# Integrating Best-Effort and Guaranteed Sessions through A Two-Level Generalized Processor Sharing Approach

by

Natanael Peranginangin

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of
Master of Science in Computer Science and Engineering
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
August 2000



Author .....
Department of Electrical Engineering and Computer Science
August 31, 2000

Certified by .....
Robert G. Gallager
Professor of Electrical Engineering
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Departmental Committee on Graduate Students

# Integrating Best-Effort and Guaranteed Sessions through A Two-Level Generalized Processor Sharing Approach

by

Natanael Peranginangin

Submitted to the Department of Electrical Engineering and Computer Science
on August 31, 2000, in partial fulfillment of the
requirements for the degree of
Master of Science in Computer Science and Engineering

## Abstract

This thesis investigates the necessary refinements for the seminal Packet-by-Packet Generalized Processor Sharing (PGPS) mechanism combined with Leaky Bucket admission control [1] with the objective of maximizing bandwidth available to the best-effort traffic without adversely affecting the guarantee of the QoS traffic.

First, a multiplexing scheme, i.e, Two-Level Generalized Processor Sharing (2L-GPS), is proposed. The scheme extends the Generalized Processor Sharing (GPS) multiplexing scheme [1] from its original one-level multiplexing architecture to a two-level architecture. Subsequently, we show that it is possible to compute performance guarantees for sources constrained by leaky buckets in a 2L-GPS server when a certain condition is satisfied.

We then propose Two-Level Packet-by-Packet Generalized Processor Sharing (2L-PGPS) which is a packet-by-packet based algorithm that approximates 2L-GPS. We analyze the lateness incurred by each session when 2L-PGPS is used to approximate the 2L-GPS scheme. The results allow us to translate session delay and buffer requirement bounds derived for a 2L-GPS server system to a 2L-PGPS server system.

Finally, we propose a grouping policy which integrates the provision of both QoS and best-effort service classes within the 2L-GPS multiplexing scheme. We show that the grouping policy improves performance (i.e., worst-case queuing delay and backlog) of the best-effort sessions. We further conjecture that aiming at such improvement in performance does not adversely affect guarantees for the QoS sessions. Finally, we provide minor modifications to the analysis of the multiple-node case in [2] so that the worst-case end-to-end session delay in

# Acknowledgements

I would like to express my deep gratitude to my advisor, Bob Gallager. Learning to do research from him is an opportunity for which I will always be thankful. His remarkable insight contributed substantially to the results of this thesis. His high standards of scholarship and intellectual integrity inspired me to challenge myself much more than I would have otherwise.

Furthermore, I would like to express my deep gratitude to PT. TELEKOMUNIKASI INDONESIA for the financial support given throughout my Master's program. In a painfully difficult financial situation at every sector of the country, PT TELKOM persisted in supporting my education.

In addition, I would like to thank Poompat Saengudomlert, Stephane Bratu and Thit Minn for putting up with my vague rambling and giving excellent suggestions despite my not so clear ideas. The time we spent at pingpong, pool, and dining table had made my days much lighter.

Finally, I would like to acknowledge my parents, whose faith has enabled me to live my dreams. This thesis is dedicated to them.

the 2L-PGPS network, where the grouping policy is applied, can be computed by applying the analysis in [3, ch. 4] in a straight-forward manner.

Thesis Supervisor: Robert G. Gallager
Title: Professor of Electrical Engineering

# Contents


1 Introduction 8

1.1 Motivation and Background ..... 8

1.1 Outline ..... 9

2 Two-Level Generalized Processor Sharing 12

2.1 2L-GPS Multiplexing ..... 13

2.2 Leaky Bucket ..... 19

2.2.1 Second-Level Leaky Bucket ..... 19

2.2.2 First-Level Leaky Bucket ..... 21

2.3 Analysis ..... 22

2.3.1 First-Level Sessions ..... 22

2.3.2 Second-Level Sessions ..... 23

2.3.2.1 Definitions and Preliminary Results ..... 25

2.3.2.2 Greedy Sessions ..... 28

2.3.2.3 2L-GPS with Infinite Incoming Link Capacities ..... 30

2.3.2.3.1 An All Greedy 2L-GPS system ..... 30

2.3.2.4 An Important Equality ..... 33

2.3.2.5 Proof of the Main Result ..... 34

2.3.2.6 The Output Burstiness $\sigma_{ij}^{out}$ ..... 37

3 Packet-by-Packet Transmission Scheme and Its Lateness 39

3.1 Two-Level Packet-by-Packet Generalized Processor Sharing ..... 40

3.1.1 Virtual Time Implementation of 2L-PGPS ..... 40

3.1.2 Algorithm ..... 43

3.2 Lateness Analysis of 2L-PGPS ..... 48

4 Integrating Best-Effort and QoS Classes of Sessions 53

4.1 A Grouping Policy ..... 54

4.2 Comparing Sessions' Performances : Grouping vs Non-Grouping ..... 56

4.3 A Conjecture ..... 64

4.4 Packet-by-Packet Two Level Generalized Processor Sharing Network under The Grouping Policy ..... 65

4.4.1 Packet Lateness for 2L-PGPS under The Grouping Policy ..... 66

4.4.2 A note on Worst-Case End-to-End Session Delay in 2L-PGPS Network under Grouping Policy ..... 67

5 Summary of the Contributions 70

# List of Figures

2-1 An example of 2L-GPS vs GPS ..... 16

2-2 A Grouping Strategy ..... 17

2-3 A Leaky Bucket ..... 20

2-4 $A_{ij}(0,t)$ and $l_{ij}(t)$ ..... 21

2-5 $A_{ij}(0,t)$, $S_{ij}(0,t)$, $D_{ij}(t)$ and $Q_{ij}(t)$ ..... 24

2-6 Graphical representation of $\sigma_{ij}^{\tau}$ ..... 26

2-7 A session $i_j$ arrival function that is greedy from time $\tau$ ..... 29

2-8 Session $i_2$ arrivals and departures after 0, the beginning of a system busy period in which all sessions are greedy ..... 31

3-1 The tree-like structure of a 2L-GPS server ..... 40

4-1 Ungrouping (GPS) vs Grouping (2L-GPS) ..... 54

# Chapter 1

# Introduction

## 1.1 Motivation and Background

In the last decade there has been a vast amount of work on providing service guarantees in integrated service networks. The need to support a large variety of applications with diverse quality of service (QOS) requirements such as voice, video-on-demand, etc., along with the development of fiber-optics technologies, have fueled the need for a flow control scheme that has the following three properties, i.e., efficiency, flexibility and analyzability [6]. In terms of an efficient flow control, we give an example of a packet switched network. In a packet switched network, a data session which typically has short bursts of high activity, followed by lengthy inactive periods, can let other sessions utilize the bandwidth unused during its inactive periods. By flexibility, we mean that the flow control scheme allows the network to treat users differently, in accordance with their desired quality of service, without compromising the fairness of the scheme, i.e., a few classes of users should not be able to degrade services to other classes, to the extent that performance guarantees are violated. Lastly, the flow control scheme needs to be analyzable to the extent that performance guarantees can be made in the first place. In an environment where short-term demand for link usage frequently exceeds the

usable capacity, each of three properties of a flow control scheme above can apparently be in conflict with the others. An approach to reconcile the conflict is proposed in [1] and its sequel [2]. This approach employs PGPS (Packet-by-Packet Generalized Processor Sharing) in the context of integrated service networks and combines this mechanism with Leaky Bucket admission control in order to provide performance guarantees in a flexible environment.

An integrated service network should allow the provision of both best-effort and QoS sessions. While not always in the forefront, as compared to the QoS traffic, the role of best-effort traffic is important in our conceptual framework for flow control. Another high-level property which needs to be integrated in the flow control scheme is therefore the capability to maximize the bandwidth available to best-effort traffic while satisfying the guarantees of the QoS class. We thus devote this thesis to investigating the necessary refinements for the seminal PGPS scheme combined with Leaky Bucket admission control in order to maximize the bandwidth available to best-effort traffic without adversely affecting the guarantees of the QoS class of traffic.

In this thesis we choose to focus on analytical rather than implementation oriented issues, which does not mean that we do not consider the latter to be important. There is currently a great pressure to choose appropriate packet schedulers for IP routers and ATM switches of the future. We hope that this thesis, and the ideas that shape it, will be of some help to those who have taken the exciting challenge of implementing these devices.

## 1.2 Outline

The remainder of this thesis can be outlined as follows. Chapter 2 extends the GPS multiplexing scheme introduced in [1] from its original one-level multiplexing architecture to a two-level architecture. The motivation, as will be apparent in Chapter 4, is to allow a greater

multiplexing flexibility so that a grouping policy can be incorporated on top of the multiplexing framework. Zhang and Bennett [4] were the first to propose the Hierarchical Generalized Processor Sharing (H-GPS) model to simultaneously support guaranteed real-time, rate-adaptive best-effort, and controlled link-sharing services. However, [4] does not provide a thorough analysis in finding worst-case queueing delay and backlog for a leaky bucket constrained session in a single H-GPS node. Following the definitions and properties of the Two-Level GPS work discipline (2L-GPS), we therefore analyze a single 2L-GPS server system in which the sessions are constrained by leaky buckets. The most important result from our analysis is that it is possible to make worst-case queuing delay and backlog guarantees for sources constrained by leaky buckets in a 2L-GPS environment, if a certain condition is satisfied.

In Chapter 3 we present a packet-by-packet based scheme that is an approximation to 2L-GPS even when the packets are of variable length. We call this packet based scheme 2L-PGPS, which stands for Two-Level Packet-by-Packet Generalized Processor Sharing. The packet-by-packet scheme adopts the H-PFQ (Hierarchical Packet Fair Queuing) framework which was first developed in [4] in order to provide a translation from the idealized fluid multilevel GPS system into a packet-by-packet system. Subsequent to describing the packet-by-packet approximation scheme, we apply the techniques developed in [1] for analyzing the lateness which is being incurred by each session when two-level HPFQ is used to approximate the 2L-GPS scheme. Results obtained in this chapter allow us to translate session delay and buffer requirement bounds derived for a 2L-GPS server system to a 2L-PGPS server system.

Chapter 4 centers around a policy that integrates the provision of both QoS and best-effort service classes within the GPS based multiplexing scheme. This policy maximizes the bandwidth available to best-effort traffic while just satisfying the guarantees of the QoS class. We shall utilize the 2L-GPS work discipline as a platform, on top of which a policy to

improve performance for the best-effort service class is proposed. We compare the performance of sessions when such a policy is implemented with the case when it is not implemented, and show that the policy improves hard performances (i.e, worst-case queuing delays and backlogs) of best-effort sessions. We further conjecture that aiming at such performance improvement does not adversely affect guarantees for the QoS class of sessions. Finally, by using the results of Chapter 3, we provide minor modifications to the analysis of the multiple node case of GPS in [2] so the end-to-end session delay in the 2L-PGPS (Two-Level Packet-by-Packet Generalized Processor Sharing) network can be computed by applying the analysis in [3, ch. 4] in a straight-forward fashion.

# Chapter 2

# Two-Level Generalized Processor Sharing

In this chapter we extend the GPS multiplexing scheme introduced in [1] from its original one-level multiplexing architecture to a two-level architecture. We call the scheme Two-Level Generalized Processor Sharing (2L-GPS). The motivation, as will be apparent in Chapter 5, is to allow a greater multiplexing flexibility so that a grouping policy can be incorporated on top of the multiplexing framework. In an integrated environment, where a high speed server treats both best effort and guaranteed class of sessions, such a grouping policy aims at improving the performance of best-effort sessions without hurting the QoS (Quality of Service) for the guaranteed sessions.

A two-level GPS multiplexing is a two-level realization of Hierarchical Generalized Processor Sharing (H-GPS) multiplexing. Zhang and Bennett [4] were the first to propose the Hierarchical Generalized Processor Sharing (H-GPS) model to simultaneously support guaranteed real-time, rate-adaptive best-effort, and controlled link-sharing services. However, [4] does not provide a rigorous analysis in finding worst-case queuing delay and backlog for a leaky bucket constrained session in a single H-GPS node.

In the the first section, we develop the definitions and properties which are inherent to a 2L-GPS server. In the following section, we extend the Leaky Bucket regulator developed in

[1] so as to accomodate traffic regulation to sources in a 2L-GPS environment. The subsequent section is devoted to the analysis of performances in the worst case for sessions that operate under leaky bucket constraints within a single server 2L-GPS. The most important result from our analysis is that it is possible to make worst-case queuing delay and backlog guarantees, for sources constrained by leaky buckets in a 2L-GPS environment, if the weight assignment to each session satisfies a certain condition.

## 2.1 2L-GPS Multiplexing

A two-level GPS multiplexing, or 2L-GPS as denoted throughout, is work conserving and distributes its rate to sessions via a two-level hierarchy. By work conserving we mean the server must be busy when there are packets waiting at the server. By distributing its rate via a two-level hierarchy, a root server with a fixed rate of $r$ does not allocate rates directly to sessions. Instead, the service distribution from the root server follows a two-level hierarchy; that is, the root server distributes its rate to a set of logical servers, each of which in turn assigns its given rate to the underlying physical sessions under its service. Favoring simplicity more than generality, the root server's rate, that is, $r$, is scaled such that $r = 1$. This assumption holds throughout the development of the single-node 2L-GPS case. Whenever necessary (i.e., in the multiple-node case), we will redefine the rate of the root server as an arbitrary positive number $r$.

A 2L-GPS server is characterized by its two-level parameters. Referring to the logical servers, which we denote interchangeably as *first-level* sessions, we define the first-level parameters as the positive real numbers : $\phi_1, \phi_2, \ldots, \phi_N$ , where $N$ is the total number of logical servers consuming service from the root server.

Referring to the physical sessions, which we denote interchangeably as *second-level* sessions, we define the second-level parameters as a set of collections : $\{\phi_{11}, \phi_{12},\ldots, \phi_{1p(1)}\}$, $\{\phi_{21}, \phi_{22},\ldots, \phi_{2p(2)}\}$, …, $\{\phi_{N1}, \phi_{N2},\ldots, \phi_{Np(N)}\}$, where $\phi_{ij}$ denotes the parameter for the $j$th session served by logical server $i$ and $p(i)$ is the total number of second-level sessions consuming service from logical server $i$. Relating parameters of the second level to the first, we have :

$$\phi_i = \sum_{j=1}^{p(i)} \phi_{ij} \tag{2.1}$$

With the assumption that $r = 1$, and furthermore without loss of generality, we set $\phi_{ij}$ for all $i,j$ as the fraction of root server's rate given to the second-level session $i_j$ at the instance when every second-level session has a positive amount of traffic being queued. Thus,

$$\sum_{i,j} \phi_{ij} = 1 \tag{2.2}$$

Combining (2.1) and (2.2), it follows :

$$\sum_{i} \phi_i = 1$$

and hence $\phi_i$, for all $i$, is the fraction of the root server's rate given to first-level session $i$ at the instants when every first-level session has a positive amount of traffic being queued.

With respect to the first-level sessions, let $S_i(\tau,t)$ be the amount of service received by logical server $i$ in time interval $(\tau,t]$. A logical server is backlogged at time $t$ if one or more of its second-level sessions have a positive amount of queued traffic. Then, a root GPS server is defined as one for which

$$\frac{S_i(\tau,t)}{S_j(\tau,t)} \geq \frac{\phi_i}{\phi_j}, j = 1,2,..,N$$

for any logical server $i$ that is backlogged in the interval $(\tau,t]$.

Summing over all logical servers $j$ :

$$S_i(\tau,t)\sum_j \phi_j \geq (t-\tau)\phi_i \tag{2.3}$$

and logical server $i$ is guaranteed a rate of $\phi_i$.

With respect to the second-level sessions, let $S_{ij}(\tau,t)$ be the amount of session $i_j$'s (i.e., $j^{th}$ session of the logical server $i$) traffic served in time interval $(\tau,t]$. A session is backlogged at time $t$ if a positive amount of that session traffic is queued at time $t$. Then, a logical GPS server is defined as one for which :

$$\frac{S_{ik}(\tau,t)}{S_{il}(\tau,t)} \geq \frac{\phi_{ik}}{\phi_{il}}, \quad l = 1,2,..,p(i)$$

for any session $i_k$ that is continuously backlogged in the interval $(\tau,t]$.

Summing over all sessions $i_l$ :

$$S_{ik}(\tau,t)\phi_i \geq S_i(\tau,t)\phi_{ik}$$

substituting from (2.3) to the rhs term :

$$S_{ik}(\tau,t) \geq \frac{\phi_{ik}}{\phi_i}\frac{\phi_i}{\sum_j \phi_j}(t-\tau)$$

$$= \phi_{ik}(t-\tau) \qquad (2.4)$$

and session $i_k$ is guaranteed a rate of $\phi_{ik}$.

An important advantage of 2L-GPS over GPS is the capability to isolate a group of sessions being served by a logical server, such that rates given by the corresponding logical server to these sessions are not directly dependent on the parameters for sessions served by other logical servers. Let us clarify the idea with an example which is represented in Fig. 2-1. Given three sessions to be served by a 2L-GPS server, we assign session $1_1$(i.e., $1^{st}$ session of logical server 1) to logical server 1 and set $\phi_{11}$ = ½. We assign the remaining two sessions to logical server 2, i.e., session $2_1$ and session $2_2$, with $\phi_{21}$ = ¼ and $\phi_{22}$ = ¼ . It follows $\phi_1 = \phi_{11}$ = ½ , $\phi_2 = \phi_{21} + \phi_{22}$ = ½ . At time 0, three packets with length 8,4,2, from session $1_1$,$2_1$,$2_2$ respectively, arrive at the 2L-GPS node. Up to time 8, session $1_1$, $2_1$ and $2_2$ are served with the rates of ½, ¼, ¼ respectively. At time 8, session $2_2$ finishes service. After time 8, session $1_1$ maintains a rate of ½ while session $2_1$ increases its rate from ¼ to ½. At time 12, the packet

from session $2_1$ finishes service and hence causes session $1_1$ to receive a full rate of 1 and therefore finishes service at time 14. The queuing delays of session $1_1$,$2_1$ and $2_2$'s packets are 14,12 and 8 units of time, respectively.

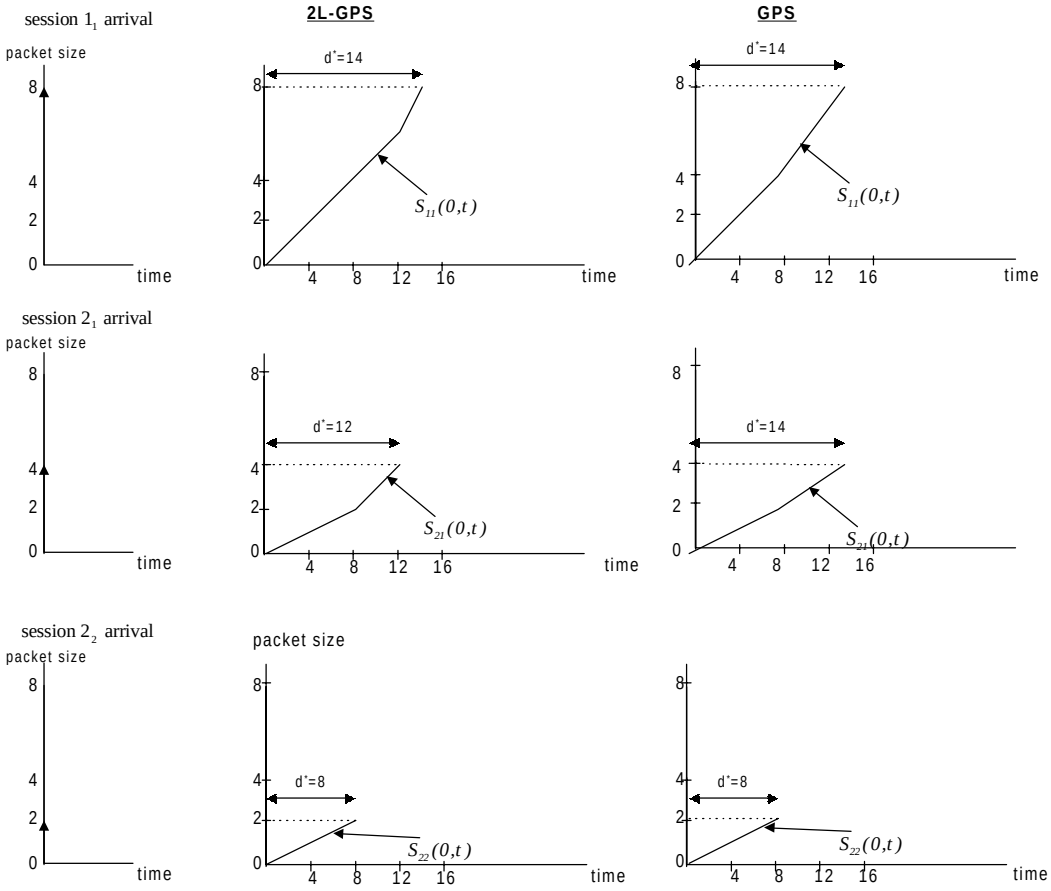



Figure 2-1: An example of 2L-GPS vs GPS

Under the GPS server with no hierarchy, session $2_2$'s packet would still finish service at time 8. Following time 8, however, session $1_1$ receives a rate of $\frac{1/2}{1/2+1/4} = 2/3$ and session $2_1$ receives a rate of $1/3$. Observe that unlike the 2L-GPS scheme, GPS allows each backlogged session to *directly* compete for its share of rate with respect to the ratio of its parameter to the sum of parameters over all the backlogged sessions. With the 2L-GPS discipline, following

time 8, each of session $1_1$ and $2_1$ competes for its share of rate with respect to the ratio of its logical server's parameter to the the sum of parameters over all backlogged logical servers (i.e., logical server 1 and 2). Continuing the example for the GPS case, both session $1_1$ and $2_1$ finish service at time 14, following the increase in rates to ⅔ and ⅓ from ½ and ¼ , at time 8. The queuing delays for session $1_1$,$2_1$, and $2_2$ are 14,14, and 8 units of time respectively. Comparing delays in the GPS case to those of 2L-GPS, it turns out that delays for session $1_1$ and $2_2$ are the same for both schemes while delay for session $2_1$ improves in the 2L-GPS scheme.

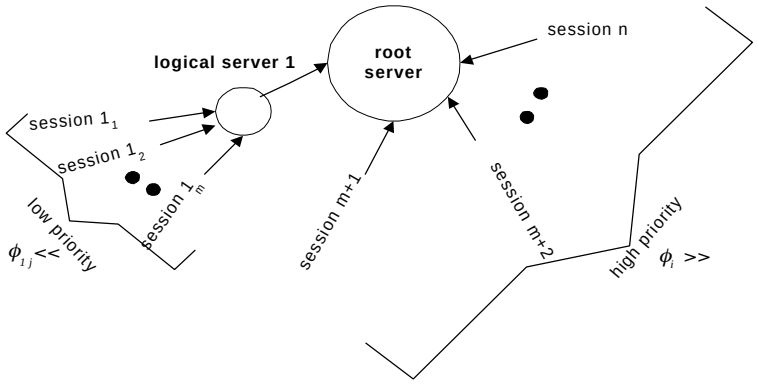



Figure 2-2 : A Grouping Strategy

The above example sheds intuition on how 2L-GPS can improve performance, i.e., queuing delay, of a session with low $\phi_{ij}$'s relative to that of GPS. In the example, we observe that the performance improvement is gained when less bursty lower-priority sessions (i.e., sessions with relatively small $\phi_{ij}$'s ) with short periods of activity, followed by lengthy inactive periods, are grouped together and further assigned to a logical server. Moreover, each of the bursty higher-priority sessions (i.e., sessions with high $\phi_{ij}$'s ) with long periods of activity gets assigned to an individual logical server (see Fig. 2-2).

Although the observation we gain is seemingly weak, due to the oversimplified example where each session only sends one packet instead of a stream of packets, we will further develop the idea of grouping later and show that such a grouping strategy will reduce worst case delays and backlogs for best effort sessions without significantly increasing delays and backlogs incurred by the QoS sessions. Note that unlike a QoS session, a best effort session does not require performance guarantees and therefore is assigned a low priority.

Notice also that a one to one assignment of individual sessions to individual logical servers reduces a 2L-GPS server to a GPS server [1], implying that more flexibility is gained by employing the 2L-GPS scheme.

Aside from the grouping policy, which aims to improve performances for the low priority sessions, 2L-GPS carries over the properties that establish GPS as an attractive multiplexing scheme. These carry-over properties of 2L-GPS are as follow :

- Define $\rho_{ij}$ to be the average rate of the $j^{th}$ session in logical server $i$. Then, as long as $\rho_{ij} \leq \phi_{ij}$, the session can be guaranteed a throughput of $\rho_{ij}$ independent of the demands of the other sessions. In addition to this throughput guarantee, a session $i_j$ backlog will always be cleared at a rate $\geq \phi_{ij}$.
- The delay of an arriving session $i_j$ bit can be upper bounded as a function of the session $i_j$ queue length, independent of the queues and arrivals of other sessions.
- Most importantly, it is possible to make worst-case network queuing delay guarantees when sessions are constrained by leaky buckets and we assign $\phi_{ij}$ to be equal or greater than $\rho_{ij}$ for each session $i_j$, where $\rho_{ij}$ denotes the average rate of the session. Note that in term of the Leaky Bucket constraint, $\rho_{ij}$ is the rate of token arrivals into the leaky bucket. We will present our results on this later.

## 2.2 Leaky Bucket

In this section, the Leaky Bucket scheme, which has its origin in [7] is adopted in order to constrain the incoming traffic of every second-level session at its source. In addition to regulating traffic from second-level sessions, the Leaky Bucket scheme is extended to virtually regulate the incoming traffic from a logical server (i.e., first-level session) at its source.

It is therefore natural to divide our discussion in two sub-sections. The first subsection discusses the second-level Leaky Bucket, which we call the physical Leaky Bucket, and the second subsection discusses the first-level Leaky Bucket, which we call the virtual Leaky Bucket.

### 2.2.1 Second-Level Leaky Bucket

Fig. 2-3 depicts the Leaky Bucket scheme that we will use to describe the traffic that enters the network. Incoming traffic from session $i_j$, that is, the $j^{th}$ session of logical server $i$, is constrained by a leaky bucket at its source. The leaky bucket is a pool of tokens or permits which are generated at a fixed rate, $\rho_{ij}$, so packets arriving at the source can be released into the network only after removing the required number of tokens from the bucket. There is no limit on the number of packets that can be buffered, but the bucket contains at most $\sigma_{ij}$ bits worth of tokens. Once the traffic is released to the network, it leaves the bucket at a maximum rate of $C_{ij} > \rho_{ij}$.

Let $A_{ij}(\tau,t)$ be the flow amount of the $j^{th}$ session in logical server $i$ that leaves the leaky bucket and enters the network in time interval $(\tau,t]$. Then, the constraint imposed by the Leaky Bucket is

$$\begin{aligned} A_{ij}(\tau,t) &\le min\{(t-\tau)C_{ij}, \sigma_{ij} + \rho_{ij}(t-\tau)\}, \ \forall t \ge \tau \ge 0 \\ &\le \sigma_{ij} + \rho_{ij}(t-\tau) \end{aligned} \tag{2.5}$$

for every session $i_j$. We say that session $i_j$ conforms to $(\sigma_{ij},\rho_{ij})$ or $A_{ij} \sim (\sigma_{ij},\rho_{ij})$

The arrival constraint above is attractive since it restricts the traffic in terms of average sustainable rate ($\rho_{ij}$), peak rate ($C_{ij}$), and burstiness ($\sigma_{ij}$, $C_{ij}$). Fig. 2-4 shows how a fairly bursty source might be characterized using the constraints.

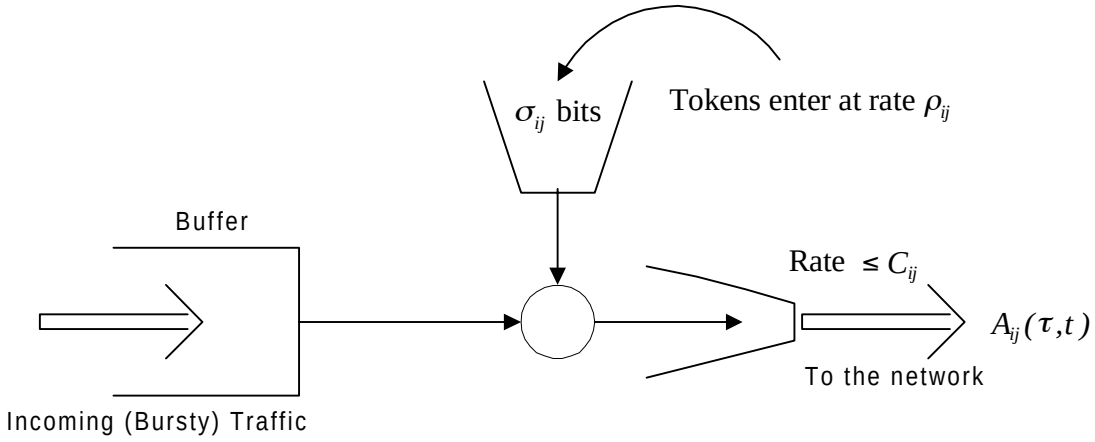



Figure 2-3: A Leaky Bucket

Represent $A_{ij}(0,t)$ as in Fig. 2-4. Let there be $l_{ij}(t)$ bits worth of tokens in the session $i_j$ token bucket at time $t$. We assume that the session starts out with a full bucket of tokens. If $K_{ij}(t)$ is the total number of tokens accepted at the session $i_j$ bucket in the interval $(0,t]$ (it does not include the full bucket of tokens that session $i$ starts out with, and does not include arriving tokens that find the bucket full), then

$$K_{ij}(t) = \min_{0 \leq \tau \leq t}\{ A_{ij}(0,\tau) + \rho_{ij}(t-\tau)\}$$

Thus, for all $\tau \leq t$

$$K_{ij}(t) - K_{ij}(\tau) \leq \rho_{ij}(t-\tau). \tag{2.6}$$

We may now express $l_{ij}(t)$ as

$$l_{ij}(t) = \sigma_{ij} + K_{ij}(t) - A_{ij}(0,t) \tag{2.7}$$

From (2.6) and (2.7), we obtain the useful inequality

$$A_{ij}(\tau,t) \le l_{ij}(\tau) + \rho_{ij}(t-\tau) - l_{ij}(t) \qquad (2.8)$$

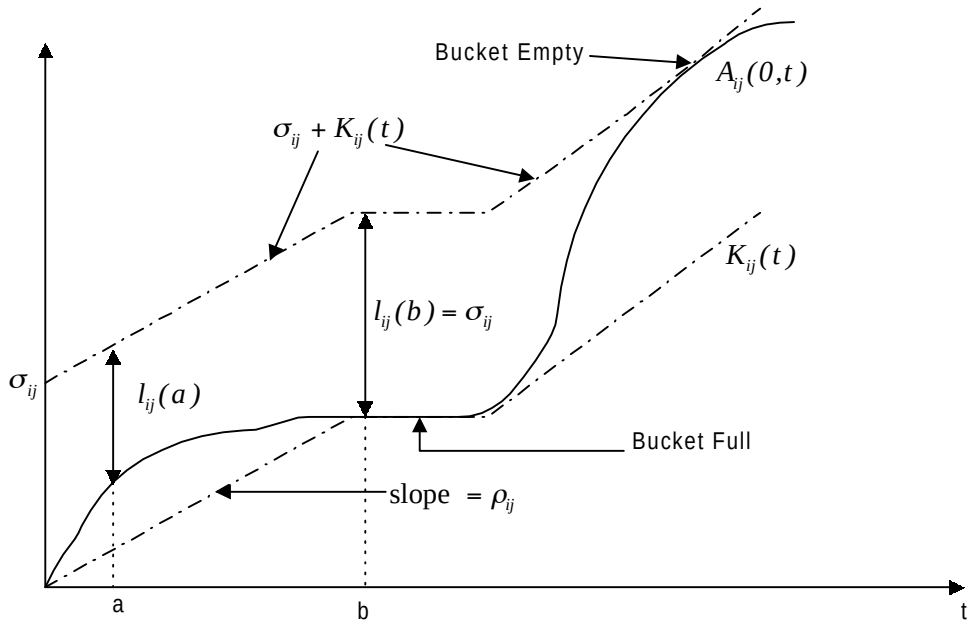



Figure 2-4 : $A_{ij}(0,t)$ and $l_{ij}(t)$

## 2.2.2 First-Level Leaky Bucket

Since every session $i_j$ is assigned to logical server $i$, $A_i(\tau,t)$, that is, the amount of flow from logical server $i$ to the root server at time interval $(\tau,t]$, is constrained such that :

$$\begin{aligned} A_i(\tau,t) &= \sum_{j=1}^{p(i)} A_{ij}(\tau,t) \\ &\le \sum_{j=1}^{p(i)} \sigma_{ij} + (t-\tau)\sum_{j=1}^{p(i)} \rho_{ij} \\ &= \sigma_i + \rho_i(t-\tau) \end{aligned}$$

where the second inequality follows from (2.5). The third equality follows after defining the *virtual* (or first-level) Leaky Bucket constraint $(\sigma_i,\rho_i)$ such that :

$$\sigma_i = \sum_{j=1}^{p(i)} \sigma_{ij} \text{ and } \rho_i = \sum_{j=1}^{p(i)} \rho_{ij}$$

We say that logical server $i$ conforms to $(\sigma_i,\rho_i)$ or $A_i \sim (\sigma_i,\rho_i)$.

Taking such a characterization of a virtual leaky bucket, as well as the definition of 2L-GPS into account, the root 2L-GPS server treats its virtually constrained first-level sessions (logical servers) in exactly the same manner as a GPS server treat its Leaky Bucket constrained sessions. Thus, the development of Leaky Bucket constraints for the first-level sessions follows the same lines as those in [1, Section V].

## 2.3 Analysis

### 2.3.1 First-Level Sessions

Incorporating the notion of virtual leaky bucket that was proposed in the previous section, the root server views each logical server as a first-level session that operates under the virtual Leaky Bucket constraints, and so the arriving traffic from a logical server, which is equivalent to the aggregate traffic arriving from the second-level sessions under its service, is consistent with $(\sigma_i,\rho_i)$.

Let there be $N$ logical servers in the 2L-GPS system. The only assumptions we make about the incoming traffic from these servers are that $A_i \sim (\sigma_i,\rho_i)$ for server $i$ , $i = 1,2,\ldots,N$ and that the system is empty before time zero. The root server is work conserving (i.e., it is never idle if there is work in the system) and operates at the fixed rate of 1.

Let $A_1^0$ , $A_2^0$ , …, $A_n^0$ be the set of arrival functions in which all the first-level sessions are greedy from time 0, the beginning of a system busy period. By greedy from time 0, we mean :

$$A_i^0(0,t) = \sigma_i + \rho_i t \quad \forall t \geq 0, \forall i = 1,\ldots,N$$

For every first layer session $p$, let $S_p^0(0,t)$ be the first layer session $p$ service function under the set of $A_i^0$'s. We adopt a Lemma from [1] that will be very useful in analyzing the

performance, i.e. worst case queuing delays and backlogs, of the second-level sessions under 2L-GPS work discipline.

**Lemma 2.1** *Suppose that time t is contained in a first-level session p busy period beginning at time* $\tau$*. Then :*

$$S_p^0(0,t-\tau) \leq S_p(\tau,t)$$

**Proof.** We notice that the first-level sessions (logical servers) are treated by the root 2L-GPS server in the identical manner as if these sessions were served by a GPS server, owing to the facts that each of the first-level sessions is constrained by a virtual leaky bucket and that, by definition, a root 2L-GPS server treats its first-level sessions in exactly the same manner as a GPS server treats its sessions. We can therefore borrow [1, Lemma 10] and its corresponding proof to show that the statement of Lemma 2.1 holds. □

The Lemma above basically asserts that the service curve of a backlogged logical server at any time interval of length, say $\tau$, is minimized by the greedy service curve that results when the arrival curves from all logical servers (first-level sessions) are greedy from 0 to time $\tau$.

With Lemma 2.1, our analysis from this point onward will be focused on finding the minimizing service curve that leads to worst case delay and backlogs for the second-level sessions. It is noteworthy that second-level sessions are the 'real' or physical sessions which are directly related to users of the network. They shall be analyzed in detail in order to arrive at their guaranteed performance, i.e., worst-case queuing delay and backlog.

### 2.3.2 Second-Level Sessions

Under logical server *i*, there are at most *p(i)* sessions receiving service. We will interchangeably denote the sessions under a logical server as either second-level sessions or

just sessions. The only assumption we make about the incoming traffic from these second-level sessions is that $A_{ij} \sim (\sigma_{ij},\rho_{ij},C_{ij})$ for $j = 1,2,\ldots,p(i)$. The logical server is work conserving and operates at a variable rate when one or more of its underlying sessions are busy. By a variable rate, we mean the service rate of the logical server at a particular instance is the portion of the root server's rate available to the backlogged logical server at that instance of time.

Denote $S_{ij}(\tau,t)$ as the amount of session $i_j$'s traffic served in the interval $(\tau,t]$. This is a continuous and non decreasing function for all $t$. Session $i_j$'s backlog at time $\tau$ is defined to be :

$$Q_{ij}(\tau) = A_{ij}(0,\tau) - S_{ij}(0,\tau)$$

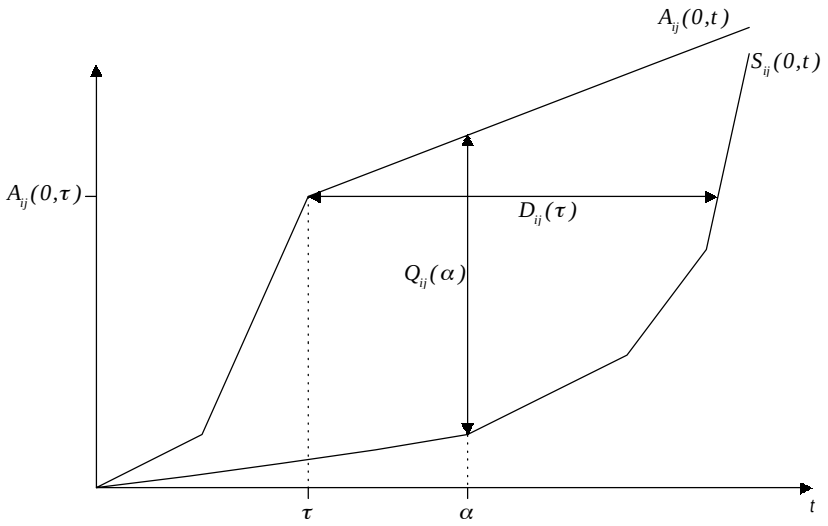



Figure 2-5: $A_{ij}(0,t)$, $S_{ij}(0,t)$, $D_{ij}(t)$, and $Q_{ij}(t)$

The session $i_j$ delay at time $\tau$ is denoted by $D_{ij}(\tau)$, and is the amount of time it would take for the session $i_j$ backlog to clear if no session $i_j$ bits were to arrive after time $\tau$. Thus,

$$D_{ij}(\tau) = inf\{t \geq \tau : S_{ij}(0,t) = A_i(0,\tau)\} - \tau$$

From Fig. 2-5, we see that $D_{ij}(t)$ is the horizontal distance between curves $A_{ij}(0,t)$ and $S_{ij}(0,t)$ at the ordinate value of $A_{ij}(0,t)$. This implies that $D_{ij}(t)$ depends on future arrivals from other sessions, i.e, arriving traffic from other sessions after time $t$.

Clearly, $D_{ij}(t)$ depends on the arrival functions $A_{ij}$ for all $i$ and $j$. We are interested in computing the maximum delay over all time, and over all arrival functions that are consistent with (2.5). Let $D_{ij}^*$ be the maximum delay for session $i_j$. Then :

$$D_{ij}^* = \max_{A_{kl} \forall k \forall l} \max_{\tau \geq 0} D_{ij}(\tau)$$

Similarly, we define the maximum backlog for session $i_j$, $Q_{ij}^*$ :

$$Q_{ij}^* = \max_{A_{kl} \forall k \forall l} \max_{\tau \geq 0} Q_{ij}(\tau)$$

The problem we will solve in the following section is :

Given $\{\phi_{11}, \phi_{12}, \ldots, \phi_{1p(1)}\}$, $\{\phi_{21}, \phi_{22}, \ldots, \phi_{2p(2)}\}$, $\ldots$, $\{\phi_{N1}, \phi_{N2}, \ldots, \phi_{Np(N)}\}$, where $\phi_{ij} \geq \rho_{ij}$ for a 2L-GPS root server of rate 1, and given $A_{ij} \sim (\sigma_{ij}, \rho_{ij})$ for $i = 1,2,\ldots,N$ ; $j = 1,2,\ldots,p(i)$ ; what are $D_{ij}^*$ and $Q_{ij}^*$ for every session $i_j$ ? We will also be able to characterize the burstiness of the output traffic for every session $i_j$, which will be especially useful in our analysis of the 2L-GPS network.

### 2.3.2.1 Definitions and Preliminary Results

We introduce definitions and derive inequalities that are helpful in our analysis. Some of these notions are general enough to be used in the analysis of any work conserving service discipline that operates on sources that are Leaky Bucket constrained.

Given $A_{ij}$ for all $i$ and $j$, let $\sigma_{ij}^\tau$ be defined for each session $i_j$ and time $\tau \geq 0$ as

$$\sigma_{ij}^\tau = Q_{ij}(\tau) + l_{ij}(\tau) \tag{2.9}$$

where $l_{ij}(\tau)$ is defined in (2.7). Thus, $\sigma_{ij}^\tau$ (see Fig. 2-6) is the sum of the number of tokens left in session $i_j$'s bucket and the session backlog at time $\tau$. We observe that $\sigma_{ij}^0 = \sigma_{ij}$ and

$$Q_{ij}(\tau) = 0 \Rightarrow \sigma_{ij}^\tau \leq \sigma_{ij} \tag{2.9a}$$

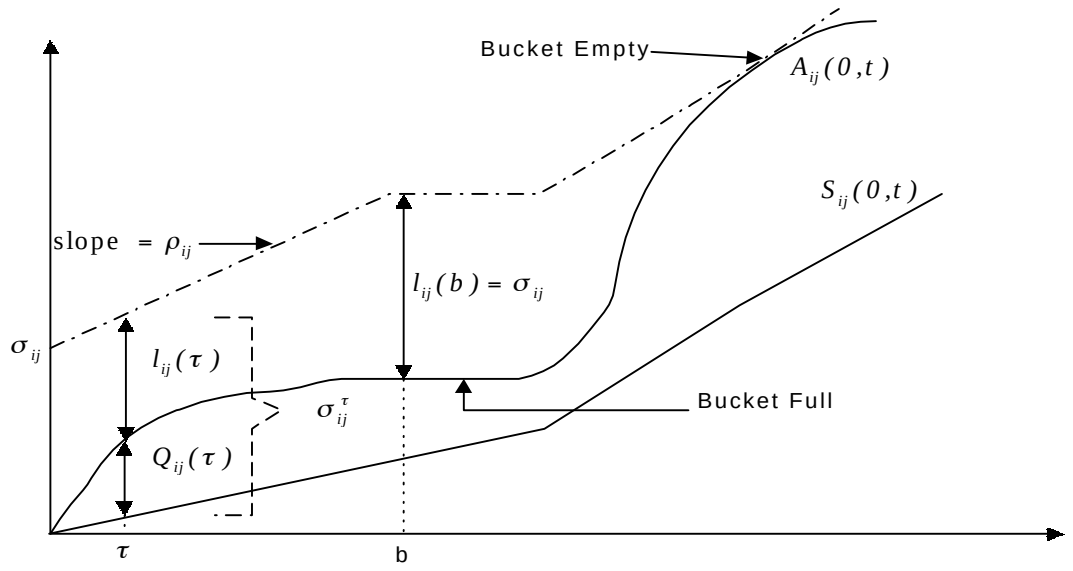



Fig. 2-6: Graphical representation of $\sigma_{ij}^{\tau}$

Recall (2.8)

$$A_{ij}(\tau,t) \le l_{ij}(\tau) + \rho_{ij}(t-\tau) - l_{ij}(t)$$

Substituting for $l_{ij}(\tau)$ and $l_{ij}(t)$ from (2.9)

$$Q_{ij}(\tau) + A_{ij}(\tau,t) - Q_{ij}(t) \le \sigma_{ij}^{\tau} - \sigma_{ij}^{t} + \rho_{ij}(t-\tau) \qquad (2.10)$$

Now notice that

$$S_{ij}(\tau,t) = Q_{ij}(\tau) + A_{ij}(\tau,t) - Q_{ij}(t) \qquad (2.11)$$

Combining (2.10) and (2.11), we establish the following useful result :

**Lemma 2.2** *For every session $i_j$, $\tau \le t$ :*

$$S_{ij}(\tau,t) \le \sigma_{ij}^{\tau} - \sigma_{ij}^{t} + \rho_{ij}(t-\tau)$$

Define a system busy period to be a maximal interval $B$ such that for any $\tau$, $t$ :

$$\sum_{i=1}^{N}\sum_{j=1}^{p(i)} S_{ij}(\tau,t) = t-\tau$$

Since the system is work conserving, if $B = [t_1,t_2]$, then

$$\sum_{i=1}^{N}\sum_{j=1}^{p(i)} Q_{ij}(t_1) = \sum_{i=1}^{N}\sum_{j=1}^{p(i)} Q_{ij}(t_2) = 0$$

Let $\alpha$ be defined as :

$$\alpha = \sum_i \sum_j \rho_{ij} \tag{2.12}$$

Now, we show the following lemma.

**Lemma 2.3** *When $\alpha < 1$, the length of a system busy period is at most*

$$\frac{\sum_{i=1}^{N}\sum_{j=1}^{p(i)} \sigma_{ij}}{1-\alpha}$$

**Proof**. Suppose $[t_1,t_2]$ is a system busy period. By assumption,

$$\sum_{i=1}^{N}\sum_{j=1}^{p(i)} Q_{ij}(t_1) = \sum_{i=1}^{N}\sum_{j=1}^{p(i)} Q_{ij}(t_2) = 0$$

Thus,

$$\sum_{i=1}^{N}\sum_{j=1}^{p(i)} A_{ij}(t_2,t_1) = \sum_{i=1}^{N}\sum_{j=1}^{p(i)} S_{ij}(t_2,t_1) = t_2 - t_1$$

Substituting from (2.5) and rearranging terms :

$$t_2 - t_1 \le \frac{\sum_{i=1}^{N}\sum_{j=1}^{p(i)} \sigma_{ij}}{1-\sum_{i=1}^{N}\sum_{j=1}^{p(i)} \rho_{ij}}$$

By (2.12), we have shown Lemma 2.3. □

Let $\alpha < 1$ be called the stability condition. The stability condition therefore requires $\alpha$, i.e., the maximum sustainable rate for the aggregate traffic arriving at the server, to be strictly less than the root server's rate. A simple consequence of Lemma 2.3 is that all 2L-GPS system busy periods are bounded if the stability condition is satisfied and may be infinite otherwise.

Note that session delay is bounded by the length of the largest possible system busy period. When the system busy period is bounded, the session delays are bounded as well.

We end this section with some comments valid only for the second-level sessions in the 2L-GPS system : Let a session $i_k$ busy period be a maximal interval $B_{ik}$ contained in a single busy period for logical server $i$ such that for all $\tau$, $t \in B_{ik}$ :

$$\frac{S_{ik}(\tau,t)}{S_{il}(\tau,t)} \geq \frac{\phi_{ik}}{\phi_{il}}, \quad l = 1,2,..,p(i)$$

Notice that it is possible for a session to have zero backlog during its busy period. However, if $Q_{ik}(\tau) > 0$ then $\tau$ must be in a session $i$ busy period at time $\tau$. We have already shown in (2.4) that :

**Lemma 2.4** *For every interval $(\tau,t]$ that is in a session $i_k$ busy period :*

$$S_{ik}(\tau,t) \geq \phi_{ik}(t-\tau)$$

### 2.3.2.2 Greedy Sessions

Session $i_j$ is defined to be greedy starting at time $\tau$, if

$$A_{ij}(\tau,t) = min\{ C_{ij}(t-\tau), l_{ij}(\tau) + (t-\tau)\rho_{ij} \}, \text{ for all } t \geq \tau \qquad (2.13)$$

In terms of the Leaky Bucket, this means that the session uses as many tokens as possible (i.e., sends at maximum possible rate) for all times $\geq \tau$. At time $\tau$, session $i_j$ has $l_{ij}(\tau)$ tokens left in the bucket, but it is constrained to send traffic at a maximum rate of $C_{ij}$ . Thus, it takes $\frac{l_{ij}(\tau)}{C_{ij}-\rho_{ij}}$ time units to deplete the tokens in the bucket. After this, the rate will be limited by the token arrival rate $\rho_{ij}$.

Define $A_{ij}^{\tau}$ as an arrival function that is greedy starting at time $\tau$ (see Fig. 2-7). From inspection of the figure and from (2.13), we see that if a system busy period starts at time zero then :

$$A_{ij}^{0}(0,t) \geq A_{ij}(0,t), \quad \forall A_{ij} \sim (\sigma_{ij}, \rho_{ij}, C_{ij}) \text{ and } t \geq 0$$

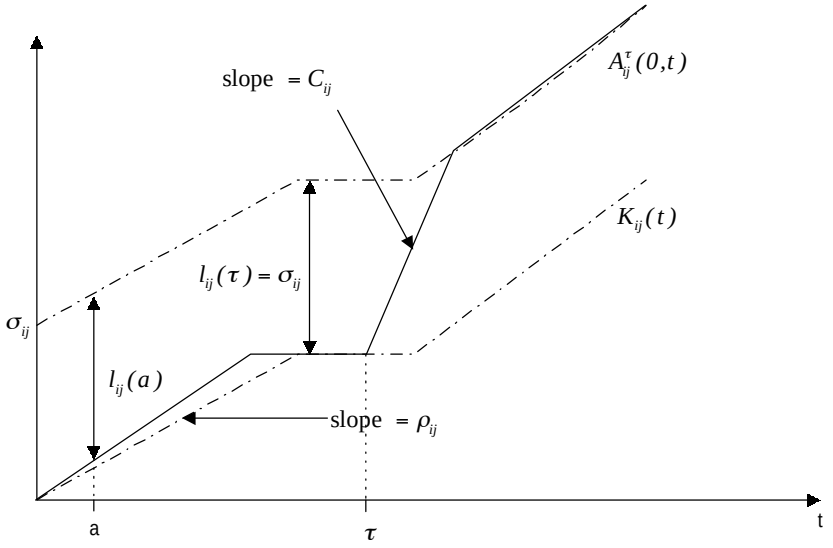



Figure 2-7: A session $i_j$ arrival function that is greedy from time $\tau$

The major result in this chapter is the following :

**Theorem 2.1** *Suppose $C_{ij} \geq r$ for every session $i_j$, where r is the rate of the root 2L-GPS server. Furthermore, the assignment of a session's parameter is restricted such that $\phi_{ij} \geq \rho_{ij}$ for all i and j. Then, for every session $i_j$, $D_{ij}^{*}$ and $Q_{ij}^{*}$ are achieved (not necessarily at the same time) when every session is greedy starting at time zero, the beginning of a system busy period. Also, $Q_{ij}^{*} = \sigma_{ij}$.*

Our approach is to prove the theorem for the case when $C_{ij} = \infty$ for all *i* and *j*, assuming initially that the stability condition, i.e., $\alpha < 1$, is satisfied - this implies the links carrying traffic to the server have infinite capacity. This is the easiest case to visualize since we do not have to worry about the input links. Further, the arriving traffic from a session, when the link

has infinite capacity, bounds the arriving traffic of the particular session when the link speed is finite in the way that is shown in (2.5) and hence any session can "simulate" a finite speed input link by sending packets at a finite rate over the link.

#### 2.3.2.3 2L-GPS with Infinite Incoming Link Capacities

When all input link speeds are infinite, the arrival constraint, as in (2.5), is :

$$A_{ij}(\tau,t) \le \sigma_{ij} + \rho_{ij}(t-\tau), \quad \forall 0 \le \tau \le t, \tag{2.14}$$

for every session $i_j$. We say that session $i_j$ conforms to $(\sigma_{ij},\rho_{ij})$ or $A_{ij} \sim (\sigma_{ij},\rho_{ij})$.

By relaxing our constraint, we allow step or jump arrivals, which create discontinuities in the arrival functions $A_{ij}$. Our convention will be to treat the $A_{ij}$ as left-continuous functions (i.e.,continuous from the left). Thus, a session $i_j$ impulse of size $\Delta$ at time 0 yields $Q_{ij}(0) = 0$ and $Q_{ij}(0^+) = \Delta$. Note also that $l_{ij}(0) = \sigma_{ij}$, where $l_{ij}(\tau)$ is the maximum amount of session $i_j$ traffic that could arrive at time $\tau^+$ without violating (2.14). When session $i$ is greedy from time $\tau$, the infinite capacity assumption ensures that $l_{ij}(t) = 0$ for all $t > \tau$. Thus, (2.8a) reduces to :

$$A_{ij}^{\tau}(\tau,t) = l_{ij}(\tau) + (t-\tau)\rho_{ij}, \text{ for all } t > \tau. \tag{2.15}$$

Note also that if the session is greedy after time $\tau$, $l_{ij}(t)=0$ for any $t > \tau$.

Defining $\sigma_{ij}^{\tau}$ as before (from 2.9), we see that it is equal to $Q_{ij}(\tau^+)$ when session $i_j$ is greedy starting at time $\tau$.

##### 2.3.2.3.1 An All-Greedy 2L-GPS system

Here we start off looking at the general case when $\phi_{ij}$ is not necessarily greater or equal to $\rho_{ij}$. This will give us some insight into why that assumption is made in Theorem 2.1.

Theorem 2.1 suggests that we should examine the dynamics of a system in which all the second-level sessions (and therefore the first-level sessions) are greedy starting at time 0, the beginning of a system busy period. This is illustrated in Fig. 2-8. For clarity of exposition, we

review the notions of first and second-level sessions, i.e., logical servers and physical sessions, respectively.

From (2.15), we know that :

$$A_{ij}^{0}(0,\tau) = \sigma_{ij} + \rho_{ij}\tau \ , \text{ for all } \tau > 0.$$

summing over all $j$,

$$A_{i}^{0}(0,\tau) = \sigma_{i} + \rho_{i}\tau \ , \text{ for all } \tau > 0.$$

Let us assume, again for clarity of exposition, that $\sigma_{ij} > 0$ for all $i$ and $j$.

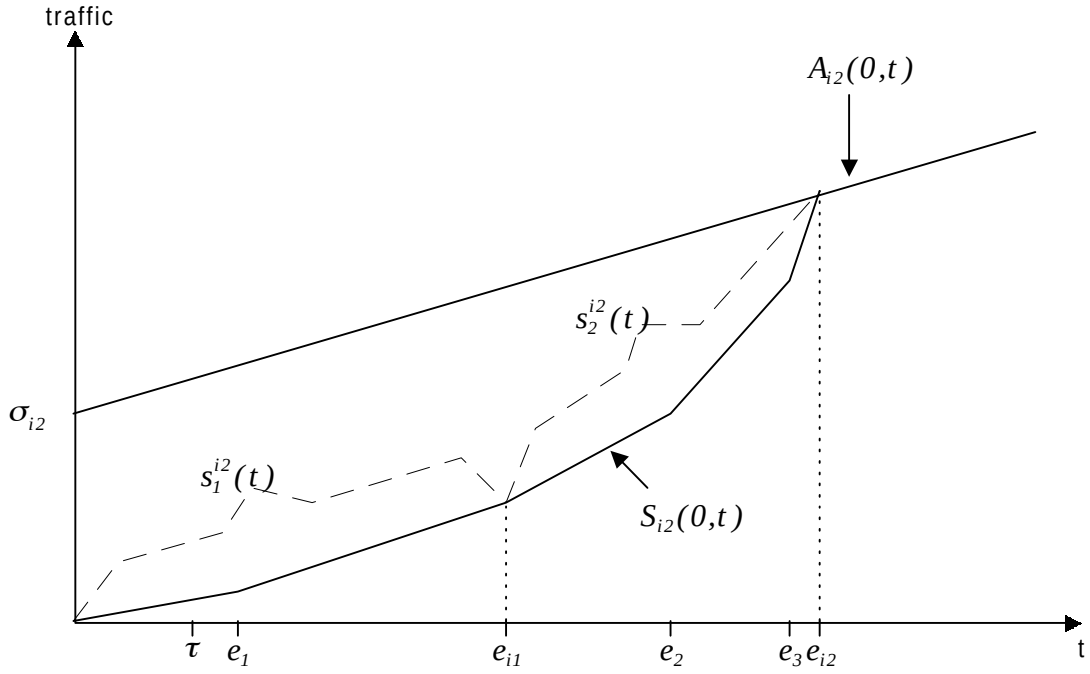



Figure 2-8: Session $i_2$ arrivals and departures after 0, the beginning of a system busy period in which all sessions are greedy.

We assume that the logical servers are ordered in the order that they empty their backlog under the all greedy regime. Similarly, assume that the second-level sessions are similarly ordered under each logical server. In a given set of $\{\phi_{ij}\},\{\rho_{ij}\}$, and $\{\sigma_{ij}\}$, let :

$e_i$ be the time when logical server $i$ goes unbacklogged.

$e_{in}$ be the time when session $i_n$ in logical server $i$ goes unbacklogged.

$s^i(t)$ be the slope of $S_i(0,t)$ (i.e., the service curve of first-level session $i$ under the all greedy regime) at time $t$.

$s_n^{ik}(t)$ be the slope of $S_{ik}(0,t)$ (i.e., the service curve of second-level session $i_k$ under the all greedy regime) at time $t \in [e_{i(n-1)}, e_{in}]$. Note that the slope could change in the interval $[e_{i(n-1)}, e_{in}]$ (see Fig. 2-8).

As illustrated in Fig. 2-8, the slopes of the various segments that comprise $S_{ik}(0,t)$ (i.e., the service curve of second-level session $i_k$ under the all greedy regime) are $s_1^{ik}, s_2^{ik}, \ldots$, where :

$$s_n^{ik}(t) = \frac{(s^i(t) - \sum_{j=1}^{n-1} \rho_{ij})\phi_{ik}}{\sum_{j=n}^{p(i)} \phi_{ij}}, \quad n = 1,2,..k \text{ and } t \in [e_{i(n-1)}, e_{in}]$$

Note that $e_{i0} = 0$. To see why the above equality is true, note that, under the greedy regime, a backlogged session, say session $i_j$, is sending traffic at rate $\rho_{ij}$. Just before session $i_j$ empties its backlog, it is served at rate $s_j^{ij}(t)$, where $s_j^{ij}(t) > \rho_{ij}$. Just after session $i_j$ empties its backlog, its service rate is equal to its arrival rate, i.e., $\rho_{ij}$ and therefore the excess rate, i.e., $s_j^{ij}(t) - \rho_{ij}$, is distributed to the remaining backlogged sessions under logical server $i$.

In the remainder of this section, we will prove a tight lower bound on the amount of service a second-level session, or session as we will denote from here on, receives when it is in a busy period.

**Lemma 2.5** *Assume that session $i_k$ is in a busy period in the interval $(\tau,t]$. Then for any subset $M$ containing $m$ sessions, $1 \le m \le p(i)$ and any time $t \ge \tau$ :*

$$S_{ik}(\tau,t) \ge \frac{(S_i^0(0,t-\tau) - (\sum_{j \notin M} \sigma_{ij}^\tau + \rho_{ij}(t-\tau)))\phi_{ik}}{\sum_{j \in M} \phi_{ij}}$$

**Proof**. For compactness of notation, let $\phi_i(lk) = \phi_{il} / \phi_{ik}$, $\forall k,l$

From Lemma 2.2,

$$S_{il}(\tau,t) \le \sigma_{il}^{\tau} + \rho_{il}(t-\tau)$$

for all $l$. Also, since the interval $(\tau,t]$ is in a session $i_k$ busy period :

$$S_{il}(\tau,t) \le \phi_i(lk) S_{ik}(\tau,t)$$

Thus,

$$S_{il}(\tau,t) \le min\{\sigma_{il}^{\tau} + \rho_{il}(t-\tau), \phi_i(lk) S_{ik}(\tau,t)\}$$

Since session $i_k$ is in a busy period in the interval $(\tau,t]$, logical server $i$ is also, and serves exactly $S_i(\tau,t)$ units of traffic in the interval $(\tau,t]$. Thus,

$$S_i(\tau,t) \le \sum_{l=1}^{p(i)} min\{\sigma_{il}^{\tau} + \rho_{il}(t-\tau), \phi_i(lk) S_{ik}(\tau,t)\}$$

$$\Rightarrow S_i(\tau,t) \le \sum_{l \notin M} \sigma_{il}^{\tau} + \rho_{il}(t-\tau) + \sum_{l \in M} \phi_i(lk) S_{ik}(\tau,t)$$

for any subset of sessions $M$. Rearranging the terms yields :

$$S_{ik}(\tau,t) \ge \frac{(S_i(\tau,t) - (\sum_{j \notin M} \sigma_{ij}^{\tau} + \rho_{ij}(t-\tau)))\phi_{ik}}{\sum_{j \in M} \phi_{ij}}$$

The interval $(\tau,t]$ is contained in logical server $i$'s busy period, and thus from Lemma 2.1 :

$$S_i^0(0,t-\tau) \le S_i(\tau,t)$$

which therefore completes the proof. □

### 2.3.2.4 An Important Inequality

In the previous section, we examined the behaviour of the 2L-GPS system when the second-level sessions are greedy. Here, we prove an important inequality that holds for any arrival functions that conform to the arrival constraint (2.14).

**Lemma 2.6** *Let $\phi_{ij} \geq \rho_{ij}$ for all i and j. Then, for any time t and any session $i_j$ under logical server i :*

$$\sigma_{ij}^t \leq \sigma_{ij}$$

**Proof**. Consider 2 cases :

Case 1 : Suppose that $Q_{ij}(t)=0$ at time $t$.

In this case, the theorem follows from (2.9a).

Case 2 : Suppose that $Q_{ij}(t)>0$ at time $t$.

Define $\tau$ to be the last time before $t$ such that $Q_{ij}(\tau)=0$. Then, session $i_j$ is in busy period in the interval $(\tau,t]$, and we have :

$$S_{ij}(\tau,t) \geq \phi_{ij}(t-\tau) \geq \rho_{ij}(t-\tau) \tag{2.16}$$

From Lemma 2.2,

$$\sigma_{ij}^t \leq \sigma_{ij}^\tau + \rho_{ij}(t-\tau) - S_{ij}(\tau,t) \leq \sigma_{ij}^\tau \leq \sigma_i$$

The second inequality follows from (2.16) and the last inequality follows from (2.9a), which completes the proof. □

### 2.3.2.5 Proof of the Main Result

In this section, we will use Lemma 2.5 and Lemma 2.6 to prove Theorem 2.1 for infinite capacity incoming links.

Let $A^0 = \{ A_{ij}^0 : i=1,..,N, j=1,..,p(i)\}$ be the set of arrival functions in which all the sessions are greedy from time 0, the beginning of a system busy period. For every session $i_j$, let $S_{ij}^0(\tau,t)$, $D_{ij}^0(t)$, and $Q_{ij}^0(t)$ be the session $i_j$ service, delay and backlog functions under $A^0$, respectively. We first show :

**Lemma 2.7** *Let $\phi_{ij} \geq \rho_{ij}$ for all i and j. Suppose that time t is contained in a session $i_j$'s busy period that begins at time $\tau$. Then :*

$$S_{ij}^0(0,t-\tau) \le S_{ij}(\tau,t)$$

**Proof.** From Lemma 2.5 :

$$S_{ij}(\tau,t) \ge \frac{(S_i^0(0,t-\tau)-(\sum_{k\notin M}\sigma_{ik}^{\tau}+\rho_{ik}(t-\tau)))\phi_{ij}}{\sum_{k\in M}\phi_{ik}}$$

for any subset $M$ of the second-level sessions under logical server $i$. Set $M$ as the set of second-level sessions (of logical server $i$) which are backlogged at time $t$-$\tau$ under $A^0$. Lemma 2.6 asserts that :

$$S_{ij}(\tau,t) \ge \frac{(S_i^0(0,t-\tau)-(\sum_{k\notin M}\sigma_{ik}+\rho_{ik}(t-\tau)))\phi_{ij}}{\sum_{k\in M}\phi_{ik}} \qquad (2.17)$$

Under $A^0$, for all $i_k \in M$ we have $S_{ik}^0(0,t-\tau) \ge \frac{\phi_{ik}}{\phi_{ij}} S_{ij}^0(0,t-\tau)$.

Furthermore, for all $i_k \notin$ M, we have $S_{ik}^0(0,t-\tau) = \sigma_{ik} + \rho_{ik}(t-\tau)$

It follows that :

$$\begin{aligned} S_i^0(0,t-\tau) &= \sum_{k\in M} S_{ik}^0(0,t-\tau) + \sum_{k\notin M} S_{ik}^0(0,t-\tau) \\ &\ge \frac{S_{ij}^0(0,t-\tau)}{\phi_{ij}} \sum_{k\in M}\phi_{ik} + \sum_{k\notin M}\sigma_{ik} + \rho_{ik}(t-\tau) \\ \Rightarrow S_{ij}^0(0,t-\tau) &\le \frac{(S_i^o(0,t-\tau)-(\sum_{k\notin M}\sigma_{ik}+\rho_{ik}(t-\tau)))\phi_{ij}}{\sum_{k\in M}\phi_{ik}} \qquad (2.18) \end{aligned}$$

Combining (2.18) and (2.17), we complete the proof. □

**Lemma 2.8** *Let $\phi_{ij} \ge \rho_{ij}$ for all i and j. For every session $i_j$ under logical server i, $D_{ij}^*$ and $Q_{ij}^*$ are achieved ( not necessarily at the same time) when every session is greedy starting at time zero. Also, $Q_{ij}^* = \sigma_{ij}$.*

**Proof.** We first show that the session $i_j$ backlog is maximized under $A^0$. Let us consider any set of arrival functions $A = \{A_{ij} : i=1,..,N , j=1,..,p(i)\}$ that conforms to (2.14), and for any session $i_j$, consider $Q_{ij}(t)$ at any $t$. Let $\tau$ be be the last time at or before $t$ such that $Q_{ij}(\tau)=0$.

From Lemma 2.7,

$$S_{ij}^{0}(0,t-\tau) \le S_{ij}(\tau,t)$$

Also,

$$A_{ij}(\tau,t) \le \sigma_{ij} + \rho_{ij}(t-\tau) = A_{ij}^{0}(0,t-\tau)$$

Thus,

$$A_{ij}^{0}(0,t-\tau) - S_{ij}^{0}(0,t-\tau) \ge A_{ij}(\tau,t) - S_{ij}(\tau,t)$$

i.e.,

$$Q_{ij}^{0}(t-\tau) \ge Q_{ij}(t)$$

Furthermore,

$$\begin{aligned} Q_{ij}^{0}(t) &= A_{ij}^{0}(0,t) - S_{ij}^{0}(0,t) \\ &= \sigma_{ij} + \rho_{ij}t - S_{ij}^{0}(0,t) \\ &= \sigma_{ij} + \rho_{ij}t - \phi_{ij}t \\ &\le \sigma_{ij} \end{aligned} \qquad (2.19)$$

(2.19) is satisfied with equality when $t = 0^{+}$, and thus $Q_{ij}^{*} = \sigma_{ij}$.

The case for delay is similar. Consider any set of arrival functions $A = \{A_{ij} : i=1,..,N, j=1,..,p(i)\}$ that conforms to (2.14). For any session $i_j$, consider $D_{ij}(t)$ at any $t$ and let $\tau$ be be the last time at or before $t$ such that $Q_{ij}(\tau)=0$. From the definition of delay in Section 2.3.2,

$$A_{ij}(\tau,t) - S_{ij}(\tau,t + D_{ij}(t)) = 0$$

Let us denote $d_{ij} = t - \tau$. From Lemma 2.7,

$$S_{ij}^{0}(0,d_{ij} + D_{ij}(t)) \le S_{ij}(\tau,t + D_{ij}(t))$$

and, since $\sigma_{ij} \ge \sigma_{ij}^{\tau}$:

$$A_{ij}^{0}(0,d_{ij}) \ge A_{ij}(\tau,t)$$

It follows,

$$\begin{aligned} A_{ij}^{0}(0,d_{ij}) - S_{ij}^{0}(0,d_{ij} + D_{ij}(t)) \ge A_{ij}(\tau,t) - S_{ij}(\tau,t + D_{ij}(t)) = 0 \\ \Rightarrow D_{ij}^{0}(d_{ij}) \ge D_{ij}(t) \end{aligned}$$

This completes the proof of Theorem 2.1 for infinite capacity incoming links with $\alpha < 1$. □

Theorem 2.1 provides an interesting observation. The stability condition (see Lemma 2.3) requires that $\alpha < 1$. However, when $\phi_{ij} = \rho_{ij}$ and therefore, by (2.2) and (2.12), $\alpha = 1$, Theorem 2.1 implies that every session $i_j$, for all $i$ and $j$, is stable. To see why the statement is true, consider the case when all sessions are greedy from time 0. Every session $i_j$ will receive service at the rate of $\phi_{ij} = \rho_{ij}$ for any time $t \geq 0$. Consequently, $Q_{ij}^0(t) = \sigma_{ij}$ and $D_{ij}^0(t) = \frac{\sigma_{ij}}{\rho_{ij}}$ for any time $t > 0$. Thus, the theorem is also true without initially assuming that $\alpha < 1$.

### 2.3.2.6 The Output Burstiness $\sigma_{ij}^{out}$

Borrowing the idea of output burstiness from [1], we now focus on determining, for every session $i_j$, the least quantity $\sigma_{ij}^{out}$ such that :

$$S_{ij} \sim (\sigma_{ij}^{out}, \rho_{ij}, r)$$

where $r$ is the rate of the server. To see that this is the best possible characterization of the output process, consider the case in which session $i_j$ is the only active session and is greedy from time zero. Then, a peak service rate of $r$ and a maximum sustainable average rate of $\rho_{ij}$ are both achieved.

By characterizing $S_{ij}$ in this manner, we can begin to analyze networks of servers. Fortunately there is a convenient relationship between $\sigma_{ij}^{out}$ and $Q_{ij}^* = \sigma_{ij}$.

**Lemma 2.9** *If $C_{ij} \geq r$ for every session $i_j$, where $r$ is the rate of the server, then for each session $i_j$ :*

$$\sigma_{ij}^{out} = Q_{ij}^* = \sigma_{ij}$$

**Proof.** First, consider the case $C_{ij} = \infty$. Suppose that $Q_{ij}^*$ is achieved at some time $t^*$, and session $i_j$ continues to send traffic at rate $\rho_{ij}$ after $t^*$. We define a set $K$ to be the set which contains all second-level sessions except session $i_j$. Further, for each session $m_n \in K$, let $t_{mn}$ be the time of arrival of the last session $m_n$ bit to be served before time $t^*$. Then, $Q_{ij}^*$ is also achieved at $t^*$ when the arrival functions of all sessions $m_n \in K$, are truncated at $t_{mn}$, i.e., $A_{mn}(t_{mn},t) = 0$, $\forall\ m_n \neq i_j$. In this case, all other session queues are empty at time $t^*$ and, beginning at time $t^*$, the server will exclusively serve session $i_j$ at rate 1 for $\frac{Q_{ij}^*}{1-\rho_{ij}}$ units of time, after which session $i_j$ will be served at rate $\rho_{ij}$. Thus,

$$S_i(t^*,t) = min\{t - t^*, Q_{ij}^* + \rho_{ij}(t^* - t)\}, \forall t \geq t^*$$

From this, we have :

$$\sigma_{ij}^{out} \geq Q_{ij}^*$$

We now show that the reverse inequality holds as well. For any $\tau \leq t$ :

$$\begin{aligned} S_{ij}(\tau,t) &= A_{ij}(\tau,t) + Q_{ij}(\tau) - Q_{ij}(t) \\ &\leq l_{ij}^{\tau} + \rho_{ij}(t-\tau) + Q_{ij}(\tau) - Q_{ij}(t) \\ &= \sigma_{ij}^{\tau} - Q_{ij}(t) + \rho_{ij}(t-\tau) \end{aligned}$$

(since $C_{ij} = \infty$ ) This implies that :

$$\sigma_{ij}^{out} \leq \sigma_{ij}^{\tau} - Q_{ij}(t) \leq \sigma_{ij}^{\tau} \leq Q_{ij}^*$$

Thus,

$$\sigma_{ij}^{out} = Q_{ij}^*$$

Now suppose that $C_{ij} \in [r, \infty)$. Since the traffic observed under the all-greedy regime is indistinguishable from a system in which all incoming links have infinite capacity, we must have $\sigma_{ij}^{out} = Q_{ij}^*$ in this case as well. By Lemma 2.8, $Q_{ij}^* = \sigma_{ij}$ and the proof is completed. □

# Chapter 3

# Packet-by-Packet Transmission Scheme and Its Lateness

A problem with 2L-GPS, as it is with GPS, is that it is an idealized discipline that does not transmit packets as entities. It assumes that the server can serve multiple sessions simultaneously and that the traffic is infinitely divisible. In the first section, we present a packet-by-packet transmission scheme that is an approximation to 2L-GPS even when the packets are of variable length. The packet-by-packet scheme adopts the H-PFQ (Hierarchical Packet Fair Queuing) framework which was first developed in [4] in order to provide a translation from the idealized fluid multilevel GPS system into a packet-by-packet system. By a packet-by-packet system, we mean only one session can receive service at a time and the minimum service unit is a packet.

Subsequent to describing the packet-by-packet approximation scheme, we apply the techniques developed in [1] for analyzing the lateness incurred by each session when two-level HPFQ is used to approximate the 2L-GPS scheme. By lateness, we mean how much later packets may depart the system under 2L-PGPS relative to 2L-GPS.

## 3.1 Two-Level Packet-by-Packet Generalized Processor Sharing

In this section, we borrow the Hierarchical Packet Fair Queuing (HPFQ) algorithm which was employed in [4] to approximate the hierarchical GPS scheme. In order to approximate the fluid 2L-GPS scheme, we shall reduce the use of HPFQ from a multiple to only a two-level packet-by-packet transmission scheme. We call the resulted packet-by-packet transmission scheme 2L-PGPS for Two-Level Packet-by-Packet Generalized Processor Sharing.

### 3.1.1 Virtual Time Implementation of 2L-PGPS

Before proceeding with the 2L-PGPS packet algorithm, we need to make a few adjustments with respect to the concept of virtual time function. In [1], the packet implementation of GPS is based on the notion of a system virtual time function $V(t)$, which is the normalized fair amount of service that all backlogged sessions should receive by time $t$ in the GPS system. In the case of a 2L-GPS system, we adopt the concept of virtual time and furthermore extend its use so that virtual time is defined for two classes of servers, i.e., root and logical server. In the following, we describe the concept of virtual time for both root server and logical server.

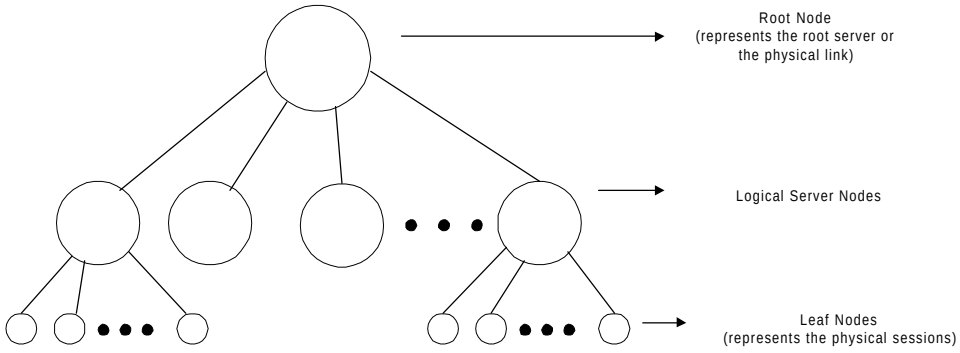



Figure 3-1 : The tree-like structure of a 2L-GPS server

Representing the 2L-PGPS server in a tree-like structure as in Fig.3.1, let us denote the root server as the root node, and the physical sessions (second-level sessions) as the leaf nodes. A logical server, which is a non-root and a non-leaf node in the tree hierarchy, can then be viewed as a parent node to the corresponding leaf nodes (physical sessions) and subsequently the root node becomes the parent node for the set of logical server nodes.

Let node $p(i)$ be the parent of node $i$. Then, in order to track the progress of each logical server node $i$ which is a child node to the root node $p(i)$ under the 2L-GPS system, we proceed with the virtual time concept for the root node. Let us assume that the root node $p(i)$ works at the rate 1. Denote as an event each packet arrival and departure from the root 2L-GPS server, and let $t_j$ be the time at which the $j^{th}$ event occurs (simultaneous events are ordered arbitrarily). Let the time of the first arrival of a busy period be denoted as $t_1 = 0$. Now observe that, for each $j = 2,3,\ldots$, the set of logical servers that are busy in the interval $(t_{j-1}, t_j)$ is fixed, and we may denote this set as $B_j$. The root server's virtual time $V_{p(i)}(t)$ is defined to be zero for all times when the root server is idle. Consider any busy period, and let the time that it begins be time zero. Then, $V_{p(i)}(t)$ evolves as follows :

$$V_{p(i)}(0) = 0 \tag{3.1}$$

$$V_{p(i)}(t_{j-1} + \tau) = V_{p(i)}(t_{j-1}) + \frac{\tau}{\sum_{i \in B_j} \phi_i} \tag{3.2}$$

$$\tau \le t_j - t_{j-1}, j = 2,3,\ldots$$

The rate of change of $V_{p(i)}$, namely $\frac{\partial V_{p(i)}(t_j + \tau)}{\partial \tau}$, is $\frac{1}{\sum_{i \in B_j} \phi_i}$, and each backlogged logical server $i$ receives service at rate $\phi_i \frac{\partial V_{p(i)}(t_j + \tau)}{\partial \tau}$. Thus, $V_{p(i)}$ can be interpreted as increasing at the marginal rate at which backlogged logical server $i$ receive service. Now suppose that the $k^{th}$ logical server $i$ packet arrives at time $a_i^k$ and has length $L_i^k$. Then, denote the virtual times at

which this packet begins and completes service as $S_i^k$ and $F_i^k$, respectively. Defining $F_i^0 = 0$ for all $i$, we have

$$S_i^k = max\{F_i^{k-1}, V(a_i^k)\}$$

$$F_i^k = S_i^k + \frac{L_i^k}{\phi_i}$$

Given that node $i$ is a leaf node, i.e., a physical session, it follows that logical node $p(i)$ is the parent of leaf node $i$. In a similar fashion to the notion of virtual time for the root node, we proceed with the notion of virtual time for each logical server node $p(i)$. Note that the notion of virtual time is used by the logical server to track the progress of its underlying physical sessions. Denote as an event each packet arrival and departure from logical server $p(i)$, and let $t_j$ be the time at which the $j^{th}$ event occurs (simultaneous events are ordered arbitrarily). Let the time of the first arrival of a busy period be denoted as $t_1 = 0$. Now observe that, for each $j = 2,3,\ldots$, the set of sessions under logical server $p(i)$ that are busy in the interval $(t_{j-1}, t_j)$ is fixed, and we may denote this set as $B_j$. Logical server $p(i)$'s Virtual time is defined to be zero for all times when the respective logical server is idle. Consider any busy period at logical server $p(i)$, and let the time that it begins be time zero. Then, $V_{p(i)}(t)$ evolves as follows :

$$V_{p(i)}(0) = 0 \tag{3.3}$$

$$V_{p(i)}(t_{j-1} + \tau) = V_{p(i)}(t_{j-1}) + \int_{t_{j-1}}^{\tau} \frac{r_{p(i)}(t)}{\sum_{j \in B_j} \phi_j} dt \tag{3.4}$$

$$\tau \le t_j - t_{j-1}, j = 2,3,\ldots$$

Given that $R(t)$ is the set of logical servers which are backlogged at time $t \in [t_{j-1}, t_j)$., $r_{p(i)}(t) = \frac{\phi_{p(i)}}{\sum_{j \in R(t)} \phi_j}$ is the rate of logical server $p(i)$ at time $t \in [t_{j-1}, t_j)$.

The rate of change of $V_{p(i)}$, namely $\frac{\partial V_{p(i)}(t_j + \tau)}{\partial \tau}$, is $\frac{r_{p(i)}(t)}{\sum_{j \in B_j} \phi_j}$, and each backlogged session $i$ receives service at rate $\phi_i \frac{\partial V_{p(i)}(t_j + \tau)}{\partial \tau}$. Thus, $V_{p(i)}$ can be interpreted as increasing at the

marginal rate at which backlogged sessions under logical server $p(i)$ receive service. Now suppose that the $k^{th}$ session $i$ packet arrives at time $a_i^k$ and has length $L_i^k$. Then, denote the virtual times at which this packet begins and completes service as $S_i^k$ and $F_i^k$, respectively. Defining $F_i^0 = 0$ for all session $i$ under logical server $p(i)$, we have :

$$S_i^k = max\{F_i^{k-1}, V(a_i^k)\}$$

$$F_i^k = S_i^k + \frac{L_i^k}{\phi_i}$$

### 3.1.2 Algorithm

Given the preceeding mechanism for updating virtual time at the level of root server and logical servers, we now embark on describing the 2L-PGPS packet-by-packet approximation algorithm. 2L-PGPS is a a two-level HPFQ which uses one-level PGPS servers as basic building blocks and organizes them in a hierarchical structure.

With the tree representation in Fig.3.1, the root node represents the physical link and a leaf node represents a physical queue from the physical sessions. Each non-root node $n$ is connected to its parent $p(n)$ by a logical queue $Q_n$. For the parent node to implement the PGPS algorithm, only the head of the logical queue is needed. Therefore, at any given time, only the reference to the packet, which is the head of the logical queue, is stored in queue $Q_n$. The actual packet remains stored in the real queue at the leaf node until the link finishes transmission of the packet. For consistency, we also define $Q_R$ for the root server to be the packet currently being transmitted. At any given time when the server is busy, there exists a path from a leaf to the root such that the logical queues of all nodes traversed by the path point to the same physical packet that is currently being transmitted. The logical queues and associated data structures at each node are updated when a packet arrives at an empty session queue at the leaf node, or when the link is picking the next packet to transmit.

Before presenting pseudocode for the algorithm, the following is an overall sketch of how the algorithm runs when either a packet arrives at a leaf node $i$ or a packet departs from the root node.

At the instant when a packet arrives at a leaf node $i$, the procedure Arrive is called. The procedure places the arriving packet into a physical queue $\hat{Q}_i$ at the leaf node $i$, sets the virtual start and finish time for this packet, and moreover calls the function Update-V to update virtual time for the parent of leaf node $i$, i.e, node $p(i)$, in accordance with (3.3) and (3.4). If the packet is at the head of $\hat{Q}_i$, then a reference for this packet is stored at $Q_i$, i.e., the logical queue for leaf node 1. When node $p(i)$, i.e., a logical server node, is idle, the procedure Restart-Node is called. Restart-Node functions as follow. First, it calls Select-Next function which causes logical server $p(i)$ to check up the logical queues of its underlying leaf nodes and moreover select a packet with the earliest virtual time. Furthermore, Restart-Node gives a new virtual start and finish times to the selected packet, and subsequently calls Update-V function to update the virtual time for the root node $p(p(i))$ (i.e., the parent of node $p(i)$) in accordance with (3.1) and (3.2). Finally, Restart-Node places a reference for that selected packet at the logical queue $Q_{p(i)}$ (i.e., the logical queue for node $p(i)$ ). When the root node $p(p(i))$ is idle, the function Restart-Node is called again to find the packet which has the smallest virtual finishing time from one of the underlying logical nodes. Given that a packet is selected, the root node calls Restart-Node function which in turn calls Transmit-Packet-To-Link to transmit the selected packet through the physical link.

At the instant when a packet originating from leaf node $i$ departs from the root node $p(p(i))$, the function Reset-Path is called. Denote $Q_{p(i)}$ and $Q_i$ as the logical queues for node $p(i)$ and leaf node $i$, respectively, and $\hat{Q}_i$ as the physical queue for leaf node $i$. Then, Reset-Path clears out references for the departing packet from $Q_{p(i)}$ and $Q_i$. Furthermore, Reset-Path dequeues the departing packet from physical queue $\hat{Q}_i$. Following the dequeue operations,

reference to the new packet at the head of physical queue $\hat{Q}_i$ is stored at logical queue $Q_i$ and the virtual time at logical server node $p(i)$ is updated. Procedure Restart-Node is then called recursively in order to select and transmit a new packet from the root node.

We now present the pseudocode for the 2L-PGPS algorithm.

Let :

$V_n(t)$ be the system virtual time function for node $n$ ;

$\phi_n$ be the service share for node $n$ ;

$Q_n$ be the logical queue for node $n$ ;

$Q_n(t)$ be the packet at head of $Q_n$ at time $t$ ;

$\hat{Q}_i$ be the real queue for the leaf node $i$ ;

$\hat{Q}_i(t)$ be the packet at head of $\hat{Q}_i$ , at time $t$ ;

$s_n(t)$ be the virtual start time of the packet $Q_n(t)$ ;

$f_n(t)$ be the virtual finish time of the packet $Q_n(t)$ ;

$L_n(t)$ be the length of the packet $Q_n(t)$ ;

$Busy_n(t)$ be true if node $n$ is backlogged at $t$ ;

$p(n)$ be parent node of node $n$ ;

$Leaf(n)$ be true if node $n$ is the leaf node.

ARRIVE($i$, Packet)

1 ENQUEUE($\hat{Q}_i$, *Packet* )
2 if $Q_i(t) = \varnothing$
3 then $Q_i(t) \leftarrow$ Packet
4 $s_i(t) \leftarrow \max (f_i(t), V_{p(i)}(t))$
5 $f_i(t) \leftarrow s_i(t) + \dfrac{L_i(t)}{\phi_i}$
6 UPDATE-$V(p(i))$
7 if $Busy_{p(i)}$ = FALSE
8 then RESTART-NODE($p(i)$)

When a packet arrives at a leaf node $i$, if session $i$'s logical queue for its parent node $Q_i$ is not empty, the packet is just appended to the end of the physical FIFO queue for the session. Otherwise, the packet also becomes the head of the logical queue $Q_i$. The arriving packet is then given its virtual start and finish times. Furthermore, the function Update-V is called in order to update the virtual time for the parent node $p(i)$. Finally, procedure Restart-Node is called with the parent node if it is currently idle.

```
RESTART-NODE (n)
1  m ← SELECT-NEXT(n)
2  if m ≠ ∅
3       then
4            ActiveChild_n ← m
5            Q_n(t) ← Q_m(t)
6            If Busy_n(t) = TRUE
7                 then s_n(t) ← f_n(t)
8                 else s_n(t) ← max (f_n(t), V_p(n)(t))
9            f_n(t) ← s_n(t) + L_n(t) / φ_n
10           Busy_n ← TRUE
11           If (n ≠ R)
12                then UPDATE-V(p(n))
13      else
14           ActiveChild_n ← ∅
15           Busy_n ← FALSE
16    If (n ≠ R) and (Q_p(n)(t) = ∅)
17      then RESTART-NODE(p(n))
18    If (n = R) and (Q_R(t) = ∅)
19      then TRANSMIT-PACKET-TO-LINK(Q_n(t))
```

A node is restarted whenever it needs to select a new packet to transmit. This occurs either when a packet arrives to an idle node or when the last packet finishes being transmitted on the physical link. If a packet arrives at an idle node $n$, the $Busy_n$ flag will be FALSE, in which case the new start time for the node is computed using $s_n(t) = \max(f_n(t), V_{p(n)}(t))$. If the node is not idle and has a packet to transmit, the new start time will be set to the previous finish time. If the node has no more packets to send, the busy flag will be cleared. If the current node is not the root node and moreover its parent does not have a packet in its logical queue $Q_{p(n)}$, the node will restart its parent node. If the current node is the root node and there is a packet in the queue, the packet will be transmitted over the link. Each non-leaf node employs PGPS [1, Section III]. By employing PGPS, each non leaf node uses the smallest virtual finish time policy in order to select a packet to transmit. Upon selecting a packet, it updates its parent's virtual time with respect to the concept of virtual time, depending on whether the node is a logical server or a root server node. These algorithms are implemented in functions Select-Next and Update-$V$.

```
RESET-PATH (n)
1  Q_n(t) ← ∅
2  if Leaf(n) = TRUE
3        then
4           DEQUEUE( Q̂_n )
5           If Q̂_n(t) ≠ ∅
6                then
7                     Q_n(t) ← Q̂_n(t)
8                     UPDATE-V(p(n))
9           RESTART-NODE(p(n))
10       else
11          m ← ActiveChild_n
12          ActiveChild_n ← ∅
```

13 RESET-PATH($m$)

When the link finishes transmitting a packet, it calls Reset-Path($R$). Reset-Path descends the tree along the path leading to the leaf node whose packet just finished transmission. At each node along the path, it resets the logical queue to be empty. When the leaf node is reached, the first packet of the queue is dequeued and its parent node is restarted. During the descent, all pointers are cleared, but not the busy flags. During the process of picking a new packet, the busy flag acts as a reminder to the Restart-Node function that a packet has just finished transmission. If there are no more packets for this node to send, Restart-Node will clear the busy flag.

## 3.2 Lateness Analysis of 2L-PGPS

A natural issue to examine at this point is the lateness of packets under 2L-PGPS. By lateness, we mean how much later packets may depart the system under 2L-PGPS relative to 2L-GPS. Below we present a theorem which applies [1,Theorem 1] to derive packet lateness at each level of the 2L-PGPS, and therefore the sum of these quantities, namely the overall packet lateness in a 2L-PGPS relative to a 2L- GPS. Note that for the analysis of packet lateness, instead of setting $r$=1, we let $r$, i.e., the rate of root server, to be any arbitrary positive number.

**Theorem 3.1** *Let* $\hat{F}_p$ *and* $F_p$ *be the times at which packet p depart under 2L-PGPS and 2L-GPS respectively. For every packet p coming from any session* $i_j$ *under logical server i :*

$$\hat{F}_p - F_p \le \frac{L_{max}}{r}\left(1 + \frac{1}{\phi_i}\right)$$

*where $L_{max}$ is the maximum packet length, r is the rate of the root server, and $\phi_i$ is the GPS weight for logical server i.*

**Proof**. Let us start from the GPS-PGPS case and further extend the lateness result to find the overall lateness for a packet originating from any session $i_j$ under logical server $i$ in the 2L-GPS case.

GPS-PGPS case :

Consider the case where the PGPS scheme is only implemented at each logical server while the root server maintains the GPS service discipline. We denote this case as the GPS-PGPS case.

Since the work discipline at each logical server is PGPS, every logical server $i$ selects a packet from one of its underlying sessions (i.e., leaf nodes) by exercising the smallest virtual finish time policy. Once a packet is chosen, the 2L-PGPS algorithm sets the pointer at the virtual queue between logical server $i$ and root server so that it refers to the chosen packet. Sustaining the GPS work discipline, the root server allows packets from individual logical servers to receive service simultaneously. Consequently, lateness in the GPS-PGPS case is caused by the packet-by-packet transmission performed at the logical servers.

We now show the associated lateness :

From (2.3), the root server with constant rate $r$ guarantees logical server $i$ a rate of $r\phi_i$. Recall that $\phi_i$ is defined as the fraction of rate that is given to logical server $i$ at the instance when every logical server has a positive amount of traffic being queued. With this lower bound on the rate at which logical server $i$ gets serviced, we apply [1,Theorem 1] as follows :

Let us denote $\tilde{F}_p$ and $F_p$ as times at which a packet $p$ departs under GPS-PGPS and 2L-GPS respectively. Then, for every packet $p$ coming from any session $i_j$ under logical server $i$, [1, Theorem 1] asserts that :

$$\tilde{F}_p - F_p \leq \frac{L_{max}}{r\phi_i} \tag{3.5}$$

where $L_{max}$ is the maximum packet length, $r$ is the rate of the root server, and $\phi_i$ is the GPS weight for logical server $i$.

2L-PGPS case :

To arrive at the 2L-PGPS case, we modify the preceeding GPS-PGPS case as follows : Instead of letting the root server operates in the GPS disciplines, let the root server operate in the PGPS discipline. It follows that all servers, i.e., root server and logical servers, employ PGPS. Since the work discipline at the root server is PGPS, the root server selects packets from its underlying logical servers by exercising the smallest virtual finish time policy. Consequently, lateness in 2L-PGPS relative to GPS-PGPS is caused by the packet-by-packet transmission performed at the root server. With $r$ fixed as a constant rate of the root server, we apply [1,Theorem 1] as follows :

Let us denote $\tilde{F}_p$ and $\hat{F}_p$ as times at which a packet $p$ departs under GPS-PGPS and 2L-PGPS, respectively. Then, for any packet $p$ :

$$\hat{F}_p - \tilde{F}_p \leq \frac{L_{max}}{r} \tag{4.6}$$

where $L_{max}$ is the maximum packet length and $r$ is the rate of the root server.

Combining (4.5) and (4.6), the statement of Theorem 3.1 follows. □

Let $S_{ij}(\tau,t)$ and $\hat{S}_{ij}(\tau,t)$ be the amount of session $i_j$ traffic (in bits, not packets) served under 2L-GPS and 2L-PGPS in the interval $[\tau,t]$.

**Theorem 3.2** *For all times $\tau$ and session $i_j$ :*

$$S_{ij}(0,\tau) - \hat{S}_{ij}(0,\tau) \le L_{max}\left(1 + \frac{1}{\phi_i}\right)$$

**Proof**. The slope of $\hat{S}_{ij}$ alternates between $r$ when a session $i_j$ packet is being transmitted, and 0 when session $i_j$ is not being served. Since the slope of $S_{ij}$ also obeys these limits, the difference $S_{ij}(0,t) - \hat{S}_{ij}(0,t)$ reaches its maximal value when session $i_j$ packets begin transmission under 2L-PGPS. Let $t$ be some such time, and let $L$ be the length of the packet going into service. Then, the packet completes transmission at time $t + \frac{L}{r}$. Let $\tau$ be the time at which the given packet completes transmission under 2L-GPS. Then, since session $i_j$ packets are served in the same order under both schemes,

$$S_{ij}(0,\tau) = \hat{S}_{ij}(0,t + \frac{L}{r})$$

From Theorem 3.1,

$$\tau \ge (t + \frac{L}{r}) - \frac{L_{max}}{r}\left(1 + \frac{1}{\phi_i}\right)$$

$$\Rightarrow S_{ij}(0,t + L - L_{max}\left(1 + \frac{1}{\phi_i}\right)) \le \hat{S}_{ij}(0,t + \frac{L}{r})$$

$$= \hat{S}_{ij}(0,t) + L$$

Since the slope of $S_{ij}$ is at most $r$, the theorem follows. □

Let $\hat{Q}_{ij}(\tau)$ and $Q_{ij}(\tau)$ be the session $i_j$ backlog (in units of traffic) at time $\tau$ under 2L-PGPS and GPS, respectively. Then it immediately follows from Theorem 3.2 that :

**Corollary 3.1**. *For all times $\tau$ and session $i_j$*

$$\hat{Q}_{ij}(0,\tau) - Q_{ij}(0,\tau) \le L_{max}\left(\frac{1}{\phi_i} + 1\right)$$

With Theorem 3.1 and Corollary 3.1, we can now translate bounds on 2L-GPS worst-case packet delay and backlog to corresponding bounds on 2L-PGPS.

# Chapter 4

# Integrating Best-Effort and QoS Classes of Sessions

This chapter centers around a policy that aims at maximizing bandwidth available to the best-effort traffic while satisfying guarantees of the QoS traffic. The motivation stems from the widely accepted observation that the best-effort service class plays an increasingly significant role in any large scale network of the future.

In Chapter 2, the analytical framework for a single-node 2L-GPS in which $\rho_{ij} \leq \phi_{ij}$ for all session $i_j$ was developed. In this chapter, we shall utilize such a work discipline as a platform, on top of which a policy to improve performance, i.e., worst-case queuing delay and backlog, for the best-effort service class is proposed.

We present the grouping policy in the first section. In the second section we show that the grouping policy improves performance of the best-effort sessions. In the following section we conjecture that aiming at such performance improvement does not adversely affect guarantees for the QoS class of sessions. In the final section we compute bounds on packet lateness and the difference in quantity of backlog for every session in a 2L-PGPS node relative to a 2L-GPS node when the grouping policy is applied. Subsequently, the results, i.e., bounds on packet lateness and the difference in quantity of backlog, are used to provide minor modifications to the analysis in [3, ch. 4] so that end-to-end session delay in the 2L-PGPS

network (assuming the grouping policy is applied) can be computed in a straight-forward manner.

## 4.1 A Grouping Policy

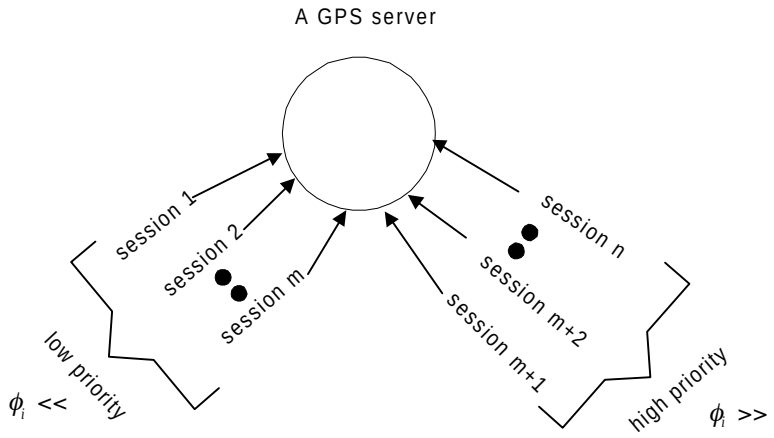
The most natural approach to integrate the provision of both QoS and best-effort service classes in a single-node 2L-GPS environment is to assign the 2L-GPS weights so that the best-effort sessions are given low priorities, i.e., low $\phi_{ij}$'s, while the QoS sessions are given high priorities, i.e., high $\phi_{ij}$'s. Adopting such an approach, we present a policy to best integrate the provision of both QoS and best-effort service classes within the single-node 2L-GPS environment. Note that, throughout the development of this chapter, the 2L-GPS weights are assumed to be such that $\phi_{ij} \geq \rho_{ij}$ for all $i,j$.

Given a set of $n$ sessions which includes low priority as well as high priority sessions, we begin by organizing the sessions in two groups, that is, by labelling low priority sessions as session 1 to $m$ and high priority sessions as session $m$+1 to $n$. Furthermore, we assume that the stability condition is satisfied, i.e., $\alpha < 1$ (see Lemma 2.3).

For each of the high priority sessions, let each of the sessions, i.e., session $m$ + 1 to $n$, be served directly by the root server. For all of the low priority sessions, we allocate a logical server, i.e., logical server 1 (without loss of generality), to serve these sessions (i.e., session 1,2,..,$m$). Following such assignment, session 1 is relabelled as session $1_1$ (i.e., the first session under logical server 1), session 2 as session $1_2$, and session $i$ ($1 \leq i \leq m$) as session $1_i$ .

Fig.4-1 represents the diagram which illustrates the assignment policy for both low and high priority sessions under the 2L-GPS environment. Such an assignment policy is called *the grouping policy*.

**Without a Grouping Policy :**




**With a Grouping Policy :**

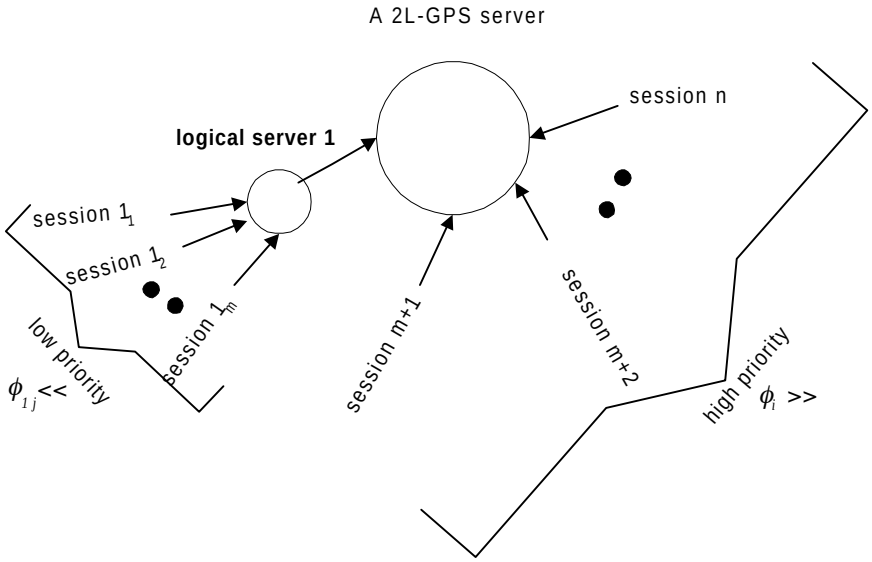



Figure 4-1: Ungrouping (GPS) vs Grouping (2L-GPS)

The leaky bucket constraints (assuming infinite link capacity) for the each of the low priority sessions, i.e., sessions $1_j$'s, arrival function is :

$$A_{1j} \sim (\sigma_{1j}, \rho_{1j}) \text{ for } j = 1,2,\ldots,m$$

It follows that the virtual leaky bucket constraint for logical server 1 is :

$$A_1 \sim (\sigma_1, \rho_1) \text{ with } \sigma_1 = \sum_{j=1}^{m} \sigma_{1j} \text{ and } \rho_1 = \sum_{j=1}^{m} \rho_{1j}$$

For each of the high priority sessions, i.e., session $m$+1 to $n$, the leaky bucket constraint (assuming infinite link capacity) is :

$$A_i \sim (\sigma_i, \rho_i) \text{ for } i = m+1, m+2, \ldots, n$$

Throughout the rest of this chapter, the 2L-GPS weights are henceforth assumed to be such that $\phi_{1j} \geq \rho_{1j}$ for $j$ = 1,2,...,$m$ and $\phi_i$ is an arbitrary positive number for $i = m+1, m+2, \ldots, n$. Note that from (2.2) :

$$\sum_{j=1}^{m} \phi_{1j} + \sum_{i=m+1}^{n} \phi_i = 1$$

In the following section, we compare performances of each session in the case when sessions are served by a non hierarchical GPS node and do not undergo grouping, with the case when sessions are served by a 2L-GPS node and grouped according to the grouping policy.

## 4.2 Comparing Sessions' Performances : Grouping vs Non-Grouping

For the purpose of comparing performances of sessions under grouping and without grouping, we define two classes of session, $H$ and $L$. Let $H$ and $L$ be defined as the set of high priority and low priority sessions, respectively.

When grouping is applied (under 2L-GPS server mechanism), the set of high priority sessions is:

$$H = \{\text{session } i\text{: } i = m+1, m+2, \ldots, n\}$$

and the set of low priority sessions is:

$$L = \{\text{session } i_j : i = 1 \text{ and } j = 1,2,..,m\}$$

Conversely, in the case when no grouping is applied (under GPS server mechanism), the set of high priority sessions is:

$$H = \{\text{session } i : i = m+1, m+2, \ldots, n\}$$

and the set of low priority session is:

$$L = \{\text{session } i : i = 1,2,\ldots,m\}$$

To avoid ambiguity, we shall name sessions according to their names in the case when no grouping is applied. For $i = 1,2,\ldots,m$ , session $i$ (i.e., a low-priority session) refers to session $i$ under no grouping and correspondingly session $1_i$ under grouping. For $i = m+1,\ldots,n$ , a session $i$ (i.e., a high-priority session) refers to session $i$ under no grouping and correspondingly session $i$ under grouping.

Furthermore, let $S_i^0(0,t)$ be defined as session $i$'s greedy service function at the interval $[0.t)$ when no grouping is applied (i.e., the service function attained when sessions are greedy from time 0 under the GPS scheme). Analogously, let $S_i^{0,G}(0,t)$ be defined as session $i$'s greedy service function at the interval $[0.t)$ when grouping is applied (i.e., the service function attained when sessions are greedy from time 0 under the 2L-GPS scheme). Recall that every session is greedy starting from time 0 when :

$$A_i^0(0,t) = \sigma_i + \rho_i t \quad \forall i \; \forall t > 0 .$$

Finally, let $e^G$ and $e$ be the instants of time when the last session in $L$ gets unbacklogged, given that every session is greedy starting from time 0, under grouping and no grouping, respectively.

Having set the above definitions, we now present the following Lemma :

**Lemma 4.1** *For all $i \in H$ and for all $t \in (0, e^G]$ :*

$$S_i^{0,G}(0,t) \le S_i^0(0,t)$$

**Proof.** Let us observe the dynamics of both 2L-GPS (with grouping) and GPS (without grouping) servers in which all the sessions are greedy starting at time 0. Furthermore, we define two types of instant, $t_1$ and $t_2$. Denote $t_1$ as the type of instant when a backlogged session $j \in L$ gets unbacklogged but logical server 1 remains unbacklogged, and $t_2$ as the type of instant when a session $j \in H$ becomes unbacklogged.

The slope of $S_i^{0,G}$ for a backlogged session $i \in H$, under the grouping scheme, at the instant of type $t_1$, remains as it was at $t_1^-$, since an unbacklogged session $j \in L$ is a second-level session. The excess rate which is available after the particular instant of type $t_1$ is only shared by sessions in $L$ (i.e., the second-level sessions) which are still backlogged at time $t_1$. Conversely, in the case of no grouping, every session $i$ is a first-level session and hence the slope of $S_i^0$ for a backlogged session $i \in H$, i.e, a first-level session, increases at the instant of type $t_1$.

At the instant of type $t_2$, the slope of $S_i^{0,G}$ for a backlogged session $i \in H$, under the grouping policy, increases. We hypothesize that such increase is *at most the same* as that in the case of no grouping. To see why the hypothesis is true, observe that $t_2$ is the type of instant when a session $j \in H$ gets unbacklogged. Under the grouping policy, before a session $j \in H$ gets unbacklogged at the instant of type $t_2$, one or more or perhaps no instants of type $t_1$ may occur. Thus, the share of excess rate which is given by such session $j$ to a backlogged session $i \in H$ at the instant of type $t_2$ is never greater than that which is received by session $i \in H$ under no grouping, concluding that the hypothesis is true.

The instant when a session $i \in H$ goes unbacklogged, with or without grouping, must be preceeded by instants of type $t_1$ or (and) $t_2$ or none of them ( i.e., when session $i \in H$ is the first to unbacklog). Each possible combination of instants that a session $i \in H$ undergoes

before it gets unbacklogged causes the slope of $S_i^{0,G}(0,t)$ to be at most the same as the slope of $S_i^0(0,t)$ for $\forall\, i \in H$ and any $t \in (0,\ e^G]$, completing the proof. □

**Lemma 4.2** For any $t > 0$:

$$\sum_{i \in L} S_i^{0,G}(0,t) \geq \sum_{i \in L} S_i^0(0,t)$$

**Proof**. From Lemma 4.1, for $i \in H$ and any time $t \in (0, e^G]$ : $S_i^{0,G}(0,t) \leq S_i^0(0,t)$. Summing over all $i \in H$, we have :

$$\sum_{i \in H} S_i^{0,G}(0,t) \leq \sum_{i \in H} S_i^0(0,t) \quad \text{for any } t \in (0,e^G] \tag{4.1}$$

Also, the workconserving property of GPS and 2L-GPS server implies :

$$\sum_{i \in H} S_i^0(0,t) + \sum_{i \in L} S_i^0(0,t) = \sum_{i \in H} S_i^{0,G}(0,t) + \sum_{i \in L} S_i^{0,G}(0,t) \text{ for any } t \in (0,e^G] \tag{4.2}$$

since $H \cup L$ is the set of all sessions entering the server. Substituting (4.1) to the rhs of (4.2) and rearranging terms,

$$\sum_{i \in L} S_i^{0,G}(0,t) \geq \sum_{i \in L} S_i^0(0,t) \quad \text{for any } t \in (0,e^G] \tag{4.2a}$$

In the case of grouping, every session $i \in L$ must already be unbacklogged at time $t > e^G$. (4.2a) shows that it may not be the case when the grouping policy is not applied, thus completing the proof. □

Let $B(t)$ and $B^G(t)$ be the set of sessions in $L$ which are backlogged before or at time $t$ in an all greedy regime, under no grouping and grouping, respectively. Analogously, let $U(t)$ and $U^G(t)$ be the set of sessions in $L$ which are unbacklogged before or at time $t$ in an all greedy regime, under no grouping and grouping, respectively. Furthermore, let $e_i^G$ and $e_i$ be the time at which session $i$ becomes unbacklogged under grouping and no grouping, respectively. By

definitions of $U^G(t)$ and $U(t)$, and Lemma 4.2, it follows that $e_j^G \le e_j \le t$, for all session $j \in U^G(t)$ under the grouping policy.

Let us define $S_L^{0,G}(0,t)$ and $S_L^0(0,t)$ such that :

$$S_L^{0,G}(0,t) = \sum_{i \in L} S_i^{0,G}(0,t) \text{ and } S_L^0(0,t) = \sum_{i \in L} S_i^0(0,t)$$

Now, we show that :

**Lemma 4.3** *Under the grouping policy, if a session $i \in L$ is backlogged in the interval (0,t] :*

$$S_i^{0,G}(0,t) \ge \frac{\phi_i}{\sum_{j \in B(t)} \phi_j} (S_L^{0,G}(0,t) - \sum_{j \in U(t)} \sigma_j + \rho_j t)$$

**Proof.** Since the total number of low priority sessions is $m$, it follows :

$$m = B^G(t) + U^G(t) \tag{4.3}$$

Given that sessions are in an all greedy regime, we suppose $|B^G(t)| > |B(t)|$. It implies that, at time $t$, there are strictly fewer sessions backlogged under no grouping policy than under the grouping policy. Thus, at time $t$, the amount of service received by the aggregate of all low priority sessions under the no grouping policy is strictly greater than that under the grouping policy. This is a contradiction, since by Lemma 4.2, $S_L^{0,G}(0,t) \ge S_L^0(0,t)$ $\forall t \ge 0$ and therefore :

$$|B^G(t)| \le |B(t)| \tag{4.4}$$

Substituting (4.4) to (4.3), it also follows :

$$|U^G(t)| \ge |U(t)|$$

Let us denote the set $D(t)$ such that :

$$D(t) = B(t) - B^G(t) = U^G(t) - U(t)$$

Further, define $s_i$ as the $i^{\text{th}}$ session that gets unbacklogged from the set $D(t)$.

Using the definition of 2L-GPS, if a session $i \in L$ (under the grouping policy) is backlogged in the interval $(0,t]$, then :

$$S_i^{0,G}(0,t) = \frac{\phi_i}{\sum_{j \in B(t)} \phi_j}\left(S_L^{0,G}(0,t) - \sum_{j \in U(t)} \sigma_j + \rho_j t\right) + \sum_{k=1}^{|D(t)|} x_k(t)$$

where :

$$x_k(t) = \frac{\phi_i}{\sum_{j \in B(t) - \{s_1, s_2, \ldots, s_k\}} \phi_j}\left(\frac{\phi_{s_k}}{\sum_{k \in B(t) - \{s_1, s_2, \ldots, s_{k-1}\}} \phi_k}\left(S_L^{0,G}(0,t) - \sum_{j \in U(t) + \{s_1, s_2, \ldots, s_{k-1}\}} (\sigma_j + \rho_j t)\right) - (\sigma_{s_k} + \rho_{s_k} t)\right), \forall k = 1, .., |D(t)| \quad (4.5)$$

From (4.5), it follows that $\sum_{k=1}^{|D(t)|} x_k(t) \geq 0$ and the proof is completed. □

**Lemma 4.4** *For all $i \in L$ and for all $t \geq 0$ :*

$$S_i^{0,G}(0,t) \geq S_i^0(0,t)$$

**Proof.** From Lemma 4.3 :

$$S_i^{0,G}(0,t) \geq \frac{\phi_i}{\sum_{j \in B(t)} \phi_j}\left(S_L^{0,G}(0,t) - \sum_{j \in U(t)} \sigma_j + \rho_j t\right) \quad (4.6)$$

when session $i \in L$ is backlogged in the interval $(0,t]$ under no grouping. By definition of GPS (see [1, eq. 1]) :

$$S_i^0(0,t) = \frac{\phi_i}{\sum_{j \in B(t)} \phi_j}\left(S_L^0(0,t) - \sum_{j \in U(t)} \sigma_j + \rho_j t\right), \quad (4.7)$$

when session $i \in L$ is backlogged in the interval $(0,t]$ under no grouping. Combining (4.7) and (4.6),

$$\begin{aligned} S_i^{0,G}(0,t) &\geq \frac{\phi_i}{\sum_{j \in B(t)} \phi_j}\left(S_L^{0,G}(0,t) - \sum_{j \in U(t)} \sigma_j + \rho_j t\right) \\ &\geq \frac{\phi_i}{\sum_{j \in B(t)} \phi_j}\left(S_L^{G}(0,t) - \sum_{j \in U(t)} \sigma_j + \rho_j t\right) \\ &= S_i^0(0,t) \end{aligned} \quad (4.7a)$$

when session $i \in L$ is backlogged in $(0,t]$ under grouping as well as no grouping. The second inequality follows from Lemma 4.2. (4.7a) implies $e_i^G \le e_i$, completing the proof. □

Now, let $D_i^*$ and $D_i^{G,*}$ be the the maximum delay of session $i$ when sessions are greedy from time zero, under no grouping and grouping, respectively, i.e.,

$$D_i^* = \max_{t \ge 0} D_i^0(t),$$

$$D_i^{G,*} = \max_{t \ge 0} D_i^{0,G}(t) .$$

In a similar manner, let $Q_i^*$ and $Q_i^{G,*}$ be the the maximum backlog of session $i$ when all sessions are greedy from time zero, under no grouping and grouping, respectively. i.e.,

$$Q_i^* = \max_{t \ge 0} Q_i^0(t),$$

$$Q_i^{G,*} = \max_{t \ge 0} Q_i^{0,G}(t)$$

**Lemma 4.5**

(i) *Under no grouping :* $D_i^*, Q_i^*$ *are session i's worst-case delay and backlog respectively*

(ii) *Under grouping :* $D_i^{G,*}, Q_i^{G,*}$ *are session i's worst-case delay and backlog respectively*

**Proof.**

(i) $S_i^0(0,t)$ is session $i$'s greedy service function at the interval $[0,t)$ when no grouping is applied (i.e., the service function attained when sessions are greedy from time 0 under the GPS scheme). Using [1, Lemma 10 and 11], the statement of Lemma 4.5 (i) follows. □

(ii) $S_i^{0,G}(0,t)$ is the session $i$'s greedy service function at the interval $[0,t)$ when grouping is applied (i.e., the service function attained when sessions are greedy from time 0 under the 2L-GPS scheme). Using Lemma 2.7 and 2.8, the statement of Lemma 4.5 (ii) follows. □

**Theorem 4.1** *For all* $i \in L$ *:*

$$\text{(i)} \quad Q_i^{G,*} \le Q_i^*$$

$$\text{(ii)} \quad D_i^{G,*} \le D_i^*$$

**Proof.**

(i) From Lemma 4.4,

$$S_i^{0,G}(0,t) \ge S_i^0(0,t), \forall t \ge 0$$

Also,

$$A_i^{0,G}(0,t) = A_i^0(0,t) = \sigma_i + \rho_i t$$

Thus,

$$A_i^{0,G}(0,t) - S_i^{0,G}(0,t) \le A_i^0(0,t) - S_i^0(0,t)$$

i.e.,

$$Q_i^{0,G}(t) \le Q_i^0(t), \ \forall t \ge 0$$

□

(ii) The case for delay is similar. Let $t$ be any time in session $i$'s busy period, then from the definition of delay in Section 2.3.2 of Chapter 2 :

$$A_i^{0,G}(0,t) - S_i^{0,G}(0,t + D_i^{0,G}(t)) = 0$$

From Lemma 4.4,

$$S_i^0(0,t + D_i^{0,G}(t)) \le S_i^{0,G}(0,t + D_i^{0,G}(t))$$

also,

$$A_i^{0,G}(0,t + D_i^{0,G}(t)) = A_i^0(0,t + D_i^{0,G}(t)) = \sigma_i + \rho_i(t + D_i^{0,G}(t))$$

Thus,

$$0 = A_i^{0,G}(0,t) - S_i^{0,G}(0,t + D_i^{0,G}(t)) \le A_i^0(0,t) - S_i^0(0,t + D_i^{0,G}(t))$$

$$\Rightarrow D_i^{0,G}(t) \le D_i^0(t)$$

□

## 4.3 A Conjecture

From Theorem 4.1 and Lemma 4.5, we summarize that the grouping policy improves performances of the low-priority sessions. A question to ask ourselves at this point is how degraded the high-priority sessions are, in term of their performances, when the grouping policy is applied. Below, we propose a conjecture which attempt to give an answer for the question.

**Conjecture 4.1** *The Grouping policy improves performances of the best-effort (i.e., low-priority) class of sessions while still satisfying performance guarantees for the QoS (i.e., high-priority) class of sessions.*

**Arguments.** Every best-effort session $i \in L$ has a low priority, i.e., low $\phi_i$. Conversely, every best-effort session $j \in H$ has a high priority, i.e., high $\phi_j$. In a gigabit or terabit networking environment, we can loosely say that the traffic from a high-priority session is served at a rate that is several order of magnitudes higher than that of a low-priority session, i.e., $\phi_j >> \phi_i$ , $\forall i \in L$ and $\forall j \in H$. With this assumption, we will show that :

$$S_i^{0,G}(0,t) - S_i^0(0,t) \approx \varepsilon \quad \text{, for all session } i \in H, \text{ some small } \varepsilon \geq 0, \text{ and } \forall t \geq 0 \qquad (4.8)$$

Given every session is greedy from time zero and the grouping policy is exercised, we shall turn our attention to the greedy service function, that is, $S_i^{0,G}(0,t)$, for all $i$. Under the grouping policy, the slope of $S_i^{0,G}$, for every backlogged session $j \in L$, increases at the instant when a session $i \in \{L$ - the last session to unbacklog in $L\}$ is unbacklogged. At that same instant, however, no increase of slope is attributed to $S_k^{0,G}$ for any backlogged session $k \in H$, since the mechanics of 2L-GPS (see Section 2.2 of Chapter 2) suggests that the excess rate is only shared among the backlogged sessions in $L$.

On the other hand, the mechanics of GPS, in which no grouping policy is exercised, suggests that a positive increase of slope is given to the greedy service function (i.e., $S_j^0$ ) of every backlogged session $j$, at the instant when a session $i \in L$ is unbacklogged. However, giving in to the assumption that $\phi_{i \in L} << \phi_{j \in H}$, we conclude that the positive increase of slope given to greedy service curve (i.e., $S_j^0$ ) of a backlogged session $j \in H$ upon having a session $i \in L$ unbacklogged, under no grouping, is negligible.

In both GPS (no grouping) and 2L-PGPS (grouping) schemes, a backlogged session $j \in H$ receives a negligible increase in service rate when any session $i \in L$ unbacklogs. Thus, we may conjecture that (4.8) is true. By using similar lines of proof in Theorem 4.1, it is not difficult to show that performances of any session $i \in H$, under grouping, tend to be equal with its performances in the case of no grouping. □

With Conjecture 4.1, we have proposed the grouping policy as a strategy to aim at improving performances of the best-effort sessions while satisfying the guarantees for the QoS sessions.

## 4.4 Packet-by-Packet Two-Level Generalized Processor Sharing Network Under The Grouping Policy

In this section we compute packet lateness and the difference in quantity of backlog for a session in a 2L-PGPS node relative to that in a 2L-GPS node when the grouping policy is applied. Subsequently, we use the results to provide minor modifications so the worst-case end-to-end session delay in a 2L-PGPS network, where the grouping policy is applied, can be computed by employing the analysis in [3, Chapter 4] in a straight-forward manner.

### 4.4.1 Packet Lateness for Two-Level Packet-by-Packet Generalized Processor Sharing under Grouping Policy

The packet-by-packet transmission scheme for 2L-GPS under the grouping policy is 2L-PGPS. Let $\hat{F}_p$ and $F_p$ be the times at which packet $p$ depart under 2L-PGPS and 2L-GPS, respectively. Furthermore, let $\hat{Q}_i(\tau)$ and $Q_i(\tau)$ be the session $i$'s backlog (in units of traffic) at time $\tau$ under 2L-PGPS and 2L-GPS, respectively. We show that :

**Corrollary 4.1** *For all packets p sent from session* $i \in H$ *:*

$$(i)\quad \hat{F}_p - F_p \le \frac{L_{max}}{r}$$

$$(ii)\quad \hat{Q}_i(0,\tau) - Q_i(0,\tau) \le L_{max}$$

**Proof**. A high priority session $i \in H$ is served directly by the root node in the 2L-GPS scheme under the grouping policy. Applying [1, Theorem 1], the statement of corrollary (4.1(i)) follows. Subsequently, by applying [1,Theorem 2], the statement of Corrollary 4.1(ii) follows. □

**Corrollary 4.2** *For all packets p sent from session* $i \in L$ *:*

$$(i)\quad \hat{F}_p - F_p \le \frac{L_{max}}{r}\left(1 + \frac{1}{\phi_1}\right)$$

$$(ii)\quad \hat{Q}_i(0,\tau) - Q_i(0,\tau) \le L_{max}\left(1 + \frac{1}{\phi_1}\right)$$

**Proof**. A low priority session $i \in L$ is served by the logical server 1 in the 2L-GPS scheme under the grouping policy. Applying Theorem 3.1, the statement of Corrollary 4.2(i) follows. Subsequently, by applying Corrollary 3.1, the statement of Corrollary 4.2(ii) follows. □

### 4.4.2 A Note on Worst-Case End-to-End Session Delay in 2L-PGPS Network under Grouping Policy

We assume the reader is familiar with [2], in particular with the notations for the network model and the analysis of worst-case end-to-end delay for a session in the PGPS network. In this subsection we will only discuss the necessary modifications so that all the results in [2] apply in a straight forward manner to the analysis of 2L-PGPS network under the grouping policy.

We assume that the network of 2L-PGPS servers under grouping policy has such weight assignments such that $\phi_1^k$ (i.e., the 2L-GPS weight for logical server 1 at node $k$) and $\phi_j^k$ (i.e., the 2L-GPS weight for first-level session $i$ at node $k$), for $i = m+1, m+2,\ldots,n$, follow a *Consistent Relative Session Treatment* in which the condition for stability in the network (see [2, Theorem 2]) is satisfied.

We know from Corrollary 4.1(ii) and Corrollary 4.2(ii) that :

$$\hat{Q}_i^1(0,\tau) - Q_i^1(0,\tau) \le L_{max} \quad \forall i \in H$$

$$\hat{Q}_i^1(0,\tau) - Q_i^1(0,\tau) \le L_{max}\left(1 + \frac{1}{\phi_1^1}\right) \quad \forall i \in L$$

for all $\tau$ where $r^m$ represents the rate of root server at node $m$, and $\hat{Q}_i^m$ and $Q_i^m$ represent session $i$'s backlog at node $m$, under 2L-PGPS and 2L-GPS, respectively. Thus,

$$\hat{Q}_i^{1,*} \le Q_i^{1,*} + L_{max} \quad \forall i \in H \tag{4.9}$$

$$\hat{Q}_i^{1,*} \le Q_i^{1,*} + L_{max}\left(1 + \frac{1}{\phi_1^1}\right) \quad \forall i \in L \tag{4.10}$$

where $\hat{Q}_i^{m,*}$ and $Q_i^{m,*}$ represent the maximum backlog for session $i$ at node $m$, under 2L-PGPS and 2L-GPS, respectively.

Also, from [1,Lemma 12] and Lemma 2.9 :

$$\sigma_i^{out} = Q_i^* \quad \forall i \in L \cup H \tag{4.11}$$

From Theorem 4.1 and (4.13), we note that, in the case of grouping, the output burstiness for each low-priority session at each node along its path is less than that in the case of no grouping. Conjecture 4.1 further claims that such improvement in the case of grouping does not increase the output burstiness of each high priority session, namely the output burstiness of each high priority session is relatively the same under both grouping and no grouping

Let $P(i)$ be a path taken by session $i$ in the graph of the 2L-PGPS network, then (4.9), (4.10) and (4.11) allow one to characterize the internal traffic at each node in $P(i)$ using essentially the same procedure in [2,Section VII]. If applying the greedy bound to a node $m \in P(i)$ (assuming 2L-GPS service) yields a bound of $\sigma_i^{out,m} = \alpha$, then the bound on this quantity under 2L-PGPS is just $\alpha + L_{max}$ for $\forall i \in H$ or $\alpha + L_{max} \frac{1}{\phi_1^m}$ for $\forall i \in L$.

The next step is to analyze delay along session $i$'s route. The following is a corrollary that allows us to relate worst-case session delay in the 2L-PGPS network to the universal service curve for the 2L-GPS network. Let :

$\hat{D}_i^{*,end-to-end}$ be the worst-case end-to-end delay for session $i$ in the 2L-PGPS network,

$G_i^K$ be the universal service curve for session $i$ and furthermore is computed using the procedure given in [2, Session X],

$r^m$ be the rate of the $m^{\text{th}}$ root server in the path taken by session $i$,

$A_i^0$ be the greedy arrival function at the node where session $i$ enters the network (i.e., the source node for session $i$)

$K$ be the total number of nodes in session $i$'s end-to-end path,

$L_i$ be the maximum length of a session $i$'s packet.

Now, we show that :

**Corollary 4.3** *For a session i in the 2L-PGPS network under grouping policy :*

*(i)* $\forall i \in H : \hat{D}_i^{*,end-to-end} \le \max_{\tau \ge 0}\{\min\{t: G_i^K(t) = A_i^0(0,\tau) + (K-1)L_i\} - \tau\} + \sum_{m=1}^{K} \frac{L_{max}}{r^m}$

*(ii)* $\forall i \in L : \hat{D}_i^{*,end-to-end} \le \max_{\tau \ge 0}\{\min\{t: G_i^K(t) = A_i^0(0,\tau) + (K-1)L_i\} - \tau\} + \sum_{m=1}^{K} \frac{L_{max}}{r^m}(1 + \frac{1}{\phi_1^m})$

**Proof**. Taking (4.9), (4.10), and (4.11) to characterize the internal traffic of session *i* at each node that it passes along its path in the 2L-PGPS network and subsequently using Corollary 4.1(i) when $i \in H$ and Corollary 4.2(i) when $i \in L$ , the proof of [2, Theorem 8] is applicable in a straight forward manner to show that the statements of Corollary 4.3(i) and 4.3(ii) follow. □

Note that the 2L-GPS network being considered here has internal characterization *identical* to the 2L-PGPS network which will in general have more bursty traffic. It is interesting to observe that as the link speeds become faster, i.e., as $r^m \to \infty$,

$$\hat{D}_i^{*,end-to-end} = D_i^{*,end-to-end}$$

Furthermore, Corollary 4.3(ii) provides an interesting observation, in that the bound on packet lateness incurred by a low priority session's packet at node *k*, $\forall k \in P(i)$, is greater by a multiplicative factor of $(1 + \frac{1}{\phi_1^k})$ than that of a high priority session's packet at the respective node. It follows that the packet lateness for each low priority session can be reduced by increasing $\phi_1^k$. From (2.1), increasing $\phi_1^k$ implies letting the number of lower priority sessions under node *k*'s logical server 1, i.e., $m^k$, to be as high as possible.

# Chapter 5

# Summary of the Contributions

This thesis investigates the necessary refinements for the seminal PGPS mechanism combined with Leaky Bucket admission control [1] with the objective of maximizing bandwidth available to the best-effort traffic without adversely affecting the guarantee of the QoS traffic.

The contributions which follow from the investigation are as follow. We propose a multiplexing scheme, i.e, 2L-GPS (Two-Level Generalized Processor Sharing), which extends the GPS multiplexing scheme introduced in [1] from its original one-level multiplexing architecture to a two-level architecture. Subsequently, we show that it is possible to compute performance guarantees for sources constrained by leaky buckets in a 2L-GPS server, if the weight assignment to each second-level session is such that that $\phi_{ij} \geq \rho_{ij}$ for all $i$ and $j$.

We then propose 2L-PGPS (Two-Level Packet-by-Packet Generalized Processor Sharing), which is a packet-by-packet based algorithm that approximates 2L-GPS even when the packets are of variable length. We analyze the lateness incurred by each session when 2L-PGPS is used to approximate the 2L-GPS scheme. The results from lateness analysis allow us to translate session delay and buffer requirement bounds derived for a 2L-GPS server system to a 2L-PGPS server system.

Finally, we propose a grouping policy which integrates the provision of both QoS and best-effort service classes within the 2L-GPS multiplexing scheme. We show that the grouping policy improves performance (i.e, worst-case queuing delay and backlog) of the best-effort sessions. We further conjecture that aiming at such improvement in performances does not adversely affect guarantees for the QoS sessions. Finally, we provide minor modifications to the analysis of the multiple-node case in [2] so that the worst-case end-to-end session delay in the 2L-PGPS network, where the grouping policy is applied, can be computed by applying the analysis in [3, ch. 4] in a straight-forward manner.

# Bibliography


[1] A.K. PAREKH AND R. GALLAGER, *A Generalized processor sharing approach to flow control - The single node case*, IEEE/ACM Trans. Networking, 1 (1993), pp. 344-357.

[2] A.K. PAREKH AND R. GALLAGER, *A Generalized processor sharing approach to flow control - The multiple node case*, IEEE/ACM Trans. Networking, 2 (1994), pp. 137-150.

[3] A.K. PAREKH, *A generalized processor sharing approach to flow control in integrated services networks*, Ph.D. dissertation, Department of Electrical Engineering and Computer Science, MIT, February 1992.

[4] J.C.R. BENNETT AND H. ZHANG, *Hierarchical packet fair queueing algorithms*, Proceedings of IEEE INFOCOM'96, pp. 120-128.

[5] R. GALLAGER. Private Communication, April 2000.

[6] D. BERTSEKAS AND R. GALLAGER, *Data Networks*, Prentice Hall, Englewood Cliffs, NJ, 1991.

[7] J. TURNER, *New directions in communications (or Which way to the information age)*, IEEE Communications Magazine, 24 (1986), pp. 8-15.