ANALYZING A PACKET SWITCH AS A NON-CONSERVATIVE
EXHAUSTIVE-SERVICE POLLING SYSTEM
·I: Cycle-Time and Capacity Analysis

by

Micha Hofri

Technical Report #527
November 1988
ANALYZING A PACKET SWITCH AS A NON-CONSERVATIVE
EXHAUSTIVE-SERVICE POLLING SYSTEM
I: Cycle-Time and Capacity Analysis

Micha Hofri — Department of Computer Science, The Technion, Haifa

November 1988

ABSTRACT
We present a queueing model of a switching element, that is a
variation on often-treated models of exhaustive-service polling
systems, except that the service mechanism is more complex: each
polling station consists of two queues that are fed by independent
arrival processes. The switch has a cycle of $N$ phases; at each phase
the two queues are served in parallel, and a phase only terminates
when both queues are simultaneously empty. There is no time-
penalty when switching between phases. The analysis employs
mostly familiar techniques, but leads to interesting functional
equations, not all of which are easy to solve.

1. INTRODUCTION AND SYSTEM DESCRIPTION

The system under consideration models a switching element that can be in $N$ distinct states; in
each of these states it handles two streams of requests in parallel. These streams can be viewed
as representing the different routing requests put to the switch, and the element handles these
requests at the address-bit level. The analysis will, for the most part, consider only two possible
states. The restriction to two switch-states is for the purpose of the analysis only, but the
methodology that we use can effect the same computations for a larger number of stages or
switch-states, in each of which two streams are served side-by-side.

The description of the switch operation uses a time axis that is slotted into equal slots. Each
switching activity (we shall refer to it as a service) requires precisely one complete slot.

One non-trivial and critical restriction of the mathematical model discussed below is that all the
input processes that describe the request-generation are assumed to be independent, each
forming an i.i.d. sequence of random variables. These rv’s specify the number of arrivals of new
requests at successive slots to the switch queues. Queued requests are kept in buffers; there is a separate, unlimited buffer for each stream. A request that arrives at a slot may be serviced at the earliest during the next slot. Hence we choose to represent the state of the buffers at "slot end"—or "slot beginning"—when all present requests are eligible for service. Figure 1 makes precise the time relationships involved. To complete the description of the queueing model we need to specify the service regime yet: for the present discussion it does not matter, unless we examine the distribution of the waiting times, at which order requests are selected for service. We may assume that requests are handled on a FCFS basis at each stream. When the two-port server is at any switching-state, it will serve in each slot one request from each of the two streams that can be handled at that state (or position). If one stream is empty — only one request is handled. If at the beginning of a slot both buffers are empty, the switching-state is changed (instantaneously, at this time-scale) and one request from each non-empty buffer that is accessible in the new position is served. The new switching-state would persist for one slot even if both buffers were empty at its beginning, and of course, no service would be done then. If there are arrivals during such an idle slot the state would be maintained, to serve them until the first time both buffers are empty, when the server would continue to handle the other streams.

It is remarkable how similar in essence this model is to standard polling systems that have been studied extensively in the past, except that the service here is non-conservative: say the switch attends to the first position, at queues #1 and 2; if at a given time-slot only buffer #1 is non-empty, a single request is served, even if there may be requests queued at buffer #4, that the currently idle port could have served if the switch were at the next station. Another way to state the same characteristic is to say that the service rate is state-dependent, a well-known nemesis of queueing analyses. A trivial example of the difficulty: In a conservative polling system (with exhaustive service and infinite buffers, as we allow), the computation of the mean cycle time, the expected time between returns of the server to begin serving a given position, is immediate (Takagi, (1988), p. 6). In our system however, determining this duration is a major effort. In fact, the first part of our analysis, reported in this installment, concerns only this variable.

---

**Figure 1:** Temporal relationships for the switch operation.
The diagrammed service is rendered only if a request was present at \( S_1 \). All the arrivals during that slot, and the requests remaining from \( S_1 \) (if any) are accounted for following \( S_2 \).
Exhaustive-Service switch...

Notation and Definitions:

\( B_i \) — phase duration, the time in slots during which the switching-state is maintained at position \( i \). The probability generating function (pgf) of this duration is denoted by \( b_i(z) \), \( i=1,2\ldots N \). The corresponding notation for a phase conditioned on the initial queue-lengths being \( r \) and \( s \) is \( B_i(r,s) \) and \( b_i(r,s) \), respectively. The cycle of the server is simply the sum of \( N \) successive phases.

\( L_j, p_j, 1 \leq j \leq 2N \) — the number of arrivals per slot to stream \( j \) and the associated probability mass function (pmf). Individual components of \( p_j \) are denoted by \( p_j(r) \), \( r \geq 0 \). The dimension of this vector is assumed unlimited; since we assume no bound on the sizes of the buffers at the switch, this will hold for all the vectors and matrices we use in this analysis. The mean value, \( \Sigma_{k \geq 1} kp_j(k) \) is denoted by \( \lambda_j \). The associated pgf is denoted by \( p_j(z) \). Note that \( p_j(0) \) has two interpretations with this notation, and they lead to the same value.

\( P_j \) — the probability transition matrix of a Markov chain that describes the number of customers at slot end in a single-server queueing system which behaves exactly as stream \( j \), if the switch never left its position at this stream (essentially, a GI/D/1 queueing system). Clearly, \( P_j(m,n) = p_j(n-m+1) \) for \( m \geq 1, n \geq m-1 \), and \( P_j(0,n) = P_j(1,n) \).

\( \pi_j \) — the steady-state queue-length distribution associated with \( P_j \). Note that \( \pi_{j,0} = 1 - \lambda_j \).

\( T_j \) — the duration of a busy-period associated with \( P_j \). Its pgf is denoted by \( t_j(z) \) and is determined by the equation

\[
 t_j(z) = zp_j(t_j(z)).
\]  

(1.1)

The analysis will show we need to consider additional quantities, which we introduce here to have the definitions all collected together.

\( C_{ij}(z) \) — a matrix, equal to \( [I - t_j(zP_i)]^{-1} \). Note that \( C_{ij}(1) \) does not exist, since \( I - t_j(P_i) \) is singular — it has the eigenvalue 0. However, it is easy to show

\[
 C_{ij}(z) = A_{ij}(z) + \gamma_{ij}(z) = [I - t_j(zP_i) + t_j(z)\bar{P}_i]^{-1} + \frac{t_j(z)\bar{P}_i}{1-t_j(z)} ,
\]  

(1.2)

where \( \bar{P}_i \) is a matrix that has all its rows equal to \( \pi_i \). It can also be obtained as \( \lim_{r \to \infty} P^r \), and it satisfies the following convenient identities:

\[
 P\bar{P} = \bar{P} P = \bar{P} \bar{P} = \bar{P} . \quad [I + \bar{P}]^{-1} = I - \frac{1}{2}P . \quad [I + a\bar{P}]^k = I + ((1 + a)^{k-1})\bar{P} .
\]  

(1.3)

The matrices \( A_{ij}(1) \) are nonsingular.

\( V_{ij}(k) \) — the duration, in time-slots, from the end of a slot when the switch was attending streams \( i \) and \( j \), stream \( j \) was empty, and the buffer of stream \( i \) held \( k \) requests, until the first slot-end when both buffers are empty. The value of \( V_{ij}(0) \) is identically zero. The associated pgf is denoted by \( v_{ij}(z) \).
2. PHASE DURATION ANALYSIS

Let us assume the special case $N=2$; this should not matter in principle. Clearly the system can be modeled as a Markov chain, embedded at slot beginnings, which has the four buffer-occupancies and the position of the server as its state. Since similar two-dimensional chains pose non-trivial computational problems, we proceed to see what can be done with more modest means. The key observation is that we can use the phase lengths $B_i$ as the basic random variables: information about the distribution of a phase length enables the computation of the initial state of the buffers when the next phase begins. As long as a phase continues, the two streams evolve entirely independently, and the characteristics of this evolution are easy to obtain (as $t_j(z)$ was shown above). In a subsequent paper we shall show how to use the distribution of the phase durations to obtain similar results for queue-lengths and waiting times.

We derive an equation for the pgf of the phase duration in the special case where the streams in one switch position have input distributions precisely as those at the other position. These streams (and the associated pmfs’s and pgf’s) will carry the indices 1 and 2 for both positions. Under this assumption the durations of phases at the two positions are identically distributed. Where this is not the case, the following analysis still applies, except—as shall become obvious when the present one is read—one more "iteration" is required; this will not change the nature of the results, but will involve heavier expressions. Precisely such an extension is required to handle more than two switch positions. The idea of the computation is elementary — we randomize on the duration of the phase in one position, hence we can use this value to condition on the initial state of the buffers at the other position. Now we compute the pgf of the conditioned sojourn duration at this second phase; when the conditioning is removed we have an implicit relation that determines $b(z)$.

The following analysis deals with steady-state distributions only, with the tacit assumption that sufficient conditions for it to exist do hold. As is usually the case, these conditions will be revealed once we obtain explicit expressions for moments of aggregate random variables. Here the expected-value of $B$ will fill this useful function.

The sojourn duration, conditioned on the initial buffer occupancies obtains its distribution as follows:

\[
\text{Prob}(B = t | i, j) = \sum_{s \geq 1} \text{Prob}(T_2^{i+j} = s) \\
\times \sum_{k \geq 0} \text{Prob} (\text{buffer #1 changes its occupancy from } i \text{ to } k \text{ during } s \text{ slots}) \\
\times \text{Prob}(V_{12}(k) = t-s).
\]  

The star stands for convolution, and $j+ = \max(1,j)$. This last notation is necessary to reflect the service policy that a phase always extends for at least one slot, even if both buffers are empty upon arrival. Considering for the moment one stream in isolation, note that the duration of a busy-period (which starts with a buffer occupancy of one) is precisely as that of a period that starts with an empty buffer, is constrained to consume the next slot, and extends until the first emptiness of the buffer.
Clearly, the probability of the event [buffer #1 changes its occupancy from \(i\) to \(k\) during \(s\) slots] is simply given by \(P^{1\leftarrow k}_{i\leftarrow k}\).

We compute now the distribution of the duration \(V_{12}(k)\). Note that throughout this duration the state of the switch does not change, hence the contents of the two buffers evolve independently. Since it starts at state \((k,0)\), and ends at \((0,0)\), it spans an integral number of busy cycles of buffer #2 (where we adopt the convenience of using this term also to denote a duration that starts and ends with an empty buffer, and which may, in fact, last only a single slot during which the server is entirely idle)—such a busy cycle has the same distribution, as we mention above, as \(T_2\). For the following derivation consider a fictitious service regime, under which the switch is fixed in the same state, and does not change it even upon total emptiness. In this case the buffers evolve without any dependence whatsoever.

Under this artificial (and temporary) assumption, denote by \(v'_{12}(k,t)\) the probability that \(t\) slots after the buffers are in state \((k,0)\) they are in state \((0,0)\) (not necessarily for the first time, this is what the \(\prime\) stands for). Its value is given by

\[
v'_{12}(k,t) = \sum_{r \geq 0} P_{r\leftarrow k} \cdot P^{1\leftarrow k}_{1\leftarrow k} ,
\]

where the term \(r=0\) only contributes for \(t=0\). This is not a pmf, but it can be assigned a gf (which is then not a pgf), and we find

\[
v'_{12}(k,z) = \sum_{t \geq 0} v'_{12}(k,t) z^t
= \sum_{t \geq 0} \sum_{r \geq 0} P_{r\leftarrow k} \cdot P^{1\leftarrow k}_{1\leftarrow k} \cdot z^t ,
\]

The pmf of the random variable \(V_{12}(k)\) can be obtained in terms of the probabilities \(v'_{12}(k,t)\) in the standard technique for Markov chains:

\[
v'_{12}(k,t) = \sum_{t \geq 1} \sum_{s \geq 1} P_{s\leftarrow k} \cdot P_{t\leftarrow k} \cdot v'_{12}(0,t-s) \
\Rightarrow v'_{12}(k,z) = v_{12}(k,z) v'_{12}(0,z) ,
\]

which we shall write explicitly as

\[
v_{12}(k,z) = \frac{[I - t_2(zP_{1})]^{-1}_{k0}}{[I - t_2(zP_{1})]^{-1}_{00}} = C_{12}(z)_{k0}/C_{12}(z)_{00} .
\]

This function is indeed a pgf. Returning now to equation (2.1) we take the associated conditional pgf (to make our intention explicit, we put in there \(a\)—the duration of the preceding phase—as well, but the distribution depends on it only through \(i\) and \(j\))

\[
b(z | i, j, a) = \sum_{t \geq 1} \sum_{t \geq 1} P_{s\leftarrow k} \cdot P_{t\leftarrow k} \cdot v_{12}(0,t-s) ,
\]

recognizing the matrix product

\[
b(z | i, j, a) = \left[ t^{i\leftarrow k}_{j\leftarrow k} (zP_{1}) \right]_{i0} C_{12}(z)_{00} .
\]

Releasing the conditioning on \(j\) provides
Exhaustive-Service switch...

\[ b(z \mid i, a) = \sum_{j=0}^{\infty} \text{Prob}(L_{2a}^Z = j) (t_{2a}^Z(zP_1)C_{12(z)})_{i0} / C_{12(z)}_{i0} \]

Since \( \text{Prob}(L_{2a}^Z = 0) = p_2^Z(0) \), we obtain

\[ b(z \mid i, a) = [(p_2^Z(0) t_{2}^Z(zP_1) + \sum_{j=1}^{\infty} \text{Prob}(L_{2a}^Z = j) t_{2}^Z(zP_1)] C_{12(z)}]_{i0} / C_{12(z)}_{i0} \]

rearranging,

\[ b(z \mid i, a) = (p_2^Z(0) t_{2}^Z(zP_1) + \sum_{j=1}^{\infty} \text{Prob}(L_{2a}^Z = j) t_{2}^Z(zP_1)) C_{12(z)}_{i0} / C_{12(z)}_{i0} \]

Collecting the terms with \( p_2^Z(0) \), and using the definition \( C_{12(z)} = (I - t_{2}^Z(zP_1))^{-1} \) we get

\[ b(z \mid i, a) = [p_2^Z(0) t_{2}^Z(zP_1)) C_{12(z)} - p_2^Z(0) I]_{i0} / C_{12(z)}_{i0} \]

To remove the condition on \( i \) in a convenient manner, note that the matrix \( D \) defined by \( D_{ij} = \delta_{i+1,j} \) (where \( \delta \) is the Kronecker symbol) satisfies the relation \( A_{10} = [D^i A]_{i0} \). Hence

\[ b(z \mid a) = [p_2^Z(0) t_{2}^Z(zP_1) C_{12(z)} - p_2^Z(0) I]_{i0} / C_{12(z)}_{i0} \] (2.7)

It is simple to verify that \( p_2^Z(D)_{00} = p_2^Z(0) \). Using this, and finally releasing the conditioning on the phase duration \( a \), that has the pgf \( b(\cdot) \) we find, for this particular symmetric case:

\[ b(z) = [ (b(p_1(D)p_2(t_{2}^Z(zP_1))) C_{12(z)})_{00} - b(p_1(0)p_2(0))] C_{12(z)}_{00} \] (2.8)

Remark: The above way of writing the pgf \( b(\cdot) \) is not well-defined until we see the structure of equation (2.7). The reason is that the normal definition of \( b(\cdot) \) would lead to \( b(AB) = \Sigma i,j b_i(A) B^j \) whereas the \( b(\cdot) \) in equation (2.8) stands for \( b(AB) = \Sigma i,j b_i A^i B^j \), and the two are not equal in general, when the quantities \( A \) and \( B \) do not commute, as is the case here with \( D \) and \( t_{2}^Z(zP_1) \).

This representation can be somewhat simplified: The structure of the matrices \( D \) and \( P_1 \) gives rise to the convenient identity \( p_1(D) = D P_1 \). Another useful observation is that while \( D \) and \( P \) are not commutative, they do satisfy the relation \( D P D = D^2 P \), which leads, for example, to \( p_2^Z(D) = D^k P_1^k \). Then, using the last equality and equation (1.1) we find \( P_1 p_2(t_{2}^Z(zP_1)) = t_{2}^Z(zP_1)/z \). Hence, finally:

\[ b(z) = [ (b(D t_{2}^Z(zP_1))/z) C_{12(z)})_{i0} - b(p_1(0)p_2(0))] C_{12(z)}_{00} \] (2.9)

The numerator in equation (2.9) has no term free of \( z \). The considerations leading to equation (2.1)—waiting for first emptiness of buffer #2, and then for both to be simultaneously empty—could be reversed; this would yield the equivalent of equation (2.9) with the indices 1 and 2 reversed. If one wishes, this may be used to get rid of the unsightly (and unknown) value \( b(p_1(0)p_2(0)) \) by eliminating it between the two expressions for \( b(z) \), giving

\[ b(z) = \frac{[b(D t_{2}^Z(zP_1))/z) C_{12(z)})_{00} - b(D t_{2}^Z(zP_1)/z) C_{21(z)})_{00}]}{C_{12(z)})_{00} - C_{21(z)})_{00}} \] (2.10)

When more than two positions exist, or even with two – but without the convenient symmetry, we would have obtained a relation very similar to equation (2.9), except that the \( b(\cdot) \) on the right-hand side needs to carry a subscript, say \( k \), and the one on the left-hand side then is \( b_{k+1}(\cdot) \).
3. THE EXPECTED VALUE OF B

As mentioned above, we expect the first moment of \( B \) to provide us with the conditions the system parameters need to satisfy, in order for steady-state to be possible. Differentiating equation (2.9) is straightforward, but some care must be exercised when the first moment is evaluated, since the matrix \( C_{12}(1) \) does not exist: \( I - t_j(P_j) \) is a singular matrix. What is possible to do however, is to use the representation given in equation (1.2), noting that it is only the \( \gamma_{12}(z) \) component that misbehaves when \( z \) approaches 1. By isolating the terms which are of second order in \((1 - t_2(z))^{-1}\) we find

\[
b'(1) = -\frac{1}{\pi_{1,0}^1} \left[ (Db'(Dt_2(P_1))(P_1t_2(P_1) - t_2(P_1))\tilde{\beta}_{1,0} + b(Dt_2(P_1))A_{12}(1)_{00}t_2'(1)\pi_{1,0} + b(p_1(0)p_2(0))t_2'(1)\pi_{1,0} \right].
\]

Note that \( [\tilde{\beta}]_{1,0} = \pi_{1,0} \). Also, both \( t_i(z) \) and \( b(z) \) vanish at \( z=0 \) (i.e. -- have no \( z \)-free term). Finally, equation (1.3) implies that \( t_2(zP_1)\tilde{\beta}_1 = t_2(z)\tilde{\beta}_1 \) and similarly that \( b(Dt_2(zP_1)/z)\tilde{\beta}_1 = b(Dt_2(z)/z)\tilde{\beta}_1 \). Using these relations we can effect some simplifications that provide finally:

\[
E[B] = \frac{([I - b(Dt_2(zP_1))A_{12}(1)_{00} + b(p_1(0)p_2(0))]}{(1 - \lambda_1(1 - 2\lambda_2))}.\tag{3.2}
\]

The factors in the denominator depend separately on the parameters of the two streams, whereas they are combined in the numerator. Hence they cannot be factored away, and the denominator may not vanish for \( E[B] \) to be finite. When we consider also the equation obtained by reversing the indices, we find that both \( \lambda_1 \) and \( \lambda_2 \) must be less than one half. We have thus proved

**Theorem:** A necessary condition for the queueing system to be stable is that

\[
\max(\lambda_1, \lambda_2) < \frac{1}{2}.\tag{3.3a}
\]

When the \( N \) input streams have different distributions, the same approach would lead to

\[
\sum_{r=1}^{N} \max(\lambda_{2r-1}, \lambda_{2r}) < 1.\tag{3.3b}
\]

The analysis implies that the conditions are also sufficient. \( \square \)

The situation here is quite unlike the usual case in queueing analysis, where the pgf's (or transforms) of variables of interest provide easily the values of the first few moments, even when we only have a functional equation for them that we cannot solve explicitly, let alone invert them. To obtain \( E[B] \) we need to work harder, and either evaluate the distribution of \( B \), or at least obtain an explicit expression for \( b(z) \). This is the business of the next section.

4. NUMERICAL TREATMENT

We know of no analytic methodology to handle the like of equations (2.9) or (2.10). Hence the
need for a numerical approach. Since the matrices and vectors involved in the calculations are inherently of infinite size, one has to turn to various truncation methods. It is best to truncate in a way that is meaningful in terms of the operation of the model, since in that case we could use our intuition as a guide for the level of truncation and determining the accuracy. When the structure of the equations below is examined closely, these "natural truncation" values become obvious. Note that things change to a very limited extent if the buffers are finite. The matrices $P_i$ are now of a finite dimension, but the all-important functions $t_i(z)$ and $b(z)$ still have an unbounded number of nonzero coefficients, nor need they be rational functions. In practice one operates as if the buffers are finite, and transforms the $P_i$ matrices by lumping all columns beyond the buffer size in the extreme right column.

The coefficients of the functions $t_i$ are simple to compute from equation (1.1). The equation $t(z) = z p(t(z))$ may be viewed as an identity in $z$, and then it gives rise to the following double recursion, using the notation $t_i(j) = [z^j] t_i(z)$:

$$t_i(0) = 0; \quad t_i(1) = p_i(0); \quad t_i(j) = \sum_{r=1}^{j-1} p_i(r) a_{i;r,j-1} = a_{i;1,j} , \quad (4.1)$$

where $a_{i;1;j} = [z^j] t_i(z) = \sum_{r=1}^{j-1} a_{i;r-1,j-1} t_i(r), \quad k > 1$.

As $t_i(0) = 0$, it follows that $a_{i;1;j} = 0$ for $k > j$. Since all the numbers involved are positive, there is little to worry about numerical error propagation. Computations of powers of $P_i$, turned out to be easy to control for precision. For example, $E[V_{12}(k)]$ can be obtained by differentiating equation (2.4) at $z = 1$, and because all lines of $P$ are identical, it is given by

$$E[V_{12}(k)] = \frac{A_{12}(1)_{00} - A_{12}(1)_{k0}}{(1 - \lambda_1)(1 - \lambda_2)} = \frac{1}{(1 - \lambda_1)(1 - \lambda_2)} \sum_{r=1}^{2 \gamma} t_2^r(P_1)_{00} - t_2^r(P_1)_{k0}. \quad (4.2)$$

Separately, each of this sums would diverge, since the terms approach $t_2(\gamma_{1,0})$, but the differences contract geometrically, and their sum turned out to be numerically quite well-behaved.

The coefficients of the pgf $b(z)$ have to be determined from equation (2.9). This equation does not, however, produce a recursion. When viewed as an identity in $z$, it produces a system of linear equations, as we now present.

Rewrite equation (2.9) as

$$b(z) C_{12}(z)_{00} = [b(Dt_2(zP_1)z)C_{12}(z)]_{00} - [b(p_1(0)p_2(0))]. \quad (4.3)$$

To equate the coefficient of $z^j$ on both sides, observe that

$$[z^j] C_{12}(z)_{00} = \sum_{r=0}^{j} [z^j] t_2^r(zP_1)_{i0} = \delta_{j0} \delta_{i0} + A_{2;j} P_1^{i0}. \quad (4.4)$$

where $A_{i;j} = \sum_{r=1}^{j} a_{i;r}$ (this is the complete column sum), and remembering the remark following equation (2.8), $b_k$ in the right-hand side appears with the coefficients

$$[z^r](D^k t_2^r(zP_1)z^{-k})_{0i} = [z^{k+r}] t_2^r(zP_1)_{ki} = a_{2;k,k+r} P_1^{k+r}. \quad (4.5)$$

Hence
Exhaustive-Service switch...

\[ [z^j] \{ b \left[ D_{1z}(z) \right] C_{1z}(z) \} \}_{00} = \sum_{k_2} b_k (a_{2,k,k+j} + \sum_{r=0}^{j-1} a_{2,k,k+r} A_{2,j-r}) P_{k_1}^{k_0}. \] (4.6)

The following set of equations obtains

\[ b_j + \sum_{i=1}^{j-1} b_i A_{2,j-i} P_{i-1}^{i-i} 00 - \sum_{i=1}^{j-1} b_i P_{i}^{i-1} 10 (a_{2,i,i+j} + \sum_{k=0}^{j-1} a_{2,i,i+k} A_{2,j-k}) = 0. \] (4.7)

This is an infinite set of equations; one has to truncate it at repeatedly higher value, until satisfactory stabilization of the obtained values is achieved. The equations are remarkably well-conditioned, in all the examples we ran. Still, unless the input rates are fairly low, this dimension is high enough to require some care in the solution of the resultant set of equations. Table 1 presents a few numbers obtained from equation (4.7). The first row could be computed with truncation at 20, whereas the last two required the solution of a set of a hundred equations (most of the time was spent in the recursion (4.1)). As is well known, the expected duration of a single server queue only depends on the first moment of the arrival process: from equation (1.1) \( E[T] = 1/(1 - \lambda) \). This is not the case for \( E[B] \), as the last two rows of the table show. The first was obtained with \( p_1 = p_2 = (0.75, 0.1, 0.15) \), whereas the second one with \( p_1 = (0.8, 0.0, 0.2) \) and \( p_2 = (0.6, 0.4) \).

<table>
<thead>
<tr>
<th>( \lambda_1 )</th>
<th>( \lambda_2 )</th>
<th>( E[T_1] )</th>
<th>( E[T_2] )</th>
<th>( E[B] )</th>
<th>( B_{.95} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.1</td>
<td>1.1111</td>
<td>1.1111</td>
<td>1.3136</td>
<td>3</td>
</tr>
<tr>
<td>0.1</td>
<td>0.4</td>
<td>1.1111</td>
<td>1.6667</td>
<td>3.6042</td>
<td>12</td>
</tr>
<tr>
<td>0.2</td>
<td>0.3</td>
<td>1.25</td>
<td>1.4286</td>
<td>2.4457</td>
<td>7</td>
</tr>
<tr>
<td>0.4</td>
<td>0.4</td>
<td>1.6667</td>
<td>1.6667</td>
<td>12.6976</td>
<td>33</td>
</tr>
<tr>
<td>0.4</td>
<td>0.4</td>
<td>1.6667</td>
<td>1.6667</td>
<td>10.9030</td>
<td>27</td>
</tr>
</tbody>
</table>

Table 1
Examples of expected length and .95 fractile of the phase duration.
The input distributions all satisfied \( p_i(r) = 0 \), for \( j \geq 3 \).

ACKNOWLEDGMENTS
The work was initiated at the IBM Research facility in Hawthorne, at the suggestion of Dr. Roch Guérin, who has previously considered different models of the switch. The sylvan calm of the back-lawn of the Koren home, in Amherst Mass., provided an ideal setting for this work.

REFERENCES