Scheduling Packet Transmissions in a Multi-hop Packet Switched Network Based on Message Length

Eytan Modiano
MIT Lincoln Laboratory
244 wood st. M/S C437
Lexington, MA 02173
email: modiano@ll.mit.edu

Abstract

This paper describes two algorithms for scheduling packets in a multi-hop network. The objective of the algorithms is to reduce end-to-end *message (not packet)* transmission delays. Both algorithms schedule packet transmissions based on the length of the original message that the packet belongs to. The first algorithm is preemptive and is based on the shortest-message-first principle and the second is based on the shortest-remaining-transmit-time principle. We develop simulation models for analyzing the algorithms. The simulations show that when message sizes vary widely, these algorithms can significantly reduce average end-to-end message delays compared to First-Come-First-Serve scheduling.

I. Introduction

Network protocols, such as IP or ATM, transport data in packets. The algorithms used by these protocols, for routing, scheduling, flow and access control, are usually designed to satisfy performance measures at the packet level (e.g., average packet delay). However, user applications exchange data messages which are not limited in size to that of a network protocol packet. This separation between the application data and the way in which networks manage that data often results in sub-optimal performance when measured in terms of application layer messages. This paper describes packet scheduling algorithms that attempt to reduce end-to-end message delays by taking into account message information in the scheduling of packets across the network.

Network layer packets are typically limited in size to hundreds or thousands of bits. As a result, data messages are fragmented into multiple packets. These packets are then sent to the network layer protocol for delivery across the network. Typically, the network layer has no knowledge of the original message to which a packet belongs. As the packets traverse the network they are usually queued and served, independently of one another, on a First-Come-First-Serve (FCFS) basis at the different nodes along their way.

Serving packets on a FCFS basis may be a reasonable strategy if one is only concerned with packet delays. It is known that in a single node, average queueing delay is the same, regardless of scheduling policy, as long as scheduling is not done on the basis of packet length[1]. If packets are scheduled on the basis of their length, then it has been shown that a scheduling policy that serves the shortest packet first results in minimum delays[2, p.146]. However, from an end user point of view, message delays rather than packet delays is a more important performance measure. These messages can be as short as a few hundreds of bits for a short e-mail, to Giga-bits for an image file. In this paper we are concerned with the end-to-end delays experienced by these higher layer messages.

In considering the scheduling of messages at a single node, it is clear that when message sizes vary widely the scheduling of messages based on their size reduces average message delays considerably. As an example, in figure 1 we show the average queueing delay for a single node system with messages arriving randomly with exponential inter-arrival times. Half of the arriving messages are one cell in length and the other half are 100 cells in length. Shown in the figure is the queueing delay for two scheduling algorithms. One algorithm serves the messages on a FCFS basis and the other gives the short messages preemptive priority over the long messages. As can be seen from the figure the average delay for the priority system is much lower than the corresponding delays for the FCFS system.

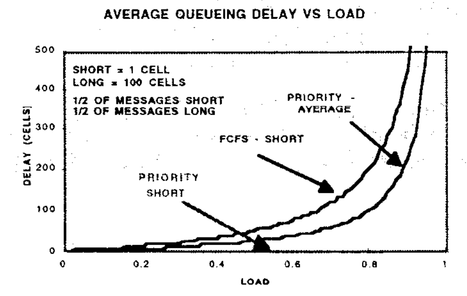

Figure 1. Average message delay in a single node system.

In general, it is known that for a single server system scheduling algorithms which give priority to shorter messages minimize average delays. In a non preemptive system the scheduling algorithm that minimizes average message delays gives priority to shorter messages over longer ones [2]. In a preemptive system the scheduling algorithm that minimizes average message gives preemptive priority to the message with the shortest remaining transmission time to completion [3].

While these scheduling algorithms were first studied in the context of processor sharing systems, they are particularly applicable to data networks. In data networks message length's are known in advance and can be used to assign priority to packets. Further, since large messages are broken down into small packets, or cells, an approximate form of preemption can be implemented by allowing messages to preempt one another only after the transmission of the current packet. So, a short message priority algorithm can be implemented by giving packets belonging to shorter messages priority over those belonging to longer ones. While this is not a pure form of preemptive priority scheduling, when packet sizes are much smaller than message sizes, it can achieve much of the benefit of a pure preemptive algorithm without incurring the added cost of preemption overhead.

In [4] this idea of scheduling messages based on the principle of giving preemptive priority to the message with the shortest remaining transmission time was applied to an Ethernet LAN. There it was shown that using this principle can significantly reduce message transmission delays. In [5,6] an exact analysis was developed for a single node network where messages are scheduled based on the shortest remaining transmission time principle. There, pure preemption was accomplished by fragmenting messages into packets whenever preemption was needed and adding a packetization overhead every time a message was preempted.

In this paper we study the implementation and performance of packet scheduling algorithms in a multi-hop network that are based on the above principles. Implementing these algorithms in a single node network is simple because the state of the system is always known (i.e., the length and remaining transmission times of all messages). However, in a multi-hop network, where messages arrive at intermediate nodes fragmented into packets so that the complete message information is not readily available, the design of scheduling algorithms based on message length information is not as straightforward.

In addition, the performance evaluation of these scheduling algorithms in a multi-hop network is complicated. In fact, even the computation of average message delays in a simple FCFS network is generally not analytically tractable. For FCFS scheduling some approximate models, such as Kleinrock's independence approximation [7] are available. However, with the addition of a complicated service discipline obtaining accurate analytical results becomes completely hopeless. We, therefore, resort to simulation to evaluate the performance of the scheduling algorithms developed in this paper.

In Section II of this paper we describe the scheduling algorithms that we developed and how they may be implemented in a multi-hop packet switched network. In section III we discuss simulation results of the algorithms in both a single node network and a simple multi-hop example network. Finally in section IV we discuss our directions for future work and conclusions .

II. Scheduling Algorithms

The scheduling algorithms we develop in this paper are based on the Shortest message preempt (SMP) and Shortest Remaining Transmission Time (SRTT) principles. All of the algorithms that we describe are preemptive in the sense that a stream of packets belonging to one message may be interrupted by packets belonging to another message. Therefore, these algorithms are preemptive at the message level, but not at the packet level. That is, no packet is interrupted in the middle of its transmission.

These algorithms are designed to be implemented at every switch in the network. However, their implementation is independent from switch to switch and requires no additional communication between the switches. Each switch implementing the algorithm is completely independent from other switches in the network. The algorithms do not rely on any information to be sent between the switches and, in fact, not all switches need to implement the algorithm.

The algorithms are based on the SRTT and SMP principles, but they are not pure SRTT or SMP. This is due to the fact that pure SRTT or SMP would require every node in the network to have the full information about each message. That is, every node would need to know the exact message to which every packet belongs and the state of that message (i.e., how much of it has already been transmitted). Since this information is not available at the network level, these algorithms approximate the behavior of SRTT and SMP by including some limited message information in the packet headers.

We discuss three algorithms. The first is simple First-Come-First-Serve (FCFS), which we use as the basis for comparison to the other two algorithms. The second algorithm is based on the Shortest-Message-Preempt (SMP) principle and the last is based on the Shortest-Remaining-Transmit-Time (SRTT) principle.

A. First-Come-First-Serve Packet Scheduling (FCFS-PS)

In FCFS-PS no effort is made to schedule the packets in a particular order. Packets are served based on the order in which they arrive at each node. Therefore, packets belonging to one message can be interrupted by those belonging to another message simply based on time of arrival. Essentially, FCFS-PS is the approach taken by packet switches where no attempt is made to schedule packets based on message information. We use the FCFS-PS algorithm as the basis for comparison of the other two algorithms.

It is very important to note that here FCFS-PS scheduling is not the same as FCFS message scheduling for which M/M/1 and M/G/1 queueing results apply. This is because in FCFS-PS packets and not messages are served on a FCFS basis. Therefore, in a multi-hop network messages naturally interrupt each other as packets belonging to different messages arrive on different input streams at overlapping time intervals.

B. Shortest Message Preempt Packet Scheduling (SMP-PS)

In pure SMP, shorter messages interrupt longer messages during their transmission and the longer messages resume their transmission once the shorter message is transmitted. Unlike pure SMP, SMP-PS only allows interruptions at the end of packet transmissions. Therefore if a packet belonging to a short message arrives during the transmission of a packet belonging to a longer message, it will not be transmitted until after that packet is transmitted. It will, however, be transmitted ahead of the rest of the packets belonging to the longer message.

In order to implement the SMP-PS algorithm, the network protocol must know the length of the message to which each packet belongs. This can simply be done with the use of a length field in the packet header. This length field will contain the length of the original message to which the packet belongs. There are many ways in which to represent the message length. The simplest approach, and the one which we use in our simulation, is to let the length field represent the number of packets contained in the message (up to a maximum number of packets). This information is usually readily available from the higher (e.g., transport) layer protocol. This approach is particularly effective in a network where all packets are of the same length (e.g., ATM).

The SMP-PS algorithm can be implemented with a priority queue where the priority of a packet is equal to the inverse of the length of the message to which it belongs. The server serves packets from their queue in accordance with their priority and newly arrived packets are inserted into the queue according to their priority.

For an example consider figure 2. On the top part of the figure the arrival of four messages is shown, according to the time in which the messages arrive. The first message to arrive (msg 1) contains 4 cells, the second message contains two cells, and the third and fourth messages each contains three cells. For the purpose of the example, in each cell we indicate the message number and the length of the message to which the cell belongs. On the bottom of the figure we show the order in which cells are served. The first cell served is the first cell of message 1. When message 2 arrives, since it is shorter than message 1, the two cells belonging to message 2 are served. Following those, the first cell of message 3, which is still shorter than message 1, is served. This cell is followed by the first cell of message 4. This is because messages 3 and 4 are of the same length and the first cell of message 4 arrives before the second cell of message 3. With similar reasoning the remaining cells of messages 3 and 4 are served in round-robin order. When messages 3 and 4 are done the last 3 cells of message 1 are served.

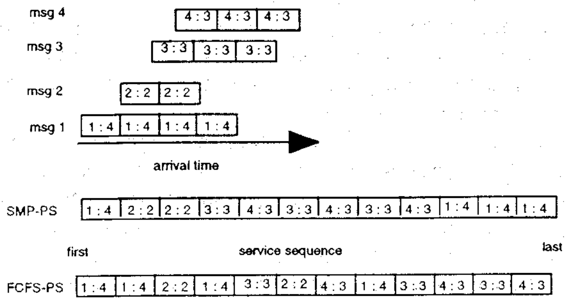

Figure 2. An example of SMP-PS scheduling.

For the purpose of comparison, we also show in figure 2 the service sequence that would have resulted from FCFS-PS. As can be seen from the figure, with FCFS-PS scheduling, message 1 would have been served sooner, but messages 2, 3 and 4 would have all taken more time. In this example, the average message delay for SMP-PS is 6.75 cells and for the FCFS-PS it is 8 cells. Although this is only an example, it illustrates the benefit of SMP-PS scheduling.

This example also illustrates one of the shortcomings of SMP-PS scheduling. Consider for instance the scheduling of messages 3 and 4. In this case the two messages are of the same length and arrive at overlapping time intervals. Since the two are of the same length and have the same priority, they are served in a round-robin order. This round-robin effect causes both messages to be unnecessarily delayed. It is obvious that if we allowed message 3 to be served to completion first, the delay for serving message 3 would have been reduced without affecting the message 4 delay.

Another related shortcoming of the SMP-PS algorithm is that messages can be interrupted (by shorter ones) at any time during their transmission. So when a long message is being transmitted, and a shorter message arrives, the shorter message interrupts the longer one even if the longer message is almost completely transmitted.

C. Shortest Remaining Transmission Time Packet Scheduling (SRTT-PS)

The obvious shortcomings of SMP-PS can be overcome by a simple variant of SMP-PS which gives priority to packets based on the remaining transmission time of the message to which they belong. In SRTT-PS scheduling, packets belonging to one message would have priority over those belonging to another message only if the remaining transmission time of the message to which they belong is shorter than the remaining transmission time of the other message. Again, as with SMP-PS, interruptions only occur at the end of packet transmissions. That is, no packet transmission is interrupted.

In order to implement a SRTT-PS algorithm, each node must have the complete message state information for every message in its buffer. This is a much more complicated task than what was required to implement SMP-PS. To implement SRTT-PS, not only must we tag each packet with its message length, but we must also tag each packet with a message identifier that will allow all nodes in the network to track the state of each message. Since this message identifier is only used for the purpose of scheduling, it does not have to be truly unique. Therefore, a simple message identifier tag can be implemented using random numbers. The length of the tag must be sufficient so that the likelihood of having two different messages with the same tag at a given node is low. However, even if two messages have the same tag the algorithm still works, except that for those two messages it reverts back to SMP-PS scheduling. Therefore it is not necessary to have a long identification tag.

In addition, for the purpose of simplifying the implementation, each packet can also contain a sequence number which gives the order of that packet within the message. This sequence number is used to determine how much of the message has already been transmitted through the node[1]. It is for use by the scheduling algorithm only and is not same as the transport layer sequence number used for the reordering of the packet stream.

Using the message identification number, the message length and the sequence number, the priority of packets belonging to a given message can be set as follows: At a given node's buffer, let Smin be the smallest sequence number of any of the remaining packet belonging to that message (i.e., all packets with smaller sequence numbers have already been transmitted) and let L be the message length. This tells us that Smin-1 packets of that message have been transmitted and L - Smin still remain to be transmitted (assuming that sequence numbers start with 0). Then the remaining transmit time associated with all of the packets belonging to the message can now be set to L - Smin and the priority of these packets will equal the inverse of the remaining transmit time.

For example consider figure 3. Shown in the figure is the same message arrival sequence as that used in figure 2. The sequence in which cells are served is indicated at the bottom of the figure. In each cell, within the service sequence, we indicate the message number followed by the remaining message transmit time. Notice that message 1, which arrived first, receives service first. Message 2, which is of length 2 cells is served next. After that, service resumes for message 1, which has a remaining service time of 3. This remaining service time is the same for messages 3 and 4, but since the message 1 cells arrived first they are served first. After message 1 is served message 3 is served. Notice again that message 3 is served to completion before we begin serving message 4. This is because as message 3 receives service the remaining transmit time is less than that of message 4 and therefore message 3 is not interrupted.

We see in this example how both the round-robin phenomenon and the interruption of nearly transmitted long messages exhibited by the SMP-PS algorithm is avoided with the use of the SRTT-PS algorithm. In this example, the average message transmit time is approximately 6.25 cells, a slight improvement over the SMP-PS algorithm.

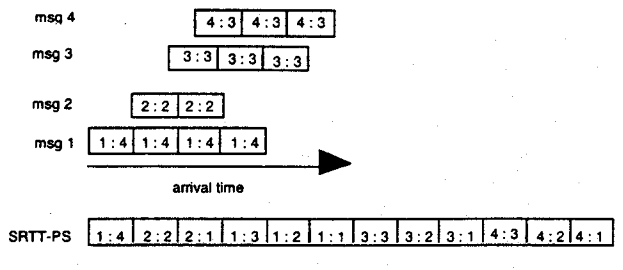

Figure 3. An example of SRTT-PS scheduling.

The example of figure 2 and 3 are useful for the purpose of illustrating the algorithms and their relative

[1] In a Virtual Circuit network, where all packets follow the same route, the sequence number can be used to determine exactly how much of the message remains to be transmitted through a node. Even in a datagram network, the sequence number can be used to approximate the remaining transmit time of a message through a given node.

advantages. In order to gain more insight into the performance of these algorithms we have to resort to simulations. In the next section we start with a discussion of simulation results for a single node network and then show simulation results for a network with a more complicated topology.

III. Performance analysis

We analyze the performance of these scheduling algorithms through the use of simulation. All of the simulations presented in this paper were developed using the Opnet simulation tool [10]. We start with a discussion of a single node system and then we describe the simulation results for an example multi-hop network.

A. Single node system simulation

The algorithms developed in the previous section, although designed for a multi-hop network, are also applicable to a single node system. In order to gain some intuitive understanding of these algorithms, we start with the simulation of a single node system. In all of our simulations we assume for simplicity that the network uses fixed size packets which we call cells. In [8] a more comprehensive discussion of message scheduling for a single node satellite broadcast system is presented. Here we simply present the simulation results for the system of figure 4, where a single source, generating messages of various lengths with exponential inter-arrival times, feeds a server with a buffer. The server, for the purpose of this example, has a service rate of one cell per second.

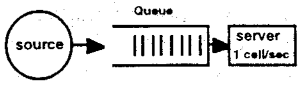

Figure 4. A single node system.

We start with a simple source which randomly generates messages of two sizes. Half of the messages are short messages which are only one cell in length and the other half are long messages which are 100 cells in length. Note that in this case, there is essentially no difference between the SRTT and the SMP scheduling algorithms because the round-robin phenomenon cannot occur in a single source network. Therefore, here we only show the results for the SRTT case. This simple case of two message sizes is clearly not realistic, but is useful for the purpose of illustrating the behavior of these algorithms. Also, it is somewhat representative of traffic in today's internet where users send short request messages and in response receive long messages (e.g., download of a long file).

Our source generates messages with exponential inter-arrival times. For the simulation results shown in figures 5-8 we use a message arrival rate of 0.0167 messages per second (inter-arrival time of 60 seconds), which is equivalent to an 85% load on the server. All of the figures show total system delay (queueing plus transmission) for complete messages over the entire simulation time of 100,000 seconds. Figure 5, shows the simulated message delay for the short messages when using FCFS. This is, perhaps, the most important figure in the sequence. What we see is the effect of having short messages queued behind the long messages. The average delay for a short (1 cell) message in the FCFS-PS system is 251 cells. In contrast figure 6 shows the equivalent delay when SRTT-PS is used. Since, with SRTT packets belonging to short messages get priority over long messages, we see in figure 6 that short messages experience minimal delay. The average short message delay for SRTT-PS is 1.31 cells (or seconds).

Despite this significant improvement in the delay for the short messages when using SRTT-PS, in figure 7 and 8 we see that the long message delay is almost the same for both systems. This is due to the fact that even though short messages preempt the long ones, the overall throughput of the short messages is insignificantly low to affect the long message delays.

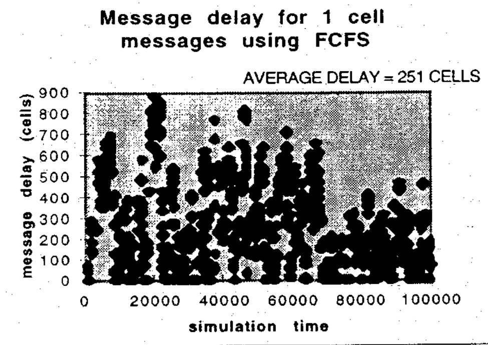

Figure 5. Short Message delay in a FCFS-PS system.

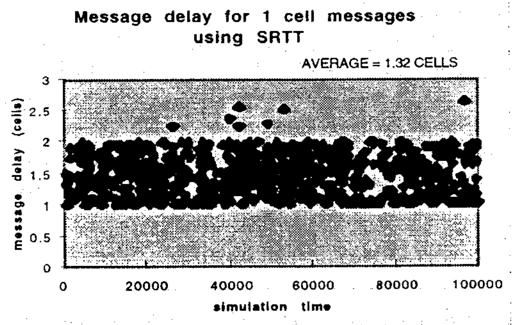

Figure 6. Short message delay in a SRTT-PS system.

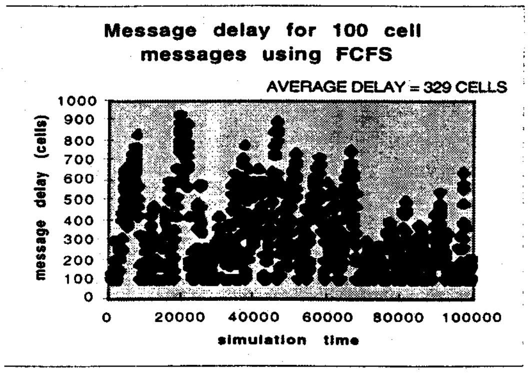

Figure 7. Long message delay in a FCFS-PS system.

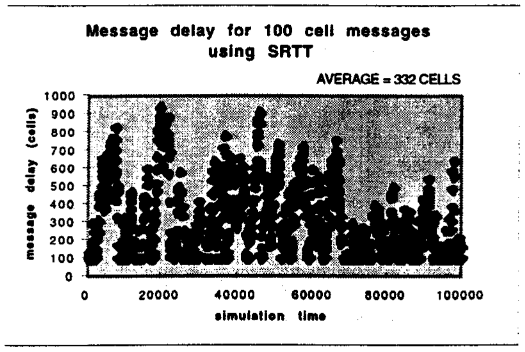

Figure 8. Long message delay in a SRTT-PS system.

The results from figures 5-8 were obtained for a system with a load of 85%. In figure 9 we show the average message delay vs. the load, for a wide range of load values, for both the short and the long messages. As can be seen from the figure, the average delay for the long messages is almost the same under the SRTT and FCFS service disciplines. However, the short message delay is a much smaller under the SRTT scheduling. This results in an overall reduction in average message delay of nearly a factor of two. Clearly these results are very encouraging for the SRTT-PS algorithm.

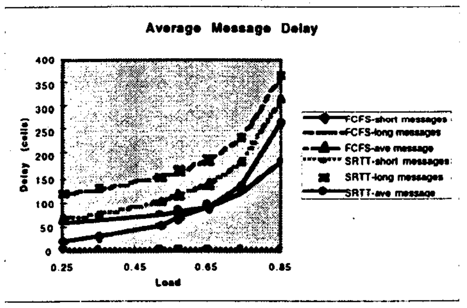

Figure 9. Average message delay vs. Load.

In order to obtain additional insight to the performance of the algorithm, we simulated the single server system with a wider range of message sizes. The simulation results shown in figures 10 and 11 are for a system with message sizes uniformly distributed between 1 and 100. That is, each arriving message is of size between 1 and 100 cells with equal probability for each size. In figure 10 we show the average delay vs. the load for all three service disciplines. As expected, the SRTT-PS algorithm results in the smallest delays, followed by the SMP-PS and the FCFS-PS algorithm. What we see from figure 10 is that the difference in the delay values is not as significant as in the two message size simulations. This can be attributed to the fact that in this simulation, short messages are a relatively small fraction of the overall traffic load. Therefore, although the delay for short messages is significantly reduced, their contribution to the overall average message delay is minimal.

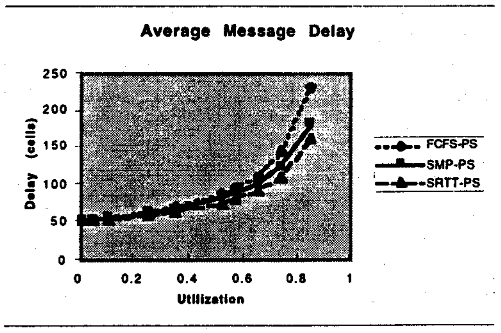

Figure 10. Average message delay vs. load for uniformly distributed message sizes.

B. Multi-Hop network simulation

While the single node network simulation shows many of the benefits of message based scheduling, it fails to capture some of the effects that occur in a multi-hop network. For example, in a single node network messages are typically assumed to arrive one at a time. However, in a multi-hop network messages arrive as packets; and different messages, traveling along different routes, can arrive at an intermediate node in overlapping time intervals. Consequently, the affects of a scheduling algorithm in a multi-hop network may not be the same as in a single node system.

In the simulations we use the network of figure 11. In this network we have 5 source nodes, all sending messages to the same destination. Each of the source nodes generates one fifth of the total traffic in the network. All of the links in the network have the same capacity, of one cell per second. As a result the bottleneck in the system is the link just before the destination node. Clearly, most of the queueing delays in this network will occur at the bottleneck link. We chose this example network in order to examine the performance of the scheduling algorithm, at an intermediate network node, where all of the traffic is in route traffic. This gives us the opportunity to examine the performance of the algorithm where messages are arriving at a node, fragmented into packets, at overlapping time intervals, rather than complete messages arriving one at a time.

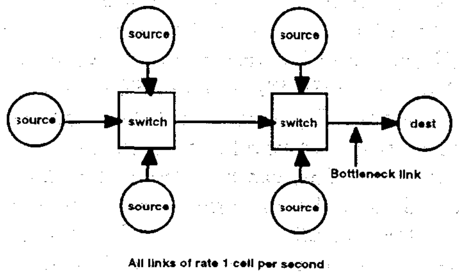

Figure 11. Network used in the simulations.

As we did with the single node system, our first simulation is for a very simple message length distribution, where each source generates messages of two sizes; short messages which are one cell each and long messages which are 100 cell each. In Figure 12 we show the average message delay vs. the network load at the bottleneck link for all three scheduling algorithms for both the long (100 cells) and the short (1 cell) messages. As expected, the average short message delay for both SMP and SRTT were essentially the same and substantially smaller than for the FCFS algorithm. For the long message delay, the SMP and FCFS algorithms performed nearly the same. The surprising result was that the SRTT resulted in the smallest average long message delay. This is despite the fact that SRTT gives preemptive priority to the shorter messages. This rather surprising result can be attributed to the fact that the SRTT algorithm eliminates the round-robin phenomenon that occurs in FCFS and SMP. While with FCFS and SMP messages that arrive in overlapping time periods are served in a round robin order, based on the arrival times of the cells belonging to those messages, in SRTT this round-robin effect is eliminated. Consequently, SRTT results in reduced message delays for both the long and the short messages.

In figures 13-14, we plot the actual simulated end-to-end delay for the short messages in the FCFS and SRTT algorithms. These plots show a short time interval during the simulation of system with an 85% load on the bottleneck link. Since the delays for SMP were very similar to SRTT and are omitted here for brevity. Similarly, the plots of long message delays for all three algorithms were very similar and are omitted for brevity. In figure 13 we see the message delay for short messages in the FCFS system. As can be seen from the figure, short messages are often delayed for 100's of cells, while they are "stuck" behind long messages in the buffer. In contrast, in figure 14 we show the short message delay for the SRTT system. Here we see that short messages take somewhere between 2 and 6 cells (seconds) to get through the network. This makes sense because the shortest route between a source and the destination is two hops.

Finally, in figure 15, we plot the average end-to-end message delay for a uniform message length distribution, where messages are uniformly distributed between 1 and 100 cells. Here, again, we see that the average message delay is minimized by the SRTT algorithm. Overall, the results that were obtained for the multi-hop network are very similar to those obtained for a single node system. These results are encouraging because they tell us that a simple implementation of the SRTT or SMP algorithms can reduce end-to-end message delays significantly.

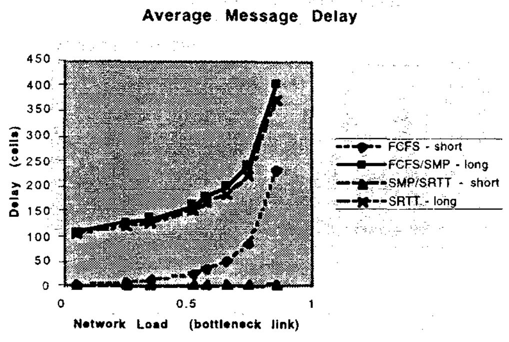

Figure 12. Average end-to-end message delay for a system with two message sizes.

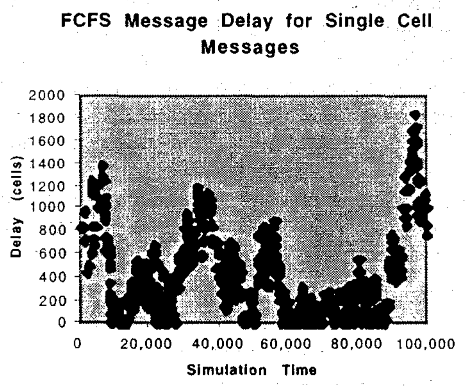

Figure 13. End-to-end short (one cell) message delay using FCFS scheduling with a load of 85%.

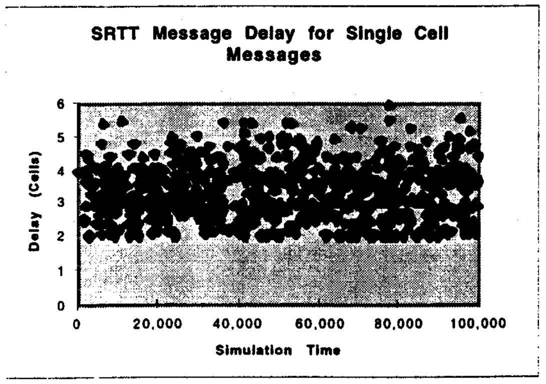

Figure 14. End-to-end short (one cell) message delay using SRTT scheduling with a load of 85%.

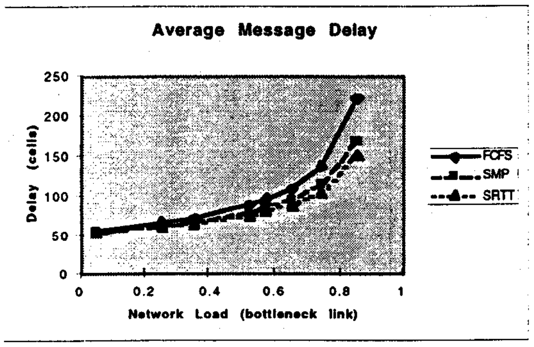

Figure 15. Average end-to-end message delay with uniformly distributed message lengths.

IV. Conclusions

This paper considers scheduling packets in a network based on the length and remaining transmission times of the messages to which they belong. We presented two algorithms for packet scheduling. The first, SMP-PS, schedules messages strictly based on their associated message length and the second, SRTT-PS, also takes into account the remaining transmit time of the message. Both algorithms are shown, through simulation, to significantly reduce the end-to-end message transmission delays, especially when message lengths in the network are widely varied. Our simulation results also show that the SRTT-PS is generally superior to the SMP-PS algorithm.

An interesting feature of both algorithms is that their implementation is done at each node independently. That is, no information needs to be exchanged between the nodes for the purpose of the scheduling algorithm. The only information that is needed to implement these algorithms is a simple tag that is added to each packet when it is initially generated at the source. In the case of the SMP-PS algorithm, this tag only contains the length of the original message. The SRTT-PS algorithm requires the tag to also contain a random message ID number and a sequence number for each packet within the message. In order to implement these algorithm in existing network layer protocols, this tag would have to be included in the network layer packet header. For example, in IP, this tag can be implemented as a header extension. A more detailed discussion of the implementation of these algorithms is beyond the scope of this paper, but is an important direction of future work.

Finally, additional work need to be done on the performance evaluation of these algorithms. At the very least, additional system simulation, with more varied traffic scenarios and more complicated network structures are needed. While analytical expressions for exact end-to-end message delays may be difficult to obtain, it would be useful to develop approximate expressions and performance bounds that give additional insight into the performance of the algorithms.

Acknowledgment

This work was sponsored by the Department of the Army under contract F19628-95-C-0002.

References

[1] Bertsekas and Gallager, Data Networks, prentice Hall, 1992

[2] L. Kleinrock, Queueing Systems, Vol. 2, John Weily and Sons, 1976.

[3] Schrage, L., "A Proof of the Optimality of the Shortest Remaining Processing Time Discipline," Operations Research 16, 1968.

[4] L. Schmickler and C. Goerg, "Performance Evaluation of a New CSMA/CD Protocol Based on the SRPT Principle," GLOBECOM '89, Dallas, TX.

[5] C. Goerg, "Evaluation of the Optimal SRPT Strategy with Overhead," IEEE Transactions on Communications, April, 1986.

[6] C. Goerg, "Further Results on a New Combined Strategy Based on the SRPT Principle," IEEE Transactions on Communications, May, 1990.

[7] L. Kleinrock, Communication Nets, McGraw-Hill, 1964.

[8] Eytan Modiano, "Scheduling Algorithms for Message Transmission in a Satellite Broadcast System," Submitted to MILCOM '97, Monterey, CA.

[9] A. Demers, S. Keshav, and S. Shenker, "Analysis and Simulation of a Fair Queueing Algorithm," Proceedings of ACM SIGCOMM '89.

[10] Opnet, MIL-3 Inc., 3400 International Drive NW, Washington, DC.