Stochastic Models, Vol. 7, No. 1, pp. 1-46, 1991

NEW RESULTS ON THE SINGLE SERVER QUEUE WITH A BATCH MARKOVIAN ARRIVAL PROCESS

David M. Lucantoni†
AT&T Bell Laboratories

ABSTRACT

The versatile Markovian point process was introduced by M. F. Neuts in 1979. This is a rich class of point processes which contains many familiar arrival process as very special cases. Recently, the *Batch Markovian Arrival Process,* a class of point processes which was subsequently shown to be equivalent to Neuts' point process, has been studied using a more transparent notation.

Recent results in the matrix-analytic approach to queueing theory have substantially reduced the computational complexity of the algorithmic solution of single server queues with a general Markovian arrival process. We generalize these results to the single server queue with the batch arrival process and emphasize the resulting simplifications.

Algorithms for the special cases of the *PH/G/*1 and *MMPP/G/*1 queues are highlighted as these models are receiving renewed attention in the literature and the new algorithms proposed here are simpler than existing ones. In particular, the *PH/G/*1 queue has additional structure which further enhances the efficiency of its algorithmic solution. Also, the two-state *MMPP/G/*1 queue, which has applications in communications modeling, has an extremely simple solution.

KEYWORDS: Matrix-analytic methodology, *N/G/*1 queue, point processes, phase-type distributions

---

† Mailing address: AT&T Bell Laboratories, Room 3K-601, 101 Crawfords Corner Rd, PO Box 3030, Holmdel, New Jersey 07733-3030. E-mail: dave@buckaroo.att.com

**** Note: This is a revised version of the original paper. Several typos have been corrected.**

1. INTRODUCTION

The versatile Markovian point process was introduced by M. F. Neuts in [1]. This is a very rich class of point processes which contains many well known arrival processes as special cases. Among them are the phase-type (*PH*) renewal process, the Markov modulated Poisson process (*MMPP*), overflows from finite Markovian queues, etc. In each case, arrivals are allowed to occur in batches where different types of arrivals can have different batch size distributions. The price paid for such generality was an elaborate notation required to keep track of the different types of arrivals. Although the notation was complex, the analysis of queues with this point process as the arrival stream proceeded, conceptually, in an analogous fashion to that of queues with simpler arrival streams. Thus it was possible to solve in a unified methodical analysis a whole class of queueing problems, unifying many results in the literature.

This was first accomplished by V. Ramaswami for the single server queue with the versatile Markovian point process as the arrival stream [2]. Since then, the infinite server, *c*-server (with deterministic service times), and finite queue versions have been solved, see [3], [4], and [5]. Although the computational algorithm suggested by Ramaswami's analysis has been shown to be numerically stable [6], in practice it has not been feasible to implement it in its full generality. The setup computations alone are a formidable burden on both CPU time and storage. Thus, until now, practical numerical solutions have been limited to particular cases of the general model.

In our analysis of a single server queue with server vacations [7], we desired the solution to the queue with a *PH*-renewal arrival process and the one with a correlated arrival stream such as an *MMPP*. As our focus was not on batch arrivals, we did not proceed with the full generality of the versatile Markovian point process, but constructed a new process which contained both *PH*-renewal and the *MMPP* processes yet whose notation was very simple. We called this process the *Markovian Arrival Process (MAP)*. This construction is easily generalized to the *Batch Markovian Arrival Process (BMAP)* to allow for batch arrivals. Although this new class of processes was originally thought to be more general than the versatile Markovian point process, we later showed that the two processes were in fact equivalent. The only difference is that the *BMAP* involves much simpler notation.

Special cases of the *BMAP/G/*1 queue have received renewed attention in the communications modeling literature. The interrupted Poisson process has long been used to approximate the overflow traffic of finite trunk systems [8]. More recently, modeling of packetized voice and data traffic has required consideration of more complicated arrival processes than the Poisson process. It is now well known ([9], [10]) that the interarrival times in the packet streams are strongly correlated. The *MMPP* was used in [10] to approximate the superposition of packetized voice processes and in [11] for a related process. The *MMPP* was chosen because it is a tractable, non-renewal stream which could match certain statistical properties of the original traffic. The *MMPP/G/*1 queue approximated the first two moments of delays as well as the tail probabilities with high accuracy. Other algorithms for solving the *MMPP/G/*1 queue are presented in [12] and [13]. For a case where the *MMPP* is obtained as the superposition of

interrupted Poisson processes see [14]. Other special cases of the *BMAP/G/*1 queue which have appeared in the literature are related to the *PH/G/*1 queue. We refer to the extended, annotated bibliography [15] for many examples and special cases.

We present here new results for the *BMAP/G/*1 queue. In particular, we show that the matrix G, which arises in the matrix analytic approach to queues of *M/G/*1 type and is the key ingredient to the computational procedures, has an exponential form. This exponential form leads to an efficient algorithm for the computation of G as well as the coefficient matrices in the transition probability matrix of the Markov chain embedded at departures. These are needed to compute the queue length distribution at departures and at arbitrary times. This key result generalizes similar results in [7] ,and [16]. The algorithms presented here allow for a general implementation of canned computer programs for solving the general model. Such a program could be used for comparing vastly different arrival processes entering a single server queue.

A further use of this algorithm is to evaluate the performance of superpositions of renewal processes entering a queue. If the renewal processes are of phase type then the superposition is a special case of the *BMAP*. Although the size of the matrices involved grows geometrically as the number of streams, for two or three streams the computations are completely feasible. The delay seen by customers in the individual streams can be derived from the results presented earlier. Similar calculations for the $MMPP/M/c/c+K$ queue were presented in [17]. These exact expressions could be used to validate various simple approximations that have been proposed, see e.g., [18] and [19].

The remainder of this paper is organized as follows. In Section 2, we define the *BMAP* and present some familiar special cases of the process. Section 3 consists of an outline of the traditional matrix-analytic approach to solving the single server queue with a *BMAP* as the arrival stream emphasizing the framework of the new notation. New results for the *BMAP/G/*1 queue are presented in Section 4. Section 5 summarizes the algorithmic simplifications for the general model, highlighting the substantial savings in both computational complexity and storage which are afforded by the new results. In Section 6 present several special cases which have particularly simple solutions. Conclusions are presented in Section 7.

2. THE BATCH MARKOVIAN ARRIVAL PROCESS

To motivate the Batch Markovian Arrival Process, *BMAP*, we first consider a Poisson process with batch arrivals. Let the rate of the Poisson process be λ and the probability that the batch size equals j be p_j, $j \geq 1$. $N(t)$ is the number of arrivals in $(0,t]$. The process $\{N(t)\}$ is then a Markov process on the state space $\{i:\ i \geq 0\}$ with infinitesimal generator of the form

$$Q = \begin{bmatrix} d_0 & d_1 & d_2 & d_3 & \cdot & \cdot \\ & d_0 & d_1 & d_2 & \cdot & \cdot \\ & & d_0 & d_1 & \cdot & \cdot \\ & & & \cdot & \cdot & \cdot \\ & & & & \cdot & \cdot \end{bmatrix}, \tag{1}$$

where, $d_0 = -\lambda$ and $d_j = \lambda p_j$ for $j \geq 1$. After an exponential sojourn (with mean λ^{-1}) in state i, the process jumps to state $i+j$ with probability p_j where the transition corresponds to an arrival and j corresponds to the size of the batch.

The Batch Markovian Arrival Process is constructed by generalizing the above batch Poisson process to allow for non-exponential times between the arrivals of batches, but still preserving an underlying Markovian structure. To accomplish this, we consider a 2-dimensional Markov process $\{N(t), J(t)\}$ on the state space $\{(i,j): i \geq 0, 1 \leq j \leq m\}$ with an infinitesimal generator Q having the structure,

$$Q = \begin{bmatrix} D_0 & D_1 & D_2 & D_3 & \cdot & \cdot \\ & D_0 & D_1 & D_2 & \cdot & \cdot \\ & & D_0 & D_1 & \cdot & \cdot \\ & & & \cdot & \cdot & \cdot \\ & & & & \cdot & \cdot \end{bmatrix}, \tag{2}$$

where D_k, $k \geq 0$, are $m \times m$ matrices, D_0 has negative diagonal elements and nonnegative off-diagonal elements, D_k, $k \geq 1$, are *nonnegative* and D, defined by

$$D = \sum_{k=0}^{\infty} D_k, \tag{3}$$

is an irreducible infinitesimal generator. We also assume that $D \neq D_0$. If $N(t)$ represents a counting variable and $J(t)$ an auxiliary state or phase variable then the above Markov process defines a batch arrival process where transitions from a state (i,j) to state $(i+k,l)$, $k \geq 1$, $1 \leq j,l \leq m$, correspond to batch arrivals of size k, and thus batch size can depend on i and j. The matrix D_0 is a stable matrix which implies that it is nonsingular and the sojourn time in the set of states $\{(i,j): 1 \leq j \leq m\}$ is finite with probability 1. This implies that the arrival process does not terminate. For future reference, we define the matrix generating function

$$D(z) = \sum_{k=0}^{\infty} D_k z^k, \quad \text{for } |z| \le 1.$$

Let $\boldsymbol{\pi}$ be the stationary probability vector of the Markov process with generator D, i.e., $\boldsymbol{\pi}$ satisfies

$$\boldsymbol{\pi} D = \mathbf{0}, \qquad \boldsymbol{\pi} \boldsymbol{e} = 1, \tag{4}$$

where $\boldsymbol{e}$ is a column vector of 1's. The fundamental arrival rate for the arrival process is then given by

$$\lambda_1'^{-1} = \boldsymbol{\pi} \sum_{k=1}^{\infty} k D_k \boldsymbol{e} = \boldsymbol{\pi} \boldsymbol{d},$$

where $\boldsymbol{d} = \sum k D_k \boldsymbol{e}$.

A constructive description of this process is useful for visualizing the evolution of the process. Assume the underlying Markov process with generator D is in some state i, $1 \le i \le m$. The sojourn time in that state is exponentially distributed with parameter λ_i. At the end of that sojourn time, there occurs a transition to another (or possibly the same) state and that transition may or may not correspond to an arrival epoch. With probability $p_i(0,k)$, $1 \le k \le m$, $k \ne i$, there will be a transition to state k *without* an arrival. With probability $p_i(j,k)$, $j \ge 1$, $1 \le k \le m$, there will be a transition to state k with a batch arrival of size j. We therefore have, for $1 \le i \le m$,

$$\sum_{\substack{k=1 \\ k \ne i}}^{m} p_i(0,k) + \sum_{j=1}^{\infty} \sum_{k=1}^{m} p_i(j,k) = 1,$$

and with this notation it is clear that $(D_0)_{ii} = -\lambda_i$, $1 \le i \le m$, $(D_0)_{ik} = \lambda_i p_i(0,k)$, $1 \le i,k \le m$, $k \ne i$, and $(D_j)_{ik} = \lambda_i p_i(j,k)$, $j \ge 1$, $1 \le i,k \le m$. The matrix D_0 thus governs transitions that correspond to no arrivals, and D_j governs transitions that correspond to arrivals of batches of size j.

If $P(t)$ represents the transition probability matrix of the Markov process $\{N(t), J(t)\}$, with generator Q, then it satisfies the Chapman-Kolmogorov equations

$$P'(t) = P(t)Q, \quad \text{for } t \ge 0, \quad \text{with} \quad P(0) = I. \tag{5}$$

2.1 The Counting Function:

Let $N(t)$ be the number of arrivals in $(0,t]$ and $J(t)$ the auxiliary phase at time t. Now let

$$P_{ij}(n,t) = P\{N(t) = n,\ J(t) = j \mid N(0) = 0,\ J(0)=i\}$$

be the (i,j) element of a matrix $P(n,t)$. If $P(t)$ is partitioned into $m\times m$ blocks then $P(n,t)$ is given by the n-th block in the first row of $P(t)$. Therefore, we see that the Chapman-Kolmogorov equations (5) with the structure of Q in (2) imply that the matrices $P(n,t)$ satisfy

$$P'(n,t) = \sum_{j=0}^{n} P(j,t)D_{n-j}, \quad n\geq 0,\ t\geq 0, \tag{6}$$

$$P(0,0) = I.$$

These equations can be derived directly by considering the possible scenarios that result in n arrivals by time $t+dt$. That is, there could be j arrivals up to t, $0\leq j\leq n$, and a batch arrival of size $n-j$ in $(t,t+dt)$. Multiplying the n-th equation in (6) by z^n, $n\geq 0$, and summing yield that the matrix generating function $P^*(z,t)$, defined by

$$P^*(z,t) = \sum_{n=0}^{\infty} P(n,t)z^n, \quad \text{for } |z|\leq 1,$$

satisfies

$$\frac{d}{dt}P^*(z,t) = P^*(z,t)D(z), \tag{7}$$

$$P^*(z,0) = I,$$

and is therefore explicitly given by

$$P^*(z,t) = e^{D(z)t}, \quad \text{for } |z|\leq 1,\ t\geq 0. \tag{8}$$

By differentiating successively in Equation (8) we may obtain expressions for the moments of the number of arrivals in $(0,t]$. (See [20] for similar calculations.)

2.2 Special Cases:

Many familiar arrival processes can be obtained as very special cases of the *BMAP*. Here is a selected sample of some of the more useful examples.

a) *The Markovian Arrival Process (MAP).* The *MAP* defined in [7] is a *BMAP* with all arrivals consisting of a batch of size 1. We therefore have $D_j = 0$, $j \geq 2$. This class contains many well known arrival processes, some of which are:

- *Poisson process.* For $D_0 = -\lambda$, $D_1 = \lambda$, the *MAP* is the ordinary Poisson process of rate λ.
- *PH-renewal process.* The phase type (*PH*) renewal process, [21], [22], with representation $(\boldsymbol{\alpha}, T)$, is a *MAP* with $D_0 = T$ and $D_1 = -Te\boldsymbol{\alpha}$. This class contains the familiar Erlang, E_k, and hyperexponential, H_k, arrival processes as well as finite mixtures of these. See [23] for other examples.
- *Markov-modulated Poisson process (MMPP)* (see, e.g., [10].) The *MMPP* with infinitesimal generator R and arrival rate matrix $\Lambda = diag(\lambda_1, \ldots, \lambda_m)$, is a *MAP* with $D_0 = R - \Lambda$, and $D_1 = \Lambda$. The *MMPP* is a particularly useful class of non-renewal processes.
- *Alternating PH-renewal process.*
- *A sequence of PH interarrival times selected via a Markov chain* [24].
- *A superposition of PH-renewal processes* [25].
- *The superposition of independent MAP's.*

We refer to [7] for additional examples and for the representations of the above examples.

b) *A MAP with i.i.d. batch arrivals.* Consider a *MAP* defined by the pair (D_0, D_1) where each arrival epoch corresponds to a batch arrival. If successive batch sizes are independent and identically distributed with probability density $\{p_j,\ j \geq 1\}$ then this process is a *BMAP* with $D_j = p_j D_1$, $j \geq 1$.

c) *A batch Poisson process with correlated batch arrivals.* Consider a batch Poisson process where the batch size distribution of successive batch arrivals is chosen according to a Markov chain. For example, let $\{q_i(k),\ k \geq 1\}$ $1 \leq i \leq m$,, be a set of m discrete density functions and let P be the transition probability matrix of an m-state, irreducible Markov chain. Let the rate of the Poisson process be λ and assume that successive batch size distributions are chosen from the set $\{q_i(\cdot),\ 1 \leq i \leq m\}$ according to P. This process is then a *BMAP* with $D_0 = -\lambda I$ and $(D_k)_{ij} = \lambda P_{ij} q_i(k)$. This example is easily extended to a *MAP* with correlated batch sizes.

d) *Neuts' versatile Markovian point process.* That process, introduced in [1], is constructively defined by starting with a *PH*-renewal process as a substratum. There are three types of arrival epochs which are related to the evolution of the *PH*-renewal process as follows. There are Poisson arrivals with arbitrary batch size distributions during sojourns in the states of the Markov process governing the renewal process. The arrival rates of the Poisson process and the batch size distributions may depend on the state of the Markov process. The underlying Markov process can change states either with or without a corresponding renewal. Each time the process changes states there is a batch arrival (the batch size may be 0) where the batch size distribution can depend on the states before and after the change as well as whether or not a renewal occurred.

It can be shown that this process is equivalent to the *BMAP*. An advantage of viewing the process in the framework of the *BMAP* is that the notation is much simplified. For example, using the notation of [1], we have the following correspondence:

$$D_0 = \Delta(\boldsymbol{\lambda})\Delta[\boldsymbol{p}(0)] - \Delta(\boldsymbol{\lambda}) + T \circ q(0) + \boldsymbol{T}^0 \boldsymbol{\alpha} \circ r(0), \tag{9}$$

$$D_k = \Delta(\boldsymbol{\lambda})\Delta[\boldsymbol{p}(k)] + T \circ q(k) + \boldsymbol{T}^0 \boldsymbol{\alpha} \circ r(k), \quad \text{for } k \geq 1.$$

Queueing systems with the versatile Markovian point process as the arrival stream are studied in [2], [3], [4], and [5].

3. THE MATRIX ANALYTIC APPROACH

In this section we outline the solution procedure based on the matrix-analytic approach to the *BMAP/G/*1 queue. This approach was pioneered by M. F. Neuts and has been used successfully to analyze a number of complicated queueing systems. (See e.g., [2], [10], [20], [26]). The main results for this section were originally proved by V. Ramaswami for the equivalent *N/G/*1 queue. We will therefore not prove the results again here, but will simply restate them in the *BMAP* notation. The purpose of including this outline is first, to show how the analysis itself serves as a recipe for the algorithmic computations of many desired performance measures; second, to have expressions for many of the intermediate quantities and performance measures in terms of the new *BMAP* notation and third, to have a benchmark to compare the new algorithms which are presented in Section 5.

3.1 Model Definition

Consider a single server queue whose arrival process is given by a *BMAP* defined by the sequence $\{D_k, k \geq 0\}$. Let the service times have an arbitrary distribution function, $\tilde{H}$, with Laplace-Stieltjes transform (*LST*), H, and finite mean μ_1'. We also make the standard independence assumptions and

assume that the traffic intensity, $\rho = \mu_1'/\lambda_1' < 1$.

The Embedded Markov Renewal Process at Departures

The embedded Markov renewal process at departure epochs is defined as follows. Define τ_k to be the epoch of the k-th departure from the queue, with $\tau_0 = 0$, and (ξ_k, J_k) to be the number in system and the phase of the arrival process at τ_k^+. Then $(\xi_k, J_k, \tau_{k+1} - \tau_k)$ is a semi-Markov process on the state space $\{ (i,j): i \geq 0, 1 \leq j \leq m \}$. The semi-Markov process is *positive recurrent* when the *traffic intensity* $\rho = \mu_1'/\lambda_1' < 1$. The transition probability matrix is given by

$$\tilde{P}(x) = \begin{bmatrix} \tilde{B}_0(x) & \tilde{B}_1(x) & \tilde{B}_2(x) & \cdots \\ \tilde{A}_0(x) & \tilde{A}_1(x) & \tilde{A}_2(x) & \cdots \\ 0 & \tilde{A}_0(x) & \tilde{A}_1(x) & \cdots \\ 0 & 0 & \tilde{A}_0(x) & \cdots \\ \cdot & \cdot & \cdot & \\ \cdot & \cdot & \cdot & \end{bmatrix}, \quad x \geq 0, \tag{10}$$

where for $n \geq 0$, $\tilde{A}_n(x)$ and $\tilde{B}_n(x)$ are the $m \times m$ matrices of mass functions defined by

$[\tilde{A}_n(x)]_{ij}$ = $P\{$ Given a departure at time 0, which left at least one customer in the system and the arrival process in phase i, the next departure occurs no later than time x with the arrival process in phase j, and during that service there were n arrivals$\}$,

$[\tilde{B}_n(x)]_{ij}$ = $P\{$ Given a departure at time 0, which left the system empty and the arrival process in phase i, the next departure occurs no later than time x with the arrival process in phase j, leaving n customers in the system$\}$.

Queues with embedded Markov renewal processes whose transition probability matrix has the structure of (10) are referred to as queues of the "$M/G/1$ type" or queues of the "$M/G/1$ paradigm"[20]. The nomenclature arises due to the similarity of (10) to its scalar analogue in the ordinary $M/G/1$ queue.

From the definition of $P(n,t)$, it is clear that

$$\tilde{A}_n(x) = \int_0^x P(n,t)\, d\tilde{H}(t). \tag{11}$$

We define the transform matrices

$$A_n(s) = \int_0^\infty e^{-sx}\, d\tilde{A}_n(x), \qquad B_n(s) = \int_0^\infty e^{-sx} d\tilde{B}_n(x),$$

$$A(z,s) = \sum_{n=0}^\infty A_n(s) z^n, \qquad B(z,s) = \sum_{n=0}^\infty B_n(s) z^n,$$

and for later use, the matrices

$$A_n = A_n(0) = \tilde{A}_n(\infty), \qquad B_n = B_n(0) = \tilde{B}_n(\infty), \tag{12}$$

$$A = A(1,0), \qquad B = B(1,0).$$

Using the properties of $P(n,t)$, it can be shown that

$$A(z,s) = \int_0^\infty e^{-sx} e^{D(z)x} d\tilde{H}(x). \tag{13}$$

From (13), we see that

$$A = \int_0^\infty e^{Dt} d\tilde{H}(t). \tag{14}$$

We note that the matrix A is stochastic, and that the stationary vector $\boldsymbol{\pi}$ defined in (4) also satisfies $\boldsymbol{\pi}A = \boldsymbol{\pi}$, $\boldsymbol{\pi}e = 1$. The vector $\boldsymbol{\beta}$, whose j-th component is the conditional number of arrivals during a service which starts with the arrival process in phase j is defined by

$$\boldsymbol{\beta} = \frac{d}{dz} A(z,0)\bigg|_{z=1} e,$$

and is given explicitly as

$$\beta = (\mu_1'/\lambda_1')e + (A-I)(e\pi+D)^{-1}d. \tag{15}$$

For all queues of the $M/G/1$ paradigm, the traffic intensity, ρ, is given by $\rho=\pi\beta$ (see, e.g., [27]) which by (15) is seen to be $\rho = \mu_1'/\lambda_1'$, as expected.

Finally, using arguments analogous to those in [7], we obtain the following expression for the matrix $B(z,s)$,

$$B(z,s) = z^{-1}[sI-D_0]^{-1}[D(z)-D_0]A(z,s), \tag{16}$$

which implies that $B = (I-D_0^{-1}D)A$. By expanding $B(z,0)$ in a power series in z, we see that, for $n \geq 0$, the matrix B_n is given by

$$B_n = -D_0^{-1}\sum_{k=0}^{n} D_{k+1}A_{n-k}. \tag{17}$$

We note that D_0 is a stable matrix, so that $-D_0^{-1}$ is nonnegative. Also, the (i,j)-entry of the matrix $-D_0^{-1}D_k$ is the conditional probability that an idle period ends with the arrival of a batch of size k and the arrival phase j, given that the idle period began with the arrival phase i. Therefore, the above formula has an obvious probabilistic interpretation.

3.2 The Stationary Queue Length at Departures

The stationary vector of the Markov chain $P = \tilde{P}(\infty)$, embedded at departures from the queue, is the joint probability density of the stationary queue length and the phase of the arrival process. From (10), we have

$$P = \begin{bmatrix} B_0 & B_1 & B_2 & \cdots \\ A_0 & A_1 & A_2 & \cdots \\ 0 & A_0 & A_1 & \cdots \\ 0 & 0 & A_0 & \cdots \\ \cdot & \cdot & \cdot & \\ \cdot & \cdot & \cdot & \end{bmatrix}. \tag{18}$$

Writing the stationary probability vector x of P in the partitioned form $x = (x_0, x_1, \ldots)$, where x_i, $i \geq 0$, are m-vectors, the system of equations, $xP = x$, can be expanded as

$$x_i = x_0 B_i + \sum_{\nu=1}^{i+1} x_\nu A_{i+1-\nu}, \qquad \text{for } i \geq 0 . \tag{19}$$

Set $\boldsymbol{X}(z) = \sum_{i=0}^{\infty} x_i z^i$. Using the expressions for $B(z)$ and equations (19), it follows that

$$\boldsymbol{X}(z)[zI - A(z)] = x_0[zB(z) - A(z)] = -x_0 D_0^{-1} D(z) A(z), \tag{20}$$

so that the generating function, $\boldsymbol{X}(z)$, is completely determined by the vector x_0.

To motivate the discussion, we note that x_{0j}, $1 \leq j \leq m$, is the stationary probability that a departure leaves the system empty with the arrival process in state j. Equivalently, it is the inverse of the expected number of transitions, between successive visits to the state $(0,j)$, in the Markov chain embedded at departures. The latter quantity is derived from the first passage time distributions for successive returns to the set $\{(0,1),\ldots,(0,m)\}$. If we define the *level* $\boldsymbol{i}$ to be the set of states $\{(i,1),\ldots,(i,m)\}$, $i \geq 0$, then from the structure of the matrix P in (18) it is clear that, in order to reach level $\mathbf{0}$ from level $\boldsymbol{i}$, $i \geq 1$, each level in between must be visited, i.e., the process is *skip-free to the left.* Moreover, the chance mechanism governing the first passage from level $\boldsymbol{i+1}$ to level $\boldsymbol{i}$ is the same for all levels with $i \geq 0$, because of the spatial homogeneity of the Markov chain. Therefore, the first passage time distributions from level $\boldsymbol{i+1}$ to level $\boldsymbol{i}$, $i \geq 0$, play a crucial role in the study of the return time distributions of the level $\mathbf{0}$.

First Passage Times from Level $\boldsymbol{i+1}$ to Level $\boldsymbol{i}$.

Define $\tilde{G}_{jj'}{}^{[r]}(k;x)$, $k \geq 1$, $x \geq 0$, as the probability that the first passage from the state $(i+r,\ j)$ to the state (i,j'), $i \geq 1$, $1 \leq j,\ j' \leq m$, $r \geq 1$, occurs in exactly k transitions and no later than time x, and that (i,j') is the first state visited in level $\boldsymbol{i}$. $\tilde{G}^{[r]}(k;x)$ is the matrix with elements $\tilde{G}_{jj'}{}^{[r]}(k;x)$.

By a first passage argument, it can be shown [27] that the joint transform matrix $G(z,s)$, defined by

$$G(z,s) = \sum_{k=1}^{\infty} \int_0^{\infty} e^{-sx} d\tilde{G}^{[1]}(k;x) z^k, \qquad \text{for } |z| \leq 1, \quad \text{Re } s \geq 0,$$

satisfies the nonlinear matrix equation

$$G(z,s) = z \sum_{\nu=0}^{\infty} A_\nu(s) G^\nu(z,s). \tag{21}$$

In the context of the *BMAP/G/*1 queue, $G(z,s)$ governs the number served during, and the duration of, the busy period. It can be shown that the joint transform matrix governing the number served during and the duration of a busy period starting with r customers, is given by $G^r(z,s)$. Equation (21) is the key equation in the matrix analytic solution to queues of the *M/G/*1 paradigm. It is the matrix analogue of Takács' equation for the busy period in the ordinary *M/G/*1 queue [28]. We also define the matrices

$$G(z) = G(z,0) = z\sum_{v=0}^{\infty} A_v G^v(z),$$

$$G = G(1) = \sum_{v=0}^{\infty} A_v G^v. \tag{22}$$

The matrix G is stochastic when $\rho \leq 1$. For $\rho < 1$, the invariant probability vector $\boldsymbol{g}$, of the positive stochastic matrix G, satisfies

$$\boldsymbol{g}G = \boldsymbol{g},\ \boldsymbol{g}\boldsymbol{e} = 1. \tag{23}$$

The vector $\boldsymbol{\mu}$ is defined by

$$\boldsymbol{\mu} = \frac{d}{dz}G(z,0)\Big|_{z=1} \boldsymbol{e},$$

and its j-th component, $1 \leq j \leq m$, is the expected number of transitions (i.e., services) from a state $(i+1,j)$ to level i. By differentiating in (21) we may derive the explicit expression

$$\boldsymbol{\mu} = (I - G + \boldsymbol{e}\boldsymbol{g})[I - A + (\boldsymbol{e} - \boldsymbol{g})\boldsymbol{\beta}]^{-1}\boldsymbol{e}. \tag{24}$$

The equality $\boldsymbol{g}\boldsymbol{\mu} = (1-\rho)^{-1}$ holds.

Computation of the Vector x_0

The quantity $(x_{0j})^{-1}$ is, by a classical property of Markov chains, the mean recurrence time of the state $(0,j)$ in the Markov chain P. If we now consider the chain P only at its visits to the level $\mathbf{0}$, and record the indices of the states visited as well as the number of transitions in P between consecutive visits to $\mathbf{0}$, we obtain an irreducible m-state Markov renewal process with transition matrix determined by the

matrix generating function $K(z)$. The matrix $K(z)$ is obtained as follows. Define the quantities $\tilde{K}_{jj'}(k;x)$, $k \geq 1$, $x \geq 0$, $1 \leq j, j' \leq m$, as the conditional probability that the Markov renewal process, starting in the state $(0,j)$, returns to the set **0** for the first time in exactly k transitions and no later than time x, by hitting the state $(0,j')$. The joint transform matrix of $\tilde{K}(k;x) = [\tilde{K}_{jj'}(k;x)]$, is defined by

$$K(z,s) = \sum_{k=1}^{\infty} \int_0^{\infty} e^{-sx} d\tilde{K}(k;x) z^k, \qquad \text{for } |z| \leq 1, \ \text{Re}(s) \geq 0.$$

A first passage argument shows that $K(z,s)$ satisfies

$$K(z,s) = z \sum_{v=0}^{\infty} B_v(s) G^v(z,s).$$

As before, we define the matrices

$$K(z) = K(z,0) = z \sum_{v=0}^{\infty} B_v G^v(z),$$

$$K = K(1) = K(1,0) = \sum_{v=0}^{\infty} B_v G^v .$$

Using (16), it can be shown that

$$K(z,s) = [sI - D_0]^{-1} [D[G(z,s)] - D_0], \tag{25}$$

where

$$D[G(z,s)] = \sum_{j=0}^{\infty} D_j G^j(z,s),$$

so that

$$K = -D_0^{-1} [D[G] - D_0] = I - D_0^{-1} D[G]. \tag{26}$$

Remark: The matrix $D[G]$ has a simple interpretation. Consider the arrival process at a time epoch

during an idle period and let its phase at that time be i. During the next infinitesimal time interval, the arrival process may remain in the phase i or could change state to k without an arrival with probability $(D_0)_{ik}dt$, *or* a batch arrival of size l may occur and the phase may change to j with probability $(D_l)_{ij}dt$. That arriving batch initiates a busy period which ends in the phase k with conditional probability $(G^l)_{jk}$. If we "excise" the time interval corresponding to the busy period, we obtain an "instantaneous" transition from i to k, whose elementary probability is given by $(\sum_{l=1}^{\infty} D_l G^l)_{ik}dt$. The matrix $D[G]$ may therefore be considered as the infinitesimal generator of a Markov process, obtained by excising the busy periods.

By arguments classical in the theory of Markov renewal processes ([29], [30]) it can be shown [20] that x_0 can be expressed in terms of the invariant probability vector $\boldsymbol{\kappa}$ of K, which satisfies $\boldsymbol{\kappa}K = \boldsymbol{\kappa}$, $\boldsymbol{\kappa}e = 1$, and the vector $\boldsymbol{\kappa}^* = K^{(1)}(1)e$, of the row-sum means of $K(z)$.

Specifically, we have

$$x_0 = \frac{\boldsymbol{\kappa}}{\boldsymbol{\kappa}\,\boldsymbol{\kappa}^*} \quad , \tag{27}$$

where $\boldsymbol{\kappa}^*$ is obtained explicitly by differentiating in (25) as

$$\begin{aligned} \boldsymbol{\kappa}^* &= \frac{d}{dz}K(z,0)\bigg|_{z=1} e \\ &= -D_0^{-1}\Big[D - D[G] + dg\Big]\Big[I - A + (e-\beta)g\Big]^{-1}e. \end{aligned} \tag{28}$$

Moments of the Queue Length at Departures

Recall from (20) that

$$X(z)[zI - A(z)] = -x_0 D_0^{-1} D(z)A(z). \tag{29}$$

Setting $z = 1$ in (29), adding $X(1)e\boldsymbol{\pi}$ to both sides and observing that $I - A + e\boldsymbol{\pi}$ is nonsingular, leads to

$$\boldsymbol{X}(1) = \boldsymbol{\pi} + -x_0 D_0^{-1} DA(I-A+e\boldsymbol{\pi})^{-1}. \tag{30}$$

The factorial moment vectors of the queue length at departures are given by the quantities $\boldsymbol{X}^{(n)}(1)$ and can be computed recursively by differentiation in (29). We present below, the final expressions for the first two moments. See [20] for the derivations.

Define $U(z) = -x_0 D_0^{-1} D(z)A(z)$ and write the derivatives $\boldsymbol{X}^{(i)} = \boldsymbol{X}^{(i)}(1)$, $U^{(i)} = U^{(i)}(1)$ and $A^{(i)} = A^{(i)}(1)$, for $i \geq 1$, and let $\boldsymbol{X} = \boldsymbol{X}(1)$. We then get

$$\boldsymbol{X}^{(1)} = (\boldsymbol{X}^{(1)} e)\boldsymbol{\pi} + \{U^{(1)} - \boldsymbol{X}[I-A^{(1)}]\}(I-A+e\boldsymbol{\pi})^{-1}, \tag{31}$$

$$\boldsymbol{X}^{(1)} e = \frac{1}{2(1-\rho)}\left\{\boldsymbol{X}A^{(2)} e + U^{(2)} e + 2\{U^{(1)} - \boldsymbol{X}[I-A^{(1)}]\}(I-A+e\boldsymbol{\pi})^{-1}\boldsymbol{\beta}\right\}, \tag{32}$$

and

$$\boldsymbol{X}^{(2)} e = \frac{1}{3(1-\rho)}\left\{3\boldsymbol{X}^{(1)} A^{(2)} e + \boldsymbol{X}A^{(3)} e + U^{(3)} e \right. \tag{33}$$
$$\left. + 3\{U^{(2)} + \boldsymbol{X}A^{(2)} - 2\boldsymbol{X}^{(1)}[I-A^{(1)}]\}(I-A+e\boldsymbol{\pi})^{-1}\boldsymbol{\beta}\right\}.$$

3.3 The Stationary Queue Length Distribution at Time t

In this section, a relationship between the stationary queue length density at an arbitrary time t to the stationary queue length at departures is given. This is accomplished by a classical argument based on the Key Renewal Theorem for Markov renewal processes ([29],[30]), and the details of the proof can be found in [2] or [20].

Let $\xi(t)$ denote the queue length and $J(t)$ be the phase of the arrival process at time t. We now consider the continuous parameter process $\{[\xi(t), J(t)], t \geq 0\}$. The time-dependent joint distribution of the queue length and the arrival phase is given by the conditional probabilities

$$Y(k,j;t) = P\{\xi(t)=k, J(t)=j \mid \xi_0=k_0, J_0=j_0\},$$

for $k \geq 0$, $1 \leq j \leq m$, $t \geq 0$. We can show that the limits

$$y_{kj} = \lim_{t\to\infty} Y(k,j;t) \ , \quad \text{for } k \geq 0,\ 1 \leq j \leq m,$$

exist and are simply related to the components of the invariant vector $\boldsymbol{x}$. For $k \geq 0$ let $y_k = (y_{k1}, y_{k2}, \ldots, y_{km})$. The vector y_0 is given by

$$y_0 = -\lambda_1'^{-1} x_0 D_0^{-1}, \tag{34}$$

and $y_0 e = 1-\rho$, as expected. The generating function, $\boldsymbol{Y}(z) = \sum_{i=0}^{\infty} y_i z^i$ is related to the generating function $\boldsymbol{X}(z)$ by the equality

$$\boldsymbol{Y}(z)D(z) = \lambda_1'^{-1}(z-1)\boldsymbol{X}(z), \qquad \text{for } |z|<1, \tag{35}$$

$$\boldsymbol{Y}(1) = \boldsymbol{\pi}.$$

By comparing the coefficients of z^i in (35), we see that the vectors $\boldsymbol{y}_i$ are related to the vectors $\boldsymbol{x}_i$ by:

$$y_{i+1} = \left[\sum_{j=0}^{i} y_j D_{i+1-j} - \lambda_1'^{-1}(x_i - x_{i+1})\right](-D_0^{-1}), \qquad \text{for } i \geq 0. \tag{36}$$

Moments of the Queue Length at an Arbitrary Time

Expressions for the moments of the queue length at an arbitrary time can be obtained by differentiation in (37). We illustrate for the first two moments of the queue length, given by $\boldsymbol{Y}^{(1)}(1)e$ and $\boldsymbol{Y}^{(2)}(1)e$, respectively. Writing the derivatives as $\boldsymbol{Y}^{(i)} = \boldsymbol{Y}^{(i)}(1)$ and $D^{(i)} = D^{(i)}(1)$, for $i \geq 1$, we have

$$\boldsymbol{Y}^{(1)} D = \lambda_1'^{-1} \boldsymbol{X} - \boldsymbol{\pi} D^{(1)}, \tag{37}$$

$$\boldsymbol{Y}^{(2)} D = 2[\lambda_1'^{-1} \boldsymbol{X}^{(1)} - \boldsymbol{Y}^{(1)} D^{(1)}] - \boldsymbol{\pi} D^{(2)}, \tag{38}$$

$$\boldsymbol{Y}^{(3)} D = 3[\lambda_1'^{-1} \boldsymbol{X}^{(2)} - \boldsymbol{Y}^{(2)} D^{(1)} - \boldsymbol{Y}^{(1)} D^{(2)}] - \boldsymbol{\pi} D^{(3)}. \tag{39}$$

Adding $\boldsymbol{Y}^{(1)} e\boldsymbol{\pi}$ to both sides of (37) and observing that $e\boldsymbol{\pi}+D$ is nonsingular, we obtain

$$\boldsymbol{Y}^{(1)} = (\boldsymbol{Y}^{(1)} e)\boldsymbol{\pi} + [\lambda_1'^{-1}\boldsymbol{X} - \boldsymbol{\pi}D^{(1)}](e\boldsymbol{\pi}+D)^{-1}. \tag{40}$$

Postmultiplying by e in (38) yields

$$\boldsymbol{Y}^{(1)} D^{(1)} e = \lambda_1'^{-1}\boldsymbol{X}^{(1)} e - \frac{1}{2}\boldsymbol{\pi}D^{(2)} e. \tag{41}$$

Postmultiplying (40) by $D^{(1)} e$ and substituting (41) leads to

$$\boldsymbol{Y}^{(1)} e = \boldsymbol{X}^{(1)} e - \frac{1}{2}\lambda_1'\boldsymbol{\pi}D^{(2)} e + [\lambda_1'\boldsymbol{\pi}D^{(1)} - \boldsymbol{X}](e\boldsymbol{\pi}+D)^{-1} D^{(1)} e. \tag{42}$$

Similar manipulations lead to

$$\begin{aligned}\boldsymbol{Y}^{(2)} e = \boldsymbol{X}^{(2)} e &- \lambda_1'\boldsymbol{Y}^{(1)} D^{(2)} e - \frac{1}{3}\lambda_1'\boldsymbol{\pi}D^{(3)} e \\ &- 2[\boldsymbol{X}^{(1)} - \lambda_1'\boldsymbol{Y}^{(1)} D^{(1)} - \lambda_1'\boldsymbol{\pi}D^{(2)}](e\boldsymbol{\pi}+D)^{-1} D^{(1)} e,\end{aligned} \tag{43}$$

where $\boldsymbol{X}(1)$, $\boldsymbol{X}^{(1)}$, and $\boldsymbol{X}^{(2)} e$, are given by (30), (31) and (33), respectively.

The generating function for the queue length at (batch) arrival epochs is given by $\lambda_1'^{-1}\boldsymbol{Y}(z)\sum_{j=1}^{\infty} D_j e = -\lambda_1'^{-1}\boldsymbol{Y}(z) D_0 e$, so that the calculation of moments for that distribution is again routine.

3.4 The Virtual Waiting Time Distribution

In this section, we state results for the virtual waiting time distribution. First, we define the following quantities

$\tilde{\boldsymbol{W}}(x)$ = $\{\tilde{W}_1(x), \cdots, \tilde{W}_m(x)\}$, where $\tilde{W}_j(x)$ is the joint probability that at an arbitrary time the arrival process is in phase j and that a *virtual* customer who arrives at that time waits at most a time x before entering service,

$\tilde{w}_v(x) = \tilde{\boldsymbol{W}}(x)\boldsymbol{e}$, the virtual waiting time distribution,

Also for use in what follows, we will need the Laplace-Stieltjes transforms

$$\boldsymbol{W}_v(s) = \int_0^\infty e^{-sx} d\tilde{\boldsymbol{W}}(x), \qquad w_v(s) = \boldsymbol{W}_v(s)\boldsymbol{e}.$$

Ramaswami [2], has shown that the Laplace-Stieltjes transform $\boldsymbol{W}_v(s)$ satisfies

$$\boldsymbol{W}_v(s) = s\boldsymbol{y}_0[sI + D(H(s))]^{-1}, \tag{44}$$

$$\boldsymbol{W}(0) = \boldsymbol{\pi},$$

from which it follows that

$$w_v(s) = s\boldsymbol{y}_0[sI + D(H(s))]^{-1}\boldsymbol{e}. \tag{45}$$

Remark: Although the analytic derivation of (44) is somewhat involved (based again on the Key Renewal Theorem for Markov renewal processes) the final results, (44) and (45), are quite elegant. Note that they are a direct generalization of the classical Pollaczek-Khinchin formula for the waiting time in the *M/G/1* queue. In particular, if $D_0 = -\lambda$ and $D_1 = \lambda$ then the *BMAP* is a Poisson process of rate λ, $y_0 = 1 - \rho$, and (45) reduces to the familiar form,

$$w_v(s) = \frac{s(1-\rho)}{s - \lambda + \lambda H(s)}.$$

Moments of the Virtual Waiting Time Distribution

We now derive expressions for the first two moments of the virtual waiting time distribution. These expressions are in a simpler form than those in [2] and although they appear quite complicated they are easily implemented for numerical computation. We begin with Equation (44) written as

$$s\boldsymbol{w}(s) + \boldsymbol{w}(s)D(H(s)) = s\boldsymbol{y}_0. \tag{46}$$

To simplify the notation and to aid in the numerical implementation of the formulas, we define

$V(s)=D(H(s))$, and write $\boldsymbol{w}^{(i)} = \boldsymbol{w}^{(i)}(0)$, $V^{(i)} = V^{(i)}(0)$ for $i \geq 1$, and let μ_i' be the i-th moment of $\tilde{H}(\cdot)$ (if it exists). Then by successively differentiating $V(s)$ we get

$$V^{(1)} = -\mu_1' D^{(1)}$$

$$V^{(2)} = (\mu_1')^2 D^{(2)} + \mu_2' D^{(1)}$$

$$V^{(3)} = -(\mu_1')^3 D^{(3)} - 3\mu_1'\mu_2' D^{(2)} - \mu_3' D^{(1)}$$

We note that $\boldsymbol{\pi} \boldsymbol{D}^{(1)} \boldsymbol{e} = \lambda_1'^{-1}$. We also define $\boldsymbol{v}^i = V^{(i)} \boldsymbol{e}$. Now, by successively differentiating in (46) we obtain, after some laborious algebra,

$$-\boldsymbol{w}^{(1)} \boldsymbol{e} = \frac{1}{2(1-\rho)} \left[2\rho + 2(y_0 - \boldsymbol{\pi} V^{(1)})(\boldsymbol{e}\boldsymbol{\pi}+D)^{-1} \boldsymbol{v}_1 + \boldsymbol{\pi} \boldsymbol{v}_2 \right], \tag{47}$$

$$\boldsymbol{w}^{(1)} = (\boldsymbol{w}^{(1)} \boldsymbol{e})\boldsymbol{\pi} - \boldsymbol{\pi} + (y_0 - \boldsymbol{\pi} V^{(1)})(\boldsymbol{e}\boldsymbol{\pi}+D)^{-1},$$

$$\boldsymbol{w}^{(2)} \boldsymbol{e} = \frac{1}{3(1-\rho)} \left[3(2\boldsymbol{w}^{(1)} + 2\boldsymbol{w}^{(1)} V^{(1)} + \boldsymbol{\pi} V^{(2)})(\boldsymbol{e}\boldsymbol{\pi}+D)^{-1} \boldsymbol{v}_1 - 3\boldsymbol{w}^{(1)} \boldsymbol{v}_2 - \boldsymbol{\pi} \boldsymbol{v}_3 \right]. \tag{48}$$

The first two moments of the virtual waiting time are thus given by (47) and (48), respectively. For example, we see that for the $M^X/G/1$ queue with arrival rate λ and batch size generating function $p(z)$, $D(z)=-\lambda+\lambda p(z)$ so that (47) reduces to

$$E(W) = \frac{\lambda[(\mu_1')^2 p^{(2)}(1) + \mu_2' p^{(1)}(1)]}{2(1-\rho)}.$$

We also note that the moments of the waiting time seen by an arrival may be obtained in terms of the vectors $\boldsymbol{w}^{(n)}$. For example, the mean waiting time of the first customer in a batch at an arrival epoch is $-(\boldsymbol{\pi}\boldsymbol{d})^{-1} \boldsymbol{w}^{(1)} \boldsymbol{d}$, etc. For single arrivals this is the actual waiting time. The actual waiting time for an arbitrary customer with batch arrivals is more complicated and will be reported elsewhere.

3.5 The Classical Algorithm

The outline of the analysis in the previous sections also serves as a recipe for the implementation of a computational algorithm. Indeed, one of the major benefits of the matrix-analytic approach to the solution of stochastic models is that intermediate quantities which arise in the analytic derivation are also needed in the numerical procedure and, due to the fact that they are derived using probabilistic arguments, they are already in a form which is suitable for numerical evaluation. That is, they often involve arithmetic operations on only nonnegative quantities, thus avoiding common sources of round-off error. Moreover, many times the obvious procedures for solving the required nonlinear matrix equations can be shown to produce monotonically increasing estimates of the unique solution so that the algorithms themselves are inherently stable. Such is the case for the current model, as it was developed in [2] and summarized more recently in [20]. One practical problem with the approaches there (besides the more complicated notation) is that the algorithms, in their complete generality, require formidable resources in both CPU time as well as storage. To give some indication of the computational complexities we briefly outline the general numerical procedure. Since this description is not meant to be implemented as a specific algorithm, it will be informal.

We first assume that the service time distribution, $\tilde{H}(\cdot)$, and the sequence $\{D_j\colon j \geq 0\}$ which specifies the arrival process, are given. Some quantities such as

$$D^{(1)}(1) = \sum_{j=1}^{\infty} jD_j,$$

may be explicitly available depending on the formulation of the problem. If they are not, then they must be numerically computed. We assume that all such setup computations have been completed.

Step 1: *Computation of the matrices* A_n. For a general service time distribution, $\tilde{H}(\cdot)$, the matrices A_n defined in (11) need to be numerically integrated. This is quite delicate since the matrices $P(n,t)$, $n \geq 0$, $t \geq 0$, are themselves computed by numerically integrating the infinite system of differential-difference equations (6). Neuts discusses in [23] a procedure which adaptively truncates (6) at both upper and lower indices as t increases. Since the A_n's are the starting point for a long series of numerical computations, they need to be computed to a high degree of accuracy and it is clear that the more accuracy required for the matrices A_n, the finer the mesh is required for solving the differential equations. Also, for each n, the matrix A_n requires m^2 numerical integrations. Once the sequence $\{A_n\colon 0 \leq n \leq M\}$, for a suitably chosen truncation index M, is computed, these matrices need to be stored. Guidelines for choosing the truncation index, M, are given in [23]. Now the matrix A, defined in (14), is computed by summation of the sequence $\{A_n\}$. This can be compared with a direct numerical integration in (14) as a check on the accuracy of computations so far. The sequence $\{A_n\}$ can also be normalized in an

appropriate way to ensure that its sum is stochastic.

Step 2: *Computation of the matrix G.* The obvious numerical procedure for computing the matrix G is by successive substitution in Equation (22), starting with $G=0$. It has been shown that this produces a sequence of nonnegative matrices which increases monotonically to the unique solution of (22). As the traffic intensity, ρ, gets moderate to large, however, the convergence gets slower. A slightly faster convergence can be obtained in the following modification,

$$G_{k+1} = \sum_{\substack{n=0 \\ n\neq 1}}^{\infty} (I-A_1)^{-1} A_n (G_k)^n,$$

starting with $G_0=0$. It was pointed out in [20] that the speed of convergence can be enhanced even further in some cases by starting the iteration with a stochastic matrix. The estimating sequence no longer possesses the monotonicity property but each iterate is itself stochastic and we have had satisfactory experience with this approach. In either case, the above (truncated) sum is computed by Horner's method.

Applying the matrix version of Newton's method, (see, e.g., [31]), to (22) results in many fewer iterations being required but a large system of linear equations needs to be solved at each iteration. This is discussed in [32] where an acceleration method based on a first order approximation to Newton's method is proposed. Experience with this approach has shown that in some cases the CPU requirement may be reduced by 50-70 percent.

Once G is computed to the desired accuracy, the stationary probability vector, $\boldsymbol{g}$, is computed by standard methods.

Step 3: *Computation of the vector* $\boldsymbol{\beta}$. The vector $\boldsymbol{\beta}$, defined in (15) is evaluated in the obvious manner.

Step 4: *Computation of the vector* $\boldsymbol{\mu}$. The system of linear equations

$$[I-A+(\boldsymbol{e}-\boldsymbol{g})\boldsymbol{\beta}]\,\boldsymbol{u} = \boldsymbol{e},$$

is solved for $\boldsymbol{u}$. Then $\boldsymbol{\mu} = (I-G+\boldsymbol{eg})\,\boldsymbol{u}$, as seen from (24). At this point, the identity $\boldsymbol{g\mu} = (1-\rho)^{-1}$, is verified with the computed estimates of $\boldsymbol{g}$ and $\boldsymbol{\mu}$. This serves as a powerful accuracy check on the numerical computations so far. Such accuracy checks are useful by-products of the matrix-analytic approach.

Step 5: *Computation of the matrix* $D[G]$. The matrix

$$D[G] = \sum_{j=0}^{\infty} D_j G^j,$$

is computed using Horner's algorithm.

Step 6: *Computation of K and* $\boldsymbol{\kappa}$. The matrix K, given by (26), is computed directly. Its stationary probability vector, $\boldsymbol{\kappa}$, is then computed by standard methods.

Step 7: *The vector* $\boldsymbol{x}_0$. The vector $\boldsymbol{\kappa}^*$ is computed from (28) using the vector $\boldsymbol{u}$ from Step 4. $\boldsymbol{x}_0$ is now obtained from (27).

Step 8: *Moments of the queue length at departures.* The first two moments of the queue length distribution at departures, $\boldsymbol{X}^{(1)}(1)\boldsymbol{e}$, and $\boldsymbol{X}^{(2)}(1)\boldsymbol{e}$, are given explicitly in terms of $\boldsymbol{x}_0$ by Equations (32) and (33), respectively.

Step 9: *The vector* $\boldsymbol{y}_0$. Once the vector $\boldsymbol{x}_0$ is obtained, $\boldsymbol{y}_0$ is computed by (34).

Step 10: *Moments of the queue length at an arbitrary time.* $\boldsymbol{Y}^{(1)}(1)\boldsymbol{e}$ and $\boldsymbol{Y}^{(2)}(1)\boldsymbol{e}$ are computed from (42) and (43), respectively.

Step 11: *Moments of the virtual waiting time distribution.* The first two moments of the virtual waiting time are given explicitly in terms of $\boldsymbol{y}_0$ by (47) and (48), respectively.

Step 12: *The distributions of the queue length at departures.* Equation (19) can be solved for $\boldsymbol{x}_{i+1}$ to get a recursion for $\boldsymbol{x}_{i+1}$ in terms of $\boldsymbol{x}_j$, $0 \le j \le i$. Unfortunately, this recursion suffers from ''catastrophic cancellation''[33] which results from subtracting small quantities of the same order. An alternative for solving (19) is to rewrite it in a form which is directly suitable for a block Gauss-Seidel iterative procedure (see e.g., [6] and [20]). Although this method was implemented in [6] and it was seen to be a numerically stable algorithm, the Gauss-Seidel procedure suffers from slow convergence, especially for high traffic intensities. The following procedure, obtained in [34], constitutes a major breakthrough in the efficient computation of the sequence $\{\boldsymbol{x}_k\}$. It is the natural extension to the matrix case of a simple device, due to P. J. Burke, to avoid loss of significance in similar computations for the $M/G/1$ queue.

Given the vector $\boldsymbol{x}_0$, the vectors $\boldsymbol{x}_i$, for $i \ge 1$, are recursively obtained from the formula

$$\boldsymbol{x}_i = \left[\boldsymbol{x}_0 \bar{B}_i + \sum_{j=1}^{i-1} \boldsymbol{x}_j \bar{A}_{i+1-j}\right](I - \bar{A}_1)^{-1}, \quad i \ge 1, \tag{49}$$

where

$$\bar{B}_\nu = \sum_{i=\nu}^{\infty} B_i G^{i-\nu}, \quad \text{and} \quad \bar{A}_\nu = \sum_{i=\nu}^{\infty} A_i G^{i-\nu}, \quad \nu \geq 0.$$

Note that all quantities in this recursion are nonnegative, thus avoiding the catastrophic cancellation suffered by other recursions. Further, as observed in [34], the implementation of (49) can be done efficiently by noting that as $i \to \infty$, $\bar{B}_i$, $\bar{A}_i \to 0$. One may therefore choose a large index i, (e.g., i can be chosen so that $\sum_{k=i+1}^{\infty} B_k e$ and $\sum_{k=i+1}^{\infty} A_k e$ have negligibly small components), and set $\bar{B}_i$ and $\bar{A}_i = 0$. The other required matrices are computed by implementing the backward recursions

$$\bar{B}_k = B_k + \bar{B}_{k+1} G, \quad \text{and} \quad \bar{A}_k = A_k + \bar{A}_{k+1} G, \quad \text{for} \quad k = i-1, i-2, \cdots, 0.$$

Note that the matrices A_k, and B_k, $k \geq 0$, still need to be computed.

Step 13: *The distributions of the queue length at an arbitrary time.* The sequence, $\{y_k, k \geq 1\}$, is computed recursively from (36) in terms of the sequence $\{x_k\}$.

Step 14: *The distribution of the virtual waiting time.* There are several methods for computing the distribution of the virtual waiting time distribution. The first is by numerical inversion of the Laplace-Stieltjes transform as given by (45). The method presented in [35] has been used successfully for inverting similar transforms. Another useful transform inversion technique is given in [36]. An alternative method is to convert (44) into the equivalent Volterra integral equation

$$\tilde{W}(x) = \tilde{W}(0) + \int_0^x \tilde{W}(u) \tilde{\Theta}(x-u) du, \qquad x \geq 0,$$

where

$$\tilde{\Theta}(x) = -\sum_{k=0}^{\infty} D_k \tilde{H}^{(k)}(x),$$

(see, e.g., [37] and [20].) There are standard methods for the numerical solution of Volterra-type equations, see, e.g., [38] and [39].

The above algorithm has been implemented successfully for several special cases of the *BMAP/G/1* queue [6]. It is clear that implementation of the algorithm in its full generality is a major task. Also, if accomplished, it would present a major burden on both CPU resources and memory requirements. In the next section, we present some new results which will eliminate the need for most of the computations and

storage requirements in the above algorithm.

4. NEW RESULTS FOR THE *BMAP/G/1* QUEUE AND THEIR IMPLICATIONS

A recent result of Sengupta [40] shows that the solution of the nonlinear matrix equation arising in the *GI/PH/1* queue has a matrix exponential form. It was immediately apparent that the solution to the corresponding equation, (22), for the *PH/G/1* queue also had a matrix exponential form. This was proved, using a probabilistic argument, for the *MMPP/G/1* queue by Neuts [16] and was extended to the *MAP/G/1* queue in [7]. The result for the *PH/G/1* queue was proved in [41] by using a duality result between the *GI/PH/1* and *PH/G/1* queues. This exponential representation leads to several explicit formulae which reduce substantially the computations involved in the algorithmic solutions of these models. We now extend this result to the *BMAP/G/1* queue and discuss the specific simplifications that occur in the algorithm.

4.1 The Matrix $G(z,s)$

By adapting methodology developed in [40], we generalize the proof given in [16], for the *MMPP/G/1* queue, to prove the following key result for the *BMAP/G/1* queue.

Theorem: For the *BMAP/G/1* queue, the matrix $G(z,s)$, satisfying (21), also satisfies the equation

$$G(z,s) = z\int_0^\infty e^{-sx}e^{D[G(z,s)]x}d\tilde{H}(x), \tag{50}$$

which readily implies that

$$G = \int_0^\infty e^{D[G]x}d\tilde{H}(x). \tag{51}$$

Remark: Since $D[G]$ is the infinitesimal generator of an irreducible Markov process, $exp(D[G]x)$ is strictly positive for $x>0$, and (51) implies that G is strictly positive.

Proof: As in [16], we consider the continuous parameter process $\{(J(t),R(t)),\ t\geq 0\}$, where $J(t)$ is the phase of the arrival process and $R(t)$ is the residual busy period at time t (or equivalently, the virtual waiting time or amount of work in the system at time t). When, at time t, the queue is empty, we set $R(t)=0$. Next, we introduce the conditional probability $\Psi_{ij}(x;k,y)$ that, given $J(0)=i$, $1\leq i\leq m$, and

$R(0)=x$, $x>0$, the (current) busy period ends before time y, $y \geq x$, with the arrival process in phase j and involves a service of k, $k \geq 0$, new customers. The $m \times m$ matrix $\Psi(x;k,y)$ has elements $\Psi_{ij}(x;k,y)$ and we define the transform $\Psi^*(x;z,s)$ by

$$\Psi^*(x;z,s) = \sum_{k=0}^{\infty} \int_{x}^{\infty} e^{-ys} d_y \Psi(x;k,y) z^k.$$

It was shown in [16] that

$$\Psi^*(x_1+x_2;z,s) = \Psi^*(x_1;z,s)\Psi^*(x_2;z,s), \qquad \text{for } x_1>0, \ x_2>0, \tag{52}$$

and by continuity, we set $\Psi^*(0;z,s) = I$.

The busy period starting with n customers is governed by the matrix $G^n(z,s)$. By conditioning on the total service time of those first n customers, we have

$$G^n(z,s) = z^n \int_0^{\infty} \Psi^*(y;z,s) d\tilde{H}^{(n)}(y), \tag{53}$$

where $\tilde{H}^{(n)}(\cdot)$ denotes the n-fold convolution of $\tilde{H}(\cdot)$. We also have

$$\Psi(x;0,y) = \begin{cases} e^{D_0 x}, & y \geq x, \\ 0, & 0 \leq y < x, \end{cases}$$

and for $k \geq 1$,

$$\Psi(x;k,y) = \sum_{j=1}^{k} \underset{(u)}{\int_0^x} du \underset{(v)}{\int_0^{y-x}} e^{D_0 u} D_j \Psi(x-u+v;k-j,y-u) d\tilde{H}^{(j)}(v).$$

Taking transforms leads to

$$\Psi^*(x;z,s) = e^{-(sI-D_0)x} + \int_{0}^{x} du \int_{x}^{\infty} e^{-sy} \int_{0}^{y-x} e^{D_0 x} \sum_{j=1}^{\infty} D_j z^j \Psi(x-u+v;z,y-u)\, d\tilde{H}^{(j)}(v)$$

$$= e^{-(sI-D_0)x} + \int_{0}^{x} dw \int_{0}^{\infty} \int_{w+v}^{\infty} e^{-s(u-w+x)}\, e^{D_0(x-w)} \sum_{j=1}^{\infty} D_j z^j \Psi(w+v;z,u)\, d\tilde{H}^{(j)}(v)$$

$$= e^{-(sI-D_0)x} + \int_{0}^{x} e^{-(sI-D_0)(x-w)} \sum_{j=1}^{\infty} D_j G^j(z,s) \Psi^*(w;z,s)\, dw$$

Premultiplying both sides of the above equation by $e^{(sI-D_0)x}$, and differentiating with respect to x leads to

$$\frac{d}{dx}\Psi^*(x;z,s) = -(sI - D[G(z,s)])\Psi^*(x;z,s),$$

with the initial condition $\Psi^*(0;z,s)=I$, which implies

$$\Psi^*(x;z,s) = e^{-(sI-D[G(z,s)])x}$$

which proves the theorem. ■

Corollary:

1. The matrix G commutes with the matrix $D[G]$.
2. The vector $\boldsymbol{g}$, defined in (23), is also the stationary probability vector of the infinitesimal generator $D[G]$.
3. The vector $-\boldsymbol{g}D_0$ is a left eigenvector of the matrix K defined in (26).
4. The vectors $\boldsymbol{x}_0$ and $\boldsymbol{y}_0$ are given explicitly in terms of $\boldsymbol{g}$ as

$$\boldsymbol{x}_0 = \lambda_1'(1-\rho)\boldsymbol{g}(-D_0), \tag{54}$$

$$\boldsymbol{y}_0 = (1-\rho)\boldsymbol{g}. \tag{55}$$

5. The Laplace-Stieltjes transform, $W_v(s)$, satisfies

$$W_v(s) = s(1-\rho)g[sI+D(H(s))]^{-1} \tag{56}$$

from which it follows that

$$w_v(s) = s(1-\rho)g[sI+D(H(s))]^{-1}e. \tag{57}$$

Proof: From the representation of G in (51), it is clear that it commutes with $D[G]$. For the second part of the corollary, let $\boldsymbol{w}$ be a stationary vector of the infinitesimal generator $D[G]$. That is, $\boldsymbol{w}$ satisfies $\boldsymbol{w}D[G] = \mathbf{0}$, $\boldsymbol{we} = 1$. From (51), it is clear that, $\boldsymbol{w}$ is a stationary vector of G. The result follows from the uniqueness of $\boldsymbol{g}$. That $-gD_0$ is a left eigenvector of K is seen from the expression for K given in (26). Therefore, (27) implies that $x_0 = -cgD_0$ for some constant c. Equation (34) then yields that $y_0 = c\lambda_1'^{-1}g$ and since $y_0 e = 1-\rho$, we have $c = \lambda_1'(1-\rho)$. This proves parts 3 and 4. Part 5 is obtained by substitution into (44) and (45).■

5. A NEW ALGORITHM FOR THE *BMAP/G/*1 QUEUE

A major consequence of the preceding results is the explicit formulas for $\boldsymbol{x}_0$ and $\boldsymbol{y}_0$ in terms of the vector $\boldsymbol{g}$. In particular, once $\boldsymbol{g}$ is computed the moments of the queue length and virtual waiting time distributions can be immediately computed from Equations (31-33), (42-43), and (47-48). Thus, many of the intermediate steps in the classical algorithm are avoided.

A further consequence of (51) is an efficient algorithm for computing the matrix G. The basic idea is to use the concept of uniformization. Basically, this says that if Q is the infinitesimal generator of a continuous time Markov process, then

$$e^{Qt} = e^{\theta t(L-I)} = e^{-\theta t}e^{\theta tL}$$

$$= \sum_{n=0}^{\infty} e^{-\theta t}\frac{(\theta t)^n}{n!}L^n, \tag{58}$$

where $\theta = \max_i(-Q_{ii})$, and $L = I+\theta^{-1}Q$ is a stochastic matrix. If we have a Poisson process of rate θ and at each Poisson epoch we make a transition in the discrete time Markov chain with transition probability matrix L, then this process is equivalent to the original Markov process with generator Q. The Poisson process with rate θ is called the *uniformizing* Poisson process for the Markov process with generator Q.

The utility of (58) is that the summation there involves only nonnegative elements and thus serves as a practical method for numerically evaluating the matrix exponential e^{Qt}.

Using this in (51) leads to

$$G = \sum_{n=0}^{\infty} \gamma_n (I+\theta^{-1}D[G])^n, \tag{59}$$

where $\gamma_n = \int_0^{\infty} e^{-\theta x} \frac{(\theta x)^n}{n!} d\tilde{H}(x)$, for $n \geq 0$. Thus, G can be computed by successively iterating in the following recursion,

$$H_{n+1,k} = [I + \theta^{-1}D[G_k]]H_{n,k} \quad n=0,1,2,..., \tag{60}$$

$$G_{k+1} = \sum_{n=0}^{\infty} \gamma_n H_{n,k}, \tag{61}$$

where $H_{0,k} = I$ and $\theta = \max_i \{(-D_0)_{ii}\}$. If we start with $G_0 = 0$, it can be shown that the successive values of G_k are monotonically increasing to the unique solution, however, the convergence can be slow especially for high values of ρ. We have found that by starting with a stochastic matrix leads to extremely fast convergence that appears to be independent of ρ. Therefore we recommend that the iteration be started with $G_0 = e\boldsymbol{\pi}$, i.e., a matrix with each row equal to $\boldsymbol{\pi}$. The matrix $D[G_k]$ is computed in each iteration using Horner's method. Let N be the truncation index on the sequence $\{D_j\}$. Then Horner's method carries out the matrix operations according to the following scheme

$$Y_0 = D_N, \qquad Y_j = D_{N-j} + Y_{j-1}G_k, \qquad \text{for } 1 \leq j \leq N.$$

Clearly, $Y_N = \sum_{j=0}^{N} D_j G_k^j$. An alternative to evaluating (59) by the iteration in (60) and (61) is to evaluate the polynomial in (59) by Horner's method. The method proposed in [42], for evaluating matrix polynomials, which requires fewer matrix multiplications than Horner's method, can also be used. The possible disadvantage with these approaches is that they compute the sum, $\sum_{n=0}^{k} \gamma_n H_n$, where k is the truncation index of the sequence $\{\gamma_n\}$, whereas (61) evaluates the sum term by term and thus the summation can be stopped when the successive terms get small. In general, all k terms need not be computed.

Note that this algorithm for computing G does not require the numerical evaluation and storage of the matrices A_n, defined in (11). When only the waiting time distribution or the moments of the queue length

are needed, then computation of the matrix G is sufficient.

We also point out that (51) can be used to derive other algorithms for computing G. In general, we prefer (60) and (61) since only the scalar sequence $\{\gamma_n\}$ needs to be computed via numerical integrations, however, in some cases other methods might be preferred. For example, if the service time distribution is a mixture of two deterministic distributions, i.e., the service time equals d_1 with probability p_1 and d_2 with probability p_2, then from (51) we see that G satisfies

$$G = p_1 e^{D[G]d_1} + p_2 e^{D[G]d_2}. \tag{62}$$

In this case, any routine which efficiently computes the matrix exponential can be used for the iteration implied by (62). See [43] for many alternative algorithms for computing matrix exponentials.

In (60) and (61), we have presented a new scheme for computing the matrix G. As a consequence of the Corollary in Section 4, we see that once G is computed, then as far as computing the moments of the queue length and waiting time distributions, we are essentially done. That is, given G, we compute $\boldsymbol{g}$ by standard methods and the moments are given explicitly by (31-33), (42-43) and (47-48). This constitutes a major reduction in the computational effort compared to the classical algorithm of Section 3. The waiting time distribution can also be obtained by inverting the transform in (57) or by solving the equivalent Volterra integral equation.

The equations for the moments of the queue length involve the moment matrices $A^{(i)}(1)$, $i=0,1,2,3$. These can be derived in a manner suitable for computation as follows. Define $V_n(t)$ to be the nth factorial moment matrix of the sequence $\{P(k,t):\ k\geq 0\}$, i.e.,

$$V_n(t) = \sum_{k=n}^{\infty} \frac{k!}{(k-n)!} P(k,t).$$

$$= \frac{d^n}{dz^n} P^*(z,t)\Big|_{z=1}.$$

Differentiating n times with respect to z in (7) and setting $z=1$ leads to

$$V_n'(t) = \sum_{j=0}^{n} \binom{n}{j} V_j(t) D^{(n-j)}. \tag{63}$$

where $D^{(n)} = \dfrac{d^n}{dz^n} D(z)\Big|_{z=1} = \displaystyle\sum_{k=n}^{\infty} \frac{k!}{(k-n)!} D_k$, $n\geq 0$. Now,

$$A^{(i)}(1) = \int_0^\infty V_i(t)\,d\tilde{H}(t), \qquad \text{for } i \geq 0.$$

Writing the summation in (63) as a matrix product, using uniformization as in (59), and writing $A^{(i)}$ for $A^{(i)}(1)$, we compute these simultaneously as the concatenated matrix

$$\left[A, A^{(1)}, A^{(2)}, A^{(3)}\right] = \sum_{n=0}^{\infty} \gamma_n L_n,$$

where $\gamma_n = \int_0^\infty e^{-\theta x} \frac{(\theta x)^n}{n!} d\tilde{H}(x)$, for $n \geq 0$, $\theta = \max_i(-D_{ii})$, L_0 is the $m \times 4m$ matrix $[I,0,0,0]$, and $L_{k+1} = L_k(I + \theta^{-1} S)$, $k \geq 0$, where

$$S = \begin{bmatrix} D & D^{(1)} & D^{(2)} & D^{(3)} \\ 0 & D & 2D^{(1)} & 3D^{(2)} \\ 0 & 0 & D & 3D^{(1)} \\ 0 & 0 & 0 & D \end{bmatrix}.$$

If the queue length distributions are needed then some additional work is required. In particular, the recursion in (49) requires the computation of the matrices A_k and B_k, $k \geq 0$. By repeating the uniformization arguments of , we can write $P(n,t)$, defined in (6) as

$$P(n,t) = \sum_{j=0}^{\infty} e^{-\theta t} \frac{(\theta t)^j}{j!} K_n^{(j)}, \tag{64}$$

where $\theta = \max_i \{(-D_0)_{ii}\}$ and $\{K_n^{(j)}\}$ is defined recursively by $K_0^{(0)} = I$, $K_n^{(0)} = 0$, $n \geq 1$, and

$$K_0^{(j+1)} = K_0^{(j)}(I + \theta^{-1} D_0) \tag{65}$$

$$K_n^{(j+1)} = \theta^{-1} \sum_{i=0}^{n-1} K_i^{(j)} D_{n-i} + K_n^{(j)}(I + \theta^{-1} D_0). \tag{66}$$

Substituting (64) into (11) leads to the following expression for A_n.

$$A_n = \sum_{j=0}^{\infty} \gamma_j K_n^{(j)}. \tag{67}$$

This representation along with the recursion (65-66) leads to an efficient algorithm for computing the A_n's. This algorithm does not require the computation of the matrices $P(n,t)$ and only involves the numerical integration of the scalar quantities γ_j, $j \geq 0$. Of course, for many service time distributions, numerical integrations may be avoided altogether.

Once a sufficient number of A_n matrices are computed, the sequence $\{B_n,\ n \geq 0\}$ is obtained directly from (17). We refer to the various truncation rules in [23] and [20] to determine how many matrices are required. Now the queue length densities can be computed using (49) and (36).

6. SOME SELECTED SPECIAL CASES

In this section we point out some simplifications in the algorithms which arise in queues without batch arrivals or when $m = 2$..

The *MAP/G/*1 Queue

For the *MAP/G/*1 queue, all arrivals are single (i.e., no batch arrivals), so that in this case $D(z) = D_0 + zD_1$. This results in several simplifications in the preceding equations. For instance, $\lambda_1'^{-1} = \pi D_1 e$,

$$D^{(1)}(1) = D_1, \quad \text{and} \quad D^{(n)}(1) = 0, \quad \text{for } n \geq 1,$$

$$B_n = -D_0^{-1} D_1 A_n, \quad \text{for } n \geq 0,$$

$$y_{i+1} = y_i D_1 + \lambda_1'^{-1}(x_i - x_{i+1}) D_0^{-1},$$

etc. The algorithm for computing G for the *MAP/G/*1 queue is given by (60) and (61) where now $D[G_k] = D_0 + D_1 G_k$. Such simplifications should be exploited in an implementation of the algorithms if only arrival streams without batch arrivals are being studied.

The *PH/G/*1 Queue

Since the *PH* renewal process is a special case of the *MAP*, the results for the *MAP/G/*1 queue hold.

In particular, if the interarrival time distribution is of phase type with representation $(\boldsymbol{\alpha}, T)$, then $D(z) = T + z\boldsymbol{T}^o\boldsymbol{\alpha}$, where $\boldsymbol{T}^o = -T\boldsymbol{e}$. Equation (51), for G, is then given by

$$G = \int_0^\infty e^{(T+\boldsymbol{T}^o\boldsymbol{\alpha}G)x} d\tilde{H}(x). \tag{68}$$

We see that if the vector $\boldsymbol{\alpha}G$ is specified, then by substituting this into the right hand side of Equation (68), the matrix G is completely determined. This suggests that we might try to compute $\boldsymbol{\alpha}G$ directly. With this in mind, let $\boldsymbol{u} = \boldsymbol{\alpha}G$. From (68) we then have

$$\boldsymbol{u} = \int_0^\infty \boldsymbol{\alpha} e^{(T+\boldsymbol{T}^o\boldsymbol{u})x} d\tilde{H}(x). \tag{69}$$

Using uniformization, as in (59), we can write $\boldsymbol{u}$ as

$$\boldsymbol{u} = \sum_{n=0}^{\infty} \gamma_n \boldsymbol{\alpha}(I + \theta^{-1}(T + \boldsymbol{T}^0\boldsymbol{u}))^n. \tag{70}$$

Thus, $\boldsymbol{u}$ can be computed by successively iterating on

$$\boldsymbol{u} = \sum_{n=0}^{\infty} \gamma_n \boldsymbol{h}_n \tag{71}$$

and

$$\begin{aligned} \boldsymbol{h}_0 &= \boldsymbol{\alpha} \\ \boldsymbol{h}_{n+1} &= \boldsymbol{h}_n(I + \theta^{-1}(T + \boldsymbol{T}^o\boldsymbol{u})) \end{aligned} \tag{72}$$

Now, for the *PH/G/*1 queue, it can be shown that $\boldsymbol{g} = (\boldsymbol{\alpha}GT^{-1}\boldsymbol{e})^{-1}\boldsymbol{\alpha}GT^{-1}$ (See e.g., [44]). Therefore, if we then solve the linear system $\boldsymbol{v}T = \boldsymbol{u}$, for $\boldsymbol{v}$, $\boldsymbol{g}$ is given by

$$\boldsymbol{g} = \frac{\boldsymbol{v}}{\boldsymbol{v}\boldsymbol{e}}.$$

Once g is computed, the moments of the queue length and waiting time distributions are readily obtained from the results in the preceding sections. If the matrix G is needed, for computing the queue length distribution for instance, it is obtained directly from

$$G = \sum_{n=0}^{\infty} \gamma_n (I + \theta^{-1}(T + \boldsymbol{T}^o \boldsymbol{u}))^n. \tag{73}$$

The implementation of the vector recursion (71-72) results in a substantial computational savings over the matrix recursion (60-61). We also note that this simplified algorithm for the *PH/G/*1 queue was obtained independently in [45].

Using the fact the for the *PH/G/*1 queue, $\boldsymbol{\pi} = -\lambda_1'^{-1} \boldsymbol{\alpha} T^{-1}$, we can reduce the recursion for the stationary probabilities at an arbitrary time, given in (34) and (36), to

$$\boldsymbol{y}_0 = -\lambda_1'^{-1} \boldsymbol{x}_0 T^{-1} \tag{74}$$

$$\boldsymbol{y}_{i+1} = \boldsymbol{x}_i \boldsymbol{e}\boldsymbol{\pi} + \lambda_1'^{-1} [\boldsymbol{x}_i - \boldsymbol{x}_{i+1}] T^{-1}, \quad i \geq 0.$$

The *GI/PH/*1 Queue.

We note that analogous results carry through for the nonlinear equation from the *GI/PH/*1 queue. In particular, if the interarrival time distribution is given by $\tilde{F}(\cdot)$, and the service time distribution is of phase type with representation $(\boldsymbol{\beta}, S)$, then the matrix R which is central to the matrix geometric solution of that model, (see, e.g., [23]), is shown in [40] to satisfy

$$R = \int_0^{\infty} e^{(S + RS^o \boldsymbol{\beta})x} d\tilde{F}(x), \tag{75}$$

where $\boldsymbol{S}^o = -S\boldsymbol{e}$. Using the same reasoning as above we define the vector $\boldsymbol{v} = R\boldsymbol{S}^o$ which satisfies

$$\boldsymbol{v} = \int_0^{\infty} e^{(S + \boldsymbol{v}\boldsymbol{\beta})x} \boldsymbol{S}^o d\tilde{F}(x). \tag{76}$$

An algorithm analogous to (71-72) can now be specified which is an improvement over both the algorithm proposed in [40] and that in [46].

Simplifications Resulting from a 2-State Arrival Process

The major computational effort in the algorithm is in computing the matrix G which satisfies (51). The algorithm presented in (59) is an efficient numerical procedure for reducing the number of numerical integrations and for computing the matrix exponential appearing in (51). In the 2-state case, a substantial savings results since we can write an explicit expression for the matrix exponential appearing in (51). Also, in the 2-state case, the matrix G has only two unknown elements, since it is stochastic. We are thus able to derive 2 scalar recursions for these unknown elements. This will lead to an extremely efficient algorithm for computing G. We present below some particularly useful special cases.

The 2-state *MAP/G/1* Queue

We write the matrices D_0 and D_1 as

$$D_0 = \begin{bmatrix} -(a+b+c) & a \\ d & -(d+e+f) \end{bmatrix}, \quad \text{and} \quad D_1 = \begin{bmatrix} b & c \\ e & f \end{bmatrix},$$

where $(a,b,c,d,e,f) \geq 0$ and without loss of generality, we assume $b \geq f$. Now for a generator S given by

$$S = \begin{bmatrix} -s_0 & s_0 \\ s_1 & -s_1 \end{bmatrix},$$

we have

$$e^{Sx} = \frac{1}{s_0+s_1} \begin{bmatrix} s_1+s_0 e^{-(s_0+s_1)x} & s_0-s_0 e^{-(s_0+s_1)x} \\ s_1-s_1 e^{-(s_0+s_1)x} & s_0+s_1 e^{-(s_0+s_1)x} \end{bmatrix}. \tag{77}$$

If we write G as

$$G = \begin{bmatrix} 1-G_0 & G_0 \\ G_1 & 1-G_1 \end{bmatrix}, \tag{78}$$

then the off-diagonal elements of (51) lead to the equations

$$G_0 = \frac{(a+bG_0+c(1-G_1))[1-H(a+bG_0+c(1-G_1)+d+e(1-G_0)+fG_1)]}{a+bG_0+c(1-G_1)+d+e(1-G_0)+fG_1} \tag{79}$$

$$G_1 = \frac{(d+e(1-G_0)+fG_1)[1-H(a+bG_0+c(1-G_1)+d+e(1-G_0)+fG_1)]}{a+bG_0+c(1-G_1)+d+e(1-G_0)+fG_1}, \tag{80}$$

where $H(\cdot)$ is the *LST* of $\tilde{H}(\cdot)$. Adding these two equations yields

$$G_0 + G_1 = 1 - H(a+bG_0+c(1-G_1)+d+e(1-G_0)+fG_1). \tag{81}$$

The commutativity of G and D_0+D_1G implies the relationship

$$aG_1+bG_0G_1+c(1-G_1)G_1 = dG_0+e(1-G_0)G_0+fG_0G_1. \tag{82}$$

These equations can be written in the following form which is suitable for iteration starting with $G_0=G_1=0$,

$$G_0 = 1 - G_1 - H(a+bG_0+c(1-G_1)+d+e(1-G_0)+fG_1), \tag{83}$$

$$G_1 = \frac{dG_0+e(1-G_0)G_0+cG_1^2}{a+c+(b-f)G_0} \tag{84}$$

This algorithm avoids computation of the sequence $\{\gamma_n\}$ and thus no numerical integrations are required. Evaluation of the matrix polynomial in (59) is also avoided. Our computational experience with this algorithm has shown it to be extremely fast and efficient.

The $E_2/G/1$ queue

The solution to the $E_2/G/1$ queue is particularly simple. The *PH*-representation for E_2 is $(\boldsymbol{\alpha}, T)$ where $\boldsymbol{\alpha} = (1,0)$ and

$$T = \begin{bmatrix} -\lambda & \lambda \\ 0 & -\lambda \end{bmatrix}.$$

Equation (83) implies that

$$G_0 = \frac{1-H(\lambda(2-G_0))}{2-G_0},$$

and using (84), we see that G can be written as

$$G = \begin{bmatrix} x & 1-x \\ x(1-x) & 1-x(1-x) \end{bmatrix},$$

and the vector $\boldsymbol{g} = (x(1+x)^{-1}, (1+x)^{-1})$, where x is the unique solution in (0,1) to

$$x = H(\lambda+\lambda x)^{1/2}. \tag{85}$$

The 2-state *MMPP/G/1* Queue

For the *MMPP/G/1* queue, we have $D_0 = R-\Lambda$, $D_1 = \Lambda$, and $D_j = 0$, $j \geq 2$. Equation (51) for G reduces to

$$G = \int_0^\infty e^{(R-\Lambda+\Lambda G)x} d\tilde{H}(x)$$

Therefore, for the m-state case, $m \geq 3$, the general algorithms for the *MAP/G/1* queue apply.

The 2-state case has received attention recently as a simple tractable process which can closely approximate much more complicated processes and predict queueing delays very accurately. (See, for instance, [10], for an application to the performance of packetized voice and data processes concentrated via a statistical multiplexer.) Equations (83) and (84) reduce to very simple forms in this case. These lead to an extremely simple and efficient algorithm for this model. To this end, we write

$$R = \begin{bmatrix} -r_0 & r_0 \\ r_1 & -r_1 \end{bmatrix}, \quad \Lambda = \begin{bmatrix} \lambda_0 & 0 \\ 0 & \lambda_1 \end{bmatrix}, \quad \text{and} \quad G = \begin{bmatrix} 1-G_0 & G_0 \\ G_1 & 1-G_1 \end{bmatrix},$$

and assume, without loss of generality, $\lambda_0 \geq \lambda_1$. Equation (83) reduces to

$$G_0 + G_1 = 1 - H(r_0+r_1+\lambda_0 G_0 + \lambda_1 G_1), \tag{86}$$

and Equation (84) reduces to

$$G_0(r_1+\lambda_1 G_1) = G_1(r_0+\lambda_0 G_0). \tag{87}$$

Solving for G_1 in this equation implies that G can be written as

$$G = \begin{bmatrix} 1-x & x \\ \dfrac{r_1 x}{r_0+(\lambda_0-\lambda_1)x} & 1-\dfrac{r_1 x}{r_0+(\lambda_0-\lambda_1)x} \end{bmatrix}, \tag{88}$$

where x satisfies

$$x = 1 - \frac{r_1 x}{r_0+(\lambda_0-\lambda_1)x} - H\left[r_0+r_1+\lambda_0 x + \frac{\lambda_1 r_1 x}{r_0+(\lambda_0-\lambda_1)x}\right]. \tag{89}$$

In many cases, the quantity x may be obtained by successive substitution in (89) starting with $x=0$. Our experience with this iteration shows it to be extremely fast and, unlike the analogous matrix iterations, the speed of convergence is *insensitive* to ρ. For instance, an example with $\rho=0.99999$ converged to 10 decimal places of accuracy in just 14 iterations. For some parameter ranges, the successive substitution scheme will oscillate. Whenever oscillation occurred, relabeling the states 0 and 1 so that $\lambda_1 \geq \lambda_0$ has solved this problem in all cases we encountered. In any case, the (unique) solution in (0,1) can be obtained easily by bisection. Although this is slightly slower, it is guaranteed to converge. in particular, if we choose $\lambda_1 \geq \lambda_0$, then it can be shown that

$$0 \leq x \leq \min(1, r_0(r_1+\lambda_1-\lambda_0)^{-1}).$$

The vector $\boldsymbol{g}$ is given by

$$\boldsymbol{g} = (g_0, g_1) = \left[\frac{G_1}{G_0+G_1}, \frac{G_0}{G_0+G_1}\right],$$

and the *LST* of the virtual waiting time, given in (45), may be written explicitly as

$$w_v(s) = \frac{s(1-\rho)[s-r_0-r_1+(H(s)-1)(g_0\lambda_1+g_1\lambda_0)]}{s^2+[(H(s)-1)(\lambda_0+\lambda_1)-(r_0+r_1)]s+(H(s)-1)[(H(s)-1)\lambda_0\lambda_1-r_0\lambda_1-r_1\lambda_0]},$$

although for computations, the matrix expression in (45) is more convenient. Once the vector g is computed, it is now routine to compute the moments of the waiting time and queue length distributions using the formulas in this paper. In particular, the expressions for the moments of the virtual waiting time given in (47) and (48), with y_0 replaced by $(1-\rho)g$, are are readily implemented and are preferred to expressions obtained by differentiating the above explicit equation for $w_v(s)$. The iteration (89) is trivial to program and seems to be the simplest solution to the 2-state *MMPP/G/*1 queue to date. (Compare this to the solution in [13].)

7. CONCLUSIONS

The *BMAP* is a natural generalization of the batch Poisson process and its notation is extremely simple. It is a wide class of arrival processes and contains as special cases many processes that have been studied in the literature. We have presented new results for the *BMAP/G/*1 queue. These results have led to simplified algorithms for computing many performance measures of interest. The new relationship (51) for the matrix G which is a key ingredient in the algorithmic solution, may lead to even further simplified numerical procedures. The algorithms presented here allow for a general implementation of canned computer programs for solving the general model. Such a program could be used for comparing vastly different arrival processes entering a single server queue.

A further use of this algorithm is to evaluate the performance of superpositions of renewal processes entering a queue. If the renewal processes are of phase type then the superposition is a special case of the *BMAP*. Although the size of the matrices involved grows geometrically as the number of streams, for two or three streams the computations are completely feasible. The delay seen by customers in the individual streams can be derived from the results presented earlier. These exact expressions could be used to validate various simple approximations that have been proposed in the literature.

Many of the previous analyses of queues with Neuts' versatile Markovian point process can now be recast in the new framework of the *BMAP* representation. This will lead to simplified expressions and algorithms for these models.

ACKNOWLEDGEMENTS

The simple solution to the $E_2/G/1$ queue was worked out by K. Sohraby, who has permitted its inclusion

here. The author thanks C. Blondia, W. Fischer, H. Heffes, K. Meier-Hellstern, M. Neuts, V. Ramaswami, and W. Whitt for helpful comments on the original manuscript.

REFERENCES

1. Neuts, M. F., A versatile Markovian point process. *J. Appl. Prob.*, **16,** 764-79, 1979.

2. Ramaswami, V., The *N/G/*1 queue and its detailed analysis. *Adv. Appl. Prob.*, **12,** 222-61, 1980.

3. Ramaswami, V. The $N/G/\infty$ Queue, Dept. of Math., Tech. Rept., Drexel University, Philadelphia, PA., Oct. 1978.

4. Neuts, M. F., The c –server queue with constant service times and a versatile Markovian arrival process. In *Applied Probability-Computer Science: The Interface,* Proc. Conf. Boca Raton, Florida, January 1981, R. L. Disney and T. J. Ott (eds), I, 31-70, 1982.

5. Blondia, C., The *N/G/*1 finite capacity queue. *Stoch. Models,* 5, No. 2, 1989.

6. Lucantoni, D. M., and Neuts, M. F., Numerical methods for a class of Markov chains arising in queueing theory, Tech. Rept. no. 78/10. Appl. Math. Inst., Univ. of Delaware, Newark, 1978.

7. Lucantoni, D. M., Meier-Hellstern, K. S., and Neuts, M. F., A single server queue with server vacations and a class of non-renewal arrival processes, to appear in *Adv. Appl. Prob.*, Sept., 1989.

8. Kuczura, A., The interrupted Poisson process as an overflow process, *Bell. Syst. Tech. J.*, **52,** 437-48, 1973.

9. Sriram, K., and Whitt, W., Characterizing superposition arrival processes in packet multiplexers for voice and data, *IEEE J. on Selected Areas in Communication, Special Issue on Network Performance Evaluation,* SAC-4, 6, 833-846, 1986.

10. Heffes, H. and Lucantoni, D. M., A Markov modulated characterization of packetized voice and data traffic and related statistical multiplexer performance. *IEEE J. on Selected Areas in Communication, Special Issue on Network Performance Evaluation,* SAC-4, 6, 856-868, 1986.

11. Heffes, H. A class of data traffic processes - Covariance function characterization and related queueing results, *Bell Syst. Tech. J.,* vol. 59, 897-929, July/Aug. 1980.

12. van Hoorn, M. H., and Seelen, L. P., The *SPP/G/*1 queue: a single server queue with a switched Poisson process as input process. *O. R. Spektrum,* **5,** 207-218, 1983.

13. Rossiter, M., The switched Poisson process and the *SPP/G/*1 queue. *Proceedings of the 12th International Teletraffic Congress,* Torino, 1988.

14. Ide, I., Superposition of interrupted Poisson processes and its application to packetized voice multiplexers, *Proceedings of the 12th International Teletraffic Congress,* Torino, 1988.

15. Neuts, M. F., Phase-type distributions: A bibliography, Department of Systems and Industrial Engineering, University of Arizona, Working Paper Nr. 89-005, February 1989.

16. Neuts, M. F., The fundamental period of the queue with Markov-modulated arrivals, *Probability, Statistics and Mathematics: Papers in Honor of Samuel Karlin,* Academic Press, 187-200, 1989.

17. Meier-Hellstern, K. S., The analysis of a queue arising in overflow models, *IEEE Trans. on Comm.,* vol. 37, no.4, 1989.

18. Holtzman, J., M., Mean delays of individual streams into a queue: The $\sum GI_i/M/1$ queue, In *Applied Probability-Computer Science: The Interface,* Proc. Conf. Boca Raton, Florida, January 1981, R. L. Disney and T. J. Ott (eds), I, 31-70, 1982.

19. Whitt, W., The queueing network analyzer, *Bell Syst. Tech. J.,* Part 1, vol. 62, no. 9, 2779-2815, Nov. 1983.

20. Neuts, M. F., *Structured stochastic matrices of M/G/1 type and their applications.* Marcel Dekker, 1989,

21. Neuts, M. F., Probability distributions of phase type. In *Liber Amicorum Prof. Emeritus H. Florin,* Department of Mathematics. Belgium: University of Louvain, 173-206, 1975.

22. Neuts, M. F., Renewal processes of phase type, *Nav. Res. Logist. Quart.,* **25,** 445-54, 1978.

23. Neuts, M. F., *Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach.* Baltimore: The Johns Hopkins University Press, 1981.

24. Latouche, G., A phase-type semi-Markov point process. *SIAM J. Alg. Disc. Meth.,* **3** , No. 1, 77-90, March 1982.

25. Neuts, M. F. and Latouche, G. The superposition of two *PH* –renewal processes. In *"Semi-Markov Models: Theory and Applications"*, J. Janssen, ed., London: Plenum Publishers, 131-77, 1986.

26. Lucantoni, D. M., *An Algorithmic Analysis of a Communication Model with Retransmission of Flawed Messages.* London: Pitman, 1983.

27. Neuts, M. F., Moment formulas for the Markov renewal branching process. *Adv. Appl. Prob.*, **8,** 690-711, 1976.

28. Takács, L., *Introduction to the Theory of Queues,* New York: Oxford University Press, 1962.

29. Çinlar, E., Markov Renewal Theory, *Adv. Appl. Prob.,* **1** , 123-87, 1969.

30. Hunter, J. J., On the moments of Markov renewal processes. *Adv. Appl. Prob.* **1** , 188-210, 1969.

31. Ortega, J. M., and Rheinboldt, W. C., *Iterative solution of nonlinear equations in several variables.* Academic Press, Inc., 1970.

32. Ramaswami, V., Nonlinear matrix equations in applied probability - solution techniques and open problems, *SIAM Review,* 30, 1988.

33. Forsythe, G. E., Malcolm, M. A., and Moler, C. B., *Computer methods for mathematical computations,* Prentice-Hall, Inc., 1977.

34. Ramaswami, V., Stable recursion for the steady state vector for Markov chains of *M/G/1* type. *Stochastic Models,* **4** 183-88, 1988.

35. Platzman, L. K., Ammons, J. C., and Bartholdi, J. J., III, A simple algorithm to compute tail probabilities from transforms, *Operations Research,* Vol. 36, No. 1, Jan.-Feb., 1988.

36. Jagerman, D., An inversion technique for the Laplace transform, *Bell Syst. Tech. J.,* vol. 61, no.8. 1995-2002, 1982.

37. Neuts, M. F., Generalizations of the Pollaczek-Khinchin integral equation in the theory of queues. *Adv. Appl. Prob.,* **18,** 952-90, 1986.

38. Churchhouse, R. F. (Editor), *Handbook of Applicable Mathematics. Vol. 3: Numerical methods.* John Wiley and Sons Ltd., 1981.

39. Davis, P. J., and Rabinowitz, P., *Methods of numerical integration.* Second edition, Academic Press, 1984.

40. Sengupta, B., Markov processes whose steady state distribution is matrix-exponential with an application to the *GI/PH/1* queue. *Adv. Appl. Prob.,* **21,** No. 1, 159-180, March 1989.

41. Ramaswami, V., From the matrix-geometric to the matrix-exponential, to appear in *Queueing Systems,* 1990.

42. Van Loan, C., A note on the evaluation of matrix polynomials, *IEEE Trans, on Aut. Con.* vol. AC-24, No. 2, 320-21, 1979.

43. Moler, C. B., and Van Loan, C., Nineteen dubious ways to compute the matrix exponential, *SIAM Rev.* vol. 20, 801-836, 1978.

44. Neuts, M. F., A new informative embedded Markov renewal process for the *PH/G/*1 queue. *Adv. Appl. Prob.,* 18, 535-557, 1986.

45. Asmussen, S. Matrix representation of ladder height distributions, submitted for publication.

46. Lucantoni, D. M. and Ramaswami, V., Efficient algorithms for solving the non-linear matrix equations arising in phase type queues. *Stochastic Models,* **1** , 29-51, 1985.