Routing and Rate Allocation to Provide End-to-End Delay Guarantees under PGPS Scheduling

Aniruddha S. Diwan
ECE Dept, IISc Bangalore
diwan@ece.iisc.ernet.in

Kumar N. Sivarajan
Tejas Networks, Bangalore
kumar@tejasnetworks.com

Abstract— **This paper considers the problem of routing sessions with Quality of Service (QoS) requirements in a network under Packet Generalized Processor Sharing (PGPS) scheduling. PGPS is a non-preemptive scheduling policy that tracks GPS. The GPS policy operates by allocating a weight ϕ_n^m for a session n whose traffic uses link m. These weights determine the rate at which the traffic from session n is served at link m and the rate in turn determines the end-to-end delay of packets belonging to session n. As a deterministic and easily computable end-to-end delay bound is available for locally stable sessions, we consider the locally stable regime in this paper. Two separate problems are considered in this paper. The first problem deals with the practically important inverse procedure of specifying appropriate weights for sessions at each link on their paths, that satisfy predetermined delay bounds, when the set of sessions to be routed is given. Here we show that the fixed routing case can be formulated as a *linear program* (LP) and the adaptive routing case can be formulated as a *mixed integer linear program* (MILP). The second problem examines the performance of PGPS scheduling policy when providing per-session QoS guarantees. We measure the performance in terms of *weighted carried traffic*. We derive an upper bound on the weighted carried traffic for *any* heuristic algorithm for admission control that operates within the locally stable domain. This upper bound can be obtained by computing a linear program (LP). By simulating a simple heuristic algorithm for admission control, we show that this upper bound is reasonably tight. Hence our upper bound can be used as a metric against which the performance of different algorithms can be compared.**

I. Introduction

The rise in the popularity of the World Wide Web (WWW) has resulted in the Web becoming the Internet's killer application and today the Web occupies more than 75% of Internet backbone traffic. However, because of the time sensitivity of audio/video and the huge bandwidth requirements of video, network delivery of audio/video posed considerable technical challenge. Until fairly recently, the only practical ways to get music/video were to play from a CD-ROM or to download a very large file across the Internet for playback at the user's desktop. But with the emergence of new technologies like *audio/video streaming* and with the tremendous increase in the link speed (e.g., optical fibers), it is now possible to deliver audio/video over the Internet directly to desktop. These developments have given rise to a plethora of real-time applications that provide users a richer and more realistic experience of multimedia. Soon streaming audio/video will occupy large portion of the Internet traffic.

However, real-time multimedia requires continuous availability of sufficiently high bandwidth in the channel. This is especially important for time-critical traffic like streaming audio/video. Variable audio/video packet delays can cause annoying audio starts and stops and picture jitters. The availability of very high-speed links hasn't solved this problem because of the explosive growth of the Internet. Today, an estimated 100 to 200 million users have access to the Internet. This calls for various network considerations and one of these is the problem of provisioning of Quality of Service (QoS) to the real-time services. Good QoS provides a guaranteed bandwidth at a constant small delay even under congested conditions.

The only way to guarantee QoS is by allocating resources (like bandwidth/buffer) at intermediate routers on per-session basis and *scheduling* the packets at the router interfaces so that a session gets its share of the router resources. The scheduling policy employed at the router interfaces governs the end-to-end delay of the packets. Traditional FIFO scheduling doesn't bound this end-to-end delay and it doesn't provide isolation as a session can block other sessions by sending a continuous stream of packets. Various scheduling policies have been proposed in the literature to provide per-session end-to-end delay (and throughput) guarantees in high-speed networks. Among these, the class of schemes based on the Generalized Processor Sharing (GPS) policy are most popular in the literature. GPS is an idealized fluid discipline with a number of desirable properties such as minimum rate guarantees with perfect isolation among sessions and deterministic and easily computable end-to-end delay bounds. Because these properties are important in the context of guaranteeing QoS in high-speed networks, the GPS discipline has become a basis for an entire range of scheduling policies that are related to GPS.

GPS scheduling was first proposed in [1], [2]. In [3] and [8], end-to-end delay bounds are derived when the session traffic is (σ, ρ)-constrained [4]. Work on analysis of GPS for statistically bounded arrivals is reported in [5] and [6]. Many scheduling policies have been proposed which aim at approximating GPS fairness in a packetized environment [2], [7]. In this paper we consider the simplest scheduling policy called Packet GPS (PGPS) [2]. PGPS is a non-preemptive scheduling policy that tracks GPS. GPS (or PGPS) works by assigning a session a weight ϕ at each link on the session's route. The weight of a session at a link on its path decides its share of the link bandwidth at that link which in turn decides the end-to-end delay. Most of the work on GPS has addressed the problem of obtaining end-to-end delay bound for prespecified session weights.

The first problem that we investigate in this paper deals with the practically important inverse procedure of determining appropriate weights for sessions at each link on their paths, that satisfy prespecified delay bounds, when the set of sessions to be routed is given. All the sessions are assumed to be leaky bucket constrained. When we assume that a session route is prespecified, it becomes the case of fixed routing. We show that the fixed routing case can be formulated as a linear program (LP). However, prespecifying a route limits the admissibility of

This work was carried out at the Indian Institute of Science and supported by a research grant from Nortel Networks.

0-7803-7097-1/01/$10.00 ©2001 IEEE

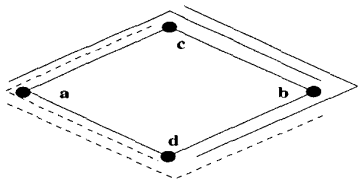

Fig. 1. Hypothetical example to illustrate advantage of adaptive routing. All links are 15 Mbps and all sessions require 5 Mbps.

a session because enough resources may not be available on that route. At the same time, another route for the same session may exist with enough resources, which is not considered in the fixed routing framework. This in turn limits the network schedulable region which calls for determining a session route instead of prespecifying one. This can considerably increase the network schedulable region. We present a motivating example below.

Fig 1 shows a simple network consisting of 4 nodes and 4 links. Assume each link has a bandwidth of 15 Mbps. Assume (a, b) and (c, d) are the only source-destination pairs (SD-pairs) on which sessions can be present. Assume all sessions require a bandwidth of 5 Mbps. Now consider the fixed routing case. Suppose the route for (a, b) is $a \rightarrow c \rightarrow b$ and that for (c, d) is $c \rightarrow b \rightarrow d$. It can be easily seen that the maximum number of sessions that the network can support simultaneously are $(3, 0), (2, 1), (1, 2)$ or $(0, 3)$, where the first number indicates the maximum on (a, b) and the second number indicates the maximum on (c, d). Now consider the adaptive routing case. There are two routes for each SD-pair now. The routes for (a, b) are $a \rightarrow c \rightarrow b$ and $a \rightarrow d \rightarrow b$ and that for (c, d) are $c \rightarrow b \rightarrow d$ and $c \rightarrow a \rightarrow d$. In this case the the maximum number of sessions supported by the network are $(6, 0), (4, 2), (2, 4)$ or $(0, 6)$. Thus the schedulable region of fixed outing case is a subset of the schedulable region of adaptive routing.

When it is required to determine a session's route as well as the weights on that route, it becomes the case of adaptive routing. We show that the adaptive routing case can be formulated as a mixed integer linear program (MILP). The advantage of both the LP and MILP is that their computational complexity is independent of the number of sessions in the networks.

The second problem in this paper investigates the performance of PGPS scheduling policy when providing per-session QoS guarantees. Here, session requests and terminations arrive at random. We measure the performance in terms of weighted carried traffic. We derive an upper bound on the weighted carried traffic for *any* heuristic algorithm for admission control that operates within the locally stable domain. This upper bound can be obtained by computing a simple linear program (LP). By simulating a simple heuristic algorithm for admission control, we show that this upper bound is reasonably tight. Thus our upper bound can be used as a metric against which the performance of different algorithms can be compared.

II. GPS Scheduling Policy

Consider N sessions contending for a link l. The GPS scheduling policy assigns N positive real numbers (weights) , $\phi_1^l, \phi_2^l, \cdots, \phi_N^l$ for the N sessions. This is called a GPS assignment for sessions at the server of link l. The weights signify the relative amount of service that each session gets, i.e., if $S_i^l(\tau, t)$ is defined as the amount of session i traffic served by the GPS server at link l during an interval $[\tau, t]$, then

$$\frac{S_i^l(\tau, t)}{S_j^l(\tau, t)} \geq \frac{\phi_i^l}{\phi_j^l} \tag{1}$$

for any session i that is continuously backlogged in the interval $[\tau, t]$. A session is backlogged at time t if a positive amount of that session's traffic is queued at time t. Note from (1) that whenever session i is backlogged it is guaranteed a minimum service rate of

$$g_i^l = \frac{\phi_i^l}{\sum_{j=1}^{N} \phi_j^l} r^l \tag{2}$$

where r^l is the rate of the link. This rate is called the session i backlog clearing rate since a session i backlog of size q is served in at most $\frac{q}{g_i^l}$ time units. Note that when all ϕ_i are equal, every backlogged queue is served at the same rate and GPS becomes a simple Processor Scheduling (PS).

III. PGPS End-to-End Delay Bound

Consider a set of sessions that are to be routed through a network. All sessions are leaky bucket constrained with parameters (σ_i, ρ_i) for session i. A session i is leaky bucket constrained with parameters (σ_i, ρ_i) if the amount of traffic that session i injects in the network, $A_i(\tau, t)$, during any interval$(\tau, t]$, is bounded as $A_i(\tau, t) \leq \sigma_i + \rho_i(t - \tau), \forall t \geq \tau \geq 0$. The network is stable if the utilization of all links is less than 1, where the utilization on link l, u^l is defined as $u^l = \frac{\sum_{j \in I(l)} \rho_j}{r^l}$, where $I(l)$ is the set of sessions that are going through link l ([2], [3]). Now consider a GPS assignment for each session at all links on its path. We focus our attention on a particular session i. This session traverses the network through a set of K links denoted as $P(i)$. Let the propagation delay along the path of session i be d_i^{prop}. At each link $l \in P(i)$, the GPS server of that link guarantees a minimum rate g_i^l at which session i traffic is served. Note that with the GPS assignment being known, the use of eqn (2) gives us g_i^l. Hence the minimum session i backlog clearing rate along its route is $g_i = \min_{l \in P(i)} g_i^l$ To take packetization into account, we consider the packetized version of GPS namely Packet GPS (PGPS) in this paper. PGPS is a non-preemptive policy that tracks GPS. Let L_i be the size of maximum length packet from session i and L be the size of maximum length packet among all sessions. A session is defined to be locally stable if $g_i > \rho_i$. Locally stable sessions have a simple and closed form bound on end-to-end delay as stated in the following theorem [3].

Theorem III.1: If $g_i \geq \rho_i$ for session i, then

$$D_i^* \leq \frac{\sigma_i + KL_i}{g_i} + \sum_{l \in P(i)} \frac{L}{r^l} + d_i^{\text{prop}}$$

where D_i^* is the maximum end-to-end delay that a session i packet may suffer while traversing the network. In other words, the GPS weight assignment provides an end-to-end delay guarantee of D_i^* for session i. Thus, given a predetermined end-to-end delay bound D^{reqd}, we can use the above expression to compute the minimum rate required to obey the delay bound for session i. For example, for a voice session having $\sigma = 100$ Bytes, $\rho = 64$ Kbps, $D^{\text{reqd}} = 50$ ms, $L_i = 100$ Bytes and $L = 1.5$ KBytes, going over a 3-link route, g_i turns out to be 85 Kbps. In this paper, the network operates under the locally stable regime.

IV. PGPS Weight Assignment Problem

A. An LP Formulation for Fixed Routing

A session i is characterized by a 4-tuple $(\sigma_i, \rho_i, D_i, L_i)$ where σ_i and ρ_i are the leaky bucket parameters of session i, D_i is the maximum end-to-end delay requirement of session i and L_i is the maximum length of a packet of session i. Sessions are grouped into traffic classes (e.g., table I). Sessions belonging to a traffic class have identical 4-tuples. Denote by C, M, L the number of traffic classes, the number of available paths and the number of links in the network respectively. A fixed routing problem is one in which the path that a session is going to use is prespecified. There is no choice as to which path should be chosen for a session. Our aim is to check the schedulability of a given set of sessions and to obtain a GPS assignment if the set of sessions is schedulable. We now formulate this problem as follows. b_{ij} is an indicator where $b_{ij} = 1$ if path i uses link j and $b_{ij} = 0$ otherwise. n_{ij} is the number of class i sessions that use path j. Without loss of generality, on a link, we assign the same ϕ to the sessions belonging to a class that use the same path. Hence we can couple the ϕ assignments for the sessions of a class i on link l and having the same path into a single ϕ. Let ϕ_{ijl} be this coupled assignment. ϕ_{ijl} is zero if class i session doesn't use path j or if link l doesn't belong to path j. Our linear formulation computes a ϕ_{ijl} assignment. We make use of the delay guarantee bound of theorem III.1 to compute the rate g_{ij} needed for a class i session on path j to achieve its delay bound. So the GPS assignment will be such that all sessions are locally stable. Denote by r^l the capacity of link l. The link capacities r^l and the b_{ij} and n_{ij} are given. The ϕ_{ijl} is to be computed in such a way that all sessions get routed while the utilization on each link remains less than 1 and the end-to-end delay requirements are obeyed. So we have the following linear formulation.

Find $\Phi = (\phi_{ijl})$ such that:

$$\phi_{ijl} r^l \geq n_{ij} b_{jl} \max(\rho_i, g_{ij}) \quad \forall i \in C, j \in M, l \in L \tag{3}$$

$$\sum_{i=1}^{C} \sum_{j=1}^{M} \phi_{ijl} = 1 \quad \forall l \in L \tag{4}$$

$$\phi_{ijl} \leq n_{ij} b_{jl} \quad \forall i \in C, j \in M, l \in L \tag{5}$$

(3) allocates ϕ so that the end-to-end delay requirements are obeyed. (4) normalizes the ϕ assignments at all links and with (3), it keeps link utilization below 1. Note that there is no loss of generality in this normalization. (5) forces ϕ_{ijl} to 0 at appropriate links. Any Φ satisfying the above constraints gives us a GPS assignment that achieves the end-to-end delay requirements of the sessions. Note that this formulation is linear in ϕ and a Φ assignment can be computed by solving a linear program (LP) with above set of constraints. Also the computational complexity of the LP is independent of the number of sessions present in the network.

TABLE I

Parameters for traffic models of type voice and video.

Class	σ(kB)	ρ(Mbps)	D(ms)	L(kB)
1: voice	0.1	0.064	50	0.1
2: video conf	10	0.5	75	1.5
3: st video	100	3	100	1.5

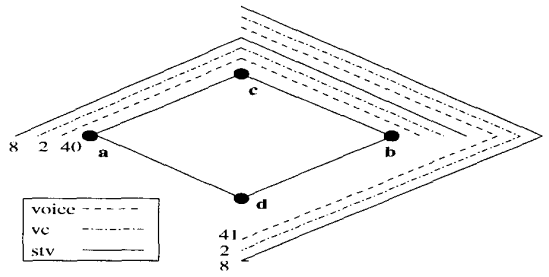

Fig. 2. Example to illustrate ϕ assignment in fixed routing case. Links are 155 Mbps and link delays are 4 ms. Sessions belong to the classes of table I. (a, b) and (c, d) are the only SD-pairs on which sessions can be present and there is only 1 route per SD-pair which is prespecified. We allow 40 voice sessions, 2 vc sessions and 8 stv sessions on SD-pair (a, b); and 41 voice sessions, 2 vc sessions and 8 stv sessions on SD-pair (c, d). With this, no more sessions can be admitted.

We illustrate the usefulness of the LP formulation for a simple network of fig 2. All links have bandwidth = 155 Mbps and link propagation delay = 4 ms. Assume (a, b) and (c, d) are the only SD-pairs on which sessions can be present. The sessions belong to the classes of table I. Session routes are fixed and suppose sessions on (a, b) follow the route $a \rightarrow c \rightarrow b$ and sessions on (c, d) is $c \rightarrow b \rightarrow d$. All routes span over two hops. The rate required by a voice session is 64 Kbps as given by the maximum of its ρ and rate computed from eqn III.1. Similarly, a video conferencing (vc) session requires a rate of 1.558 Mbps and a stored video (stv) session requires a rate of 8.9716 Mbps. We consider the following set of sessions: On SD-pair (a, b), 40 voice sessions, 2 vc sessions and 8 stv sessions; on SD-pair (c, d), 41 voice sessions, 2 vc sessions and 8 stv sessions, that is these are the $n_i j$ for our example, see fig 2. By solving the LP, it can be seen that the above n_{ij} set is a feasible set and the respective ϕ weights given by the LP are shown in table II. We also note that no more sessions can be admitted on any SD-pair; if admitted, total session rate on any link exceeds the link rate. In the next section, we show how adaptive routing can be used to increase the schedulable region.

TABLE II

ϕ ASSIGNMENT FOR THE FIXED ROUTING EXAMPLE.

session type	ϕ on link		
	(a,c)	(c,b)	(b,d)
voice	0.0165	0.0334	0.0169
video conf	0.0201	0.0402	0.0201
st video	0.9634	0.9264	0.9630

TABLE III

ϕ ASSIGNMENT FOR THE ADAPTIVE ROUTING EXAMPLE.

session type	ϕ on link			
	(a,c)	(c,b)	(b,d)	(d,a)
voice	0.0206	0.0334	0.0211	0.0083
video conf	0.1105	0.0402	0.1104	0.1806
st video	0.8689	0.9264	0.8685	0.8111

B. An MILP Formulation for Adaptive Routing

In the adaptive routing case, it is not known which session uses which path. But the source and destination of a session are known. Also the paths that are available for a SD-pair are known. The adaptive routing problem is to select paths for the sessions such that the end-to-end delay requirements are obeyed. Hence the adaptive routing formulation is based on a class-SD-pair formulation. We define s_{ij} and a_{ij} as follows. s_{ij} is the number of class i sessions whose SD-pair is j. $a_{ij} = 1$ if path i is for SD-pair j. $g_{ij}, b_{ij}, n_{ij}, \Phi, r^l$ are defined as in section IV-A. Note that, here Φ as well as N are to be computed in such a way that all sessions get routed while the utilization on each link remains less than 1 and the end-to-end delay requirements are obeyed. N gives the routing information and Φ gives the GPS assignment. So we have the following linear formulation.

Find N and Φ such that:

$$s_{ij} = \sum_{k=1}^{M} n_{ik} a_{kj} \quad \forall i \in C, j \in P \tag{6}$$

$$\phi_{ijl} \geq n_{ij} b_{jl} \max(\rho_i, g_{ij}) \quad \forall i \in C, j \in M, l \in L \tag{7}$$

$$\sum_{i=1}^{C} \sum_{j=1}^{M} \phi_{ijl} = 1 \quad \forall l \in L \tag{8}$$

$$\phi_{ijl} \leq n_{ij} b_{jl} \quad \forall i \in C, j \in M, l \in L \tag{9}$$

Note that in the above formulation ϕ are real nonnegative variables and n_{ij} are integer variables. (6) is the only additional constraint which decides the routing of sessions. The above formulation is linear in ϕ and n_{ij}. Since n_{ij} is an integer variable, Φ and N assignments can be computed by a mixed integer linear program (MILP) with above set of constraints. Note that the computational complexity of the MILP is independent of the number of sessions present in the network.

We continue with the example of last section to illustrate the advantage of adaptive routing. We allow one more route per SD-pair as follows. Sessions on (a,b) can also use $a \rightarrow d \rightarrow b$ and sessions on (c,d) can also use $c \rightarrow a \rightarrow d$. We add 10 voice sessions, 9 vc sessions and 7 stv sessions on the new route for SD-pair (a,b). Similarly we add 10 voice sessions, 9 vc sessions and 7 stv sessions on the new route for SD-pair (c,d) as in fig 3. So on SD-pair (a,b) total voice sessions are 50, total vc sessions are 11 and total stv sessions are 15. On SD-pair (c,d) total voice sessions are 51, total vc sessions are 11 and total stv sessions are 15. This gives us the s_{ij} and n_{ij} sets. By solving the MILP, it can be seen that this is a feasible set. The MILP also gives the ϕ weights which are shown in table III. Also, we can add no more sessions of any class on any SD-pair. Hence this set is an extreme point. Thus with adaptive routing we were able to admit more sessions.

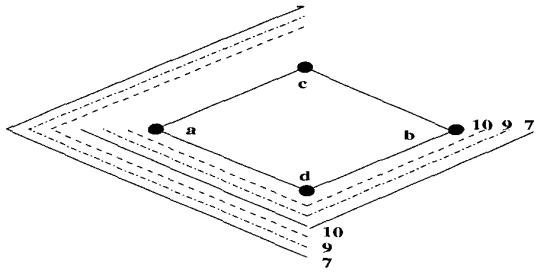

Fig. 3. Fixed routing example continued to illustrate routing & ϕ assignment in adaptive routing case. Here 2 routes per SD-pair are allowed. Because of this, we can 10 voice sessions, 9 vc sessions and 7 stv sessions on the new route for SD-pair (a,b) and 10 voice sessions, 9 vc sessions and 7 stv sessions on the new route for SD-pair (c,d). This makes total 50 voice sessions, 11 vc sessions and 15 stv sessions on SD-pair (a,b) and total 51 voice sessions, 11 vc sessions and 15 stv sessions on SD-pair (c,d).

Note that in both routing cases, $\max(\rho_i, g_{ij})$ rate is allocated to an admitted session. When $g_{ij} > \rho_i$, additional bandwidth $= g_{ij} - \rho_i$ is allocated for a class i session. This additional bandwidth remains unused and because of such sessions, network utilization may remain low. We call such sessions delay constrained sessions. Thus a voice session of table I becomes delay constrained when it has to go over more than 2 hops. Both video sessions of table I are delay constrained for any number of hops. Let us consider 3 hop case. The rate required for voice is 0.085 Mbps, for video conferencing 1.85 Mbps and for stored video it is 9.53 Mbps. So 0.021 Mbps per voice session, 1.35 Mbps per video conf. session and 6.53 Mbps per stored video session bandwidth remains unused. We note that this is the price one has to pay for achieving delay guarantees under PGPS scheduling.

V. PERFORMANCE OF PGPS SCHEDULING

A. The Upper Bound

In this section we investigate the performance of PGPS policy under the locally stable regime. We assume the sessions arrive and depart at random. The goal is to maximize the weighted carried traffic, for a given offered load (in Erlangs) of respective classes on each SD-pair. The weight reflects in some sense the amount of resources (bandwidth) required for that particular traffic class. The maximization problem is formulated as an Integer Linear Program (ILP). However, solving an ILP is computationally expensive. So we relax the integer

constraints of the ILP to get a simple LP. The LP solution is an upper bound on the ILP solution. We then go on to show that this is a tight upper bound for *any* heuristic algorithm operating in the locally stable domain.

Note that in the case of PGPS routing & weight assignment problem, we were able to formulate fixed routing case as an LP even though it was a special case of adaptive routing case. In this section we only consider the adaptive routing case because there is no gain in formulating the two cases separately. Fixed routing case is treated as a special case of adaptive routing.

Before proceeding further we define a few terms here. $C, M, P, L, (a_{ij}), (b_{ij})$ are defined as before. $\rho_i, \sigma_i, D_i, L_i$ are the traffic descriptors and g_{ij} is the rate required to obey end-to-end delay bound for class i session on path j as before. Let λ denote the total offered load in Erlangs. The (p_{ij}) matrix gives the distribution of offered load among different classes for the SD-pairs. So $p_{ij}\lambda$ is the offered load of class i traffic on SD-pair j. The offered load for the static case (a fixed set of sessions) is the number of sessions that are available to be routed. In the dynamic case, it is the expected number number of sessions that would be in progress if one could successfully route all session arrivals. Let s_{ij} denote the number of class i sessions carried between SD-pair j. Also n_{ij} denotes the number of class i sessions carried on path j. In the dynamic case, s_{ij} and n_{ij} are the expectations of the respective quantities.

We measure the performance of PGPS policy under locally stable regime in terms of weighted carried traffic. The weight reflects in some manner the amount of resources (bandwidth) required for that particular traffic class. We fix weight for traffic class i equal to ρ_i. We want to find the optimal algorithm for rate allocation so that the end-to-end delay bounds of admitted sessions are obeyed and the weighted carried traffic is maximized. In the practically important dynamic case, where sessions arrive at random, hold bandwidth along their paths for random durations and then depart, we are interested in maximizing the weighted expected carried traffic. Then, the optimal algorithm for the bandwidth allocation problem is found by by solving the following ILP (integer linear program) whose value we denote by $I(\lambda, p)$.
(Maximize weighted carried traffic)

$$I(\lambda, p) = \max \sum_{i=1}^{C} \rho_i \sum_{j=1}^{P} s_{ij}$$

subject to

$$s_{ij} = \sum_{k=1}^{M} n_{ik} a_{kj} \quad \forall i \in C, \forall j \in P \quad (10)$$

$$s_{ij} \leq p_{ij}\lambda \quad \forall i \in C, \forall j \in P \quad (11)$$

$$\sum_{i=1}^{C} \sum_{j=1}^{M} g_{ij} n_{ij} b_{jl} < r^l \quad \forall l \in L \quad (12)$$

where $s_{ij}, n_{ij} \geq 0$ and are integer variables.

Note that *any* heuristic algorithm for allocating rates under the locally stable PGPS regime has to satisfy the constraints of the above ILP. So the solution of above ILP produces an upper bound on the weighted carried traffic of *any* heuristic algorithm that operates within the locally stable domain. But solving ILP may be computationally expensive and hence we are interested in an easily computable upper bound. If we relax the integer constraints of above ILP, we get an upper bound on the solution of ILP and hence an upper bound on the performance of any heuristic algorithm for rate allocation that operates within the locally stable domain. Let us denote this upper bound by $L(\lambda, p)$. Though computationally inexpensive, the LP bound is really useful if it is reasonably tight. In the next section, we consider a heuristic algorithm for bandwidth allocation and we show by simulations that the LP upper bound is indeed reasonably tight. This LP upper bound can be used as a metric against which the performance of different algorithms can be compared.

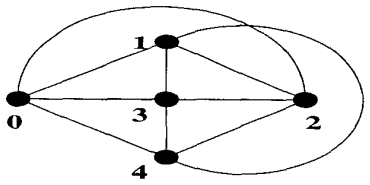

Fig. 4. Network to illustrate the performance of PGPS rate allocation. It has 5 nodes and 8 links. Links are 155 Mbps and link delays are 4 ms.

B. Comparison with a Heuristic Rate Allocation Algorithm

Here we show the utility of the LP upper bound that is derived in the last section. The network is shown in fig 4 and the traffic descriptors are as in table I. We consider the following shortest-path heuristic bandwidth allocation algorithm. The set of shortest paths between an SD-pair is ordered in some manner. The required rate to provide end-to-end delay guarantee for the traffic of an arriving session is computed for the first path on the list. If there is enough rate available along the path, the session is admitted on that path. Otherwise, we check for the second path of the list and so on. If no path can be found, the session is considered blocked. We simulated the performance of this simple algorithm for traffic where session requests are assumed to arrive according to a Poisson process and last for a duration that is exponentially distributed. We consider only the case of uniform traffic, i.e., $p_{ij} = 1/(CP)$ here although the upper bound is applicable to *any* traffic pattern.

Fig 5–6 show the weighted carried traffic as a function of offered traffic for above heuristic algorithm and their upper bound. For example, in Fig 5 we compare the performance of above heuristic where at most 2 shortest paths are allowed between any SD-pair. It can be seen that the performance of the heuristic is close to that achievable by *any* algorithm for this network up to reasonably high offered load. It is possible to conclude thus because our upper bound is reasonably tight here. Moreover, we have observed that the offered load where the upper bound is tight has reasonably low blocking probability. We would indeed like to operate in low blocking probability region. The part where the upper bound is not so tight is not of much interest and hence we can safely conclude that over region of interest, our upper bound is reasonably tight. Now consider fig 7 which shows the performance bound of various

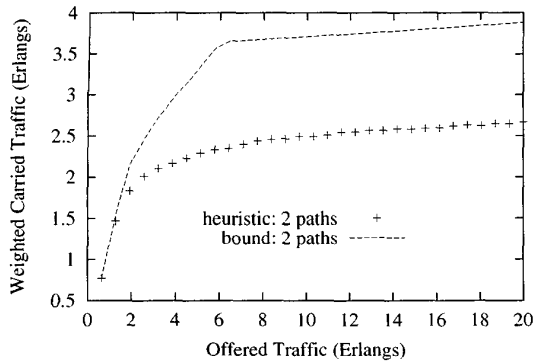

Fig. 5. Weighted carried traffic of heuristic and its bound computed by the LP are plotted. At most 2 paths between any SD-pair are considered. Heuristic performs well for low offered traffic.

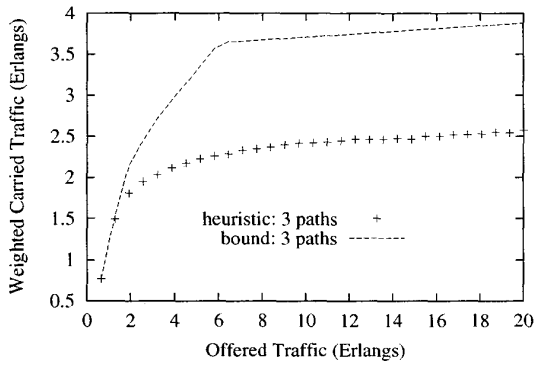

Fig. 6. Weighted carried traffic of heuristic and its bound computed by the LP are plotted. At most 3 paths between any SD-pair are considered. Heuristic performs well for low offered traffic.

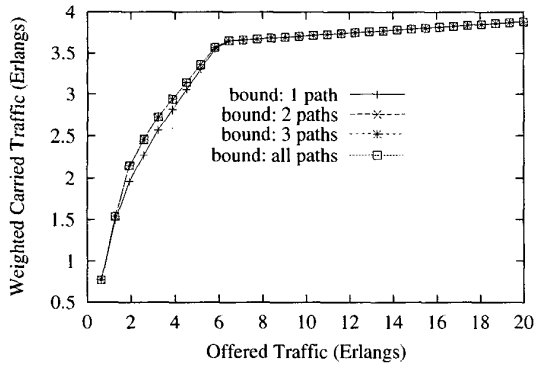

Fig. 7. Bounds on the weighted carried traffic of *any* heuristic algorithms are plotted. Observe that the gain in considering more paths is not much.

optimal algorithms. It can be seen that we don't gain much by considering complicated algorithms which try to use many paths between an SD-pair. In our example, the performance, when at most 2 paths between any SD-pair are allowed, is almost the same as that obtained by considering all paths between an SD-pair.

VI. Conclusion

We have addressed the problem of routing sessions with delay and rate requirements in a network under PGPS scheduling. In the first problem that we have considered, we try to find a GPS weight assignment (ϕ assignment) for a given set of sessions so that their delay requirements are obeyed. Fixed routing case and adaptive routing case are dealt separately. It is shown that the fixed routing ϕ assignment can be obtained by computing an LP. In the case of adaptive routing, it is shown that the routing information and ϕ assignment can be obtained by computing an MILP. The computational complexity of both the LP and MILP is independent of the number of sessions present in the network. It is observed that for delay constrained sessions, PGPS allocates more additional bandwidth to achieve delay guarantees. The links remain underutilized because of this unused extra bandwidth.

The second problem that is addressed here is the performance of PGPS scheduling policy. We obtain an upper bound on the performance of *any* heuristic bandwidth allocation algorithm that operates within the locally stable domain. The bound can be obtained by solving an LP. The utility of our upper bound is shown by comparing it with the performance of a simple shortest path heuristic algorithm. The performance of this simple shortest path heuristic is close to that achievable by *any* algorithm. We can conclude thus because our upper bound is reasonably tight here. Hence our upper bound can be used as a metric against which the performance of different algorithms can be compared. Moreover, it can be seen from the upper bound curves that the gain in weighted carried traffic is not much for more number of paths between an SD-pair.

References

[1] A. Demers, S. Keshav and S. Shenker, "Analysis and Simulation of a Fair Queueing Algorithm," *Proc. ACM SIGCOMM'89*, pp. 3–12, 1989.

[2] A. K. Parekh and R. G. Gallagher,"A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single Node Case," *IEEE Trans. on Networking*, Vol. 1, No. 3, pp. 137–150, 1993.

[3] A. K. Parekh and R. G. Gallagher,"A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Multiple Node Case," *IEEE Trans. on Networking*, Vol. 2, No. 2, pp. 347–357, 1993.

[4] R. L. Cruz, "A Calculus for Network Delay, Part I: Network Elements in Isolation," *IEEE Trans. Inform. Theory*, Vol. 37, pp. 114–131, 1991.

[5] O. Yaron and M. Sidi, "Generalized Processor Sharing Networks with Exponentially Bounded Burstiness Arrivals," *Proc. IEEE INFOCOM*, June 1994.

[6] Z. L. Zhang, D. Towsley and J. Kurose, "Statistical Analysis of the Generalized Processor Sharing Scheduling Discipline," *IEEE JSAC*, Vol. 13, No. 6, pp. 1071–1080, Aug. 1995.

[7] S. J. Golestani, "A Self-Clocked Fair Queueing Scheme for Broadband Applications," *Proc. IEEE INFOCOM*, pp. 636–646, 1994.

[8] Z. L. Zhang, Z. Liu and D. Towsley, "Closed-Form Deterministic End-to-End Performance Bounds for the Generalized Processor Sharing Scheduling Discipline," *Journal of Combinatorial Optimization*. Special Issue on Scheduling.