AT&T Bell Laboratories Technical Journal
Vol. 63, No. 6, July–August 1984
Printed in U.S.A.

Queueing and Framing Disciplines for a Mixture of Data Traffic Types

By A. G. FRASER* and S. P. MORGAN*

(Manuscript received September 22, 1983)

Packet-switched data networks are constructed from switching nodes interconnected by trunks. Trunk queueing delays for short messages can be reduced at the expense of long messages by having the trunk server take no more than a fixed number of bytes from each message before going on to the next message. We report analysis and simulations of two partial-service disciplines, namely Round Robin (RR) and Priority first-in first-out followed by Round Robin (PR + RR), for a mixture of traffic types. The PR + RR discipline permits short messages to experience finite mean delay at traffic levels where longer messages see infinite mean delay. Information is transmitted over the trunk in frames, where a frame may contain parts of several messages. At the far end of the trunk, the contents of a frame are not transmitted further until the end of the frame has arrived. We simulate two framing algorithms that work effectively with the PR + RR queueing discipline to achieve acceptably low frame overhead together with short delays for short messages. In addition, queueing plus framing delays for longer messages are substantially reduced, at a given overall traffic intensity if the access lines run more slowly than the trunk.

I. INTRODUCTION

Packet-switched data networks are constructed from switching nodes interconnected by trunks. For long-distance communication, 56-kb/s trunks are used. Traffic enters the network from computers

* AT&T Bell Laboratories.

and terminals that are connected to the nearest node by access lines operating at speeds ranging from 75 b/s to 1 Mb/s or more.

In a network based on the *DATAKIT** Virtual Circuit Switch,[1] traffic flows from several message sources through a node and out over a trunk line, as shown in Fig. 1. Messages generated by each source are transmitted over the access line character by character at the speed of the line. At the node the data are stored in one or more first-in first-out queues from which they are eventually taken for transmission over the trunk. The data from each source are transmitted on a different logical channel of the trunk. However, physical transmission over the trunk takes place in frames, where each frame may contain data from several sources in addition to framing information.

It is interesting to know the delay performance of the network under various operating conditions and with different choices of queue service discipline and framing discipline. Low delay is particularly critical for messages from interactive terminals and for the various control messages used in high-level protocols. Fortunately, messages that require low delay are usually only a few characters long, and we can use a queue service discipline that gives preference to short bursts of transmission. An advantage of using such a discipline is that it allows a trunk design that is appropriate regardless of user protocol.

Queueing delays for short messages generally can be reduced at the expense of long messages by using a separate first-in first-out queue for each channel, and by having the trunk server take no more than a fixed number of bytes, which we call a packet, from each queue before going on to the next queue. A short message following a long message in the same channel is necessarily delayed if we wish to preserve sequence, but this is not expected to create difficulties in practice. Two partial-service disciplines, shown in Fig. 2, are studied in this paper:

1. Round Robin (RR)—When a channel has data to transmit, the channel number is added to the end of a single list of channel numbers and works its way to the front, at which time one packet is transmitted from the given channel. If this does not empty the per-channel queue, the channel number is returned to the end of the list.

2. Priority first-in first-out followed by Round Robin (PR + RR)—This is a two-list discipline in which arriving channel numbers join a priority first-in first-out list, and then join a round robin if they require more than one packet of service. After transmitting each packet, the server checks to see if there are any channel numbers waiting in the priority list, and if so it attends to them first.

Successive packets are assembled by the trunk server into frames,

* Trademark of AT&T.

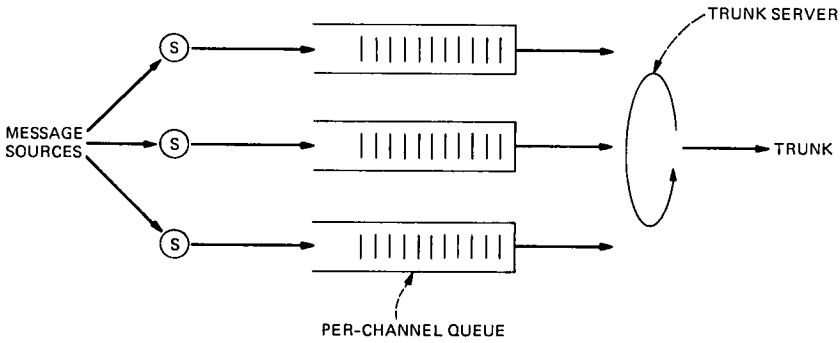

Fig. 1—Message queues at a trunk node.

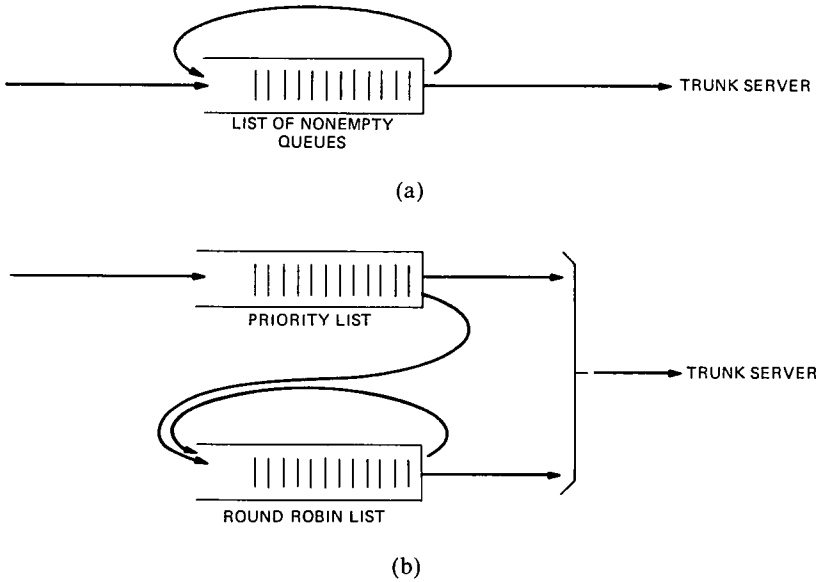

Fig. 2—Queueing disciplines. (a) Round robin. (b) Priority first-in first-out followed by round robin.

where a frame may contain data from several different channels. Each frame also includes a fixed number of framing bytes. A frame is terminated when it reaches a preset length, or when there are for the time being no data to send. At the far end of the trunk, the contents of the frame are held in a buffer until the end of the frame has arrived and the check bits have been verified. Optimal choice of frame length requires balancing the increased trunk overhead that is associated with short frames against the increased delay for short messages that is associated with waiting for the arrival of the end of long frames.

In this paper we are concerned only with the queueing plus framing delay at a trunk node. We do not include the times required to transmit the message over the access line and over the trunk; these partially concurrent times also contribute, of course, to the end-to-end delay of the network. Furthermore, we assume for simplicity that the access lines do not run faster than the trunk. The queueing plus framing delay is defined as the time that elapses between the arrival of the last character of the message in the per-channel queue (Fig. 1), and the time when the end of the frame containing the last character disappears into the trunk. In the limit of vanishing utilization and the absence of any packet or frame overhead, the delay so defined would be zero. A slightly more complicated definition of delay is needed if the access lines run faster than the trunk.

An analytical recipe for computing mean queueing delays for a mixture of message types, when the access lines run at the same speed as the trunk, has been given by Wolff[2] for the RR discipline. A similar recipe is given for PR + RR in the Appendix of the present paper. However, the analytical approach gives only mean delays and not the distribution of delay. Furthermore, it does not include the effects of framing, and it is not applicable to the practically important case where the access lines run more slowly than the trunk. Accordingly, the analysis has been augmented by extensive simulations.

In the numerical examples, we assume that the input traffic is a mixture of three types of messages, chosen to approximate single-character terminal-to-host transmissions from asynchronous terminals (10 percent), time-sharing host-to-terminal responses (40 percent), and host-to-host file transfers (50 percent). The assumed message characteristics are shown in Table I. On a 56-kb/s trunk, one character time is 0.143 ms.

In Section II we calculate mean trunk queueing delays, in the absence of framing, as a function of load when access speed is equal to trunk speed, for each message type. Most of the calculations are for a single packet size and overhead, since the qualitative effects of different packet sizes and overheads are reasonably easy to foresee. Three queueing disciplines are compared, namely First-In First-Out (FIFO), RR, and PR + RR.

Table I—Assumed traffic characteristics

Message Type	Length Distribution	Mean Length (bytes)	Relative Arrival Rate	Traffic Fraction
1	Constant	1	1	0.099
2	Exponential	40	1/10	0.395
3	Exponential	512	1/100	0.506

In ordinary FIFO or message-at-a-time service, where entire messages are transmitted in order of arrival, all messages see the same mean first-character waiting time regardless of length. The mean last-character delay is greater for longer messages if there is nonzero packet overhead. Disciplines that break messages up into smaller parts typically treat short messages better than long messages. In the case of PR + RR, sufficiently short messages are served entirely from the PR list, and they can experience finite mean delay at traffic levels where the RR is saturated by longer messages. These results appear both from the analytic solutions and from the simulations.

In Section III we simulate combined queueing and framing delay as a function of load, assuming the PR + RR discipline with access speed equal to line speed, and the traffic mix of Table I. For PR + RR, it is possible to impose a shorter average length on frames that contain data from the PR list than on frames that do not. We simulate two framing algorithms that accomplish this in slightly different ways, and we determine mean and 95th-percentile queueing plus framing delays for each message type.

Finally, in Section IV we consider the effect of low-speed access lines on queueing plus framing delays. For a given total load on the trunk, low-speed access lines smooth out the incoming traffic flow and reduce the mean last-character delay for long messages in the per-channel queue. We obtain mean queueing plus framing delays for each message type by simulating the PR + RR queueing discipline with one of the above-mentioned framing algorithms, for a nominal trunk utilization of 60 percent excluding overhead, and for various access speeds assuming that all access lines run at the same speed.

We conclude in Section V that the PR + RR queueing discipline, followed by either of the two framing disciplines, subjects single-character messages at a 56-kb/s trunk node to mean queueing plus framing delays of less than 20 milliseconds, together with acceptably low frame overhead, even when the trunk is heavily loaded with longer messages; and low-speed access lines reduce substantially the mean queueing plus framing delays for longer messages by smoothing out the access traffic. The numerical results are for a particular line speed and a specific traffic model, since it is notoriously difficult to explore a multidimensional parameter space by simulation; but the qualitative conclusions appear to be of much broader applicability.

II. QUEUEING DELAY

In this section we consider pure queueing delay on an unframed trunk, and assume that the access lines run at the same speed as the trunk. Then each message may be regarded as ready for transmission in its entirety as soon as its first character reaches the per-channel

queue. If the merged arrival process is Poisson, and if each message arrives in a separate channel so that it is not delayed behind another message in the same per-channel queue, then the messages constitute an M/G/1 queue of "customers" subject to whatever discipline the trunk server imposes.

Under FIFO service, the mean first-character waiting time is independent of message length and is given by the well-known formula

$$W = \frac{\lambda E(S^2)}{2(1 - \rho)},$$

where λ is the merged arrival rate, $E(S^2)$ is the second moment of message length in the merged message stream, and ρ is the effective traffic intensity (ratio of mean data plus overhead rate to raw trunk speed). The effect of per-packet overhead, as shown in the Appendix, is to increase ρ and $E(S^2)$ above the nominal values that are associated with the incoming message stream.

Figure 3 shows the mean first-character waiting time for message-at-a-time service on a 56-kb/s trunk, when the traffic mix of Table I arrives on 56-kb/s access lines. Mean waiting time in milliseconds is plotted against nominal utilization ρ', where ρ' is the ratio of mean user data rate to raw trunk speed. The three curves correspond, respectively, to 64-byte packets with no overhead (in which case the packet size is irrelevant), to packets with 64 bytes of data and 2 bytes of overhead, and to packets with 16 bytes of data and 2 bytes of overhead.

The curves of Fig. 3 have the form constant/$(1 - \rho'/\rho_0)$, where ρ_0 is the nominal utilization at which the sum of mean user data rate plus overhead rate is equal to the raw trunk speed. ρ_0 depends, of course, on the traffic mix and on the ratio of overhead bytes to data bytes in a packet. In the absence of overhead, $\rho_0 = 1$. For (64 + 2)-byte packets and the assumed traffic mix, $\rho_0 = 0.807$; and for (16 + 2)-byte packets, $\rho_0 = 0.757$. In general, ρ_0 will be substantially less than the ratio of data bytes to total bytes in a full packet, because if there are many short messages, there will be many packets with fewer than the maximum number of data bytes. Regardless of the number of data bytes, each packet contains a fixed number of overhead bytes.

Message-at-a-time service subjects long and short messages to the same mean first-character waiting time. Mean last-character delays depend on message length in the presence of per-packet overhead, since longer messages must wait for more overhead bytes to be transmitted; but the maximum difference is 9.3 ms in the numerical examples of Fig. 3, according to Table II of the appendix.

Rudin[3] has pointed out that breaking messages up into packets on the access lines, and then transmitting the access packets out of a

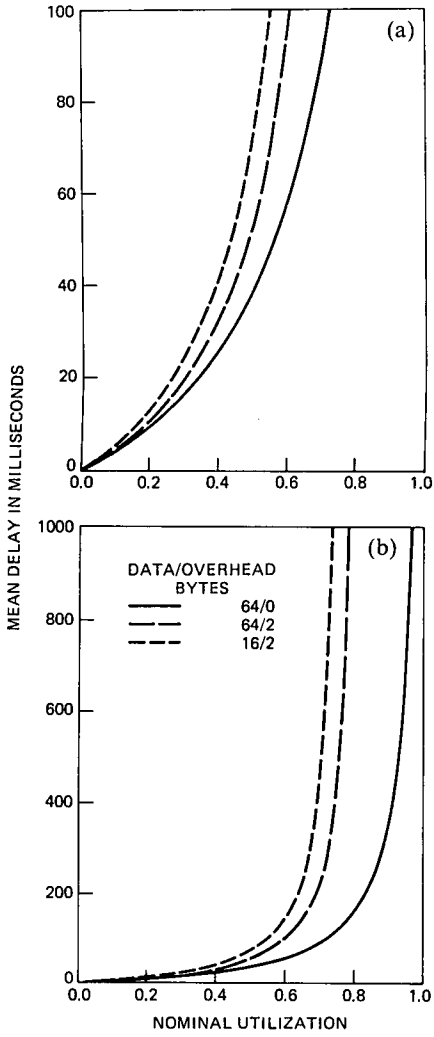

Fig. 3—Mean first-character waiting time for message-at-a-time service. (a) Light utilization. (b) Heavy utilization.

single FIFO, favors short messages to some extent; but in the present case it does not permit single-character messages to go in a few milliseconds as they should, while delaying file transfers for several seconds if necessary. To obtain the desired orders-of-magnitude discrimination, one must apparently resort to some such discipline as RR or PR + RR.

Wolff[2] has derived an infinite system of linear equations whose solution yields the mean delay for each message type in an M/G/1

queue with RR service. A similar system of equations is derived in the appendix for an M/G/1 queue with PR + RR service. In practical cases an approximate numerical solution may be obtained by truncating the infinite system to a finite system. In the unrealistic case of zero overhead, it is easy to solve the limiting case of infinitesimal packet size ("processor sharing"). Unfortunately this limit is of little interest in the present context, because data packets are not indefinitely divisible, and even if they were, the delays would go to infinity as the information per packet went to zero if the overhead per packet were a fixed nonzero amount.

Theoretical last-character mean delays are shown in Fig. 4 (dashed curves) for each message type under RR and PR + RR service, assuming (64 + 2)-byte packets and the traffic mix of Table I. As we expected, RR treats short messages better and long messages worse than ordinary FIFO, but the delays for all message types eventually saturate at the same point. PR + RR, on the other hand, pushes the delay curves farther apart; and the single-character delay does not go to infinity at the same point as the delay for longer messages. Only the RR saturates at this point, and single-character messages never see the RR.

Figure 5 shows the theoretical mean delays (dashed curves) for PR + RR with (16 + 2)-byte packets. Putting fewer data bytes in a packet with fixed overhead increases the difference in behavior of long and short messages and decreases the effective capacity of the trunk, because the average number of overhead bytes per message is increased. More generally, with the PR + RR discipline one could take different packet sizes for channels in different lists. Exploratory calculations of mean delays have not so far revealed any particular advantages to doing so, at least for the present traffic mix.

To deal with more complicated issues in the trunking of mixed data traffic, it appears necessary to simulate the desired disciplines. A simulator was therefore written to apply a variety of queueing and framing algorithms to a traffic model somewhat different from the M/G/1 queue assumed in the theoretical analysis.

The input to the trunk queueing simulator consists of a specification of one or more classes of access lines, in which n_i lines of class i each carry independent and identically distributed (i.i.d.) messages of mean length l_i separated by i.i.d. gaps of mean length g_i at a line speed s_i. The assumed configuration is like that of Fig. 1, so that under heavy loads a short message has some probability of being delayed behind a long message in the same channel. The message lengths may be deterministic (constant) or exponentially distributed; gap lengths are exponentially distributed. A number of other parameters such as trunk speed, packet sizes, overhead, queueing and framing discipline, and

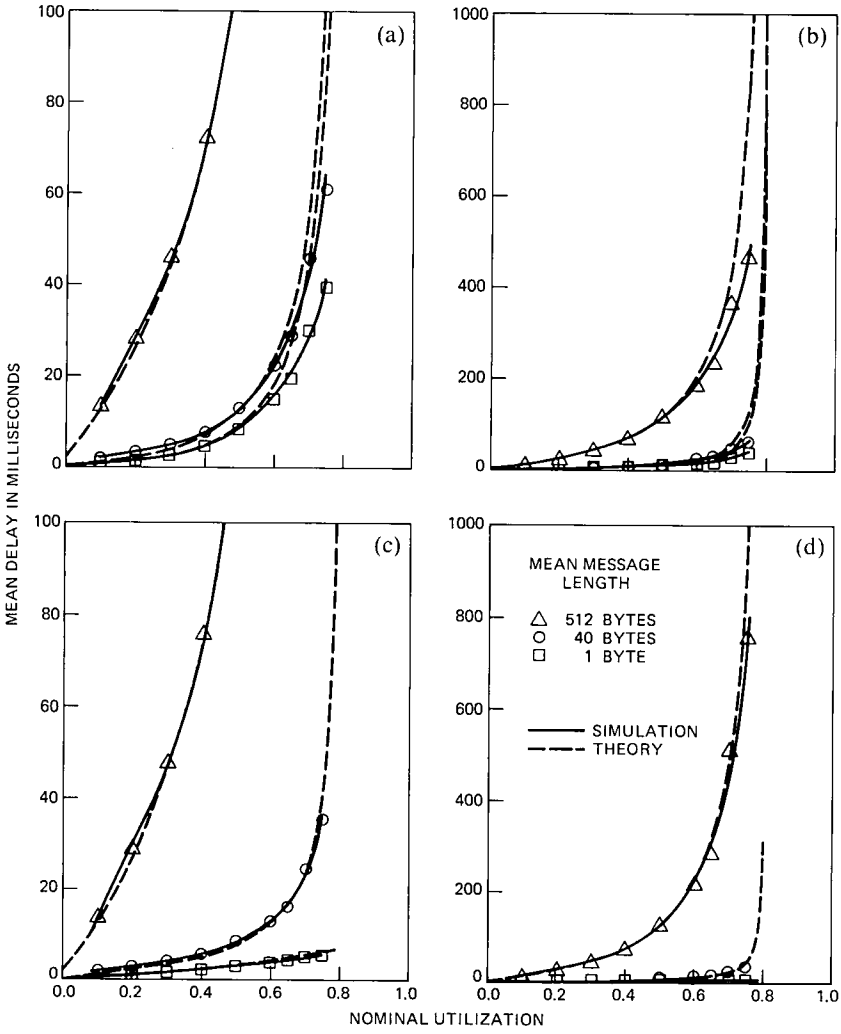

Fig. 4—Mean queueing delays for partial service disciplines. (a) Round robin low utilization. (b) Round robin high utilization. (c) Low utilization for priority first-in first-out followed by round robin. (d) High utilization for priority first-in first-out followed by round robin.

simulation time are set by flags or default options. The simulator starts with no messages in the system, and stops at the first regeneration (= empty and idle) point[4] after the specified simulation time has elapsed. Simulation outputs used in the present study are the mean delay for messages ot type *i*, and the 95th-percentile delay for messages of type *i*. The simulator can produce histograms of delay distributions for messages of each type, but we have not made use of the histograms.

A simulation is a probabilistic experiment. In order to obtain con-

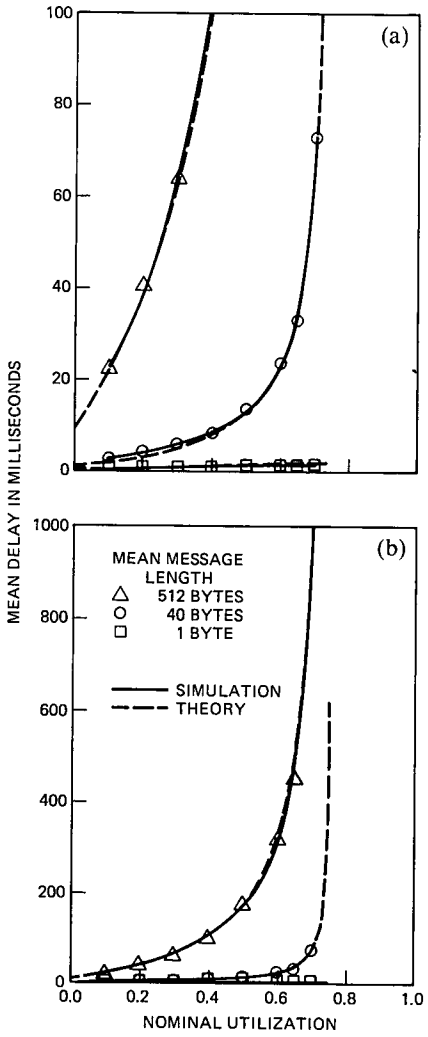

Fig. 5—Mean queueing delays for priority first-in first-out followed by round robin service with (16 + 2)-byte packets. (a) Low utilization. (b) High utilization.

fidence intervals, it is useful to divide the simulation time into a number of shorter simulations and look at the scatter of the results. We use the fact that the mean delay for a particular message type, averaged over a sufficiently large number of regeneration epochs, is asymptotically normally distributed.[5] Similarly, any particular quantile, such as the 95th percentile, of an arbitrary distribution is asymptotically normally distributed. Assuming that the mean delays from n simulations under "identical" conditions (except for the initial seeds of the random number generator) are normally distributed, we form

from the n sample means and their variance a variable having a t distribution with $n - 1$ degrees of freedom. The mean of sample means is taken as the estimate of the population mean, and a confidence interval is constructed from the t distribution. An estimate and a confidence interval for the 95th-percentile delay are calculated in a similar way.

In the simulations we more or less arbitrarily assumed 100 access lines for each of the message types of Table I (300 access lines altogether), and we adjusted the gap lengths to achieve the desired ratios of arrival rates and the desired nominal trunk utilization. In the longest runs, simulation times were chosen so that approximately 4000 type 3 messages would arrive (that is, approximately 40 messages on each type 3 access line) during the course of a single run. Twenty simulations were done for each set of parameter values, and the mean of the 20 sample means was taken as the final estimate of the mean delay (similarly for the 95th-percentile delay). The 90-percent confidence interval for this estimate was constructed from the variance of the sample of 20 means.

Not surprisingly, the delays for type 3 messages have the greatest uncertainty, with 90-percent confidence intervals as wide as ±5 percent (total width 10 percent) of the estimated value when access speed is equal to trunk speed and the trunk utilization is high. Furthermore, the absolute width of the confidence interval seems to be more or less independent of access line speed, so the relative uncertainty is greater for low-speed access lines.

To put matters in perspective, a single simulation run that includes about 4000 type 3 message arrivals takes about an hour of time on a large minicomputer, so the 20 runs necessary to get one point on the attached curves take about 20 hours and delineating the shape of a curve with 10 points may require 200 hours. Such a computation actually produces six curves, including both the mean and the 95th-percentile delays for all three traffic types, but it is still a substantial undertaking.

Why does it take so long? Basically, because the mean delay is proportional to the second moment of message length. In our case, almost the entire contribution to the second moment comes from type 3 messages, which constitute less than 1 percent of the total number of messages. We have to simulate long enough to see a substantial number of type 3 messages, while also doing all the bookkeeping for types 1 and 2 messages. As a check on our results, we have looked at published formulas[4] for confidence intervals in simulations of the mean delay in an M/G/1 queue, and have found that the predicted simulation times to achieve specified confidence intervals are quite comparable to the simulation times required in the present study.

Variance-reduction techniques, such as the method of control variates,[5] should probably be investigated if further simulations of this type are done.

How should the uncertainty in the results of the simulations be represented? One way would be to draw broken-line curves between the simulated points, and to plot the 90-percent confidence interval at each point. A more esthetic, if less scientific, visual impression is obtained by plotting the simulated points and using least squares to fit splines to the simulated points and to the upper and lower endpoints of the confidence intervals. (In cases where a function is expected to have a pole, such as queueing delays do at $\rho' = \rho_0$, the pole is taken out and least-squares fitting is applied to the numerator.) It must be emphasized that the upper and lower curves resulting from this procedure are *not* curves between which the mean delay has been proved to lie with 90-percent probability; they are only a qualitative indication of the uncertainty in the mean delay. "Bounds" are not drawn in most of the figures because they would essentially coincide with the mean value.

Results of simulating mean queueing delays, as a function of nominal utilization, for the RR and PR + RR disciplines on a 56-kb/s trunk with 56-kb/s access lines, are plotted as solid curves in Figs. 4 and 5. For those delays that saturate as a function of load, the pole is assumed to be at the same place as for message-at-a-time service.

Agreement between theory and simulation in Figs. 4 and 5 is good for light and moderate loads. For high loads, the simulated delays are less than the theoretical delays, especially for the RR discipline. This is understood qualitatively as follows: In the theory, the merged arrival process is assumed to be Poisson. In the simulations, the interarrival intervals on a single access line, being sums of an exponentially distributed or deterministic message length and an exponentially distributed gap, are not themselves exponential. The per-line arrival process is less bursty (fewer short intervals) than a Poisson process and would therefore be expected to lead to smaller queueing delays. The Palm-Khintchine theorem[6] states that the superposition of a sufficiently large number of arbitrary arrival processes approaches a Poisson process. However, Albin[7] has shown that under heavy traffic loads the Poisson limit may be approached very slowly. Some simulations with the same overall traffic divided among 600 access lines gave results closer to the Poisson values and suggested that we have encountered such an effect here.

III. FRAMING DELAY

The effect of framing on trunk delay is twofold. Framing overhead reduces the effective speed of the trunk for data transmission; and in

addition, at the far end of the trunk the last byte of a given message must wait to be transmitted further until the entire frame has arrived and the check bits have been verified. If there are a constant number of framing bytes per frame, the overhead effect is reduced by long frames and the waiting effect is reduced by short frames. For a given message length and a given trunk utilization, there is evidently a frame length that minimizes the combined queueing plus framing delay. The combined delay for long messages is minimized by long frames, and the combined delay for short messages is minimized by short frames. It would be desirable, therefore, to shorten just those frames that contain short messages.

We cannot recognize short messages *per se*, but we can achieve somewhat the same effect by shortening the frames that include data from the PR list. Two algorithms that accomplish this have been simulated.

3.1 Mixed frames

In this algorithm, complete packets are transmitted until the number of transmitted bytes exceeds a preset maximum. The framing bytes are then added and the frame is closed. The current frame is also closed if there are no packets ready to send. Different maximum frame lengths are imposed depending on whether the frame contains any data from the PR list. Thus a typical frame contains a mixture of PR and RR packets, but, on the average, frames that contain some PR packets are shorter than frames that contain only RR packets.

3.2 Sorted frames

In this algorithm, a flag is set whenever a frame contains any PR packets, and no further RR packets are added to such a frame. Thus the general form of a frame is a series of packets from the RR list followed by a series of packets from the PR list. In addition, maximum frame lengths are imposed, which may differ for frames that contain some PR packets than for frames that contain only RR packets.

Figure 6 shows the mean queueing plus framing delays for the PR + RR discipline with (64 + 2)-byte packets, as obtained by simulating each of the framing algorithms just described, and Fig. 7 shows the 95th-percentile delays. For mixed frames a nominal maximum of 64 bytes is imposed on frames containing any packets from the PR list, and 256 bytes on frames containing only packets from the RR list. Note that the actual maximum frame lengths will be greater than the nominal values because the last packet is not divided, and 5 framing bytes are added at the end. For sorted frames a nominal maximum of 256 bytes is imposed on both kinds of frame.

To see the effect of frame delay, one may compare Fig. 6 with the

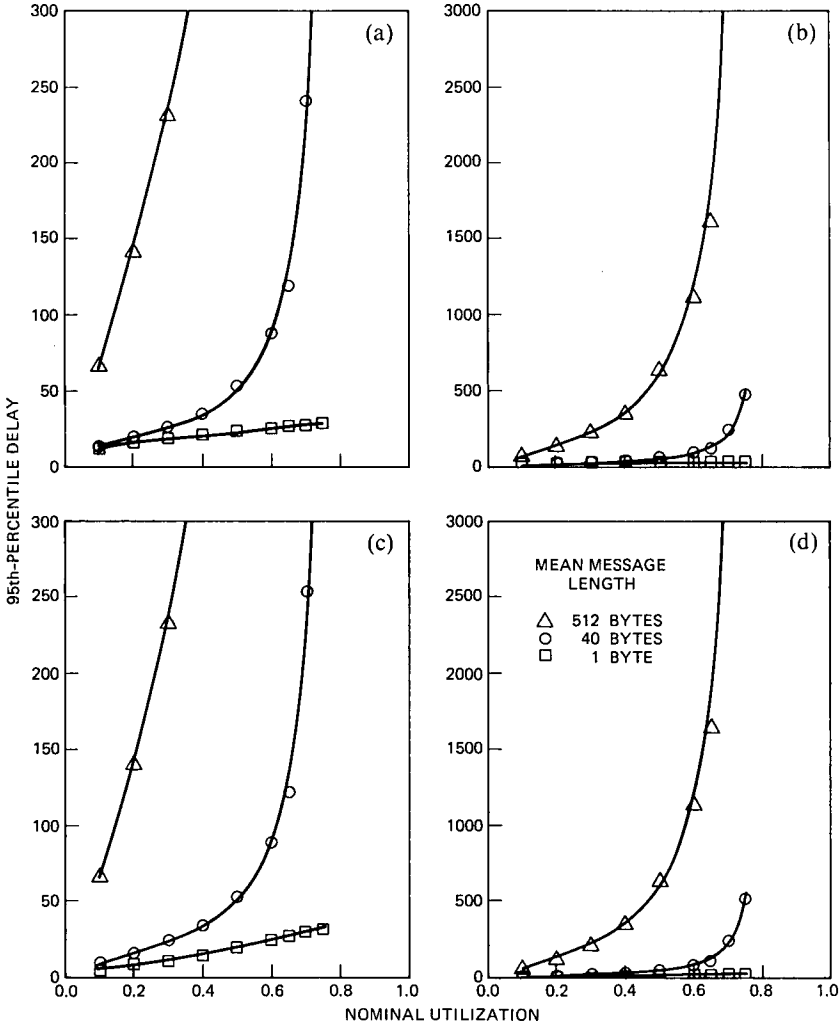

Fig. 6—Mean queueing plus framing delays. (a) Low utilization for mixed frames. (b) High utilization for mixed frames. (c) Low utilization for sorted frames. (d) High utilization for sorted frames.

PR + RR plots in Fig. 4, which show mean queueing delay separately. The addition of frame delay increases the total delay, most noticeably for single-character messages; but single-character messages still do not saturate at the same point as longer messages. (We do not have a theoretical value for the vertical asymptote in the presence of frame delay. Arbitrarily placing the pole at $\rho_0 = 0.80$ for the plots of Figs. 6 and 7 led to attractive curves using least-squares smoothing of the numerator.) The "sorted" algorithm with 256/256 limits looks about

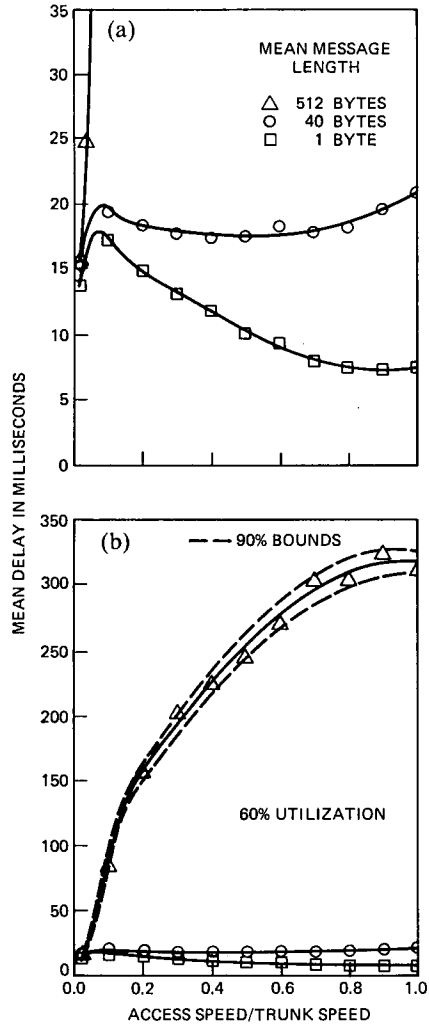

Fig. 7—95th-percentile queueing plus framing delays. (a) Low utilization for mixed frames. (b) High utilization for mixed frames. (c) Low utilization for sorted frames. (d) High utilization for sorted frames.

the same as the "mixed" algorithm with 64/256 limits for the present traffic mix. Conceptually, the sorted algorithm may be a little simpler.

IV. LOW-SPEED ACCESS LINES

Up to now we have assumed that access lines run at the same speed as the trunk. In practice, however, access lines will often be slower than the trunk, and for a given total load on the trunk the effect of

low-speed access lines will be to smooth out the traffic flow and substantially reduce the delay in the per-channel queue of the last character of a typical message. However, the end-to-end message delay generally will be increased, since the access-line transmission time is increased at the same time that the trunk queueing delay is decreased. In a sense the access line itself is being used as a buffer for the trunk queueing module. Rudin[3] and Anick, Mitra, and Sondhi[8] have considered the reduction in buffering requirements permitted by low-speed access lines.

Figure 8 shows simulated mean queueing plus framing delays for the PR + RR discipline with sorted frames, assuming the traffic mix of Table I, for various access speeds when all the access lines run at the same speed. A nominal trunk utilization of 60 percent was assumed together with (64 + 2)-byte frames. Cubic splines were fitted to the simulated points by least squares, and "90-percent bounds," as discussed in Section II, are shown for the type 3 delays.

Figure 8 shows that the delay for long messages falls off with decreasing access line speed. It falls off faster with decreasing access line speed for utilizations less than 60 percent, and more slowly for utilizations greater than 60 percent. As Rudin has pointed out, the curve would be expected to have a knee near the point where the mean time for transmission of a message over the low-speed access line is equal to the mean queueing time that would exist at the trunk for full-speed access lines.[3]

At the same time, the delay for short messages rises, up to a point, with decreasing access line speed. The reason for this is that if the access lines run much more slowly than the trunk, the trunk server generally empties the per-channel queue on each pass and, not recognizing that more of the same message is coming, allows the channel number to reappear in the PR list when it does not deserve to do so. Thus, nearly everything gets served from the PR list and distinctions between long and short messages are eroded.

One could discourage the same message from reappearing in the PR list so often by making each channnel number pass through the RR list after every service. The trunk server would delete the channel number from the RR list if there were nothing in the per-channel queue the next time the server got around to looking at it; and only then would the channel number be eligible to reappear in the PR list. This "PR + RR with hysteresis" discipline would treat single-character messages somewhat better at low access speeds, but it would also cause all three message types to saturate at the same load, like ordinary RR. The choice between PR + RR and PR + RR with hysteresis may ultimately depend on more detailed knowledge of the traffic to which the trunk nodes will be subjected.

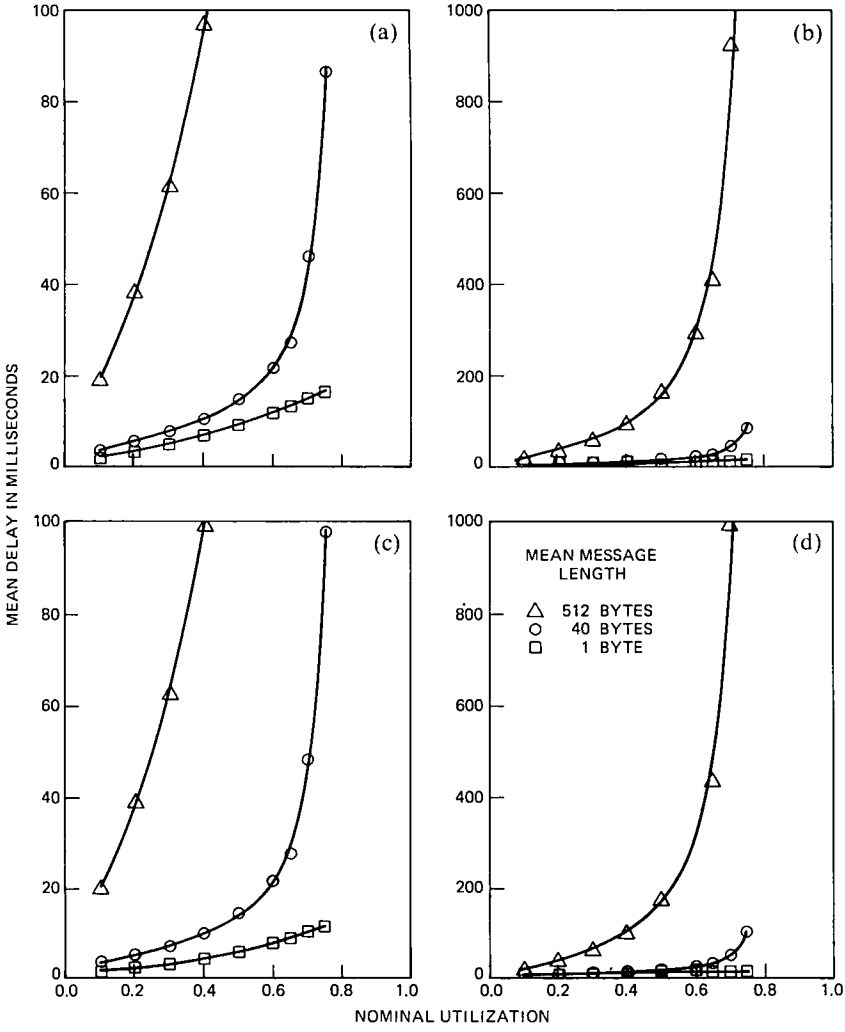

Fig. 8—Mean queueing plus framing delays for different access speeds. (a) Low utilization for sorted frames. (b) High utilization for sorted frames.

V. CONCLUSIONS

Some kind of partial-service discipline is essential if we want to transmit and receive single-character messages expeditiously in the presence of longer messages. The PR + RR discipline is particularly attractive because the RR list saturates before the PR list, so that single-character messages can still experience finite mean delay while longer messages see infinite mean delay.

Acceptable framing delays for both long and short messages can be

achieved by imposing different maximum lengths on frames that do and do not contain items from the PR list. For the assumed traffic mix, the PR + RR queueing discipline with 64-byte packets, together with the sorted framing algorithm with 256-byte frames, leads to mean queueing plus framing delays for single-character messages of less than 20 milliseconds at a 56-kb/s trunk node, even when the trunk is heavily loaded.

Trunk queueing plus framing delays for longer messages are substantially reduced if the access lines run more slowly than the trunk. At 60-percent nominal utilization, the reduction amounts to a factor of about 2 if the access speed is 20 percent of the trunk speed. The relative reduction is greater if the utilization is lower. Furthermore, increasing the trunk speed while holding the access speed and the trunk utilization constant reduces the trunk queueing plus framing delay by a more than proportional factor. In a practical system it would be advantageous to have the trunks run as fast as possible.

VI. ACKNOWLEDGMENTS

We are especially indebted to G. G. Riddle, who proposed both the PR + RR and PR + RR with hysteresis disciplines for the *DATAKIT* network, and with whom we have had extensive discussions. Thanks are also due to Ward Whitt for constructive comments on an earlier draft of this paper, and to a referee for drawing our attention to Ref. 3.

REFERENCES

1. A. G. Fraser, "Towards a Universal Data Transport System," IEEE J. Selected Areas in Commun., *SAC-1*, No. 5 (November 1983), pp. 803–16.
2. R. W. Wolff, "Time Sharing With Priorities," SIAM J. Appl. Math., *19*, No. 3 (November 1970), pp. 566–74.
3. H. Rudin, Jr., "Buffered Packet-Switching: A Queue With Clustered Arrivals," Int. Switching Symp. Rec., MIT (1972), pp. 259–65.
4. S. S. Lavenberg and D. R. Slutz, "Introduction to Regenerative Simulation," IBM J. Res. Develop., *19* (September 1975), pp. 458–62.
5. A. M. Law and W. D. Kelton, *Simulation Modeling and Analysis*, New York: McGraw-Hill, 1982.
6. D. P. Heyman and M. J. Sobel, *Stochastic Models in Operations Research*, Vol. I, New York: McGraw-Hill, 1982.
7. S. L. Albin, "On Poisson Approximations for Superposition Arrival Processes in Queues," Management Sci., *28*, No. 2 (February 1982), pp. 126–37.
8. D. Anick, D. Mitra, and M. M. Sondhi, "Stochastic Theory of a Data-Handling System With Multiple Sources," B.S.T.J., *61*, No. 8 (October 1982), pp. 1871–94.
9. R. W. Wolff, "Poisson Arrivals See Time Averages," Oper. Res., *30*, No. 2 (March–April 1982), pp. 223–31.

APPENDIX

Mean Delays for PR + RR Discipline

We shall derive a system of linear equations satisfied by the mean

delays in the PR + RR queueing discipline. The approach is modeled on Wolff's analysis[2] of the RR discipline.

In the mathematical model, jobs arrive in a Poisson stream. There are two queues served by a single server. Queue A has nonpreemptive priority over Queue B; the server does not start a job in Queue B if work is waiting in Queue A. An arriving job joins the end of Queue A, and when it reaches the server it receives a service quantum of up to δ_1. If additional service is required, the job joins the end of Queue B, works up to the server, receives service of at most δ_2, returns if necessary to the end of Queue B, on the next pass receives service of at most δ_3, and so on. Eventually the job completes service and leaves the system.

We allow for deterministic overhead at each service by modifying the service-time distribution of the incoming jobs. Finally, we assume that the incoming stream is a superposition of K independent Poisson streams, each with its own service-time distribution, and we write down expressions for the mean delay experienced by each job stream. Numerical results for exponential and deterministic distributions appear in Figs. 4 and 5.

A.1 Notation

λ = the arrival rate of jobs.
S = the service time of a job.
G = the distribution function of S: $G(t) = P\{S \leq t\}$.
G^c = the complement of G: $G^c(t) = 1 - G(t)$.
μ = the service rate, i.e., $E(S) = 1/\mu$.
ρ = λ/μ, where we assume $\rho < 1$.
G_e = the equilibrium distribution of G: $G_e(t) = \mu \int_0^t [1 - G(u)]du$.
S_e = a random variable with distribution $G_e(t)$. Note that $E(S_e) = E(S^2)/2E(S)$. Assume $E(S^2) < \infty$.
δ_j = the amount of time allocated to a job on its jth pass, $j = 1, 2, \cdots$.
Δ_j = the total time allocated to a job on its first j-passes:

$$\Delta_j = \sum_{i=1}^{j} \delta_i \qquad \text{for} \quad j = 1, 2, \cdots, \qquad \text{and} \quad \Delta_0 = 0^-.$$

j-job = a job that is completed on the jth pass, i.e., a job for which $\Delta_{j-1} < S \leq \Delta_j$.
p_j = the probability that a job is a j-job: $p_j = P\{\Delta_{j-1} < S \leq \Delta_j\} = G(\Delta_j) - G(\Delta_{j-1})$.
j-pass = a job that is either waiting in queue or in service and has completed $j - 1$ passes. Note that a j-job can be a $(j - 1)$-pass, but it is impossible for a $(j - 1)$-job to be a j-pass.

Q_j = the expected number of j-passes waiting in queue, in the time-average sense.

r_j = the expected delay (wait in queue) of a j-pass just prior to making the jth pass.

d_j = total expected delay (in queue) of a j-job: $d_j = \sum_{i=1}^{j} r_i$.

ν_j = the *virtual work* of a j-pass, meaning the expected amount of additional time required to complete processing a j-pass in queue (including, if necessary, time drawn from subsequent passes). Hence,

$$\nu_j = \frac{\int_{\Delta_{j-1}}^{\infty} G^c(t)dt}{G^c(\Delta_{j-1})}, \qquad j = 1, 2, \cdots. \tag{1}$$

ω_j = the expected amount of work performed on an $\{i: i \geq j\}$-job on the jth pass:

$$\omega_j = \frac{\int_{\Delta_{j-1}}^{\Delta_j} G^c(t)dt}{G^c(\Delta_{j-1})}, \qquad j = 1, 2, \cdots. \tag{2}$$

A.2 Equations satisfied by mean delays

Consider a particular job (the "tagged" job) arriving at Queue A. Because Poisson arrivals see time averages,[2,9] on arrival the tagged job encounters the following expected values:

Q_1 1-passes in Queue A

Q_j j-passes in Queue B, where $j = 2, 3, \cdots$.

$\rho E(S_e)$ residual service time of the job in service.

The expected delay before the tagged job reaches the server for the first time is

$$r_1 = Q_1\omega_1 + \rho E(S_e) - O, \tag{3}$$

where O, the expected overage, is the expected amount of work that will remain to be performed on the job in service when it is interrupted.

Consider the job in service at the instant the tagged job arrives ("job in service" always means this particular job). The job in service is a j-pass for some value of j. The arrival rate of jobs that will complete δ_j of service on the jth pass is $\lambda G^c(\Delta_j)$. This class of jobs receives a total of $\lambda G^c(\Delta_j)\delta_j$ service per unit time on the jth pass, so the probability that at a random instant a job is in service that is about to complete δ_j of service on its jth pass is $\lambda G^c(\Delta_j)\delta_j$. The expected overage for such a job is ν_{j+1}, as defined by (1). Hence the expected overage for the actual job in service, which can have any value of j, is

$$O = \sum_{j=1}^{\infty} \lambda G^c(\Delta_j)\delta_j \nu_{j+1} = \lambda \sum_{j=1}^{\infty} \delta_j \int_{\Delta_j}^{\infty} G^c(t)dt. \tag{4}$$

If a j-pass completes its service quantum δ_j, the probability that it will reach the kth following pass, for $k = 1, 2, \cdots$, is $G^c(\Delta_{j+k-1})/G^c(\Delta_j)$, and the expected service that it will receive on the kth following pass is, from (2),

$$\frac{G^c(\Delta_{j+k-1})\omega_{j+k}}{G^c(\Delta_j)}. \tag{5}$$

Hence the expected service that the job in service will receive on the kth following pass is

$$\sum_{j=1}^{\infty} \lambda\delta_j G^c(\Delta_{j+k-1})\omega_{j+k}\,. \tag{6}$$

Substituting (4) into (3) gives for the average delay of the tagged job in Queue A:

$$r_1 = Q_1\omega_1 + \rho E(S_e) - \lambda \sum_{j=1}^{\infty} \delta_j \int_{\Delta_j}^{\infty} G^c(t)dt. \tag{7}$$

Immediately after service in Queue A, the tagged job expects to find

$\lambda(r_1 + \delta_1)$	1-passes in Queue A
$Q_1G^c(\Delta_1) + Q_2$	2-passes in Queue B
Q_j	j-passes in Queue B, where $j = 3, 4, \cdots$
Job in service	(6) with $k = 1$ in Queue B.

By the time the tagged job has waited an expected additional time r_2 for its first service in Queue B, an expected λr_2 more 1-passes will have arrived and will have been served in Queue A. It follows that

$$r_2 = \lambda(r_1 + r_2 + \delta_1)\omega_1 + [Q_1G^c(\Delta_1) + Q_2]\omega_2 + \sum_{j=3}^{\infty} Q_j\omega_j + \lambda \sum_{j=1}^{\infty} \delta_j G^c(\Delta_j)\omega_{j+1}. \tag{8}$$

Now let us define

$$\begin{aligned} s_2 &= r_1 + r_2, \qquad s_j = r_j \quad \text{for} \quad j \neq 2, \\ Q_2' &= Q_1G^c(\Delta_1) + Q_2, \qquad Q_j' = Q_j \quad \text{for} \quad j \neq 2. \end{aligned} \tag{9}$$

Then eqs. (7) and (8) become

$$s_1 = Q_1'\omega_1 + \rho E(S_e) - \lambda \sum_{j=1}^{\infty} \delta_j \int_{\Delta_j}^{\infty} G^c(t)dt, \tag{10}$$

$$s_2 - s_1 = \lambda(s_2 + \delta_1)\omega_1 + \sum_{j=2}^{\infty} Q_j'\omega_j + \lambda \sum_{j=1}^{\infty} \delta_j G^c(\Delta_j)\omega_{j+1}. \qquad (11)$$

Now suppose that the tagged job has reached the server in Queue B for the lth time, where $l \geq 2$. Ahead of it on this circuit there were expected to be:

$\lambda(s_{l+1} + \delta_l)$	1-passes in Queue A

and the following in Queue B:

$\lambda(s_l + \delta_{l-1})G^c(\Delta_1)$	2-passes
$\lambda(s_{l-1} + \delta_{l-2})G^c(\Delta_2)$	3-passes
$\vdots$	
$\lambda(s_2 + \delta_1)G^c(\Delta_{l-1})$	l-passes
$Q_2'G^c(\Delta_l)/G^c(\Delta_1)$	$(l + 1)$-passes
$Q_3'G^c(\Delta_{l+1})/G^c(\Delta_2)$	$(l + 2)$-passes
$\vdots$	
$Q_{n+1}'G^c(\Delta_{l+n-1})/G^c(\Delta_n)$	$(l + n)$-passes
$\vdots$	
Job in service	(6) with $k = l$.

Adding all the expected times together gives, for $l \geq 2$,

$$s_{l+1} = \lambda \sum_{m=1}^{l} (s_{l+2-m} + \delta_{l+1-m})G^c(\Delta_{m-1})$$

$$+ \sum_{m=2}^{\infty} \frac{Q_m'G^c(\Delta_{l+m-2})\omega_{l+m-1}}{G^c(\Delta_{m-1})} + \lambda \sum_{m=1}^{\infty} \delta_m G^c(\Delta_{m+l-1})\omega_{m+l}. \qquad (12)$$

The next step is to express Q_j in terms of r_j. The arrival rate of j-passes is $\lambda G^c(\Delta_{j-1})$, and so by Little's law,

$$Q_j = \lambda G^c(\Delta_{j-1})r_j, \quad j = 1, 2, \cdots. \qquad (13)$$

It follows from (9) and the fact that $G^c(\Delta_0) = 1$ that

$$Q_j' = \lambda G^c(\Delta_{j-1})s_j, \quad j = 1, 2, \cdots. \qquad (14)$$

Substituting (14) into (10), (11), and (12), we obtain after some rearrangement

$$s_1 = A_1 s_1 + B_1,$$

$$s_2 - s_1 = A_1 s_2 + \sum_{j=2}^{\infty} A_j s_j + B_2,$$

$$s_i = \sum_{j=2}^{i} A_{i-j+1}s_j + \sum_{j=2}^{\infty} A_{i+j-2}s_j + B_i \quad \text{for} \quad i = 3, 4, 5, \cdots, \qquad (15)$$

where

$$A_i = \lambda G^c(\Delta_{i-1})\omega_i = \lambda \int_{\Delta_{i-1}}^{\Delta_i} G^c(t)dt, \quad i = 1, 2, \cdots, \tag{16}$$

and

$$B_1 = \lambda \int_0^\infty tG^c(t)dt - \lambda \sum_{j=1}^{\infty} \delta_j \int_{\Delta_j}^\infty G^c(t)dt,$$

$$B_i = \sum_{j=1}^{i-1} \delta_j A_{i-j} + \sum_{j=1}^{\infty} \delta_j A_{i+j-1}, \quad \text{for} \quad i = 2, 3, \cdots. \tag{17}$$

Note that if $\delta_j = \delta$ for all j,

$$B_i = \delta \sum_{j=1}^{\infty} A_j = \delta\lambda \int_0^\infty G^c(t)dt = \rho\delta, \quad i = 2, 3, \cdots. \tag{18}$$

Having solved (15) for the quantities s_j, one obtains the waiting times r_j easily from (9).

A.3 Mean delays for RR discipline

Wolff[2] has given the equations for the ordinary RR; they are

$$r_1 = \sum_{j=1}^{\infty} A_j r_j + B_1,$$

$$r_i = \sum_{j=1}^{i-1} A_{i-j} r_j + \sum_{j=1}^{\infty} A_{i+j-1} r_j + B_i \quad \text{for} \quad i = 2, 3, \cdots, \tag{19}$$

where the A's and B's are given as before by (16) and (17).

Note that one cannot get eqs. (19) by setting $\delta_1 = 0$ in eqs. (15) and renumbering. The two disciplines are slightly different even when zero service is given in Queue A of the two-queue discipline. In PR + RR, arriving jobs have to queue up in Queue A when a job is in service in Queue B, and they join Queue B immediately behind the job in service if it returns to the end of Queue B, even if the jobs that were queued in Queue A get no service there as a result of their wait. In RR, arriving jobs join the queue ahead of the job in service in case the latter has to return to the end of the queue. In practice one would hardly set up a two-queue discipline with zero service quantum in the first queue.

A.4 Overhead

Let us suppose now that the jth time the server attends to a particular job, the job gets up to δ_j' units of service and there are δ_j'' units of overhead. Define

$$\delta_j = \delta_j' + \delta_j'',$$

$$\Delta_j = \Delta_j' + \Delta_j'' = \sum_{i=1}^{j} \delta_i' + \sum_{i=1}^{j} \delta_j''. \tag{20}$$

Suppose that the job brings in the intrinsic service time distribution $F^c(t)$, where $F^c(t)$ is the probability that the service time S' excluding overhead exceeds t. If $G^c(t)$ is the probability that the effective service time S, including overhead, exceeds t, a little thought shows that

$$\begin{aligned} G^c(t) &= F^c(\Delta_j') \quad \text{for} \quad \Delta_j \leqslant t \leqslant \Delta_j + \delta_{j+1}'', \\ G^c(t) &= F^c(t - \Delta_{j+1}'') \quad \text{for} \quad \Delta_j + \delta_{j+1}'' \leqslant t \leqslant \Delta_{j+1}, \end{aligned} \tag{21}$$

for $j = 0, 1, 2, \cdots$.

It is now straightforward to express various quantities that we need for numerical calculations. For example,

$$\begin{aligned} E(S) &= \int_0^\infty G^c(t)dt \\ &= \sum_{j=0}^{\infty} \left[\int_{\Delta_j}^{\Delta_j + \delta_{j+1}''} F^c(\Delta_j')dt + \int_{\Delta_j + \delta_{j+1}''}^{\Delta_{j+1}} F^c(t - \Delta_{j+1}'')dt \right] \\ &= \sum_{j=0}^{\infty} \left[\delta_{j+1}'' F^c(\Delta_j') + \int_{\Delta_j'}^{\Delta_{j+1}'} F^c(t)dt \right] \\ &= E(S') + \sum_{j=0}^{\infty} \delta_{j+1}'' F^c(\Delta_j'). \end{aligned} \tag{22}$$

Similarly,

$$\begin{aligned} E(S^2) &= 2 \int_0^\infty tG^c(t)dt \\ &= E(S'^2) + \sum_{j=0}^{\infty} \left[(2\Delta_j \delta_{j+1}'' + \delta_{j+1}''^2) F^c(\Delta_j') \right. \\ &\qquad \left. + 2\Delta_{j+1}'' \int_{\Delta_j'}^{\Delta_{j+1}'} F^c(t)dt \right], \end{aligned} \tag{23}$$

$$\int_{\Delta_j}^\infty G^c(t)dt = \int_{\Delta_j'}^\infty F^c(t)dt + \sum_{l=j}^{\infty} \delta_{l+1}'' F^c(\Delta_l'), \tag{24}$$

and

$$A_j = \lambda \int_{\Delta_{j-1}}^{\Delta_j} G^c(t)dt = \lambda \int_{\Delta_{j-1}'}^{\Delta_j'} F^c(t)dt + \lambda \delta_j'' F^c(\Delta_{j-1}'). \tag{25}$$

These expressions simplify somewhat if δ_j' and δ_j'' are independent of j.

A.5 Merged job streams

Suppose that the incoming job stream consists of K streams of jobs arriving according to independent Poisson processes. Suppose that the arrival rate for the ith class is λ_i, and the intrinsic service time distribution for the ith class is $F_i^c(t)$. Then the merged arrival rate is

$$\lambda = \sum_{i=1}^{K} \lambda_i, \tag{26}$$

and the merged intrinsic service time distribution is

$$F^c(t) = \frac{1}{\lambda} \sum_{i=1}^{K} \lambda_i F_i^c(t). \tag{27}$$

This expression for $F^c(t)$ can go into all the previous machinery.

Finally, we would like to compute the mean waiting time in queue for the ith job class. Since the mean waiting time for a j-job is

$$d_j = \sum_{l=1}^{j} r_l, \tag{28}$$

the mean waiting time in queue for the ith job class is just

$$W_i = \sum_{j=1}^{\infty} d_j [F_i^c(\Delta_{j-1}') - F_i^c(\Delta_j')], \tag{29}$$

where the expression in square brackets is the probability that a random job from the ith stream is a j-job.

Note that in the presence of overhead the effective mean service time $E(S_i)$ will not be the same as the intrinsic mean service time $E(S_i')$. Hence the mean delay of the ith job class due to both queueing and overhead is

$$D_i = E(S_i) - E(S_i') + W_i, \quad i = 1, 2, \cdots, K. \tag{30}$$

A.6 Numerical cases

If we assume any particular form for the intrinsic message-length distributions $F_i^c(t)$, it is straightforward to calculate the A_i's and the B_i's from (16) and (17). For example, exponential and deterministic (constant-length) streams with intrinsic mean length $1/\mu_i'$ are given, respectively, by

$$F_i^c(t) = e^{-\mu_i' t},$$

$$F_i^c(t) = \begin{cases} 1, & 0 \leq t < 1/\mu_i', \\ 0, & t \geq 1/\mu_i'. \end{cases} \tag{31}$$

The relationship between the nominal server utilization ρ' and the effective utilization ρ in the presence of overhead is given by

$$\rho' = \lambda E(S'), \quad \rho = \lambda E(S), \tag{32}$$

where $E(S')$ and $E(S)$ are related by (22).

In the numerical calculations we have assumed three message streams as in Table I, and have taken the nominal utilization ρ' as the independent parameter. Trunk packets are 64 bytes or 16 bytes, and the overhead is 2 bytes. It turns out that for these numbers:

$$\begin{aligned} &\delta' = 64, \quad \delta'' = 2, \quad \rho' = 0.807\rho; \\ &\delta' = 16, \quad \delta'' = 2, \quad \rho' = 0.757\rho. \end{aligned} \tag{33}$$

It is perhaps less surprising that so much of the trunk capacity is consumed by overhead if we recall that every single-character message looks like a 3-character message when it is put on the trunk.

The relationship between intrinsic and effective mean message lengths in the presence of overhead is shown in Table II. Message lengths are expressed in milliseconds, using the fact that one byte time = 0.143 milliseconds on a 56-kb/s trunk.

Some words about the numerical solution of eqs. (15) and (19) are in order. The coefficient matrix of these equations is not sparse in the technical sense (that is, not mostly zeros); however, the coefficients do approach zero more or less exponentially with increasing distance from the main diagonal. Also, the contributions of high-order partial delays to the average waiting time in eq. (29) fall off exponentially. This suggests that we truncate the infinite system (15) or (19) to an $n \times n$ system where $e^{-n\mu_K'\delta'} \ll 1$, assuming that $1/\mu_K'$ is the longest average message length.

Not surprisingly, the numerical problem is easy if long messages typically fit into a few trunk packets ($\mu_K'\delta' \approx 1$), and hard if long messages require many packets ($\mu_K'\delta' \ll 1$). Since solving a system of n linear equations takes time proportional to n^3, halving the packet size multiples the solution time by 8. In the numerical examples of

Table II—Intrinsic and effective message lengths

Packet Size (bytes)			Mean Length (ms)		
Data δ'	Overhead δ''	Message Type	Intrinsic $E(S_i)$	Effective $E(S_i)$	Difference
64	2	1	0.143	0.429	0.286
		2	5.71	6.07	0.36
		3	73.1	75.6	2.4
16	2	1	0.143	0.429	0.286
		2	5.71	6.58	0.87
		3	73.1	82.4	9.3

Figs. 4 and 5, we found that $n = 75$ was an appropriate number of equations to solve for $\delta' = 64$, and $n = 300$ for $\delta' = 16$. Various checks indicate that the mean waiting times calculated with these truncations are in error by no more than 0.5 percent. One could solve larger systems if necessary, but in view of the simplified traffic model that we are using, there seems little reason to refine the computations any further.

AUTHORS

A. G. Fraser, B.Sc. (Aeronautical Engineering), 1958, Bristol University; M.A. (Computing Science), 1966, Cambridge University; Ph.D. (Computing Science), 1969, Cambridge University; Ferranti, Ltd. (now ICT Ltd.), 1959–1966; AT&T Bell Laboratories, 1969—. At Cambridge University, Mr. Fraser wrote the file system for the Atlas 2 computer. Since joining AT&T Bell Laboratories, his personal research interests have been the architecture of data communication networks and high-level languages for integrated circuit design. Since 1982 he has been Director of the Computing Science Research Center. Member, ACM, IEEE.

Samuel P. Morgan, B.S., 1943, M.S., 1944, Ph.D. (Physics), 1947, California Institute of Technology; AT&T Bell Laboratories, 1947—. Mr. Morgan is a member of the Computing Science Research Center. A research mathematician, he was originally concerned with applications of electromagnetic theory to waveguide and radar problems. From 1959 to 1967 he was Head, Mathematical Physics Department, and from 1967 to 1982 he was Director, Computing Science Research Center. His current interests include queueing and congestion theory in computer-communication networks. Member, American Physical Society, ACM, SIAM.