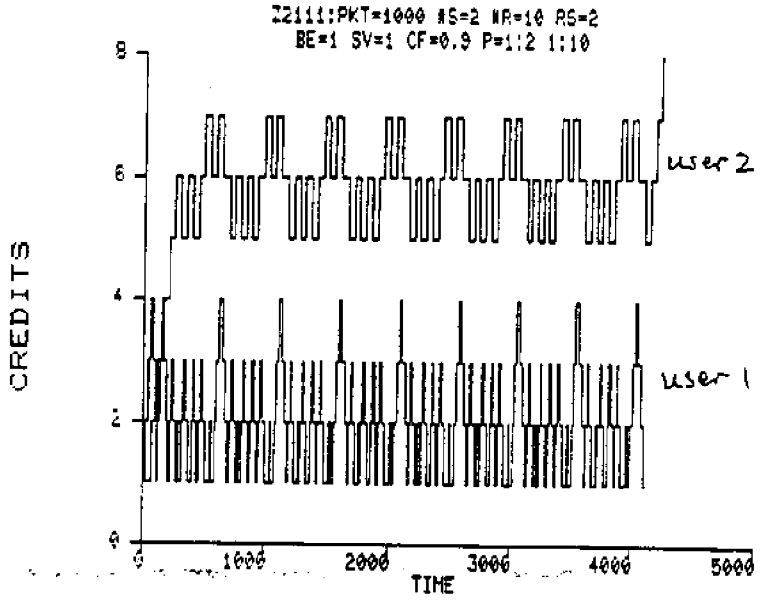

Figure 4: Behavior of window size with Configuration 1

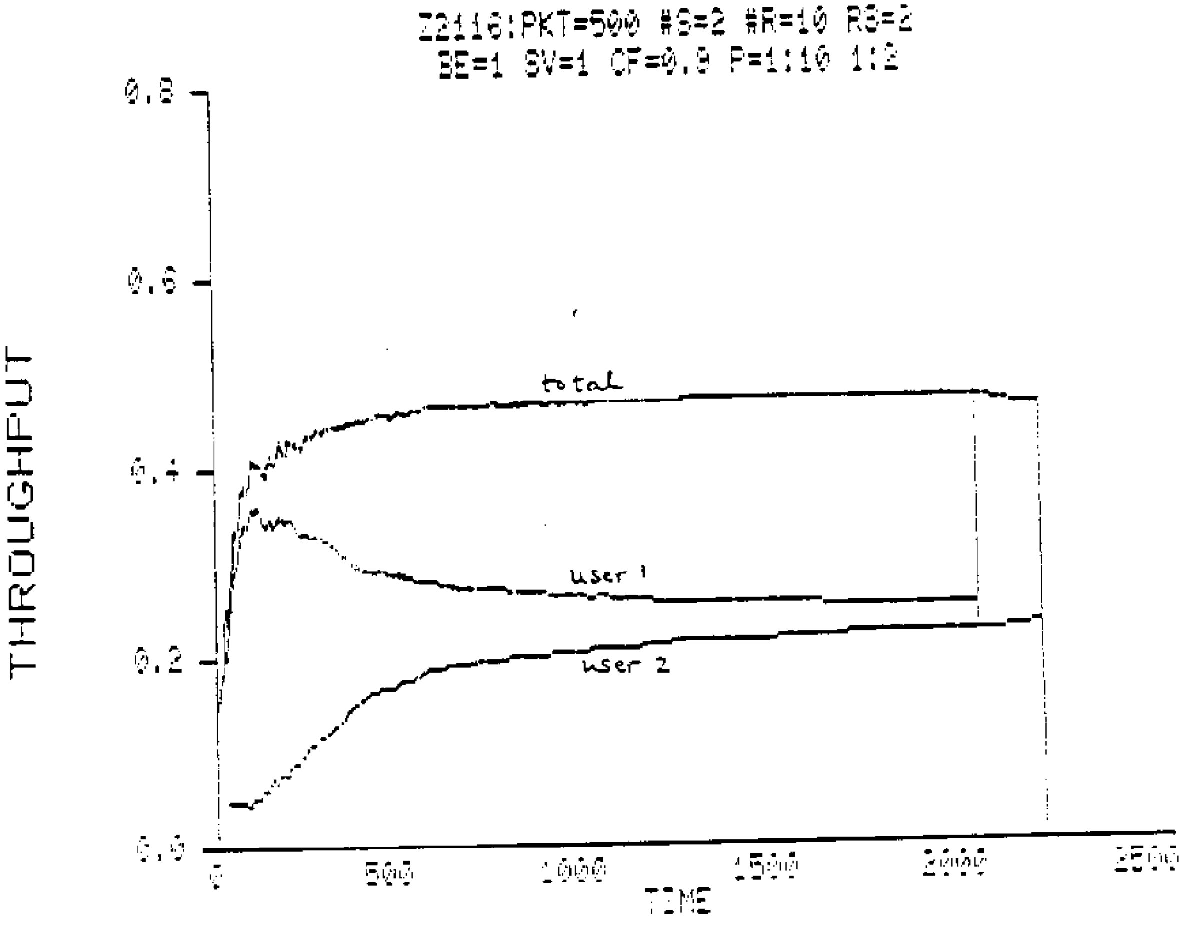

Figure 5: Behavior of throughput with Configuration 1

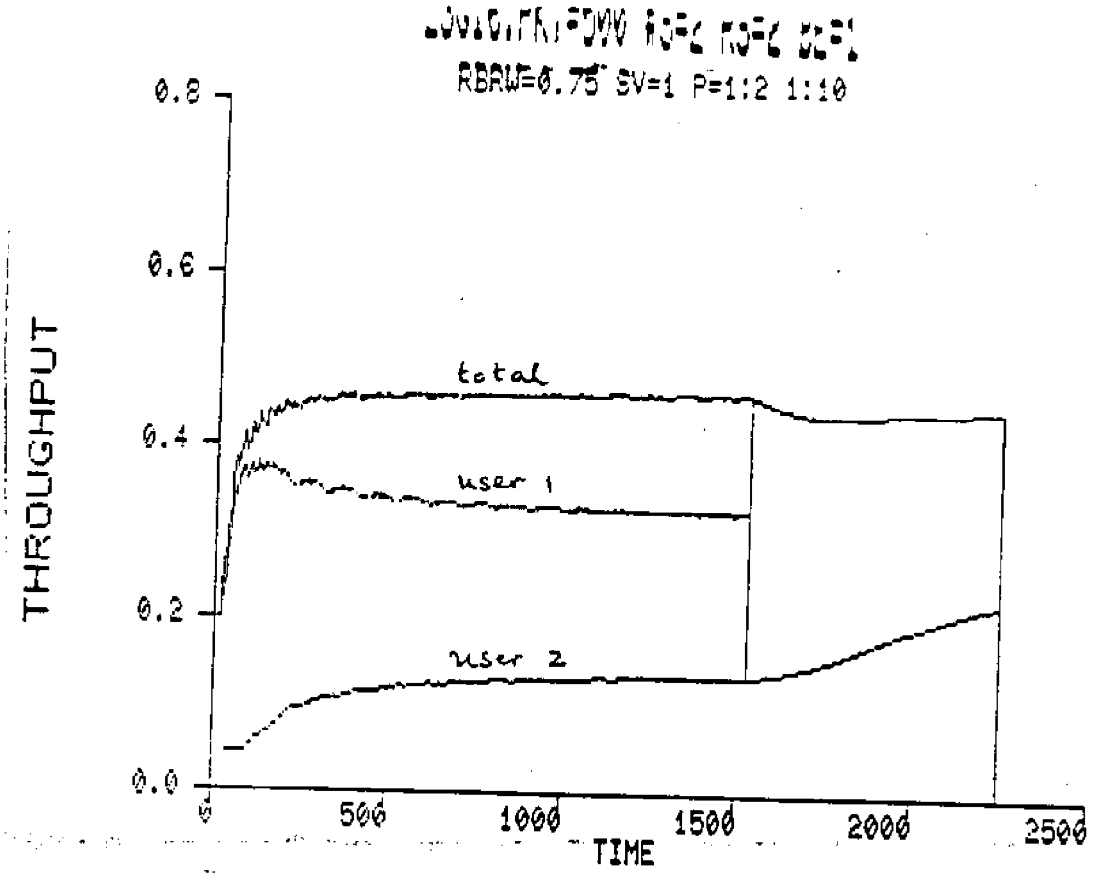

Figure 6: Throughput with Multiple Users, Capacity Factor=0.5, Configuration 1.

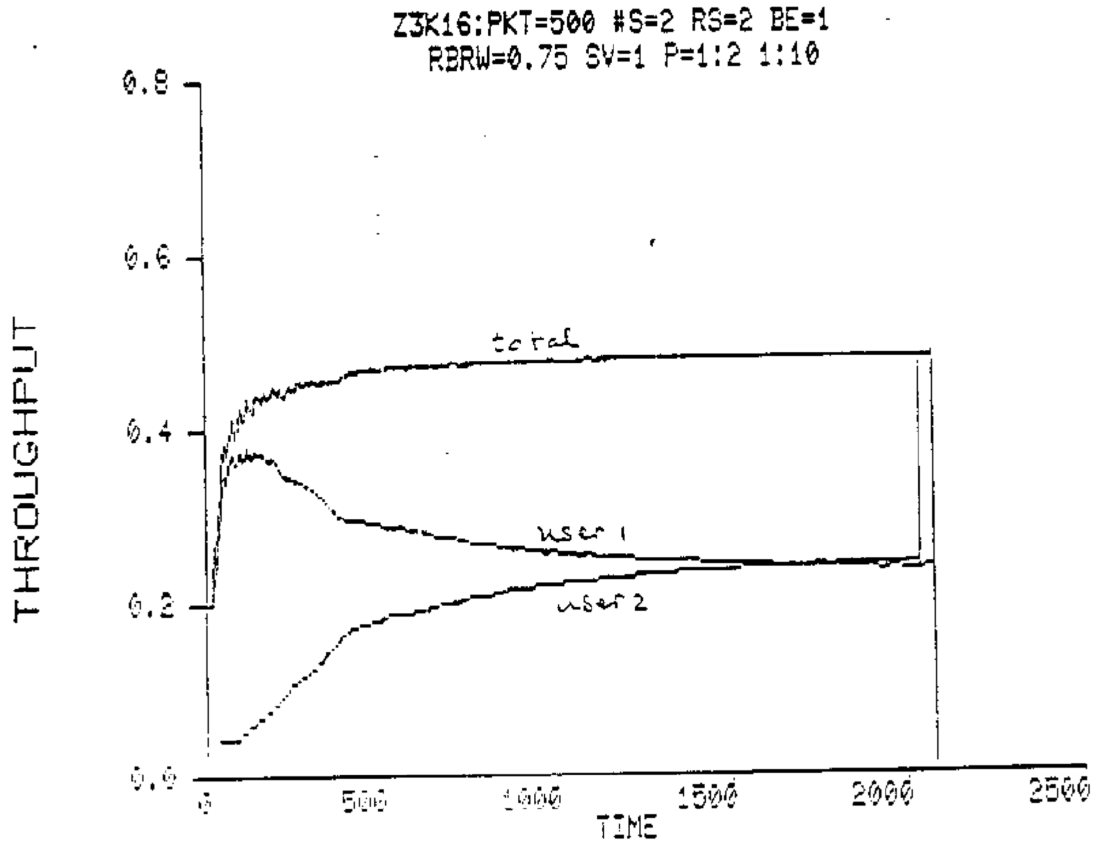

Figure 7: Throughput with Multiple Users, Capacity Factor=0.9, Configuration 1.

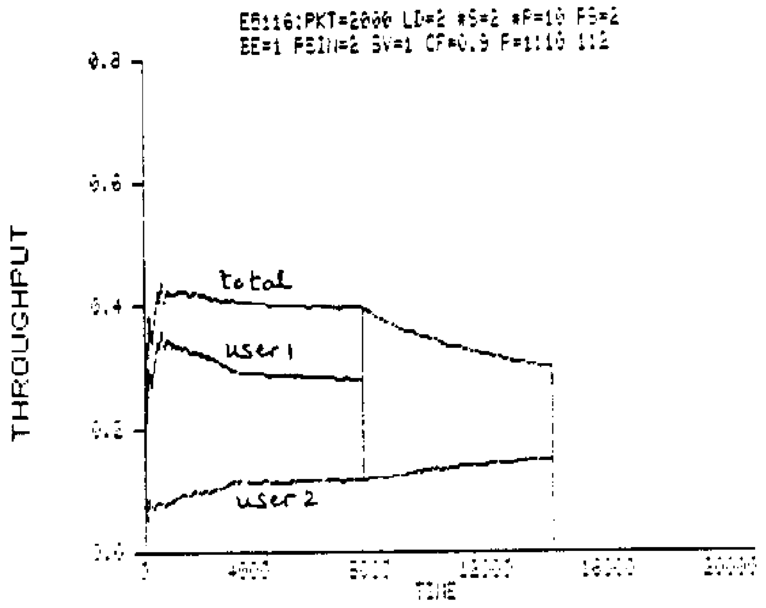

Figure 8: Behavior of throughput - exponential packet sizes, Configuration 1.

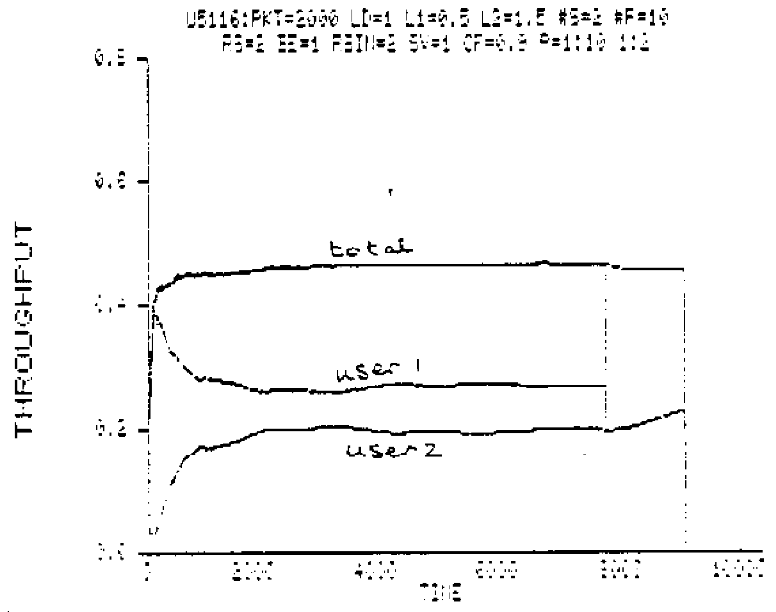

Figure 9: Behavior of throughput - uniform packet sizes, Configuration 1.

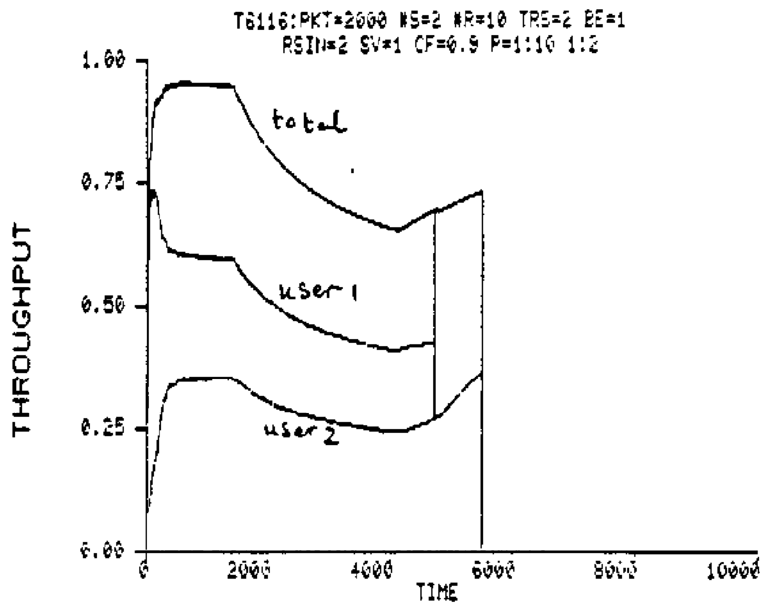

Figure 10: Behavior of throughput - bottleneck service time transient, Config.1.

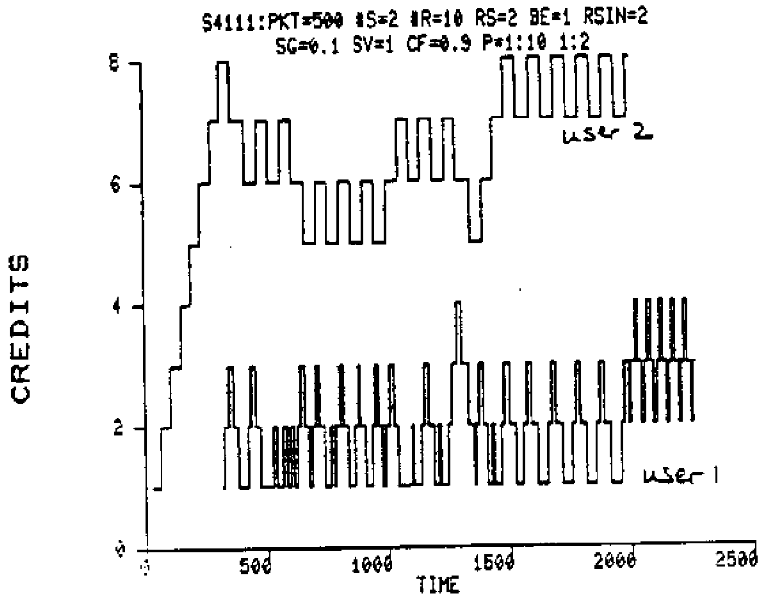

Figure 11: Behavior of window size - transient user demand:staggered start, Config.1.

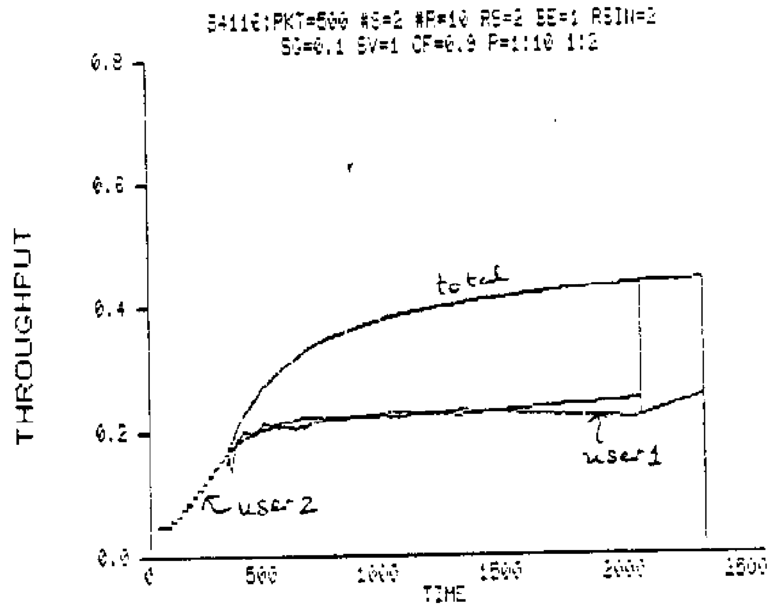

Figure 12: Behavior of throughput - transient user demand:staggered start, Config.1.

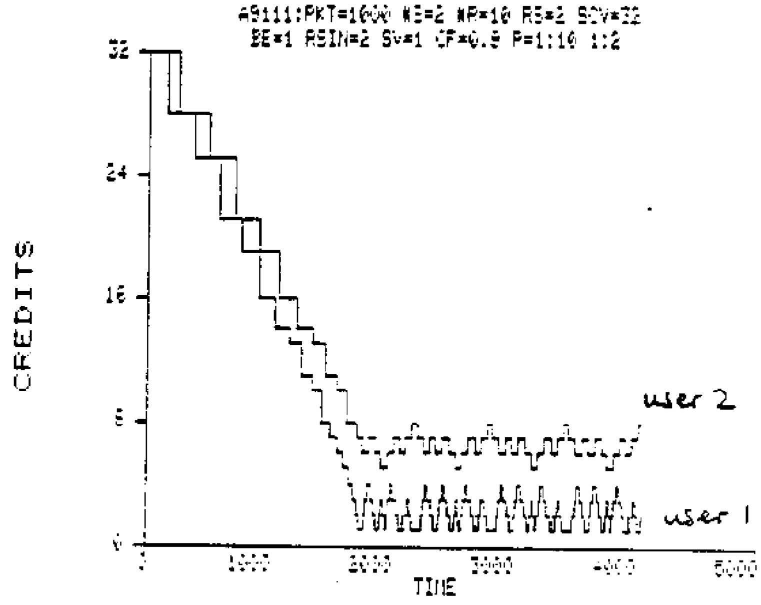

Figure 13: Behavior of window - users starting at arbitrary window sizes, Config.1.

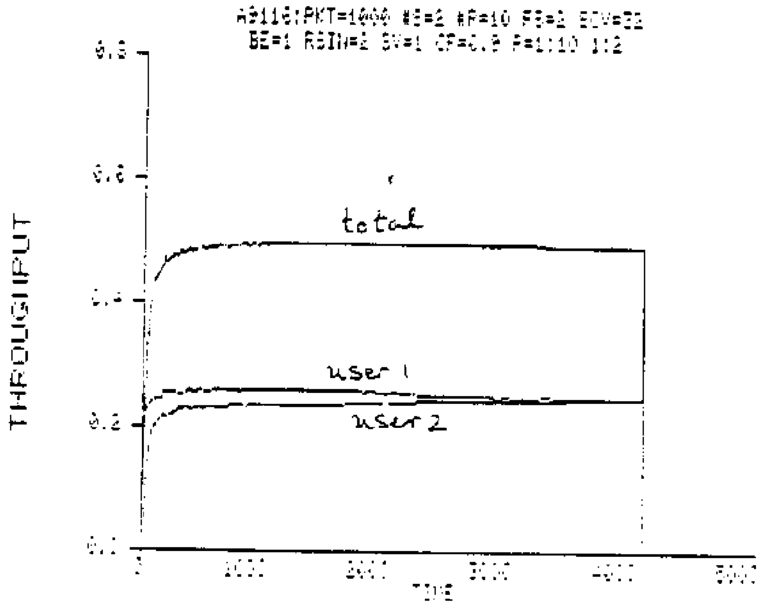

Figure 14: Behavior of throughput - users starting at arbitrary window sizes, Config.1.

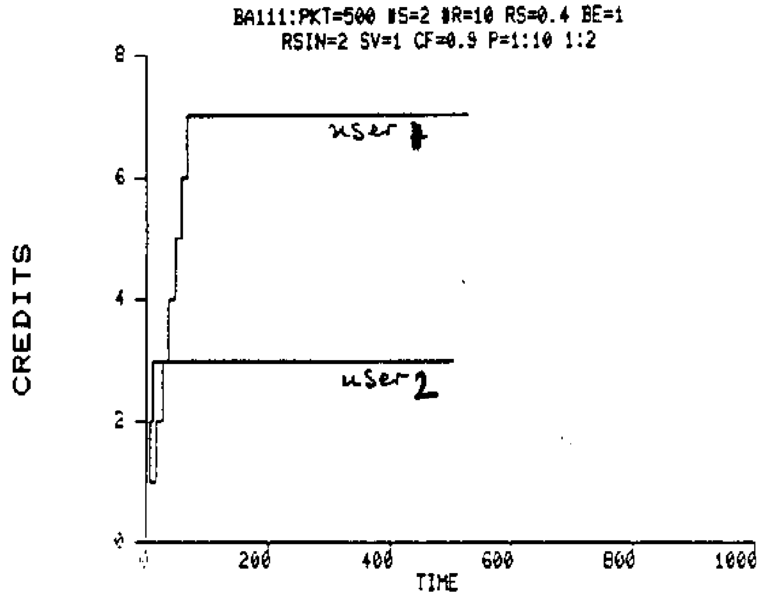

Figure 15: Behavior of Window with a User Bound Configuration

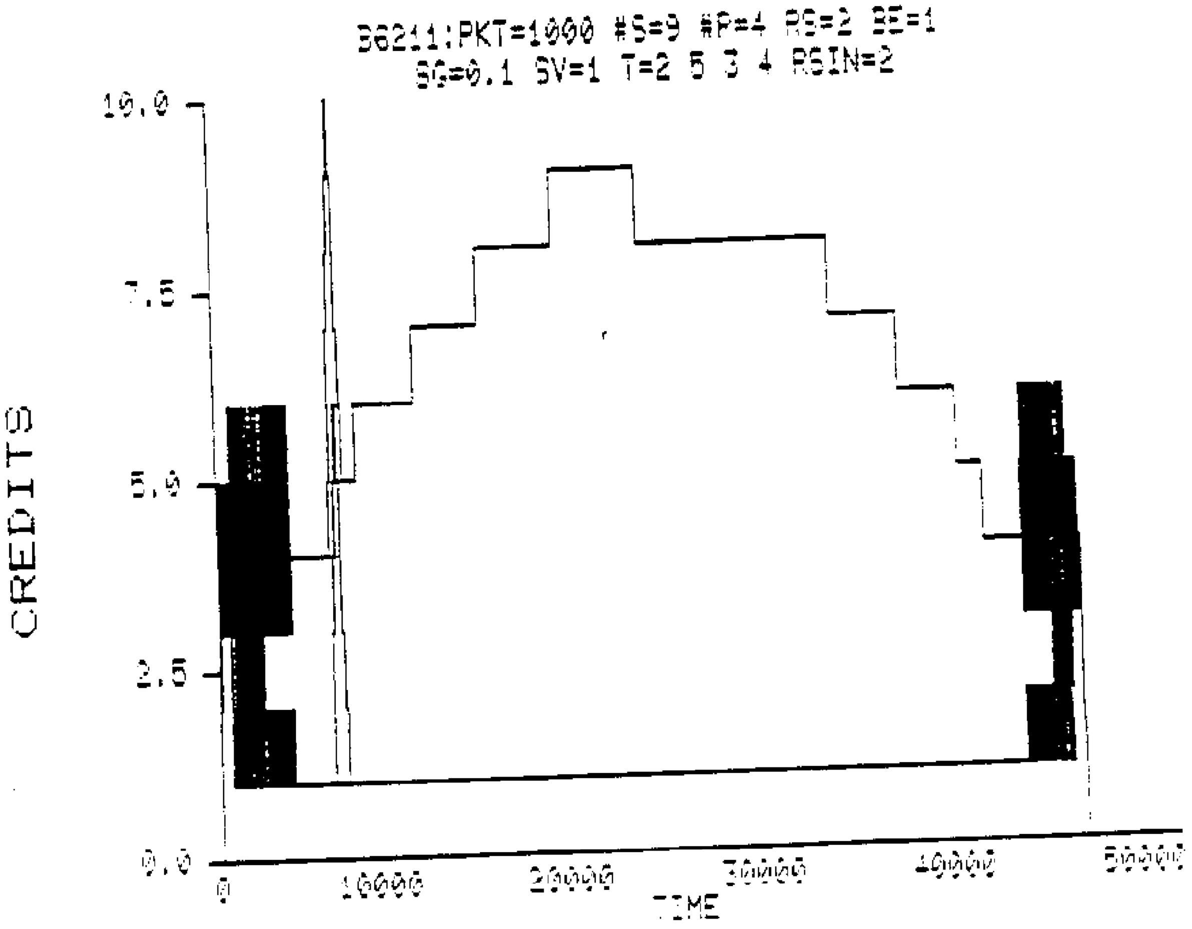

Figure 16: Behavior of Window with an Overloaded Network Configuration