Inequality Comparisons and Traffic Smoothing in Multi-Stage ATM Multiplexers

Michael J. Neely
MIT -- LIDS
mjneely@mit.edu

Charles E. Rohrs
Tellabs Research Center and MIT
crohrs@trc.tellabs.com

***Abstract* -- In this paper we examine the queuing behavior of a multi-stage multiplexer with fixed length packets flowing through the stages. The system consists of two components: a *front end* multi-input, multi-output device as a preliminary stage, and a single-server, deterministic service time queue system (multiplexer) as a final stage. We treat arbitrary exogenous arrival patterns and examine sample path characteristics of packet occupancy in the system. Under identical inputs, we compare the multi-stage system to the corresponding single-stage system without the front end. Treating both the infinite buffer (unlimited capacity) and finite buffer (fixed capacity) cases, we prove a two-part Multi-Stage Multiplexing Theorem. From the first part, we conclude that any type of multi-staging is "sub-optimal." However, from the second part we find that deterministic service time queues--if they need to be installed as front ends for a larger network--actually improve upon or *smooth* the data traffic for downstream nodes.**

I. Introduction

Deterministic service time queues are often used as models for packet switch multiplexers in communication networks that handle fixed length data packets. In particular, networks operating under ATM standards segment input data streams into fixed bit-length packets called *cells*. These cells have a deterministic service time T at each queuing node of the network, where T is inversely proportional to the server processing speed.

Often, a single server is used to provide service to many variegated data streams. Packets are multiplexed together and wait in a queue for their opportunity to be processed at the server. In large networks, data streams may pass through a *front end* consisting of several initial stages before being multiplexed together at the final node. In this paper, we examine these multi-stage systems and compare them to single-stage systems with the same inputs. We show that any type of *black box* front end device can only increase aggregate packet congestion and hence is *sub-optimal* in comparison to the single stage multiplexer alone. However, we also demonstrate that front ends consisting of deterministic service time queues naturally act to *smooth* traffic--making it better for downstream nodes to receive.

Much analysis of multi-stage deterministic service time systems has concentrated on tandem chains of queues and memoryless inputs ([1]-[3]). A broader class of inputs were treated in [4] for discrete time tandems. In [5], an average queue length analysis was performed for a more general tree network using the properties of memoryless inputs. Their approach relied on a lemma from [6] that used combinatorical analysis. A contribution in [7, 8] was an equivalent model theory for exact analysis of deterministic service time tree networks with arbitrary exogenous input patterns. The smoothing results in this paper are based upon the equivalency relationships developed in [7, 8], although this paper can be read independently.

The topic of traffic smoothing has been addressed in a different context in [9, 10], where the authors analyze tandems of queues with arbitrary input patterns and with leaky bucket traffic control throttles on each input link. In [7, 8], a concavity result for tree networks introduced the notion of traffic smoothing by demonstrating that fewer queues in parallel smooth traffic more than many queues in parallel. Several results in this paper build upon these ideas and formalize the notion of traffic improvement.

The inequality results developed here are based upon sample path analysis of both infinite and finite buffer multiplexing systems. This analysis is important for establishing analytical bounds on packet occupancy and overflow probabilities in large networks in terms of results for simpler one-queue systems. It also develops in a rigorous manner a qualitative understanding of how queue congestion changes as an existing network is modified or enlarged.

II. Multi-Stage Sub-Optimality

Consider a single-server, deterministic service time queuing system which acts as a packet switch multiplexer by processing incoming fixed length packets one at a time and sending them out over a high speed communication link (Fig. 1a). Now suppose that physical constraints require the packets to first pass through a front end device (Fig. 1b). The front end processes the packets in some manner before passing them to the multiplexer at the final stage. This front end could represent a portion of a data network that the packets first travel through, or it could represent preliminary queuing stages performing necessary aggregation of the input streams. It could also represent some *intelligent* device which senses the packet occupancy of the final queue as well as predicts future inputs, and holds packets back until the final queue is ready to receive them.

We wish to compare the two systems. What are the effects of including the front end? How do the total number of buffer

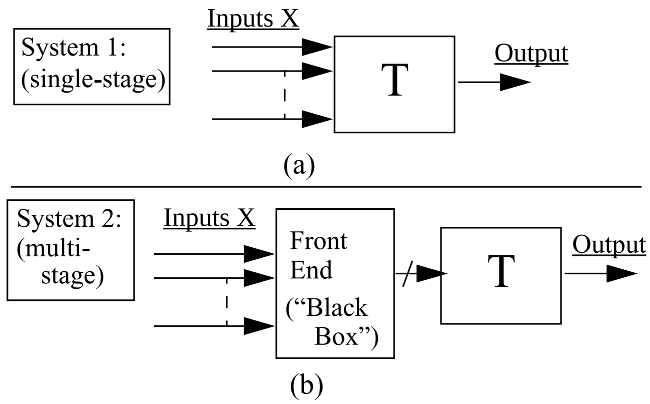

Fig. 1: A single and multi-stage system with identical inputs.

slots required in each system compare? Here we show that--under identical sets of inputs X--adding any form of front end device as a pre-stage is *sub-optimal* to the single-stage system. We make the following definitions for the two systems--System 1 and System 2--of Fig. 1:

$A(t)$ = The total number of packets that have entered the system by time t (same for both systems).
$D_i(t)$ = The total number of departures from System i by time t (i = 1, 2).
$L_i(t)$ = The total number of packets in System i at time t. (i = 1, 2).

Theorem 1: (*Infinite Capacity Multi-Stage Multiplexing Congestion Theorem*): If the systems of Figs. 1a and 1b have infinite buffer capacity and identical inputs, then the number of packets in the single-stage System 1 is always less than or equal to the number of packets in the multi-stage System 2, i.e., $L_1(t) \leq L_2(t)$ for all t. Consequently, $D_1(t) \geq D_2(t)$.

The proof of this Theorem can be found in [8]. ❑

From Theorem 1 we find that for infinite capacity systems, both the packet occupancy and the average waiting time in the multi-stage system is bounded below by the singe-stage characteristics. Often, infinite capacity analysis is used to determine buffer requirements. However, dynamical behavior of finite buffer (*packet-dumping*) systems can be quite different from their *non-packet-dumping* counterparts--particularly when buffer sizes are small. We are led to question whether the multi-stage sub-optimality notions gained from Theorem 1 hold for systems with limited buffering. To address this issue, we consider the two systems shown in Fig. 1 as finite buffer systems. We make the following definition:

$G_i(t)$ = The total number of packets that have been dropped into the "garbage" by system i up to time t (i=1,2).

Relating all of these sample functions, we clearly have the following *packet conservation equality:*

$$G_i(t) = A(t) - D_i(t) - L_i(t) \quad , (i = 1, 2). \tag{1}$$

Theorem 2: (*Finite Capacity Multi-Stage Multiplexing Sub-Optimality Theorem*): If the total number of buffer spaces available in the combined multi-stage system is less than or equal to the number available in the single-stage system, then:
(a) $D_1(t) \geq D_2(t)$ for all t.
(b) $G_1(t) \leq G_2(t)$ for all t.
Furthermore, if the number of buffer spaces available in the multi-stage system is *strictly less* than the number in the single-stage system, then the inequality in the loss functions is strict after the first packet loss, i.e.: $G_1(t) \leq G_2(t)$ for all times t, with equality *iff* no packets have been lost up to time t.

Proof: Suppose that the single-stage System 1 has a finite capacity of B packets. Suppose now that the multi-stage System 2 has a total number of memory slots for less than or equal to B packets.

Claim 1: Let t^* be some fixed point in time. If $D_1(t) \geq D_2(t)$ for all $t \leq t^*$, then $G_1(t) \leq G_2(t)$ on the same time interval.
Pf: If $D_1(t) \geq D_2(t)$ for all $t \leq t^*$ then from (1) we have:

$$G_1(t) + L_1(t) \leq G_2(t) + L_2(t) \quad (\text{for all } t \leq t^*). \tag{2}$$

Notice that at any time t_x when System 1 looses a packet, we know that the number of packets currently in that system must be B (i.e., the system must be completely full). Hence, at these times, from (2) we have:

$$G_1(t_x) + B \leq G_2(t_x) + L_2(t_x) \leq G_2(t_x) + B \tag{3}$$

Thus, $G_1(t_x) \leq G_2(t_x)$ at whatever times $t_x \leq t^*$ that System 1 looses a packet. Because $G_1(t)$ can only change when a packet is lost, and because the $G_i(t)$ functions are non-decreasing, we know that $G_1(t) \leq G_2(t)$ for all times $t \leq t^*$. This proves Claim 1. ❑

Claim 2: $D_1(t) \geq D_2(t)$ for all times t.
Pf: Let us define a *simultaneous renewal* as the event that Systems 1 and 2 both become empty. It suffices to prove the claim over a single renewal period (possibly infinite in duration). The first packet after a renewal event will cause $D_1(t)$ to increase either before or as $D_2(t)$ increases. We now show that there can be no *crossings*. For a crossing to occur, there must be some time t^* when $D_2(t)$ increases to become equal to $D_1(t)$, and the next departure after time t^* is from System 2 while $D_1(t)$ remains unchanged. At a candidate time t^*, there are two possible cases to consider:
A) $L_1(t^*) > 0$: In this case, since System 1 is not empty, it is currently serving a packet that will depart within the next T seconds. However, since System 2 just experienced a depar-

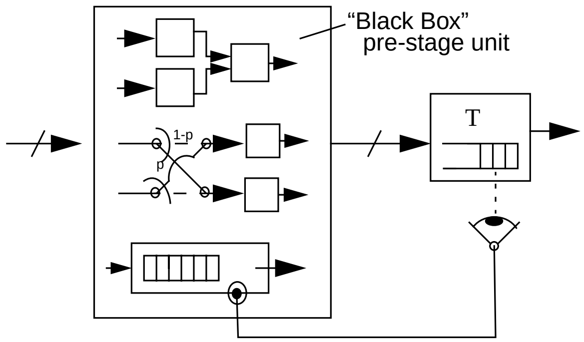

Fig. 2: A multi-stage multiplex system where the first stage is a "black box" with arbitrary packet processing devices.

ture, the next departure can only come after a period of time greater than or equal to T seconds. Hence, the $D_2(t)$ function *cannot increase first*, and so a crossing cannot occur.

B) $L_1(t^*) = 0$: In this case, from (1) we have:

$$L_2(t^*) + G_2(t^*) = L_1(t^*) + G_1(t^*) = G_1(t^*) \tag{4}$$

But from Claim 1 we know that $G_1(t^*) \leq G_2(t^*)$, and hence it must be that $L_2(t^*) = 0$. Thus, in this case both Systems 1 and 2 are empty, we have a simultaneous renewal, and we are done. This proves Claim 2. ❑

Claims 1 and 2 together prove parts (a) and (b) of the Theorem. Note that the final remark of Theorem 2 can be shown by repeating the above proof using strict inequalities. ❑

Notice that in Theorems 1 and 2, the black box is any arbitrary device which sends inputs to outputs. The box itself could consist of a single or multi-stage unit of queues or switches (see Fig. 2). Furthermore, the box could have some sensing device which looks at the state of the final stage queue, and signals the pre-stage to hold incoming packets back until the final queue is ready for them. In all of these examples, Theorems 1 and 2 apply and indicate that any such pre-stage device creates more total congestion and is hence sub-optimal.

III. Traffic Smoothing

Here we again consider Fig. 1, and concentrate on comparing the single-stage system to the final queue in the multi-stage system. As in the previous section, we continue to work with arbitrary input processes, although here we shall need to assume that all exogenous input processes from the different input lines are independent and stationary. Also, we replace the *black box* front end in the setup of Fig. 1 with a collection of tree networks of deterministic service time queues, each with service time T (Fig. 3b).

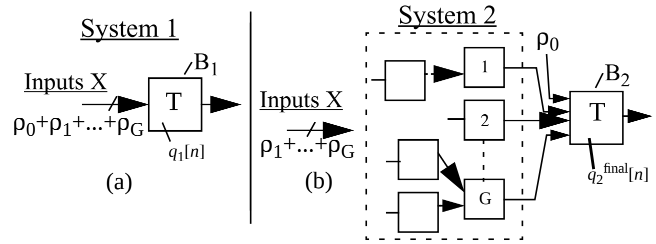

Fig. 3: The single-stage System 1 compared to the final node of a multi-stage tree network.

With this tree-type front end, we demonstrate that the queue at the last stage of the multi-stage System 2 will be less congested than the single-node System 1 handling the same traffic (Fig. 3). Thus, the preliminary stages of deterministic service time queues act as smoothing devices on the input data traffic and make it easier for the final node to handle.

As in Section II, we approach the problem using both infinite and finite buffer analysis. We find that infinite capacity analysis is a valuable tool even for analyzing finite buffer systems. We shall thus utilize the equivalence theory of tree networks with infinite capacity that was presented in [7, 8].

Equivalent Models: (From Theorem 1 in [7]) The 2-stage deterministic service time tree system in Fig. 4a below can be equivalently modeled as a single queue in tandem with a pure time T_1 delay or *observation window.* Here we assume $T_1 \leq T_2$.

Equivalence of the two systems is in terms of the input-output behavior. With identical inputs, the two systems of Figs. 4a and 4b produce identical outputs. It follows that at every instant of time, the number of packets in the entire two-stage System A is the same as the number of packets in the entire System B (observation window plus the final node).

Tree Reduction Principle: (From Theorem 2 in [7]) Consider multi-stage tree systems with deterministic, non-decreasing service times, such as the example in Fig. 3b. If arrivals from the different exogenous input lines are independent and stationary, then analysis of the final node in the tree is equivalent to the analysis of the final node in a 2-stage system where the original exogenous inputs are replaced by superpositions of these same inputs (Fig. 5b below).

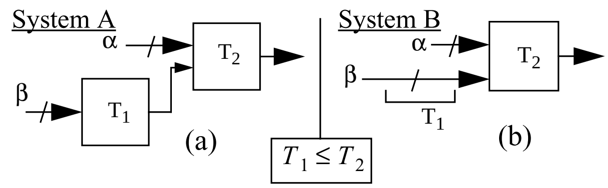

Fig. 4: Equivalent models for 2-stage deterministic service time systems with service times $T_1 \leq T_2$.

Using these properties of tree networks, we compare the two systems of Figs. 3a and 3b. We first assume all queue buffers have infinite capacity, and we define the steady state complementary occupancy probabilities $q_1[n]$ and $q_2^{final}[n]$ as the probability of finding more than n packets in the single-stage System 1 and in the final node of the multi-stage System 2, respectively.

Theorem 3: (*Infinite Capacity Smoothing Theorem*) If all nodes of the tree networks shown in Fig. 3 have infinite capacity, and if exogenous inputs are stationary and independent, then $q_1[n] \geq q_2^{final}[n]$ for all n.

Proof: We use the Tree Reduction Principle to reduce the tree configuration of Fig. 3b to the 2-stage system of Fig. 5b. We next use the equivalence theory to represent this system by an equivalent model, as shown in Fig. 5a.

By equivalence, the number of packets in the two systems of Figs. 5a and 5b is the same at every instant of time. However, it is also clear that the number of packets in the set of first stage queues in Fig. 5b is always greater than or equal to the number in the observation window of the equivalent model (Fig. 5a). It follows that the number of packets in the final stage of the system in Fig. 5b is always less than or equal to the number in the final stage of the equivalent system in Fig. 5a.

Now notice that this final stage of our equivalent model is exactly the same as the original single stage system of Fig. 3a when inputs $\rho_1,..., \rho_G$ are delayed by T seconds. Because inputs are independent and stationary, the stochastic nature of the input processes are not affected by such a time shift, and the delay is probabilistically unimportant. Hence, the single-stage system probabilistically has at least as many packets as the final node in the multi-stage system, i.e., $q_1[n] \geq q_2^{final}[n]$ for all n. ❑

We now turn our attention to the finite capacity case. Rather than showing that the number of packets in the final node of the multi-stage system is probabilistically less than the number in the single-stage system (which in general is *not* true for the finite capacity case), we focus on the number of packets that are dropped.

Consider again the two systems of Fig. 3, and suppose that

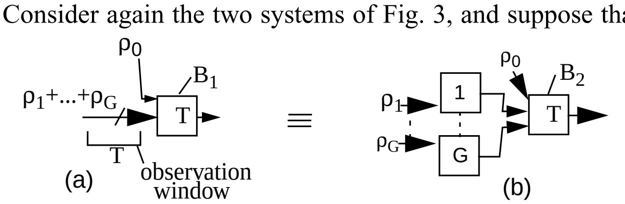

Fig. 5: The canonical 2-stage reduced tree system (b), and its equivalent model (a).

the single stage system and the final node in the multi-stage system have buffer sizes B_1 and B_2, respectively. We assume that the remaining pre-stage nodes of Fig. 3b have infinite capacity[1]. Further, we assume that inputs from the different lines in the single-stage system are independent and stationary, and that these same inputs are applied to the multi-stage system.

Theorem 4: (*Finite Capacity Smoothing Theorem*) For the finite capacity systems described above and illustrated in Fig. 3, if B_2>1 and if $B_1 \leq B_2$, then the packet drop rate in the single-stage system is lower bounded by the drop rate in the final node of the multi-stage system. Thus, to achieve a given drop rate threshold, the number of buffer slots required in the single-stage system is greater than or equal to the number needed in the final node of the multi-stage system.

Setup: We shift the inputs $\rho_1,..., \rho_G$ of the single stage system in Fig. 3a by T seconds by first passing them through an observation window, as shown in Fig. 5a. Because the input lines are independent and stationary, this timeshift does not affect the statistical dynamics of the queuing system. Furthermore, we assume that the multi-stage System 2 in Fig. 3b is first reduced to its equivalent 2-stage model (Fig. 5b) via the tree reduction principle. In this way, we convert our comparisons of the systems in Fig. 3 to comparisons of the two systems in Fig. 5.

As in Theorem 2, we define the monotonically increasing functions $A_1(t)$, $D_1(t)$, and $G_1(t)$ to be the total number of arrivals, departures, and packet drops, respectively, into and from the node of Fig. 5a during the time interval from 0 to t. Note that the arrivals from the $\rho_1,..., \rho_G$ streams first pass through the observation window at the first stage before they are counted as part of the $A_1(t)$ accumulated arrivals to the node. We also define $L_1(t)$ to be the current number of packets in this node (excluding any packets in the observation window). Likewise, we define $A_2(t)$, $D_2(t)$, $G_2(t)$, and $L_2(t)$ to be the corresponding values for the final node of the multi-node system in Fig. 5b. Note that (1) holds for these sample functions.

As a final definition for our setup, we say that the preliminary stages of the two systems we are comparing (the observation window of the system in Fig. 5a, and the G preliminary queues of the system in Fig. 5b) are in the *same state* when they have the same number of packets in them and the next departures from both will occur simultaneously.

[1] This is the most difficult case to consider. Clearly, if we change from an infinite capacity pre-stage to a finite capacity pre-stage, then the accumulated packet drops $G_2(t)$ at the final node cannot increase.

Proof: We show simultaneously that $D_1(t) \le D_2(t)$, and $G_1(t) \ge G_2(t)$ for all t.

Claim 1: If the preliminary stages of both systems are in the same state at time t=0, and if $D_1(t) \le D_2(t)$ for $0 \le t \le t^*$ for some t^*, then $G_1(t) \ge G_2(t)$ for the same time interval.

Pf: The proof of this claim is the same as the proof of Claim 1 in Theorem 2. ❑

Claim 2: $D_1(t) \le D_2(t)$ for all time t.

Pf: We define a *renewal event* to be the event that the second stage of the multi-node system empties, i.e., when the $L_2(t)$ function decreases to 0. By default, we consider time t=0 to be the first renewal event. Furthermore, we define a new *virtual process* $\tilde{D}_1(t)$ to be a version of the single-node system departure process that we modify in the following way: Whenever a renewal event occurs for the multi-node system, we immediately *shift out* all packets currently in the node at the second stage of the single-node system by sending them out as departures with zero service time. In this way, we ensure that at renewal times, the final stages of both the multi-node system and the modified single-node system are empty, and that the preliminary stages are in the same state.

Clearly this modified departure process $\tilde{D}_1(t)$ sends out packets more rapidly than the original, and we have:

$$D_1(t) \le \tilde{D}_1(t) \quad \text{for all time } t. \tag{5}$$

We now show that $\tilde{D}_1(t) \le D_2(t)$ for all time by checking that their incremental changes over every renewal period satisfies the same inequality. We define the incremental changes $a_2(t)$, $d_2(t)$, $g_2(t)$, and $l_2(t)$ to be the change in value of the arrivals, departures, packet drops, and packet occupancy in the final stage of the multi-node system from renewal time t_1 and for all time t up to and including the next renewal time t_2. For example, for $a_2(t)$ we have: $a_2(t) = A_2(t) - A_2(t_1)$ for $t_1 \le t \le t_2$. Let $\tilde{a}_1(t), \tilde{d}_1(t), \tilde{g}_1(t)$, and $\tilde{l}_1(t)$ be the corresponding incremental values for the modified single-node system. Notice that $\tilde{l}_1(t) = \tilde{L}_1(t)$, $l_2(t) = L_2(t)$, and also that the packet conservation equation (1) holds for these incremental functions. These incremental values can also be thought of as cumulative values of an unmodified system on the interval $t_1 \le t < t_2$, treating t_1 as if it were time 0. For this reason, Claim 1 holds true for the incremental functions, and thus we know that if $\tilde{d}_1(t) \le d_2(t)$ for all t such that $t_1 \le t < t_2$, then $\tilde{g}_1(t) \ge g_2(t)$ on the same time interval.

Notice now that whether or not any packets are dropped in the final node of the finite buffer multi-node system during the renewal interval in question, the incremental departures $d_2(t)$ on this interval will be the identical to what the departures would be if the system had infinite capacity and started out in the same state at time t_1. This is true because a finite buffer only alters a departure process at the time when a system empties after a packet has been dropped.

Because of the equivalence theory, this departure process $d_2(t)$ is also identical to what the departures would be in the single-node system if *it* were an infinite capacity system. However, these departures are lower bounded by the actual departures $\tilde{d}_1(t)$ for $t_1 \le t < t_2$ (because of the finite capacity B_1 in the single-node system). Hence:

$$d_2(t) \ge \tilde{d}_1(t) \quad \text{for} \quad t_1 \le t < t_2 \tag{6}$$

$$\tilde{g}_1(t) \ge g_2(t) \quad \text{for} \quad t_1 \le t < t_2 \tag{7}$$

where (7) follows from (6) by Claim 1.

It remains only to check the departure process inequality for time t_2. Here we check that inequality (6) also holds at time t_2 when the remaining packets in the node of the system of Fig. 5a are shifted out. At this time, we have:

$$\tilde{d}_1(t_2) = \tilde{d}_1(t_2^-) + \tilde{l}_1(t_2^-) \tag{8}$$

$$= \tilde{a}_1(t_2^-) - \tilde{g}_1(t_2^-) \tag{9}$$

$$= a_2(t_2^-) - \tilde{g}_1(t_2^-) \tag{10}$$

$$= d_2(t_2^-) + l_2(t_2^-) + g_2(t_2^-) - \tilde{g}_1(t_2^-) \tag{11}$$

$$= d_2(t_2) + g_2(t_2^-) - \tilde{g}_1(t_2^-) \tag{12}$$

$$\le d_2(t_2) \tag{13}$$

Equations (8) and (9) follow from the definitions of the incremental sample functions and of the virtual departure process $\tilde{d}_1(t)$. (Note that no arrival can occur at time t_2 because otherwise the multi-node system would not be empty then). Equation (10) follows because $\tilde{a}_1(t_2^-) = a_2(t_2^-)$ at any time t_2 when the final node of the multi-node system empties, since no packets arrived within the past T seconds and the arrival patterns are "re-synchronized" at this time. Equation (12) follows because at time t_2, when the final node of the multi-node system empties, the $d_2(t)$ value in (11) increases by 1 while the $l_2(t)$ value decreases by 1. Inequality (13) is obtained from (12) and (7).

Thus, the incremental departure process $d_2(t)$ is greater than or equal to the modified incremental departure process $\tilde{d}_1(t)$ at every time within an arbitrary renewal period. It follows that the total accumulated departures obey the same inequality, and we have:

$$D_2(t) \ge \tilde{D}_1(t) \ge D_1(t) \tag{14}$$

This proves Claim 2. ❑

Claims 1 and 2 together prove both that $D_1(t) \le D_2(t)$ and $G_1(t) \ge G_2(t)$ for all time *t*. ❑

Theorem 4 above proves that in order to ensure a certain low packet drop rate, more buffer slots are needed in a node whose traffic has not been smoothed than in a node whose traffic has been smoothed by deterministic service time tree networks.

IV. Conclusions

In this paper we have developed several ways of bounding the performance of large networks in terms of simple one-queue systems. We have used finite and infinite capacity analysis to prove a general, two-part Multi-Stage Multiplexing statement involving comparisons between single and multi-stage systems with identical inputs. The first part dealt with sub-optimality of multi-staging, and we showed that attaching any type of front end device to a deterministic service time queue cannot decrease aggregate waiting times or packet congestion. Rather, the aggregate congestion and the requisite number of buffer slots in the multi-stage system upper bounds those corresponding quantities in the single-stage system.

The second part of the Multi-Stage Multiplexing statement dealt with traffic smoothing. We demonstrated that if the front end of the multi-stage system is composed of a collection of deterministic service time tree networks of queues with identical service times, then the queues *smooth* or improve traffic for downstream nodes. In particular, the complementary occupancy probabilities of nodes with smoothed inputs lower bound those of nodes without the smoothing.

We found that the theory of infinite capacity systems was a valuable tool for understanding and analyzing systems with finite capacity. Our proof of the smoothing result for finite capacity systems relies heavily upon these tools. We showed using sample path arguments that a single-stage node with a finite amount of buffer space and a time shifted version of the original input stream deterministically drops more total packets than the same node with the tree network as a first stage. This shows that the drop rate and buffer requirement in a single-stage node upper bound the drop rate and buffer requirement in the same node within a multi-stage tree.

The proof techniques in this paper--those of defining renewal periods over sample path inputs--are very powerful for attacking queuing problems of a very general nature. The statements as well as the proofs provide insights into the dynamics of queuing networks.

Our smoothing results applied to multi-stage systems with fixed length packets. The processing speeds at each queuing node were assumed to be identical, and hence identical service times *T* were often assumed. All of our results are the same for the case of monotonically increasing service times $T_1 \le T_2 \le \ldots \le T$. However, decreasing service times complicate analysis and have not been addressed. We conjecture that deterministic service times do much to smooth traffic even in these cases. It would be interesting to discover the manner in which this smoothing takes place in decreasing service time systems, and the type of assumptions needed on the exogenous input processes in order for it to occur.

REFERENCES:

[1] M. Shalmon and M.A. Kaplan, "A tandem network of queues with deterministic service and intermediate arrivals," Operations Research, vol. 32, no. 4, July-August 1984.

[2] J. Morrison, "Two discrete-time queues in tandem," IEEE Trans. Comm., vol. COM-27, no. 3 March 1979.

[3] O.J. Boxma, "On a tandem queueing model with identical service times at both counters,I and II," Adv. Appl. Prob., vol. 11, pp. 616-643, 644-659, 1979.

[4] S.K. Walley and A.M. Viterbi, "A tandem of discrete-time queues with arrivals and departures at each stage," Queuing Systems, vol. 23, pp. 157-176, 1996.

[5] E. Modiano, J.E. Wielselthier, A. Ephremides, "A simple analysis of average queueing delay in tree networks," IEEE Trans. Info Th., vol. IT-42, no. 2, pp. 660-664. March 1996.

[6] J.A. Morrison, "A combinatorical lemma and its application to concentrating trees of discrete-time queues," Bell Sys. Tech. J., vol. 57, no. 5, pp. 1645-1652, May-June 1978.

[7] M. J. Neely and C. E. Rohrs, "Equivalent models and analysis for multi-stage tree networks of deterministic service time queues," Proceedings on the 38th Annual Allerton Conference on Communication, Control, and Computing, Oct. 2000.

[8] M.J. Neely with C.E. Rohrs, *Queue Occupancy in Single Server, Deterministic Service Time Tree Networks*. Masters Thesis, MIT LIDS, March 1999.

[9] A.W. Berger and W. Whitt, "The impact of a job buffer in a token-bank rate-control throttle," Commun. Statist.--Stochastic Models, 8(4), 685-717 (1992).

[10] A.W. Berger and W. Whitt, "The pros and cons of a job buffer in a token-bank rate-control throttle," IEEE Trans. Comm, vol. 42, no. 2/3/4, Feb/Mar/Apr 1994.

[11] Gallager, Robert G. *Discrete Stochastic Processes*. Boston: Kluwer Academic Publishers, 1996.

[12] D.P. Bertsekas and R. Gallager, *Data Networks*. Englewood Cliffs, NJ: Prentice-Hall, 1987.