Tracking Switch Fluid Policies: Bounding Lookahead

by

Michael John Girone

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2002

Author .....
Department of Electrical Engineering and Computer Science
January, 31 2002

Certified by .....
Vahid Tarokh
Associate Professor of Electrical Engineering and Computer Science
Thesis Supervisor

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

Tracking Switch Fluid Policies:

Bounding Lookahead

by

Michael John Girone

Submitted to the Department of Electrical Engineering and
Computer Science
on January, 31 2002, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

Media-access arbitration policies capable of guaranteeing a wide range of qualities of service (QoSs) will play a vital role in future systems. Such systems include integrated services networks and advanced wireless networks. A general approach for developing these policies begins by first considering media access as infinitesimally divisible, or fluid. Fluid policies are convenient for design and analysis but are often not realizable. The second step is to develop a granular, and therefore realizable, method of approximating the fluid policy. Approximating some fluid policies requires lookahead, knowledge of the policies' future behavior. We prove a lower bound of order N on the lookahead required for granular policies to adequately approximate, or track, fluid policies. We also present preliminary results in our study of an upper bound on lookahead, above which all fluid policies can be tracked. We introduce a particular granular policy and conjecture that with N lookahead it tracks any fluid policy.

Thesis Supervisor: Vahid Tarokh
Title: Associate Professor of Electrical Engineering and Computer Science

Acknowledgments

The author would like to acknowledge his thesis advisor Professor Vahid Tarokh for the indispensable role he played in bringing this work to fruition. This research and its educational utility to the author would not have been possible without his guidance and insight. The author would also like to acknowledge his wife Irene. Her love, support, and encouragement have made his graduate work possible.

This work was funded by the Office of Naval Research (ONR) under grant N00014-99-1-1019. The author would like to thank the project's principle investigator, Professor Vincent Chan, for his support.

Contents

1 Introduction **5**

1.1 A Common Framework 5

1.2 Prior Work and Motivation 9

2 Preliminaries **15**

3 A Lower Bound on Lookahead **17**

3.1 The N-H Fluid Policy 17

3.2 Conditions for Tracking Loss 18

3.3 The Lower Bound 19

4 The Greedy Policy **22**

4.1 Notation and Assumptions 22

4.2 Worst-Case Fluid Policies 24

5 Conclusion and Future Work **27**

1 Introduction

Arbitrating access to shared media is of great importance to many engineering systems. This is illustrated by the following three examples. In the first, an integrated services network serves N data flows that enter a router. The router must transmit each session's packets onto one of its M output ports, the shared media. In the second example, N mobiles share a wireless channel for transmission to a base station. If we assume the use of a frequency-division-multiple-access (FDMA) scheme, the base station must indicate the frequency bands in which each mobile may transmit. For the third example, consider N processes running on a single CPU. The operating system must decide the order and duration for servicing each process.

1.1 A Common Framework

Examples such as those above can be placed into a common framework. The term *media* refers to that which use must be arbitrated, like output ports, wireless channels, and CPU-access. A medium is characterized by one or more *axes* which span the *media space.* A medium's axes are the principle dimensions along which arbitration occurs. For example, axes are time and frequency for time division multiple access (TDMA) and FDMA, respectively. The media spaces of TDMA- and FDMA-accessed media are all time and all frequencies, respectively. Devices like the router, base station, and operating system that perform media-access arbitration can be referred to as *switches.*

Switches have *input ports* through which media-access requests, or *work*, can enter. Examples of input ports include router input lines and mobile transmitters. Each input port has a queue that stores work that has *arrived* but has not yet been serviced. The time sequence of work that enters through an input port is called a *session.* Network data flows, mobile transmissions, and processes are examples of sessions. A session with queued work is said to be *backlogged.* At each time, a session's work is associated with a particular set of requirements, called *quality-of-service* (QoS) *requirements*, concerning the manner in which it is to be performed. A session can be

a time-multiplexed set of *sub-sessions* whose only common feature is that they share an input queue. Work is measured in *units of work*. For communications systems, a unit of work can be a bit for example. Medium access is parameterized by a *rate* that is in units of unit work per unit time. Work sometimes occurs in indivisible units called *blocks*. In communications networks, packets serve as blocks. Sessions whose work is quantized into blocks are said to be *block-based*. While a switch is performing a session's work, the current block, if any, and its session are said to be being *serviced* by the switch.

Switches generate *schedules* that determine when and how sessions will access media. A switch employs a *policy* to determine the manner in which it generates schedules. Policies can sometimes consider the medium space as granular, divided into *slots*. Slots' properties can be constrained by a policy. For example, in FDMA systems, the slots are frequency bands which may have a fixed size. The act of scheduling a session for a particular medium use involves the specification of a *window*, a section of the media space. A window is defined by the information that a switch requires to service a session for a particular media use. Such information can include the access's duration of time, band of frequency, and spreading code. When all of the windows that will include at least part of particular slot have been specified, the slot is said to have been *allocated*. Each medium use is associated with a *capacity*, the largest rate at which the medium may be accessed. Capacity can depend on both which session is accessing the medium and the window it is using. For example, a TDMA-accessed wireless channel may have a capacity parameterized by the time and the transmitting mobile. Table 1.1 lists some of the framework's terms and some specific examples.

A switch's scheduling task can become non-trivial with the introduction of QoS requirements. Switches that provide QoS guarantees must serve each session in accordance with the session's own desired level of service. For example, in the communications context, one session may contain streaming video data which requires that its packets be delivered on time but can tolerate intermittent packet losses. Another session may be a file transfer. In this case, a maximum delay guarantee is not

Term	TDMA	FDMA	CDMA	Processor Sharing
media	wireless, wireline channel			CPU
axis/axes	time	frequency	code properties	time
media space	all time	all frequencies	all spreading codes	all time
switch	base station, router			operating system
session	transmission, network flow			process
input port	router input port, mobile transmitter			
unit of work	bit			operation
block	packet			operation
slot	time slot	frequency band	set of codes	processor cycle
rate units	bits/s			operations/s
schedule	transmit signals at these times	transmit signals at these frequencies	transmit signals with these codes	service processes at these times
window	time period	frequency band	spreading code	time period

Table 1: Framework terms and specific examples

required, but packet loss is unacceptable.

A well-known approach for developing scheduling algorithms begins by initially assuming that switches can set window properties with arbitrary granularity, or *fluidly.* An $N \times M$ *fluid scheduling policy* is capable of scheduling N sessions access to M media with arbitrarily granular windows. For example, a TDMA $N \times M$ fluid policy can assign each session an arbitrarily small duration of time for medium access. It can therefore effectively service up to N blocks simultaneously. This flexibility makes it relatively simple for a switch to provide sessions with their desired quality of service[1].

This is demonstrated by the following example. Assume that a TDMA switch employing a fluid policy can provide medium access at a maximum rate C. Also, assume that each session $i \in \{1, 2, \ldots, N\}$ requires service at a rate r_i where $\sum_i r_i \leq C$. Let T be an arbitrarily small length of time. Then, to meet the QoS requirements of all N sessions, every T units of time, the switch must transmit $r_i T$ information units for each session i.

[1]This assumes that a switch is not oversubscribed, i.e. that the QoS requirements do not exceed the switch's ability to provide service even if one assumes perfect scheduling.

Employing a fluid policy can be particularly helpful when the capacity varies along the media axes and/or with the serviced session. The switch can then grant sessions capacity according to their QoS requirements. For example, sessions with high QoS requirements can be assigned windows in which the media has large capacity. Such a session in an FDMA system may be assigned a band of frequencies in which the channel's frequency response is relatively large.

In many cases, it is impossible or impractical to implement a fluid-policy switch. It may be that, due to hardware or protocol constraints, windows must be specified in whole slots. The following examples illustrate possible sources of this granularity. For packet-based sessions passing through TDMA routers, it is impossible for the router to service an arbitrary fraction of a packet. For FDMA switches, limitations in modulation and coding schemes can make it undesirable or impossible to provide arbitrarily granular frequency allocation. For code-division-multiple-access (CDMA) switches, spreading code construction limitations can make assigning codes with arbitrarily granular properties (such as processing gain and correlation) undesirable or impossible.

A fluid policy's ease of design and QoS-provisioning effectiveness can still be utilized even when a switch will not employ one directly. This is accomplished by developing a slot-based method of approximating the fluid policy. Because such *approximating policies* are slot-based, they can sometimes be realized when fluid policies cannot.

Approximating policies can involve simulating a fluid policy on the side. This simulation is then used to generate a slot-based schedule. Such algorithms may generate the slot-based schedule on a slot-by-slot basis as follows:

1. Choose a *current* slot, say slot k_0.

2. Simulate the fluid policy until it has allocated slots $k_0, k_1, \ldots, k_H$.

3. Decide which session to assign slot k_0 in the slot-based policy by taking into account the fluid policy's scheduling of slots $k_0, k_1, \ldots, k_H$.

Slots $k_1, k_2, \ldots, k_H$ are called the *lookahead window*. They are the information about the fluid policy's schedule that an approximating policy *with lookahead H* requires in addition to the current slot. It is of central importance to our work that approximating policies require lookahead in order to adequately approximate fluid policies.

1.2 Prior Work and Motivation

A well-known $N \times 1$ fluid policy is Generalized Processor Sharing (GPS). A GPS server arbitrates *GPS sessions'* access to a single-axis medium. A GPS session is a session whose QoS requirements are constant with time and can be expressed by a constant real number called its weight. The server provides each session service directly proportional to its weight. Formally, a GPS switch is defined as that for which

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

where τ and t are both locations on the medium's axis, i is a session that is continuously backlogged on the interval $(\tau, t]$, $S_i(\tau, t)$ is the amount of session i work performed in an interval $(\tau, t]$, and ϕ_i is the weight of session i. Note that equality in the above expression holds when both sessions i and j are backlogged on $(\tau, t]$. When GPS is used in a router, the delay and throughput that a session experiences are coupled and are directly related to the session's weight.

Under GPS, session i is guaranteed a rate of

$$r_i = \frac{\phi_i}{\sum_{j \in B(t)} \phi_j} r \tag{1}$$

where $B(t)$ is the set of backlogged sessions at time t and r is the rate at which work is performed by the switch.

Parekh and Gallager applied GPS to packet switching in [4, 5]. They demonstrated that GPS can provide end-to-end delay bounds to leaky-bucket-constrained sessions in networks using GPS at each switch. In this case, GPS is unrealizable. They, therefore, present an implementable approximation to GPS called *packet-by-packet*

GPS (PGPS), also known as Weighted Fair Queuing (WFQ). This approximation preserves GPS's delay-bounding utility.

PGPS simulates GPS and bases its packet service order on GPS's packet service completion times. Assume that a PGPS server is idle at time τ. At the next time slot during which a packet is queued for service, PGPS is defined as the policy that services the first packet that would complete service under GPS if no additional packets were to arrive after time τ.

PGPS is work-conserving [2] and typically services packets in the same order as GPS. The order is not preserved when the next packet that GPS completes arrives too late to be serviced in order. Consider the following example. Assume that a switch services 2 equally-weighted sessions, sessions 0 and 1. Packet p_0 of session 0 arrives at time 0 and requires 10 time units of service, and packet p_1 of session 1 arrives at time 1 and requires 1 unit of service. Between times 1 and 3, GPS will provide each session with a rate of 0.5, and p_1 will therefore complete GPS service at time 3. p_0 will complete GPS service after time 3. In order for PGPS to preserve the packet service order, it should service p_1 first. At time 0, though, it must serve p_0 because it is the only packet in the system. p_0 will complete PGPS service at time 10, and p_1 will complete PGPS service at time 11. The order is, therefore, not preserved.

Parekh and Gallager also prove the following two facts concerning GPS and PGPS:

- *Fact 1.* $S_{j,\text{GPS}}(0,\tau) - S_{j,\text{PGPS}}(0,\tau) \leq L_{\max}$ for all times τ and sessions j
- *Fact 2.* There is no $c \geq 0$ such that $S_{j,\text{PGPS}}(0,\tau) - S_{j,\text{GPS}}(0,\tau) \leq cL_{\max}$ for all sessions j over all traffic patterns.

where $S_{j,\text{GPS}}(0,\tau)$ and $S_{j,\text{PGPS}}(0,\tau)$ are the amount of session j work performed in an interval $(0,\tau]$ by GPS and PGPS, respectively; and $L_{\max}$ is the maximum packet size. Fact 1 states that at any time at which PGPS has provided a session with less service than GPS, the deficit can never be greater than the maximum packet size. Fact 2 states that the amount of PGPS service that a session receives in excess of its

[2] Work-conserving policies are policies that are always servicing a packet if a packet is queued for service.

GPS service cannot be upper-bounded in general. Therefore, PGPS is guaranteed to "keep up" with GPS, but PGPS can get arbitrarily "ahead" of GPS.

In [1], Bennett and Zhang present another GPS approximating policy called Worst-case Fair Weighted Fair Queuing (WF^2Q). Let τ be a time at which a WF^2Q server is idle. WF^2Q is defined as the policy that then considers the set of packets that have at least begun GPS service and, of those packets, services the one that completes GPS service the soonest, assuming that no additional packets arrive after time τ. Because WF^2Q restricts itself to servicing packets that GPS has at least begun servicing, the extent to which it can "get ahead" GPS is bounded. In fact, [1] proves that

$$S_{j,\mathrm{WF^2Q}}(0,\tau) - S_{j,\mathrm{GPS}}(0,\tau) \leq (1 - \frac{r_j}{r})L_{\max} \tag{2}$$

where $S_{j,\mathrm{WF^2Q}}(0,\tau)$ and $S_{j,\mathrm{GPS}}(0,\tau)$ are the amount of session j work performed in an interval $(0,\tau]$ by WF^2Q and GPS, respectively; r_j is session j's guaranteed minimum serviced rate under GPS as defined in (1); and r is the maximum switch rate. Because both Fact 1 and (2) applies to WF^2Q, WF^2Q is guaranteed to better approximate GPS than PGPS. In fact, using a term that will be defined later in this section, WF^2Q *tracks* GPS, and WFQ does not.

Many other GPS-approximating policies have been developed by other researchers with varying degrees of implementation complexity and approximation accuracy. GPS-approximating policies are referred to as *Packet Fair Queuing* (PFQ) algorithms.

Despite GPS's usefulness at providing delay guarantees, it does possess some shortcomings. The five drawbacks that we consider arise from the fact that its weights are constant. First, sessions can be time-multiplexed sets of sub-sessions, yielding time-varying QoSs. GPS is unable to handle sessions with time-varying QoSs.

The second drawback is the fact that an $N \times 1$ GPS switch can service at most N differently weighted GPS sessions. A general switch is also limited to N sessions, but these sessions can be made of any number of sub-sessions.

The dynamic environment described by these drawbacks may become common in

future communications systems. Such an environment could exist in an integrated services network in which input lines carry large numbers of network flows, each with their own QoS. Another such environment is discussed in [10] and concerns wireless mobiles transmitting on a common channel. Because of the nature of the wireless channel, the link from each mobile to the base station will have a different time-varying capacity. The base station could use its knowledge of the channel to intelligently change sessions' weights. Mobiles' weights can be directly proportional to their transmission capacities, for example. Also, mobiles that have received less than their desired QoS because of a fade can be compensated with a larger weight when their channel improves. GPS does not support this kind of flexibility.

A third drawback of GPS is discussed in [8]. Stamoulis and Liebeherr note the fact that backlogged sessions will experience a sudden decrease in rate whenever a previously un-backlogged session becomes backlogged. This is due to the sudden change in the summation of (1). These sudden changes can create abrupt increases in delay and jitter (i.e. delay variance). They propose an algorithm called Slow-Start GPS (S^2GPS) that avoids this effect by slowly decreasing the service rate of previously backlogged sessions. S^2GPS accomplishes this by using time-varying weights.

In [6], Stamoulis and Giannakis bring to light a fourth drawback: delay and bandwidth guarantees are coupled under GPS and PFQ. Because both derive from a scalar, constant weight, it is difficult to independently control a session's guaranteed delay and bandwidth. To decouple these guarantees, they propose a queuing algorithm that uses deterministically time-varying weights.

A fifth drawback of GPS is presented in [3]. Duffield et al. question the manner in which GPS and PFQ allocate excess resources in communications networks. Such excess resources exist when at least one session is not backlogged. These algorithms allocate unused resources to backlogged sessions proportional to their weights. They argue that it may better serve sessions to redistribute these resources in a more QoS-sensitive manner. For example, to reduce jitter, excess resources could be assigned to sessions with the longest delays. Packet-loss probabilities could be decreased if excess resources were assigned to sessions whose buffers are closest to overflowing. We believe

that such flexibility could be obtained by considering GPS with time-varying weights.

To overcome GPS's drawbacks, we consider *general fluid policies* (GFPs), GPS policies with time-varying weights. In [9], Tabatabaee et al. consider $N \times M$ GFPs using FCFS input-queued switches and fixed-sized packets. A goal of their work is to discover conditions that imply the existence of *tracking policies*. A tracking policy of a particular fluid policy π_f is defined as an approximating policy of π_f that emulates π_f consistent with a specific accuracy criterion.

Note the following assumptions and notation from [9] given a particular fluid policy π_f and a particular packetized policy π_p:

- All packets are length 1 and the switch can input and output at most 1 packet from each input and output port, respectively.
- Let $w_{ij}[k]$ denote the amount of traffic that π_f transmits from port i to port j at time k. $w_{ij}[k]$ possesses the following properties:
  - $w_{ij}[k] \geq 0$
  - $\sum_j w_{ij}[k] \leq 1, \ \forall i \in \{1, 2, \ldots, N\}$
  - $\sum_i w_{ij}[k] \leq 1, \ \forall j \in \{1, 2, \ldots, M\}$
- Let $W[k]$ denote an $N \times M$ matrix with elements $W_{ij}[k] = \sum_{l=1}^{k} w_{ij}[l]$
- Let $J[k]$ denote an $N \times M$ sub-permutation matrix with elements $J_{ij}[k]$ such that

$$J_{ij}[k] = \begin{cases} 0, & \pi_p \text{ services 0 packets from input port } i \text{ to output port } j \text{ at time } k \\ 1, & \pi_p \text{ services a packet from input port } i \text{ to output port } j \text{ at time } k \end{cases}$$

  $J_{ij}[k]$ possesses the following properties:
  - $\sum_j J_{ij}[k] \leq 1, \ \forall i \in \{1, 2, \ldots N\}$
  - $\sum_i J_{ij}[k] \leq 1, \ \forall j \in \{1, 2, \ldots M\}$
- Let $I[k]$ denote a matrix with elements $I_{ij}[k] = \sum_{l=1}^{k} J_{ij}[l]$.

Tabatabaee et al. define the term tracking policy such that π_p is a tracking policy of π_f if and only if π_p

- never serves a packet after the end of the slot in which it completes fluid-policy service and
- never serves a packet before the slot during which it begins fluid policy service.

Because they considered FCFS queues, this is equivalent to

$$\lfloor W_{ij}[k] \rfloor \leq I_{ij}[k] < W_{ij}[k] + 1 \quad \forall i, j. \tag{3}$$

Therefore, the solution to the following problem is a tracking policy of π_f.

Problem 1 *Given $W[l]$ for some values of l, find a sequence of sub-permutation matrices $J[k]$ for $k = 0, 1, \ldots$ such that (3) holds.*

The result of [9] that is the most relevant to our work is the fact that a solution exists to Problem 1 that uses no lookahead for the case that $N = M = 2$. Specifically, it is proven that given $W[k]$, $J[k]$ can be found by the following algorithm such that π_p tracks π_f through time k for any k:

Algorithm

1. Choose $J[k]$ such that

$$J_{ij}[k] = \max\{\lfloor W_{ij}[k] \rfloor - I_{ij}[k-1],\ 0\} \quad \forall i, j.$$

2. If for some column or row, say row 1, it holds that $I_{i1}[k-1] = \lfloor W_{i1}[k] \rfloor$ for all i and $W_{i1}[k]$ is non-integer for all i, modify $J[k]$ by setting $J_{m1}[k] = 1$ for some m such that the modified $J[k]$ is still a sub-permutation matrix.

Because of this result, one might conjecture that if an $N \times N$ packetized policy employed sufficient lookahead, it could track any $N \times N$ GFP. However, in Appendix

B of [2], Bonuccelli and Clò give the following 4×4 counter-example.

$$w[k] = \begin{bmatrix} 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ \frac{1}{3} & \frac{1}{2} & \frac{1}{6} & 0 \\ \frac{1}{3} & 0 & \frac{1}{2} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{6} & 0 & \frac{1}{2} \end{bmatrix} \quad \forall k$$

This fluid policy cannot be tracked even with infinite lookahead. This therefore implies the existence of an $N \times N$ GFP for any $N \geq 4$ that is unable to be tracked.

The focus of this thesis is to identify some of the requirements of tracking $N \times 1$ GFPs. We derive a lower bound on the required lookahead, below which there exists at least one GFP that is not trackable. We also present several lemmas that describe the behavior of $N \times 1$ GFPs.

This thesis is organized as follows. Notation is introduced in Section 2. A lower bound on lookahead is presented in Section 3. We present a particular packetized policy and preliminary analysis in Section 4. We conclude the thesis and discuss possible future extensions in Section 5.

2 Preliminaries

We consider $N \times 1$ packet switches, that is network multiplexers that can service up to N sessions. Each port employs a unique FCFS queue and packets are of a fixed size 1. Each time slot $k \in \mathbb{Z}$, each port j can serve at most 1 packet where $j \in \{1, 2, \ldots N\}$.

A fluid policy is capable of assigning each session at each time slot k a fraction of a packet. $w_j[k] \geq 0$ denotes the (possibly non-integer) number of packets that pass through port j during time slot k under a fluid policy. Due to the constraint that each port can service at most a single packet,

$$\sum_j w_j[k] \leq 1. \tag{4}$$

Let $W[k]$ denote a length-N vector with elements $W_j[k]$. $W_j[k] \in \mathbb{R}$ is the cumulative (possibly non-integer) number of packets that have passed through port j on the time interval $[1, k]$ under a fluid policy. That is,

$$W_j[k] = \sum_{l=1}^{k} w_j[l] \quad \forall j. \tag{5}$$

A packetized policy can service 0 or 1 packets for each time slot k. $J[k]$ is a length-N vector with elements $J_j[k]$ equal to the number of packets that pass through port j during time slot k under a packetized policy. Packetized policies may allow each port to serve at most a single packet during each time slot. Thus,

$$\sum_{j} J_j[k] \leq 1. \tag{6}$$

Let $I[k]$ denote a length-N vector with elements $I_j[k]$. $I_j[k] \in \{0, 1, \dots\}$ is the cumulative number of packets that have passed through port j on the time interval $[1, k]$ under a packetized policy. That is,

$$I_j[k] = \sum_{l=1}^{k} J_j[l] \quad \forall j. \tag{7}$$

As specified in [9] and stated earlier for the $N \times M$ case, an $N \times 1$ packetized policy π_p is said to *track* an $N \times 1$ fluid policy π_f if only if π_p:

- never serves a packet after the end of the slot in which it completes fluid-policy service and
- never serves a packet before the slot during which it begins fluid policy service

Because we consider FCFS queues, this definition can be restated. A packetized policy described by the vector sequence $I[0], I[1], \dots$ tracks a fluid policy described

by the vector sequence $W[0], W[1], \ldots$ if and only if

$$\lfloor W_j[k] \rfloor \leq I_j[k] < W_j[k] + 1 \quad \forall k, j. \tag{8}$$

Define *a packetized policy with H lookahead* to be a packetized policy that uses knowledge of $W[k+h]$ for $h \in \{0, 1, 2, \ldots, H\}$ to calculate $J[k]$ for each k.

3 A Lower Bound on Lookahead

We now present a lower bound on the lookahead required to track an $N \times 1$ GFP. We consider a particularly difficult-to-track fluid policy called the N-H policy, denoted $\pi_f(N, H)$.

3.1 The N-H Fluid Policy

The *N-H* fluid policy is specified by N and H where $H \leq N - 3$. It possesses the following properties:

Property 1. $w_j[k] = 0 \quad \forall j$ and $k \leq 0$.

Property 2. $w_j[k] = \frac{1}{N} \quad \forall j$ and $1 \leq k \leq H$.

Property 3. For $0 \leq k \leq N - 2$, if $J_{j_x}[k] = 1$ for any $j_x \in \{1, 2, \ldots, N\}$, then $w_{j_x}[k + l + H + 1] = 0 \quad \forall l$ where $0 \leq l \leq N - 2 - k$.

Property 4. For $H + 1 \leq k \leq H + N - 1$, $w_j[k] = \frac{1}{N+H-k}$ for each $w_j[k]$ not made zero by Property 3.

Figure 1 is an example of $\pi_f(5, 3)$.

A fluid policy with these properties is very difficult to track because the packetized policy's H lookahead time slots provide minimally useful and often misleading information. Fluid policy service is evenly distributed, maximizing the number of ports that appear to require service most urgently. By Property 3, ports that will have no service under the fluid policy do so one time slot beyond the lookahead window. This hides this potentially useful information from the packetized policy. Most importantly, this fluid policy works to nullify the actions of the packetized policy by

$$
\begin{aligned}
w[1] &= [\ (\tfrac{1}{5})\ \ \tfrac{1}{5}\ \ \tfrac{1}{5}\ \ \tfrac{1}{5}\ \ \tfrac{1}{5}\ ] \\
w[2] &= [\ \tfrac{1}{5}\ \ (\tfrac{1}{5})\ \ \tfrac{1}{5}\ \ \tfrac{1}{5}\ \ \tfrac{1}{5}\ ] \\
w[3] &= [\ \tfrac{1}{5}\ \ \tfrac{1}{5}\ \ (\tfrac{1}{5})\ \ \tfrac{1}{5}\ \ \tfrac{1}{5}\ ] \\
w[4] &= [\ \tfrac{1}{5}\ \ \tfrac{1}{5}\ \ \tfrac{1}{5}\ \ (\tfrac{1}{5})\ \ \tfrac{1}{5}\ ] \\
w[5] &= [\ 0\ \ \tfrac{1}{4}\ \ \tfrac{1}{4}\ \ \tfrac{1}{4}\ \ (\tfrac{1}{4})\ ] \\
w[6] &= [\ 0\ \ 0\ \ \tfrac{1}{3}\ \ \tfrac{1}{3}\ \ \tfrac{1}{3}\ ] \\
w[7] &= [\ 0\ \ 0\ \ 0\ \ \tfrac{1}{2}\ \ \tfrac{1}{2}\ ] \\
w[8] &= [\ 0\ \ 0\ \ 0\ \ 0\ \ 1\ ]
\end{aligned}
$$

Figure 1: An example $\pi_f(N,H)$: $\pi_f(5,3)$. $w[k] \equiv [w_1[k], w_2[k], \ldots, w_N[k]]$. If $w_j[k]$ is in parentheses, $J_j[k] = 1$

no longer servicing those links previously serviced by the packetized policy. The fluid policy therefore maximizes the likelihood of tracking loss by concentrating service on only those links that have not yet been serviced by the packetized policy.

3.2 Conditions for Tracking Loss

Under this policy, at any time $H+1 \leq k \leq N+H-1$, for an unserviced port v where $v \in \{1, 2, \ldots, N\}$,

$$W_v[k] = \sum_{a=N+H-k}^{N-1} \frac{1}{a} + \frac{H+1}{N}. \tag{9}$$

We assume that the packetized policy always makes the best decision with the available information. Therefore, one port is serviced each time slot. Tracking is lost if and only if at any time the number of unserviced ports for which $W_v[k] \geq 1$ exceeds 1, the number that can be serviced in one time slot. Let k^* denote the earliest time at which $W_v[k] \geq 1$ for a port that was previously unserviced. Because of symmetry, $W_v[k^*] \geq 1$ for *all* ports v that are not serviced before time k^*. The number of

unserviced ports at time k^* is $N - k^*$. Therefore, tracking is lost if and only if

$$\begin{aligned} N - k^* > 1, \text{ or equivalently} \\ k^* \leq N - 2. \end{aligned} \tag{10}$$

Because $W_j[k]$ is a non-decreasing function of k, we can restate the necessary and sufficient condition for tracking loss as

$$1 \leq W_v[k^*] \leq W_v[N-2] \tag{11}$$

$$W_v[N-2] = \sum_{a=H+2}^{N-1} \frac{1}{a} + \frac{H+1}{N} \geq 1 \tag{12}$$

where $H \leq N - 3$.

3.3 The Lower Bound

The following lemma will aid the interpretation of (12).

Lemma 1

a. *If there exists an $N_0 \times 1$ fluid policy that no $N_0 \times 1$ packetized policy with lookahead H is able to track, there exists a fluid policy that no $N \times 1$ packetized policy with H lookahead is able to track where $N \geq N_0$.*

b. *If there exists an $N \times 1$ fluid policy that no $N \times 1$ packetized policy with lookahead H_0 is able to track, there exists a fluid policy that no $N \times 1$ packetized policy with lookahead $H \leq H_0$ is able to track.*

c. *If there exists a tracking policy with lookahead H_0 of an $N \times 1$ fluid policy, there exists a tracking policy with any lookahead $H \geq H_0$ of the fluid policy.*

By Lemma 1, if (12) holds for $\pi_f(N_0, H_0)$, an un-trackable fluid policy exists for $N \geq N_0$ and lookahead $H \leq H_0$. If an (N_0, H_0) pair does not satisfy (12), there exists

a packetized policy that can track $\pi_f(N_0, H_0)$. Then by Lemma 1, $\pi_f(N_0, H_0)$ is also trackable by packetized policies for which $H \geq H_0$.

Figure 2 shows the relative value of $W_v[N-2]$ and 1 over the N-H plane. The upper-left region corresponds to those (N, H) pairs for which (12) does not apply because $H > N - 3$. The lower-right region corresponds to those (N, H) pairs for which $W_v[N-2] \geq 1$. Such pairs, therefore, describe packetized policies for which there exists an un-trackable fluid policy. The third, middle region consists of (N, H) pairs for which $W_v[N-2] < 1$. All pairs within and, by Lemma 1, above this region correspond to packetized policies that are able to track the N-H fluid policy.

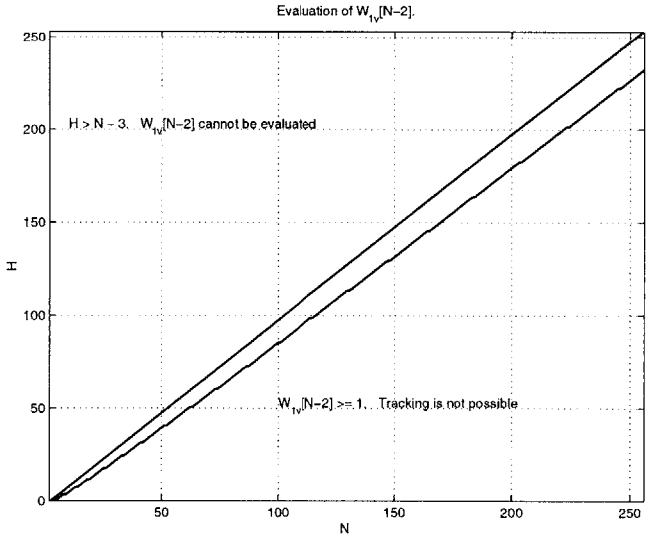

Figure 2: Evaluation of $W_v[N-2]$.

We now present a closed-form lower bound.

Theorem 1 *If $N \geq N_c$, there exists a fluid policy that cannot be tracked by any packetized policy with only $H = cN - 1$ lookahead where c is any real number in $[c_{min}, 1)$,*

$$N_c \equiv \frac{1 + e^{1-c}}{1 - ce^{1-c}} \ , \tag{13}$$

and $c_{min} \approx \frac{13}{72} \approx 0.18556$ is defined as the solution to $c(1 + 2e^{1-c}) = 1$

Proof: First, note the following lower bound

$$\sum_{j=a}^{b} \frac{1}{j} \geq \int_{a}^{b} \frac{1}{x} dx \tag{14}$$

Applying (14) to (12) and evaluating the integral we obtain

$$W_v[N-2] \geq \ln\left(\frac{N-1}{H+2}\right) + \frac{H+1}{N} \tag{15}$$

Assume that $H = cN - d$ for real $\frac{d}{N} \leq c < 1$ and $d \geq 0$. Then, $\pi_f(N, H)$ will be un-trackable if

$$\ln\left(\frac{N-1}{cN-d+2}\right) + \frac{cN-d+1}{N} \geq 1 \tag{16}$$

After some algebra this becomes

$$\frac{N-1}{cN-d+2} \geq \exp\left(1-c+\frac{d-1}{N}\right) \tag{17}$$

The choice of $d = 1$ makes (17)'s right-hand-side independent of N. This choice provides the following condition for the non-existence of a $\pi_f(N, cN-1)$-tracking packetized policy

$$N \geq \frac{1+e^{1-c}}{1-ce^{1-c}} \equiv N_c \tag{18}$$

We wish to restrict the range of c to include only values of interest.

Such values of $c \in [\frac{1}{N}, 1)$ meet the following two conditions:

- $H = cN - 1 \geq 0$
- For some range of N, denoted $\mathbf{N}$, there exists an un-trackable $\pi_f(N, cN-1)$

Assume that $N \in \mathbf{N}$, and therefore, by (18), $N \geq N_c$. Because $cN - 1 \geq cN_c - 1$ and $cN_c - 1$ is a monotonically increasing function of $c \in [\frac{1}{N}, 1)$, we can restrict our

attention to those values of c for which $cN_c - 1 \geq 0$, or, equivalently,

$$c\left(\frac{1+e^{1-c}}{1-ce^{1-c}}\right) \geq 1$$

$$c \geq \frac{1-ce^{1-c}}{1+e^{1-c}} \tag{19}$$

where we used the fact that $\frac{1-ce^{1-c}}{1+e^{1-c}} > 0 \; \forall c \in [0,1)$. Note that in (19) for $c \in [0,1)$, the left-hand-side is monotonically increasing and the right-hand-side is monotonically decreasing. Therefore, (19) holds for all values of c greater than c_{min}, the value for which (19) holds with equality. Algebra demonstrates that c_{min} can be defined as the solution to $c(1+2e^{1-c}) = 1$ which is approximately $\frac{13}{72} \approx 0.18556$. Therefore, the domain of c is restricted to $[c_{min}, 1)$. ■

Because N_c is monotonically increasing in $c \in [c_{min}, 1)$, its minimum value occurs at $N_{c_{min}} \approx 5.54$. Therefore, Theorem 1 only applies to $N \times 1$ GFPs for which $N \geq 6$.

4 The Greedy Policy

We now present the preliminary results in our study of an upper bound on lookahead, above which any fluid policy can be tracked. Our work has led us to the following conjecture.

Conjecture 1 *There exists a packetized policy that with N lookahead could track any $N \times 1$ GFP.*

4.1 Notation and Assumptions

Given a fluid policy whose actions before time l are described by $W[l]$ and a packetized policy whose actions before time $k-1$ are described by $I[k-1]$, define

$$a_j[k,l] = W_j[l] - I_j[k-1], \quad l \geq k. \tag{20}$$

The following observations follow from this definition.

- $a_j[k,k] \geq 1$ implies that a choice of $J_j[k] = 0$ will cause the packetized policy to fall behind the fluid policy at time k. That is, the lower bound of (8) will be violated.

- $a_j[k,k] > 0$ implies that the packetized policy will not get ahead of the fluid policy regardless of the choice of $J_j[k]$. That is, the upper bound of (8) will hold.

- $a_j[k,k] \leq 0$ implies that

  - if $J_j[k] = 0$, the packetized policy will not fall behind the fluid policy. That is, the lower bound of (8) will hold.

  - if $J_j[k] = 1$, the packetized policy will get ahead of the fluid policy. That is, the upper bound of (8) will be violated.

The following notation is used to define k^i and $\mathbf{P}^i$ at time k.

- Let A_j be an infinitely long vector defined at time k by

$$A_j = [\ a_j[k,k],\ a_j[k,k+1],\ a_j[k,k+2],\ \ldots\ ]$$

- Define n_j such that the n_j^{th} element of A_j is the first that is not less than 1. Then, define $m_j = n_j + k - 1$, the time that corresponds to the n_j^{th} element.

- Define the set $K = \{m_j : 1 \leq j \leq N\}$. Let $k^1, k^2, \ldots, k^N$ be unique members of K. Define the set $K^i = \{k^1, k^2, \ldots, k^i\} \subseteq K$ for $1 \leq i \leq N$, and let $K^0 = \emptyset$.

Define k^i for $1 \leq i \leq N$ at time k by

$$k^i = \min_{\{j\,:\,m_j \notin K^{i-1}\}} m_j$$

Let $\mathbf{P}^i$ be the set of ports for which $a_j[k,k^i] \geq 1$ and, if $k^i > k$, $a_j[k,k^i - 1] < 1$.

Let i' be the least value of i for which $a_j[k,k] > 0$ for at least one port $j \in \mathbf{P}^i$. Let $\mathbf{N}$ be the set of ports that are the *among the neediest* at time k. $\mathbf{N}$ consists of those ports $j \in \mathbf{P}^{i'}$ such that $a_j[k,k] > 0$. Let $k_{\mathbf{N}} = k^{i'}$.

Assume the following packetized policy, called the *greedy policy* and denoted by π_g, which does the following at time k with lookahead H:

1. If $k^1 > k + H$,

   (a) If $a_j[k,k] > 0$ for at least 1 value of j, choose $J_i[k] = 1$ for port i such that $i = \arg\max_j a_j[k,k]$.

   (b) If $a_j[k,k] \leq 0$ for all j, choose $J_j[k] = 0$ for all j.

2. If $k^1 \leq k + H$,

   (a) If $a_j[k,k] \leq 0$ for all j, choose $J_j[k] = 0$ for all j.

   (b) Otherwise, choose $J_p[k] = 1$ where $p = \arg\max_{j \in \mathbf{N}} a_j[k, k_{\mathbf{N}}]$. That is, serve the neediest port.

In this algorithm, when two or more ports meet the criteria for service simultaneously, the tie is broken by some deterministic method.

Conjecture 2 *π_g with lookahead N can track any $N \times 1$ GFP.*

The following is additional notation. Let π_x^y denote a fluid policy where x is any symbol or symbols except "g" and y is any symbol or a blank. π_x^y generates the vector sequence $w^y[0]$, $w^y[1], \ldots$. Let $W^y[k] = \sum_{l=1}^{k} w^y[l]$. When π_g approximates π_x^y, it generates vectors $J^y[k]$ for all k for which $I^y[k] = \sum_{l=1}^{k} J^y[l]$. Subscripts index the elements of these vectors. π_g tracks π_x^y before time t^y and loses tracking at time t^y. Additionally, let $a_j^y[k,l] = W_j^y[l] - I_j^y[k-1]$. If π_x^y is parameterized by a time n, then let π_x^y also be denoted by $\pi_x^y[n]$.

4.2 Worst-Case Fluid Policies

We now present some results that may be useful in analyzing π_g. Assume that there exists a set $\mathbf{S} \neq \emptyset$ of fluid policies that π_g cannot track. Let $\pi_f^1, \pi_f^2, \ldots, \pi_f^S$ be the

S members of set $\mathbf{S}$. For $i \in [1, S]$, π_f^i, is associated with a time t^i at which π_g is unable to track it. That is, t^i is the smallest value of k for which (8) does not hold where $I[k] = I^i[k]$ and $W[k] = W^i[k]$ for all k. Define $t = \min_i t^i$, and let π_f be a fluid policy in $\mathbf{S}$ that π_g is unable to track at time t. That is, π_f is a *worst-case* fluid policy.

We now focus on π_g approximating π_f and assume that π_g uses $H = N$ lookahead. Assume that π_g and π_f generate the sequences of vectors $J[1], J[2], \ldots$ and $w[1], w[2], \ldots$, respectively. Also, because tracking is lost at time t, there exists a set of ports $\mathbf{M}$ of size $m \geq 2$ such that $a_j[t, t] \geq 1$ for $j \in \mathbf{M}$.

Lemma 2 *For any $N \times 1$ GFP that generates the vector sequence $w[0], w[1], \ldots$, if $\sum_{j=1}^N w_j[k] = 1$ for all $k \geq 1$, then for all $k \geq 1$ $\sum_{j=1}^N a_j[k, k] = 1$ and $J_j[k] = 1$ for some port.*

Proof: The proof is by induction. At time $k = 1$, $\sum_{j=1}^N a_j[1, 1] = \sum_{j=1}^N w_j[1] = 1$. Because $\sum_{j=1}^N a_j[1, 1] > 0$, it must be that $a_j[1, 1] > 0$ for at least 1 value of j. Therefore, $J_j[1] = 1$ for one such value.

Assume that $\sum_{j=1}^N a_j[k, k] = 1$ for some $k > 1$ and $J_j[l] = 1$ for some port for each $1 < l \leq k$. Then

$$\begin{aligned}
\sum_{j=1}^{N} a_j[k+1, k+1] &= \sum_{j=1}^{N} (W_j[k+1] - I_j[k]) && (21)\\
&= \sum_{j=1}^{N} (W_j[k] + w_j[k+1] - I_j[k-1] - J_j[k]) && (22)\\
&= \sum_{j=1}^{N} a_j[k, k] + \sum_{j=1}^{N} w_j[k+1] - 1 && (23)\\
&= 1 && (24)
\end{aligned}$$

Because $\sum_{j=1}^N a_j[k+1, k+1] > 0$, it must be that $a_j[k+1, k+1] > 0$ for at least 1 value of j. Therefore, $J_j[k+1] = 1$ for one such value. ■

Lemma 3 *For any worst-case $N \times 1$ GFP that generates the vector sequence $w[0], w[1], \ldots$, if for at least one time $l \geq 1$, $\sum_{j=1}^N w_j[l] < 1$, then there exists a fluid policy π^b*

such that

- $\sum_{j=1}^{N} w_j^b[k] = 1$ *for* $k \geq 1$ *and*
- $t^b = t$.

That is, if π_f *services less than 1 packet for at least 1 time slot, then there exists a fluid policy* π^b *that services 1 packet each time slot and for which* π_g *loses tracking at the same time as it does when approximating* π_f.

Proof: π^b is constructed as follows. Let $\delta_l = 1 - \sum_{j=1}^{N} w_j[l]$. For each time l for which $\sum_{j=1}^{N} w_j[l] < 1$, let $w_j^b[l] = w_j[l] + \frac{\delta_l}{N}$. By this construction, $\sum_{j=1}^{N} w_j^b[k] = 1$ for $k \geq 1$. Note that by Lemma 2, $J_j^b[k] = 1$ for some port each time k.

We consider two cases.

Case 1. $a_j[k,k] > 0$ for at least 1 value of j for each time $k \geq 1$:
In this case, $J_j[k] = 1$ for some port each time $k \geq 1$. Therefore, $J_j[k] = J_j^b[k]$. That is, adding the length-N vector $[\frac{\delta_l}{N}, \frac{\delta_l}{N}, \ldots, \frac{\delta_l}{N}]$ to some fluid-policy-service vectors will not change the actions of π_g.

Consider the event that adding the vector makes $a_j^b[l,l] \geq 1$ for more than one value of j and some time $l \leq t$. This would cause π_g to lose tracking before time t when it approximates π^b. This is a contradiction, though, of the assumption that π_f is a worst-case fluid policy. Therefore, this event cannot occur and $t^b = t$.

Case 2. $a_j[k,k] \leq 0$ for all j and $k \in K \neq \emptyset$:
For each $k \in K$, $J_j[k] = 0$ for all j. By Lemma 2, $J_j^b[k] = 1$ for some port. That is, adding the length-N vector $[\frac{\delta_l}{N}, \frac{\delta_l}{N}, \ldots, \frac{\delta_l}{N}]$ to some traffic vectors will change the actions of π_g.

This case, however, cannot occur because it implies a contradiction. If $a_j[k,k] \leq 0$ for all j for some time k, there exists a fluid policy π^a constructed by the following algorithm.

1. $w^a[l] := \begin{cases} 0, & l < k \\ w[l], & l \geq k \end{cases}$

2. For each port j:

   (a) If $a_j[n,n] \geq 0$ for some $n > k$,

      i. Let $c_j > k$ be the earliest time at which $a_j[c_j, c_j] \geq 0$.

      ii. $x := k$

      iii. $s := 0$

      iv. While $s < a_j[k,k]$:

         A. If $w_j[x] \geq a_j[k,k] - s$,

            - $w_j^a[x] := w_j[x] - (a_j[k,k] - s)$

            - $s := a_j[k,k]$

         B. else,

            - $w_j^a[x] := 0$

            - $s := s + w_j[x]$

         C. $x := x + 1$

   (b) else, $w_j^a[m] := 0$ for $m \geq k$

3. $w^a[l] := w^a[l - k - 1]$

Step 1 makes it such that $a_j^a[k,k] = 0$. Because $a_j[k,k] < 0$, we must subtract $a_j[k,k]$ from the fluid policy service that port j receives prior to the earliest time n at which $a_j[n,n] \geq 0$. This is accomplished by Step 2. Step 3 shifts the fluid policy service in time.

The algorithm generates fluid policy service vectors that will cause π_g to lose tracking at time $t - k - 1$. Therefore, this case cannot occur because it implies the contradiction that π_f is not a worst-case fluid policy. ■

5 Conclusion and Future Work

We considered the problem of arbitrating users' access to a shared medium consistent with quality of service (QoS) requirements. This problem arises in the designing of

a wide variety of systems, collectively called switches, that include packet routers, computer operating systems, and wireless networks. We have focused on a fluid-policy-based solution. In this case, a switch first employs a fluid policy to generate a medium access schedule. Because they treat medium access as infinitesimally divisible, fluid policies are typically easier to design and less computationally intensive than granular, or packetized, policies. However, in many cases of interest, such as packet routers, fluid policy schedules are unrealizable because medium access must be granular. Therefore, a switch then employs a granular policy that generates a realizable schedule by approximating the fluid policy schedule. If the granular and fluid policy schedules are similar enough, consistent with a specific criterion, the granular policy is said to track the fluid policy.

Tracking a fluid policy can require lookahead, or future knowledge of its schedule. This lookahead can be obtained by operating the tracking policy at a fixed delay. We derived a lower bound on the lookahead required to track $N \times 1$ fluid policies of $\mathcal{O}(N)$. We also presented preliminary results in our study of an upper bound on lookahead, above which all fluid policies can be tracked. These results are an important first step in understanding the requirements of tracking $N \times 1$ fluid policies.

There are many interesting open questions for future research on this topic. An upper bound on the lookahead required to track any $N \times 1$ fluid policy would be very useful to granular-policy designers. Whether or not this upper bound is finite and, if it is, its value are currently unknown. Additionally, if a finite upper bound exists, one could quantify the adverse effects of using lookahead below this bound. This problem could also be studied statistically. Perhaps there is a relationship between the quantity of lookahead and the probability that a particular granular policy tracks any fluid policy at any given time. In addition, for cases in which the quantity of lookahead used is below the upper bound, the effects on tracking of estimating the future behavior of a fluid policy could be studied.

References

[1] Bennett, J. C. R., H. Zhang. WF^2Q: Worst-case Fair Weighted Fair Queueing. INFOCOM '96, March 1996.

[2] Bonuccelli, M. A., M. C. Clò: EDD Algorithm Performance Guarantee for Periodic Hard-Real-Time Scheduling in Distributed Systems. *IEEE IPPS/SPDP 1999.* pp. 668-677.

[3] Duffield, N.G., T.V. Lakshman, D. Stiliadis. On Adaptive Bandwidth Sharing with Rate Guarantees. INFOCOM '98. *Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings.* IEEE, 1998, pp. 1122-1130, vol. 3.

[4] Parekh, A. K., R. Gallager. A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single-Node Case. *IEEE/ACM Transactions on Networking*, June 1993, pp 344-357, vol. 1.

[5] Parekh, A. K., R. Gallager. A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Multiple Node Case. *IEEE/ACM Transactions on Networking*, April 1994. pp. 137-150, vol. 2.

[6] Stamoulis, A., and G.B. Giannakis. Deterministic Time-Varying Packet Fair Queueing for Integrated Services Networks. Global Telecommunications Conference, 2000. GLOBECOM '00. IEEE. pp. 621-625, vol.1.

[7] Stamoulis, A., G.B. Giannakis. Packet Fair Queueing Scheduling Based on Multirate Multipath-Transparent CDMA for Wireless Networks. INFOCOM 2000. *Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings.* IEEE, 2000. pp. 1067-1076, vol. 3.

[8] Stamoulis, A., and J. Liebeherr. GPS Scheduling with Graceful Rate Adaptation. Global Telecommunications Conference, 1998. GLOBECOM 1998. The Bridge to Global Integration. IEEE, Vol. 4, 1998. pp. 2489-2494, vol. 4.

[9] Tabatabaee, V., L. Georgiadis, L. Tassiulas. QoS Provisioning and Tracking Fluid Policies in Input Queueing Switches. INFOCOM 2000. *Nineteenth Annual Joint Conf. of the RENEE Computer and Communications Societies Proceedings*, pp 1624 - 1633 vol. 3.

[10] S. Tsao. Extending Earliest-Due-Date Scheduling Algorithms for Wireless Networks with Location-Dependent Errors. Vehicular Technology Conference, 2000. IEEE VTS Fall VTC 2000. 52nd, pp. 223-228, vol. 1.