#180

ANALYSIS OF BUFFER ALLOCATION SCHEMES IN A MULTIPLEXING NODE*

Parviz Kermani and Leonard Kleinrock
Computer Science Department
University of California, Los Angeles
Los Angeles, California 90024

SUMMARY

We consider input streams of data sharing a buffer which is emptied by a single output channel and study different buffer allocation schemes in such a multiplexing node. These schemes range from Complete Sharing (CS) to Sharing with Minimum Allocation and Maximum Queue Length (SMAMXQ). In Complete Sharing a pool of buffers is shared by input streams without discrimination, whereas in Sharing With Minimum Allocation and Maximum Queue Length, besides a number of buffers allocated to each stream permanently, there is a common pool of buffers which can be shared by all input streams in such a way that there is a limit on the number of messages from each class that can be present in the shared pool simultaneously. For each scheme we present either explicit expressions or numerical algorithms to calculate different performance measures, namely, blocking probability, system throughput, utilization and delay.

1. INTRODUCTION

We consider a multiplexing node in which streams of messages from one or more input channels are buffered and merged into a single outgoing channel. As a result of finite storage size, some messages may be lost.

This type of system has been analyzed by others in one form or another, but almost all of the previous studies were based on the assumption that as far as allocation of buffer storage is concerned, there is no discrimination among messages from different input channels (which we call different classes of messages). In other words, a pool of buffers is completely shared among different classes of messages. The purpose of this paper is to analyze and compare some other storage sharing and/or allocation schemes.

The simplest scheme is Complete Sharing (CS), in which an arriving message is accepted if any buffer storage is available. In this scheme, all classes of messages are treated uniformly. The other extreme of buffer allocation, Complete Partitioning (CP), partitions the storage into fixed blocks of buffers and assigns each block to a class. In this scheme a message is rejected if the storage allocated to its class is full, even if the entire storage is not completely used. Intuitively, as long as throughput or probability of blocking is of concern, CS results in a better performance. That is because as long as there is a buffer available, it can be used by any incoming message, regardless of its class. CS results in a higher throughput and utilization; however, it suffers from two drawbacks. First, if input rates are asymmetric, the classes with higher input rates dominate the system and comsume most of the storage, and although the total throughput and utilization of the system is high, the contribution of the low input classes to this throughput may be low. In other words, the high input classes deny the others use of the system.[1] Another drawback is, that even with uniform input rates, if all input rates are scaled up simultaneously by a factor (say δ), as $\delta \to \infty$, in contrast to intuition, there is a nonzero probability that any one of the classes of messages will be absent from storage. This phenomena may not be desirable in certain situations, as we may require that in the case of infinite input rates there should be one or more messages of each class present in the system at all times.

The above considerations lead us to impose some restriction on the contention for space. The first drawback may be remedied by imposing a limit on the number of buffers that can be occupied by any class simultaneously. The scheme which incorporates this idea will be called Sharing with Maximum Queue (SMXQ). Even with SMXQ we are not guaranteed to have remedied the second drawback, the nonzero probability of one or more classes being absent from the system at a very high input rate. In order to remove this deficiency, we must allocate a minimum number of buffers to each class. This leads us to a scheme which has been called, Sharing with Minimum Allocation (SMA). In this scheme, besides the minimum allocated buffers to each class, there is a pool of buffers to be shared by different classes of messages. If this sharing is done without limitation, the scheme will be simply called SMA. If, however, we impose some limitation on the number of messages of each class that can be present in the system, then the scheme will be called Sharing with Minimum Allocation and Maximum Queue (SMAMXQ).

Statistical multiplexors have been modeled and analyzed by others (Chu[2] and the references therein), but the problem we are addressing here has not been studied before. Rich and Schwartz[3] and Drukey[4] made a preliminary study of buffer allocation in a Store and Forward (S/F) node. A comprehensive study of buffer sharing in S/F nodes was carried out by Kamoun and Kleinrock[1,5]. Although most of the terminology used in this paper is from these last works, the present study is different from theirs since in a S/F node, each class of input stream has a dedicated output channel, whereas in a statistical multiplexor a single output channel is shared by all input messages. This introduces major differences in the analysis.

In Section 2 we analyze these schemes and present either explicit expressions or numerical algorithms to calculate different performance measures, namely, probability of blocking, system throughput, etc. Finally in Section 3 we compare the performance of these schemes.

2.1 The Model

Our model consists of a single server (i.e., one channel) queueing system with finite waiting room of size B. The input to the system consists of R independent Poisson streams with rates λ_r $(r=1,2,\ldots,R)$. In order to analyze the system we require that all message lengths be distributed exponentially with the average length $1/\mu$ bits. Assuming the output channel speed to be C, the service time is exponentially distributed with rate μC. We further assume that arriving messages which are not accepted leave the system without service and the accepted ones are served on a first-come-first-served basis and that there is no priority involved.

If we characterize the states of the system by vector $\underset{\sim}{n} = (n_1, n_2, \ldots, n_R)$, where n_r is the number of class r messages in the system, and notice that the entire system is a birth-death process[6], then we are able to

*This research was supported by the Advanced Research Projects Agency of the Department of Defense under Contract DAHC 15-73-C-0368.

derive the steady state joint probability distribution. This distribution obeys the product form solution of the network of queues[7,8]. To save space we present only the final solution. For a derivation see [9].

$$P[\underset{\sim}{n}] = C_x n! \prod_{r=1}^{R} \frac{\rho_r^{n_r}}{n_r!} \quad \underset{\sim}{n} \in F_x \; ; \quad \sum_{r=1}^{R} n_r = n \tag{1}$$

where $\rho_r = \lambda_r/\mu C$ and $x \in \{1,2,3,4,5\}$.

The subscript x refers to the scheme we use. The integers 1 to 5 refer to CS, CP, SMXQ, SMA, and SMAMXQ respectively, and F_x is the set of possible states for for scheme x. C_x is a normalization constant defined so that the probabilities $P[\underset{\sim}{n}]$ sum to one.

$$C_x^{-1} = \sum_{\underset{\sim}{n} \in F_x} \left[n! \prod_{r=1}^{R} \frac{\rho_r^{n_r}}{n_r!} \right] \; ; \quad n = \sum_{r=1}^{R} n_r \tag{2}$$

Notice that C_x is also the probability that the system is empty.

In the following we will first calculate C_x and then other quantities of interest. In contrast to the analysis in [1], we will see that, except for the CS scheme, values of C_x cannot be found explicitly; as an alternative, however, we give efficient algorithms for their calculations.

Throughout the remainder of this section we will use the following notation:

a_r = Number of buffers allocated to class r messages (this is used in CP and SMA and SMAMXQ).

b_r = Maximum number of buffers of the shared storage which can be occupied by class r messages at any time.

B = Total number of buffers

B_s = Total number of shared buffers.

Notice that we have the following relationships: $B = B_s$ in CS; $B = \Sigma a_r$ and $B_s = 0$ in CP; $B_s = B - \Sigma a_r$ in SMA and SMAMXQ.

For each scheme we will consider two special cases, and for each case we will study the limiting behavior of the system. These cases are (1) $\lambda_r = \delta\lambda_r^o$, $r=1,2,\ldots,R$ and $\delta\uparrow\infty$; and (2) only one of the input rates, λ_i, grows to infinity.

2.2 Complete Sharing (CS)

The set of feasible states for this scheme is

$$F_1 = \{\underset{\sim}{n} \mid 0 \le n_r \le B, \quad r = 1,2,\ldots,R\} \tag{3}$$

The equations describing behavior of the system are well-known[6] (an M/M/1 queueing system with finite waiting room) and we report them here.

$$C_1^{-1} = P(0)^{-1} = \frac{1-\rho}{1-\rho^{B+1}} \tag{4}$$

$$PB = \Pr[\Sigma n_r = B] = \frac{1-\rho}{1-\rho^{B+1}} \rho^B \tag{5}$$

where $\rho = \lambda/\mu C$ and $\lambda = \Sigma \lambda_r$.

Notice that probability of blocking is the same for all classes of messages.

The marginal distributions can easily be found[9]

$$\Pr[n_r = k] = P(0) \sum_{n=k}^{B} \left[\binom{n}{k} \rho_r^k (\rho - \rho_r)^{n-k} \right] \tag{6}$$

The total average number of messages in the system is

$$\overline{n} = \frac{\rho}{1-\rho} \; \frac{1-(B+1)\rho^B + B\rho^{B+1}}{1-\rho^{B+1}} \tag{7}$$

and by using Little's result[10], the average time spent in the system for the accepted messages is

$$T = \frac{1/\mu C}{1-\rho} \; \frac{1-(1+B)\rho^B + B\rho^{B+1}}{1-\rho^B} \tag{8}$$

For the marginal statistics we have the following:

$\overline{n}_r$ = E[number of class r messages in the system]

$$\overline{n}_r = \frac{\lambda_r}{\lambda} \overline{n} \tag{9}$$

and

$$T_r = T \tag{10}$$

Special case (1): $\lambda_r = \delta\lambda_r^o$, $r = 1,2,\ldots,R$ and $\delta\uparrow\infty$.

We have the following results:

$$\lim_{\delta\uparrow\infty} \Pr[\underset{\sim}{n}] = \begin{cases} 0 & \text{if } n = \Sigma n_r \ne B \\ \dfrac{B!}{\rho^{oB}} \displaystyle\prod_{r=1}^{R} \frac{(\rho_r^o)^{n_r}}{n_r!} & \text{if } n = \Sigma n_r = B \end{cases} \tag{11}$$

In particular

$$\lim_{\delta\uparrow\infty} PB = \lim_{\delta\uparrow\infty} \Pr[\Sigma n_r = B] = 1$$

also

$$\lim_{\delta\uparrow\infty} \Pr[n_r = k] = \binom{B}{k} (\rho_r^o/\rho^o)^k (1-\rho_r^o/\rho^o)^{B-k} \tag{12}$$

Further, we have

$$\lim_{\delta\uparrow\infty} \Pr[n_r = 0] = (1-\rho_r^o/\rho^o)^B \ne 0 \tag{13}$$

Eq. (13) reveals an interesting fact, namely, even if the input rates are scaled up to infinity, there is a nonzero probability that one (or more) of the classes of the messages are absent from the system. This is a distinct drawback of the CS scheme.

Special Case (2): $\lambda_i\uparrow\infty$. From Eqs. (4 - 6) we get the following results:

$$\lim_{\lambda_i\uparrow\infty} \Pr[\Sigma n_r = k] = \begin{cases} 0 & k \ne B \\ 1 & k = B \end{cases} \tag{14}$$

$$\lim_{\lambda_i\uparrow\infty} \Pr[n_r = k] = \begin{cases} 1 & k = 0 \\ 0 & k \ne 0 \end{cases} , \; r \ne i \tag{15}$$

$$\lim_{\lambda_i\uparrow\infty} \Pr[n_i = k] = \begin{cases} 0 & k \ne B \\ 1 & k = B \end{cases} \tag{16}$$

$$\lim_{\lambda_i\uparrow\infty} \lambda_r' = \lim_{\lambda_i\uparrow\infty} \lambda_r'(1-PB) = \begin{cases} 0 & r \ne i \\ \mu C & r = i \end{cases} \tag{17}$$

These results show how the system can be dominated by a single class of high-rate messages.

2.3 Complete Partitioning (CP)

The set of possible states will be

$$F_2 = \{\underset{\sim}{n} \mid 0 \le n_r \le a_r , \quad r = 1,2,\ldots,R\} \tag{18}$$

and we have $B = \Sigma a_r$.

In this case the computation of the normalization term C_2 is not as simple as C_1, and in fact, no explicit expression for this constant has been derived. Notice that the summation in

$$C_2^{-1} = \sum_{n\epsilon F_2} n! \prod_{r=1}^{R} \frac{\rho_r^{n_r}}{n_r!} \quad , \quad n = \Sigma n_r$$

is taken over $\sum_{n=0}^{B} \binom{R = n - 1}{n} = \binom{R + B}{B}$ possible system states and a direct summation is out of the question. Unfortunately, because of the n! term we cannot even use the generating function approach as in [1,11,12], and we must resort to numerical techniques. These techniques have been studied by others and interested readers can refer to [13,14,15,16,17]. However, the problem at hand is different from theirs and we must modify their techniques.

We define the following set of states:

$$S(n,m) \overset{\Delta}{=} \{\underset{\sim}{n} = (n,n_2,\ldots,n_m) \mid \sum_{r=1}^{m} n_r = n,\ 0 \le n_r \le a_r,\ r = 1,2,\ldots,m\} \quad (19)$$

S(n,m) as defined above is the set of all system states in which only the first m classes may be present in the system and the total number of messages in the system is n. Notice that $F_2 = \bigcup_{n=0}^{B} S(n,R)$ and we have

$$C_2^{-1} = \sum_{n=0}^{B} \left[\sum_{\underset{\sim}{n}\epsilon S(n,R)} n! \prod_{r=1}^{R} \frac{\rho_r^{n_r}}{n_r!} \right]$$

we also define an auxiliary function u(n,m) as

$$u(n,m) \overset{\Delta}{=} \sum_{\underset{\sim}{n}\epsilon S(n,m)} \left[n! \prod_{r=1}^{m} \frac{\rho_r^{n_r}}{n_r!} \right] 0 \le n \le a_m^* \text{ and } 0 \text{ if } n > a_m^*$$

where $a_m^* = \sum_{r=1}^{m} a_r$, and clearly we have $B = a_R^*$.

C_2 is related to u(n,m) according to the following equation:

$$C_2^{-1} = P(0)^{-1} = \sum_{n=0}^{B} u(n,R) \quad (20)$$

We also have

$$Pr\left[\sum_{r=1}^{R} n_r = n\right] = P(0)u(n,R) \quad (21)$$

In what follows we will present an algorithm to calculate iteratively u(0,R), u(1,R)... , U(B,R). Following the approach of [13] we have

$$u(n,m) = \sum_{k=0}^{n} \left[\sum_{\substack{\underset{\sim}{n}\epsilon S(n,m) \\ n_m=k}} u(n,m) \right]$$

However, because n ε S(n,m) and $n_m = k$, then $k \le a_m$ and we have

$$u(n,m) = \sum_{k=1}^{\lfloor n,a_m \rfloor} \left[\sum_{\substack{n\epsilon S(n,m) \\ n_m=k}} u(n,m) \right]$$

where $\lfloor a,b \rfloor$ = inf(a,b). Factoring out the constant terms, we observe that for m > 1

$$u(n,m) = \sum_{k=0}^{\lfloor n,a_m \rfloor} \left[\binom{n}{k} \rho_m^k u(n-k,\ m-1) \right]$$

However, by our definition we should have $(n-k) \le a_{m-1}^*$ or $k \ge (n-a_{m-1}^*)$ and we have for m > 1

$$u(n,m) = \begin{cases} \displaystyle\sum_{k=\lceil 0,n-a_{m-1}^* \rceil}^{\lfloor n,a_m \rfloor} \left[\binom{n}{k} \rho_m^k u(n-k,\ m-1) \right] & 0 < n < a_m^* \\ 0 & n > a_m^* \end{cases} \quad (22)$$

and for m = 1,

$$u(n,1) = \rho_1^n \quad 0 \le n \le a_1 \quad (23)$$

and

$$u(0,m) = 1 \quad 1 \le m \le R \quad (24)$$

Eqs. (22 - 24) provide us with an interative scheme to calculate u(n,R). One has to first initialize the first column and row of maxtrix u according to Eqs. (23) and (24), and each element of columns 2 to R is calculated according to Eq. (22).

Having calculated the matrix u, we can find expressions for other statistics of interest. C_2 and $Pr[\Sigma n_r=n]$ have been defined already in Eqs. (20) and (21). For the total average number of messages in the system, we have

$$\overline{n} = P(0) \sum_{n=1}^{B} u(n,R)n \quad (25)$$

and for the marginal distribution of the Rth class, we have the following (for the derivation see [9])

$$Pr[n_R=k] = P(0) \sum_{n=k}^{k+a_{R-1}^*} \left[\binom{n}{k} u(n-k,\ R-1)\ \rho_R^k \right] \quad (26)$$

$$Pr[n_R=0] = P(0) \sum_{n=0}^{a_{R-1}^*} [u(n,\ R-1)] \quad (27)$$

and, in particular, the probability of blocking of the Rth class will be

$$PB_R = Pr[n_R=a_R] = P(0) \sum_{n=a_R}^{B} \left[\binom{n}{a_R} u(n-a_R,\ R-1) \right] \rho_R^{a_R} \quad (28)$$

From Eq. (26), we can find other measures of interest such as throughput of the Rth class, λ_r'; average number of class R messages in the system, $\overline{n}_R$, etc. In order to find statistics of other classes of messages, we must rearrange the ordering of classes and position the class of interest as the Rth.

Special Case (1): $\lambda_r = \delta\lambda_r^o$, r = 1,2,...R , and $\delta\uparrow\infty$

$$\lim_{\delta\uparrow\infty} Pr[\Sigma n_r=n] = 0 \text{ if } n \ne B \ ;\ 1 \text{ if } n = B \quad (29)$$

$$\lim_{\delta\uparrow\infty} P_r[n_r=k] = 0 \text{ if } k \ne a_r \ ;\ 1 \text{ if } k = a_r \quad (30)$$

Comparing Eq. (30) with Eq. (13) we notice that the CS scheme does not display this property.

For the limiting throughput, we have

$$\lim_{\delta\uparrow\infty} \lambda' = \mu C \quad (31)$$

$$\lim_{\delta\uparrow\infty} \lambda_r' = \lim_{\delta\uparrow\infty} \lambda_r(1 - PB_r) = \frac{a_R}{B}\ \mu C \quad (32)$$

which shows that in the limit, each class will use a portion of the output channel equal to the fraction of the storage it is allocated.

Special Case (2): $\lambda_i\uparrow\infty$. In contrast to the CS scheme the limiting throughput of other classes is not zero. The expressions for the limiting probabilities of blocking and other measures are very complicated. In

[9], these expressions are derived for the case $R = 2$, which we report below.

$$\lim_{\lambda_2 \uparrow \infty} PB_r = \frac{\binom{B}{a_2}\rho_1^B}{\sum_{n=a_2}^{B} \binom{n}{a_2}\rho_1^n} \text{ if } r = 1 \text{ ; } 1 \text{ if } r = 2 \tag{33}$$

$$\lim_{\lambda_2 \uparrow \infty} \lambda_r' = \begin{cases} \dfrac{\sum_{n=a_2}^{B-1} \binom{n}{a_2} \rho_1^{n+1}}{\sum_{n=a_2}^{B} \binom{n}{a_2} \rho_1^n} \mu C & r = 1 \\[2ex] \dfrac{\rho_1^{a_2} + \sum_{n=a_{2+1}}^{B} \binom{n-1}{a_2-1} \rho_1^n}{\sum_{n=a_2}^{B} \left(\binom{n}{a_2} \rho_1^n\right)} \mu C & r = 2 \end{cases} \tag{34}$$

The above expressions show that if one of the input rates grows to infinity, it does not prevent the other class from using the system. This was not the case for the CS scheme.

2.4. Sharing with Maximum Queue Length (SMXQ)

The set of feasible states for this scheme is

$$F_3 = \{\underset{\sim}{n} \mid \sum_{r=1}^{B} n_r \leq B \text{ ; } 0 \leq n_r \leq b_r, \; r = 1,2,\ldots,R\} \tag{35}$$

where b_r is the maximum queue length for class r mes- messages. Clearly, if $\Sigma b_r \leq B$ this scheme reduces to CP and if $b_r > B$, $r = 1,2,\ldots,R$, it will be identical to CS.

To calculate the normalization constant C_3 and other statistics as we did for the CP scheme, we must resort to numerical techniques. The algorithm for this calculation, except for minor differences, is similar to the one for the CP scheme which we will briefly explain.

The sets $S(n,m)$ and the auxiliary function $u(n,m)$ will be defined as follows:

$$S(n,m) \triangleq \{\underset{\sim}{n} = (n_1, n_2, \ldots, n_m) \mid \sum_{r=1}^{m} n_r = n, \; 0 \leq n_r \leq b_r, \; r \varepsilon [1,m]\}$$

$$u(n,m) \triangleq \sum_{\underset{\sim}{n} \varepsilon S(n,m)} \left[n! \prod_{r=1}^{m} \frac{\rho_r^{n_r}}{n_r!} \right]$$

Let $b^*_m = \sum_{r=1}^{m} b_r$; then by using an approach very similar to CP, we have for $m > 1$

$$u(n,m) = \begin{cases} \sum_{k=\lceil 0, n - \lfloor B, a^*_{m-1} \rfloor \rceil}^{\lfloor n, b_m \rfloor} \left[\binom{n}{k} \rho_m^k u(n-k, m-1) \right] & n \leq \lfloor B, b^*_m \rfloor \\ 0 & n > \lfloor b, b^*_m \rfloor \end{cases}$$

and for $m = 1$,

$$u(n,1) = \rho_1^n \qquad 0 \leq n \leq \lfloor B, b_1 \rfloor$$

$$u(0,m) = 1 \qquad m = 1,2,\ldots,R$$

where, as before, $\lfloor a,b \rfloor = \inf(a,b)$ and $\lceil a,b \rceil = \sup(a,b)$.

Having calculated the matrix u, we can find the statistics of interest. In particular

$$C_3^{-1} = P(0)^{-1} = \sum_{n=0}^{B} u(n,R) \tag{36}$$

$$\Pr\left[\sum_{r=1}^{R} n_r = n\right] = P(0)u(n,R) \tag{37}$$

$$\bar{n} = E[\text{number of messages in the system}] = P(0) \sum_{n=1}^{B} u(n,R)n \tag{38}$$

For the marginal statistics of class R, we have

$$\Pr[n_R = k] = \frac{\sum_{n=k}^{\lfloor B, b^*_{R-1} + k \rfloor} \left[\binom{n}{k} u(n-k, R-1) \right] \rho_R^k}{\sum_{n=0}^{B} u(n,R)} \qquad 0 \leq k \leq \lfloor B, b_R \rfloor \tag{39}$$

In particular

$$\Pr[n_R = 0] = \frac{\sum_{n=0}^{\lfloor B, b^*_{R-1} \rfloor} u(n,R-1)}{\sum_{n=0}^{B} u(n,R)} \tag{40}$$

The blocking probability of class R messages is the probability that either the entire storage is full, or n_R is equal to b_R. This leads to

$$PB_R = \frac{u(B,R) + \sum_{n=b_R}^{\lfloor B-1, b^*_R \rfloor} \left[\binom{n}{b_R} u(n-b_R, R-1) \right] \rho_R^{b_R}}{\sum_{n=0}^{B} u(n,R)} \tag{41}$$

In order to find marginal statistics of other classes, we must rearrange the ordering of the classes. Now we consider two special cases:

Special Case (1): $\lambda_r = \delta \lambda_r^o$, $r = 1,2,\ldots,R$ and $\delta \uparrow \infty$. In this case we have $\lim_{\delta \uparrow \infty} PB_r = 1$, $r = 1,2,\ldots,R$ and

$$\lim_{\delta \uparrow \infty} \Pr[n_R = k] = \begin{cases} 0 & \text{for } 0 \leq k < B - b^*_{R-1} \\ \neq 0 & \text{for } k \geq B - b^*_{R-1} \end{cases} \tag{42}$$

In particular, $\lim_{\delta \uparrow \infty} \Pr[n_R = 0] = 0$ if $B > b^*_{R-1}$. This shows that, in order to guarantee that at infinite input rate ($\delta \uparrow \infty$) there is always one or more packets of a certain class of messages (say r) present in the system, we should have $B > \sum_{i=1, i \neq r}^{B} b_i$. This is one of the motivations behind using the SMA scheme. For the throughput of the class R messages, we have for $b^*_R > B$

$$\lim_{\delta \uparrow \infty} \lambda_R' = \frac{h(B-1,r) - \binom{B-1}{b_R} h(B-1-b_R, R-1) \rho_R^{o\, b_R}}{h(B,R)} \lambda_R^o \tag{43}$$

where $h(n,m) = u(n,m) \mid_{\rho_i = \rho_i^o}$. And for $b^*_R \leq B$

$$\lim_{\delta \uparrow \infty} \lambda_r' = b_r / b^*_r \;\; \mu C \tag{44}$$

which is the same as for the CP scheme.

Special Case (2): $\lambda_i \uparrow \infty$. If $b_i \geq B$, then intuitively $\lim_{\lambda_i \uparrow \infty} PB_r = 1$, $r = 1,2,\ldots R$ and

$$\lim_{\lambda_i \uparrow \infty} \lambda_r' = \begin{cases} 0 & r \neq 1 \\ \mu C & r = i \end{cases} \tag{45}$$

This behavior is similar to the CS scheme. If, however, $b_i < B$, then

$$\lim_{\lambda_i \uparrow \infty} PB_r = \begin{cases} 1 & r = i \\ < 1 & r \neq i \end{cases} \tag{46}$$

In fact, b_i buffers will be occupied by class i messages with probability one and the B-b_i remaining buffers will be shared by the other classes (according to SMXQ). The expressions for the limiting probabilities are complicated and we only present the expressions for the case, R = 2.

$$\lim_{\lambda_2 \uparrow \infty} PB_r = \frac{\binom{B}{b_2}\rho_1^B}{\sum_{n=b_2}^{B}\binom{n}{b_2}\rho_1^n} \quad \text{if } r=1 ; \quad 1 \text{ if } r=2 \qquad (47)$$

and

$$\lim_{\lambda_2 \uparrow \infty} \lambda'_r = \frac{\sum_{n=b_2}^{B-1}\binom{n}{b_2}\rho_1^{n+1}}{\sum_{n=b_2}^{B}\binom{n}{b_2}\rho_1^n}\mu C, \ r=1 ; \quad \frac{\rho_1^{b_2} + \sum_{n=b_2+1}^{B}\binom{n-1}{b_2-1}\rho_1^n}{\sum_{n=b_2}^{B}\binom{n}{b_2}\rho_1^n}\mu C \quad \text{if } r=2 \qquad (48)$$

Comparing Eqs. (47) and (48) with Eqs. (33) and (34) we notice that in this case the behavior is like the CP scheme when a pool of B buffers is partitioned into $(B - b_2)$ and b_2 buffers.

2.5. Sharing with Minimum Allocation (SMA)

As Eq. (42) suggests, in order to guarantee that at infinite input rates one or more packets of a certain class of messages will be always in the system, at least one buffer storage must be permanently allocated to that class of messages. This is the idea behind the SMA scheme in which, out of a pool of B buffers, a_r, $(r = 1,2,\ldots,R)$ buffers are permanently allocated to class r messages, and the remaining $B_s = B - \Sigma a_r$ buffers, similar to the CS scheme, are shared by all of the classes. As a result, the set of possible states will be

$$F_4 = \{\underset{\sim}{n} = (n_1, n_2, \ldots, n_R) \mid \sum_{r=1}^{R} \lceil 0, n_r - a_r \rceil \le B$$
$$0 \le n_r \le B_s + a_r, \ r = 1,2,\ldots,R\}$$

where $B_s = B - \Sigma a_r$ and $\lceil a,b \rceil = \sup(a,b)$

Calculation of the normalization factor C_4 is more complex than for the previous schemes, as we will see shortly. We start with ordering the classes in an arbitrary manner and define the following sets:

$$T(b,n,m) \triangleq \{n = (n_1, n_2, \ldots, n_m) \mid \sum_{r=1}^{m} n_r = n;$$
$$\sum_{r=1}^{m} \lceil 0, n_r - a_r \rceil = b; \ 0 \le n_r \le b + a_r, \ r = 1,2,\ldots,m\}$$

where $0 \le b \le B_s$, $0 \le n \le B$, $1 \le m \le R$.

T(b,n,m) is the set of states where no messages of classes m+1, m+2, ..., R are in the system, the total number of messages in the system is n and the number of messages in the shared storage is b. Instead of a two-dimensional matrix u, we must use a three-dimensional matrix

$$t(b,n,m) \triangleq \sum_{\underset{\sim}{n} \in T(b,n,m)} \left[n! \prod_{r=1}^{m} \frac{\rho_r^{n_r}}{n_r!} \right] \quad n = \sum_{r=1}^{m} n_r$$

The elements of matrix t can be calculated according to the following equations. (For a derivation, see [9].)

For $1 \le m \le R$

$$t(b,n,m) = \begin{cases} \sum_{k=\lceil 0, n-b-a^*_{m-1} \rceil}^{\lfloor n, a_m \rfloor} \left[\binom{n}{k} \rho_m^k \, t(b, n-k, m-1) \right] \\ + \rho_m^{a_m} \sum_{k=1}^{n-a_m, b} \binom{n}{k + a_m} \rho_m^k t(b-k, n-(k+a_m), m-1) & \text{if} \quad n \le b + a^*_m \\ 0 & \text{if} \quad n > b + a^*_m \end{cases}$$

$$t(b,0,m) = 1 \ \text{ if } b = 0 \ ; \ 0 \text{ if } b > 0$$

$$t(0,n,1) = \rho_1^n \text{ if } 0 \le n \le a_1 \ ; \ 0 \text{ if } n > a_1$$

$$t(b,n,1) = 0 \text{ if } n \ne b + a_1 ; \ \rho_1^n \text{ if } n = b + a_1, \quad b > 0$$

Using the matrix t, we can find values of interest.

$$C_4^{-1} = P(0)^{-1} = \sum_{b=0}^{B_s} \sum_{n=0}^{b+a_r} t(b,n,R) \qquad (50)$$

$$Pr[\Sigma n_r = n] = P(0) \sum_{b=0}^{B_s} t(b,n,R) \qquad (51)$$

For the marginal distribution of class R we have

$$Pr[n_R = k] = P(0) \sum_{b=\lceil 0, k-a_R \rceil}^{B_s} \sum_{n=k}^{a^*_{R-1} + b - \lceil 0, k-a_R \rceil} \left[\binom{n}{k} \rho_R^k \, t(b - \lceil 0, k-a_R \rceil, \ n-k, \ R-1) \right] \qquad (52)$$

In particular,

$$Pr[n_R = 0] = P(0) \sum_{b=0}^{B_s} \sum_{n=0}^{b+a^*_{R-1}} t(b,n,R-1) \qquad (53)$$

The probability of blocking of class R is the probability that the shared storage is full and that $n_R \ge a_R$. This gives us

$$PB_R = P(0) \sum_{k=a_R}^{B_s + a_r} \rho_R^k \left\{ \sum_{n=k}^{B-k} \left[\binom{n}{k} u(B_s - (k-a_R), \ n-k, \ R-1) \right] \right\}$$

Expressions for the limiting probabilites when $\lambda_r = \delta\lambda_r^o$, $r = 1,2,\ldots,R$ and $\delta\uparrow\infty$ are very complex and are not presented here. We can show that $\lim_{\delta\uparrow\infty} PB_r = 1$, $r = 1,2,\ldots,R$ and $\lim_{\delta\uparrow\infty} Pr[n_r = k] = 0$ if $0 \le k < a_r$ and $\ne 0$ if $a_r \le k \le B_s + a_r$

For the special case, $\lambda_i \uparrow\infty$ we can show that

$$\lim_{\lambda_i \uparrow \infty} Pr[n_r = k] \begin{cases} > 0 & 0 \le k \le a_r , \ r \ne i \\ = 0 & k > a_r , \ r \ne i \end{cases} \quad \text{and}$$

$$\lim_{\lambda_i \uparrow \infty} Pr[n_i = k] = 0 \text{ if } k \ne B_s + a_i ; = 1 \text{ if } k = B_s + a_i$$

The analysis of SMAMXQ is quite identical to the SMA scheme and to save space we do not present it here. The interested reader may refer to [9].

3. COMPARISON OF SCHEMES AND CONCLUSION

Here we present some performance curves and compare different sharing schemes. In all of the cases we study, we assume R=2 (i.e., there are two input classes).

Figs. 1(a) and (b) show the behavior of the (normalized) throughput and probability of blocking with respect to

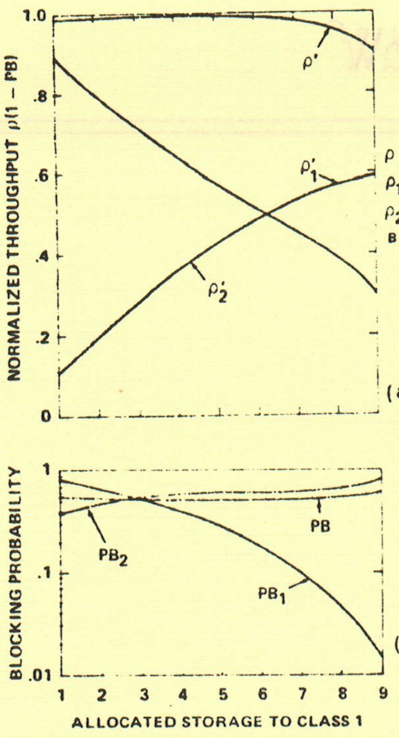

Fig. 1. Effect of Allocated Storage in CP

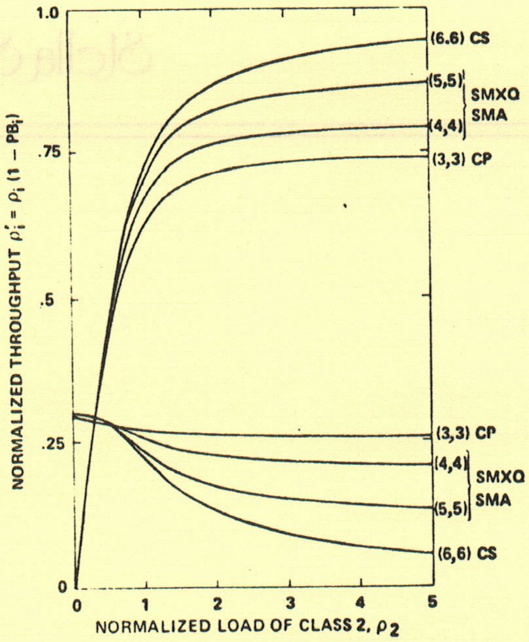

Fig. 2. Comparison of Different Schemes When One of the Input Rates Increases

the allocated storage when the CP scheme is used. In these figures a pool of 10 buffers is shared by two classes of messages. The horizontal axis represents the number of buffers allocated to the first class, a_1, (hence $a_2 = 10 - a_1$). It can be seen that there is an optimal allocation of storage which maximizes the total throughput (or minimizes the blocking probability). The optimal partitioning of storage depends on the values of the input rates. A good approximation to this partitioning is to divide the storage according to relative values of the input rates (i.e., $a_1 = \lambda_1/(\lambda_1 + \lambda_2)B$ and $a_2 = \lambda_2/(\lambda_1 + \lambda_2)\ B$). The performance curve is flatter for larger input rates. This means that deviation from the optimal partitioning does not affect the total throughput severely. Therefore, as can be seen from the curves for ρ_1' and ρ_2', one is able to control the contribution of each class to the output by changing the partitioning of the storage with little degradation of the performance. For example, in Fig 1(a), $a_1 = 6$ and $a_2 = 4$ result in an almost equal contribution to the output by each class and the drop in the maximum throughput (which is achieved at $a_1 = 3$ and $a_2 = 7$) is negligible.

In Fig. 2, a pool of B=6 buffers is shared according to the SMXQ scheme. Notice that we have the following:

1. If $b_1 = b_2 = 3$ then SMXQ is equivalent to CP.
2. If $b_1 = b_2 = 4$ then SMXQ is equivalent to SMA ($a_1 = a_2 = B_s = 2$)
3. If $b_1 = b_2 = 5$ then SMXQ is equivalent to SMA ($a_1 = a_2 = 1$, $B_s = 4$).
4. If $b_1 = b_2 = 6$ then SMSQ is equivalent to CS.

The curves show the normalized throughput of each class when λ_1 remains fixed at 0.3 and λ_2 varies. It is seen how ρ_1' degrades when $b_1 = b_2 = 6$ (CS). In fact, as $\lambda_2 \uparrow \infty$, according to Eq. (17), $\rho' \downarrow 0$. Fig. 2 shows how one can prevent this phenomena by permanently allocating some buffers to class 1 messages. The asymptotic values of ρ_1' and ρ_2' can be calculated according to Eq. (48).

Fig. 3 shows the probability of being absent from the system, $P_i(0)$ as a function of the input rate to the system. In this figure the two input rates are identical (i.e., $\lambda = \lambda_1 + \lambda_2 = 2\lambda_1 = 2\lambda_2$). The horizontal axis represents the total input rate λ. Again, a pool of B=6 buffers is shared according to SMXQ as in Fig. 2. As λ increases, all curves except the CS curve eventually decrease to zero. The CS curve, however, does not decrease to zero. Its asymptotic value as $\lambda \uparrow \infty$ can be calculated according to Eq. (13).

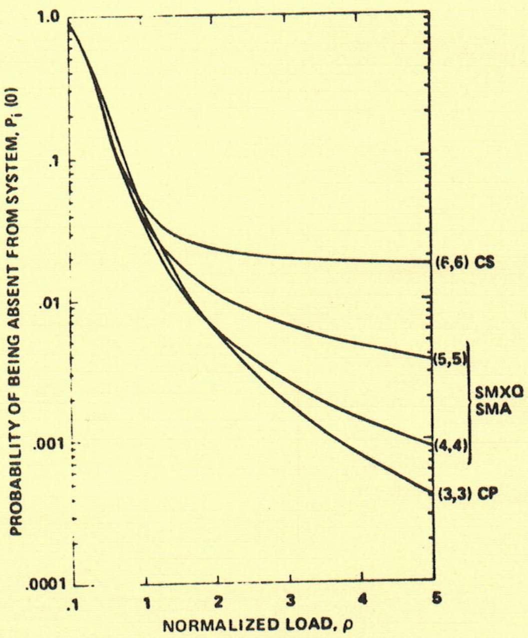

Fig. 3. Comparison of Different Schemes: Probability of Being Absent from the System

As we mentioned at the opening of this paper, if the measure of performance is the total throughput or the probability of blocking of the system, then the CS scheme results in a better performance. This fact is shown in Fig. 4, in which a total of B=6 buffers are shared as in Fig. 2. The two input rates are identical, $(\lambda_1 = \lambda_2)$. In this figure the normalized throughput of class i is plotted as a function of its input rate, λ_i. Notice that the total throughput is equal to $2\rho_1'$. For completeness, we have also presented the total average delay of each class in Fig. 5.

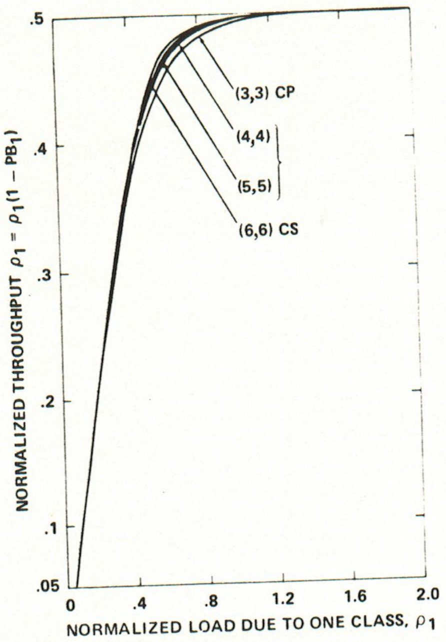

Fig. 4. Comparison of Different Schemes: Throughput

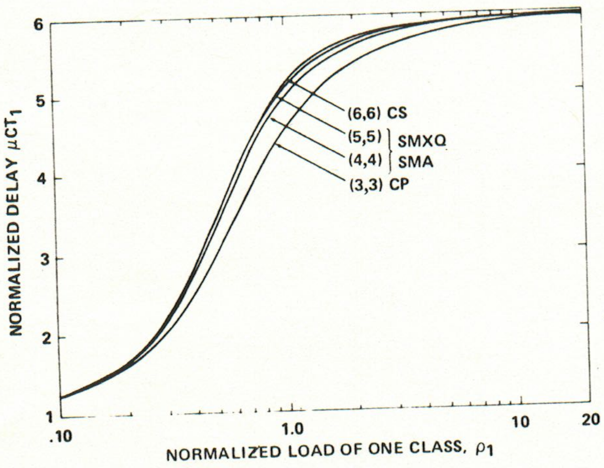

Fig. 5. Comparison of Different Schemes: Delay

We conclude that although the CS scheme always results in higher throughput and lower blocking probability, in situations where other measure of performance are important (e.g., throughput of individual message classes), one should consider other buffer sharing schemes as well. In general the final decision should be based on the particular parameters of environment.

ACKNOWLEDGMENTS

The authors would like to thank George Ann Hornor for her assistance in typing the massive equations.

REFERENCES

[1] Kamoun, F, "Design Considerations for Large Computer Communication Networks," Sch. of Engr. and Appl. Sci., Univ. of California, Los Angeles, CA Rep. No. UCLA-ENG-7642, April 1976.

[2] Chu, W. and L. Liang, "Buffer Behaviour for Mixed Input Traffic and Single Constant Output Rate," *IEEE Trans. on Commun.*, Vol. COM-20, No. 2, April 1972, pp. 230-235.

[3] Rich, M., and M. Schwartz, "Buffer Sharing in Computer Communication Network Nodes," *ICC 75 Conf. Rec.*, Vol. III, June 1975, pp. 33-17 to 33-20.

[4] Drukey, D., "Finite Buffers for Purists," TRW, Inc. Redondo Beach, CA, Systems Rep. 75.6400-10-97, 1975.

[5] Kamoun, F., and L. Kleinrock, "Analysis of Shared Storage in a Computer Network Environment," *Proc. of the 9th Hawaii Int. Conf. on Sys. Sci.*, Jan., 1976, pp. 89-92.

[6] Kleinrock, L., *Queueing Systems, Vol. I: Theory*, Wiley-Interscience, New York, 1975.

[7] Jackson, J., "Networks of Waiting Lines," *Oper. Res.*, Vol. 5, 1957, pp. 518-521.

[8] Baskett, F., K. Chandy, R. Muntz and F. Palacios, "Open, Closed and Mixed Networks of Queues with Different Classes of Customers," *J. of the ACM*, Vol 22, No. 2, April 1975, pp. 248-260.

[9] Kermani, P. "Switching and Flow Control Techniques in Computer Communications Networks," Ph.D. Dissertation, Sch. of Engr. and Appl. Sci., Univ. of California, Los Angeles, 1977.

[10] Little, J., "A Proof of the Queueing Formula L=λW," *Opers. Res.*, Vol. 9, No. 2, March 1961, pp. 383-387.

[11] Moore, F., "Computational Model of a Closed Queueing Network with Exponential Servers," *IBM J. of Res. and Dev.*, Vol. 16, No. 6, Nov. 1972, pp. 567-572.

[12] Williams, A. and R. Bhandiwad, "A Generating Function Approach to Queueing Network Analysis of Multiprogrammed Computers," Mobil Oil Corp, Princeton, N.J., April 1974.

[13] Buzen, J., "Computational Algorithms for Closed Queueing Networks with Exponential Servers, *Comm. of the ACM*, Vol. 16, No. 9, Sept., 1973, pp. 527-531.

[14] Kobayashi, H., and M. Reiser, "Queueing Networks with Multiple Closed Chains: Theory and Computational Algorithms," *IBM J. of Res. and Dev.*, Vol. 19, NO. 3, May 1975, pp. 283-294.

[15] Reiser, M. and H. Kobayashi, "On the Convolutional Algorithm for Separable Queueing Networks," IBM T.J. Watson Res. Center, Yorktown Heights, New York, Rep. (RC 5914 (#25202), Jan. 1976.

[16] Reiser, M., "Numerical Method in Separable Queueing Networks, IBM T.J. Watson Res. Center, Yorktown Heights, N.Y., Rep. RC 5842(#25286), Feb., 1976.

[17] Wong, J., "Queueing Network Models for Computer Systems," Sch. of Eng. and Appl. Sci., Univ. of California, Los Angeles, Rep. UCLA-ENG-7579, October 1975.