Queueing Syst (2006) 53:127–145
DOI 10.1007/s11134-006-8306-0

A matrix-analytic solution for the *DBMAP/PH/*1 priority queue*

Ji-An Zhao · Bo Li · Xi-Ren Cao · Ishfaq Ahmad

Received: 18 September 2002 / Revised: 2 November 2005

Abstract Priority queueing models have been commonly used in telecommunication systems. The development of analytically tractable models to determine their performance is vitally important. The discrete time batch Markovian arrival process (*DBMAP*) has been widely used to model the source behavior of data traffic, while phase-type (*PH*) distribution has been extensively applied to model the service time. This paper focuses on the computation of the *DBMAP/PH/*1 queueing system with priorities, in which the arrival process is considered to be a *DBMAP* with two priority levels and the service time obeys a discrete *PH* distribution. Such a queueing model has potential in performance evaluation of computer networks such as video transmission over wireless networks and priority scheduling in ATM or TDMA networks. Based on matrix-analytic methods, we develop computation algorithms for obtaining the stationary distribution of the system numbers and further deriving the key performance indices of the *DBMAP/PH/*1 priority queue.

Keywords Markovian arrival process · Phase-type distribution · $M/G/1$-type Markov chain · Matrix-analytic methods

AMS subject classifications: 60K25 · 90B22 · 68M20

*The work was supported in part by grants from RGC under the contracts HKUST6104/04E, HKUST6275/04E and HKUST6165/05E, a grant from NSFC/RGC under the contract N_HKUST605/02, a grant from NSF China under the contract 60429202.

Ji-An Zhao · Bo Li
Department of Computer Science
e-mail: zhaojian@ieee.org
e-mail: bli@cs.ust.hk

Xi-Ren Cao (✉)
Department of Electronic Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong, China
e-mail: eecao@ee.ust.hk

Ishfaq Ahmad
Computer Science and Engineering Department, The University of Texas at Arlington, Arlington, TX 76019-0015, USA
e-mail: iahmad@cse.uta.edu

1. Introduction

The Markovian arrival process (*MAP*) [9] and the discrete time Markovian arrival process (*DMAP*) were introduced to model source behavior with correlations, which is a distinctive feature of data traffic. *MAP* and *DMAP* were extended to the batch arrival case, called the batch Markovian arrival process (*BMAP*) [10, 11] and the discrete-time batch Markovian arrival process (*DBMAP*) [3], respectively. *BMAP* and *DBMAP* were introduced to capture the bursty nature of traffic source behavior. They were thus well suited to model multimedia data [2]. There have been extensive studies on queueing systems with *BMAP* or *DBMAP* input. For instance, the *BMAP/G/*1 single server queue was studied in [10], and the *DBMAP/G/*1/*N* single server queue was investigated in [3]. Phase-type (*PH*) distributions were shown to approximate many general distributions, particularly in modeling the service time [13]. A *DBMAP/PH/*1 queue with different service disciplines was studied in [5]. However none of these models consider priority in their queueing systems.

Priority queueing models are commonly used in telecommunications systems. The development of analytically tractable models to determine the performance of such models is essential. Traditional $M/G/1$ priority queues with Poisson arrivals were studied in [16], in which average performance measures were obtained. Queue length distribution

for continuous-time $M/M/1$ priority queues were obtained in [12]. The *DMAP/PH/*1 priority queue was especially studied in [1], in which the arrival process was modeled as a single arrival *MAP* with two priority levels, and the service process was modeled to follow discrete time *PH*-type distribution. Based on the quasi-birth-death (*QBD*) type of transition probability matrix of the underlying Markov chain, a matrix-geometric solution was derived. Furthermore, based on the upper triangular structure of the R measure (defined below in Section 4.1), computation algorithms were developed for the queue length and waiting time distributions. This paper considers the computation of a queueing system consisting of a *DBMAP* arrival process with two priority levels and a single server with a *PH*-type service distribution, which we call the *DBMAP/PH/*1 priority queue. Our work builds upon the existing *DMAP/PH/*1 queueing model with priority support by taking into account the fact that batch arrival accurately captures the bursty and correlation nature of data traffic generated in computer communication networks.

The main contributions of this paper are summarized as follows. We extend the *DBMAP* definition [3] to support different types of arrivals. The extension is based on Markov chains with marked transitions [8]. We derive the transition probability matrices for the underlying Markov chains of the *DBMAP/PH/*1 priority queue under both non-preemptive and preemptive disciplines. We show that the related Markov chains are $M/G/1$-type [14]. The transition probability matrix for the Markov chain has a block structure with an infinite number of blocks and each block element has infinitely large dimensions. It is known that there is no general solution for this type of Markov chain, where the block elements of the transition probability matrix have infinite dimensions. The matrix-analytic methods [6] can handle the case only when the block elements are of finite size. In this paper, we develop an analytical solution for the problem. We firstly study the relationships among the R measures [19] for the related Markov chains. We then prove that the $R_{0,k}$ and Φ_0 measures are in upper-Hessenberg form and have row-repeating structures. We proceed to show that the R_k measures have upper-triangular form structures with row-repeating properties. Based on these special structures and the relationships among R_k, R_{0k} and Φ_0, we develop algorithms to compute these measures and present detailed computation steps. Subsequently, we solve the underlying Markov chains of the *DBMAP/PH/*1 priority queues by adopting matrix-analytic methods, and we obtain the stationary queue length distributions for both the high and low priority queues. We also analyze the time and memory complexity of the computation. Finally, we offer numerical results and discuss possible applications of the *DBMAP/PH/*1 priority queues in modeling computer and communication networks.

In what follows, we introduce the *DBMAP* process in Section 2.1 and the *PH*-type distribution in Section 2.2. In Section 3, we discuss the basic model of the *DBMAP/PH/*1 queueing system under both non-preemptive and preemptive disciplines. We then describe the computation algorithms that solve the *DBMAP/PH/*1 priority queue in Section 4. Finally, we provide numerical examples in Section 5 and present concluding remarks in Section 6.

2. The *DBMAP* arrival process and *PH* distribution

2.1. The *DBMAP* arrival process

In this subsection, we give a brief introduction to the discrete-time batch Markovian arrival process (*DBMAP*) [3]. Consider a sequence of $n \times n$ substochastic matrices, D_k, $k = 0, 1, 2 \ldots \infty$, and an n state discrete-time Markov chain with transition probability matrix D, which is the sum of the D_k elements, i.e., $D = \sum_{k=0}^{\infty} D_k$. Suppose that, at time t, $t \geq 0$, the Markov chain is in state j, $1 \leq j \leq n$; then, at time epoch $t+1$, with conditional probability $D_k(j, j')$, $k \geq 0$, the arrival process enters state j', $1 \leq j' \leq n$, and triggers a batch of k arrivals. Therefore, the D_0 matrix corresponds to state transitions with no arrival, and the D_k matrix with $k \geq 1$, corresponds to state transitions with a batch arrival of size k. In other words, when the state of the arrival process changes from j to j', the batch size distribution of the triggered arrival is given by

$$[D_0(j, j'),\ D_1(j, j'),\ D_2(j, j'),\ D_3(j, j'), \ldots].$$

Obviously, we have

$$\sum_{k=0}^{\infty} \sum_{j'=1}^{n} D_k(j, j') = 1, \quad \text{for all } 1 \leq j \leq n,$$

which is equivalent to

$$\sum_{k=0}^{\infty} D_k e = De = e,$$

where e denotes a column vector with all elements being 1[1]. Assume the process is in stationary state, and let α be the steady state probability vector for D, we have

$$\begin{cases} \alpha D = \alpha, \\ \alpha e \ = 1. \end{cases}$$

The arrival rate λ of a stationary *DBMAP* process is given by

$$\lambda = \alpha(\sum_{k=0}^{\infty} k D_k)e.$$

[1] In this paper, we always let e denote a column vector with all 1s, with proper dimensions according to the context.

The probability of having an arrival, regardless of the batch size, is given by $\alpha \sum_{k=1}^{\infty} D_k e$.

Let $N(t)$ be the number of arrivals in the duration $(0, t]$, let the random variable $J(t)$ be the state of the underlying Markov chain immediately after time t, the bivariate sequence $(N(t), J(t)), t \geq 0$ is a two dimensional discrete time Markov chain on the state space $\{(i, j), i \geq 0, 1 \leq j \leq n\}$. The states can be listed in the form of $(level, phase)$ pairs as follows:

$$\begin{array}{ccccc} (0,0), & (0,1), & (0,2), & \cdots & (0,n), \\ (1,0), & (1,1), & (1,2), & \cdots & (1,n), \\ (2,0), & (2,1), & (2,2), & \cdots & (2,n), \\ \vdots & \vdots & \vdots & \cdots & \vdots \ . \end{array}$$

The transition probability matrix for the above Markov chain is given by

$$\begin{bmatrix} D_0 & D_1 & D_2 & D_3 & \cdots \\ 0 & D_0 & D_1 & D_2 & \cdots \\ 0 & 0 & D_0 & D_1 & \cdots \\ \vdots & \vdots & \ddots & \ddots & \ddots \end{bmatrix}. \tag{1}$$

The above two-dimensional Markov chain (1) completely describes the corresponding *DBMAP* process. The transition from state (i, j) to state $(i + k, j')$, where $k \geq 1$, $1 \leq j \leq n$, $1 \leq j' \leq n$, corresponds to an arrival with batch size k.

It has been shown that many traditional arrival processes, such as the Bernoulli arrival process, the discrete-time Markov modulated Bernoulli process, and the batch Bernoulli process with correlated batch arrivals, etc., are all special cases of the *DBMAP* process [3].

2.2. The phase-type distribution

Consider an $m + 1$ state Markov chain with initial probability vector (β, β_{m+1}) and transition probability matrix

$$P = \begin{bmatrix} S & S^0 \\ 0 & 1 \end{bmatrix},$$

where β is a row vector with m elements, (β, β_{m+1}) is a row probability vector with $m + 1$ elements, $S^0 = e - Se$, S is substochastic and $I - S$ is nonsingular[2]. A discrete phase-type (*PH*) distribution is defined as the time until absorption of state $m + 1$ in the finite discrete Markov chain defined by the above stochastic matrix P. The probability density function p_k of the *PH*-type distribution is given by

$$p_0 = \beta_{m+1}, \ \ p_k = \beta S^{k-1} S^0, \ \ k \geq 1,$$

and the mean equals to $\beta(I - S)^{-1}e$. The pair (β, S) is called a *representation* of the *PH*-type distribution [13] and m is called the *dimension* of the *PH*-type distribution in this paper.

3. The *DBMAP/PH/1* priority queueing model

In this section, we establish the *DBMAP/PH/1* priority queueing model for both the non-preemptive and preemptive disciplines. We firstly define the *DBMAP* arrival process with priorities in Section 3.1, and then introduce the service disciplines in Section 3.2. We formulate the *non-preemptive DBMAP/PH/1* priority queue in Section 3.3, and the *preemptive DBMAP/PH/1* priority queue in Section 3.4.

3.1. The *DBMAP* arrival process with priorities

The *DBMAP* process introduced in Section 2.1 describes only a single class or a single priority of arrivals. To extend the *DBMAP* process to distinguish more than one class or more than one priority of arrivals, we take a similar approach as the Markovian chain with marked transitions model in [7, 8] and extend the *one-dimensional* index for the D_k matrices in Section 2.1 to a *two-dimensional* index below.

We consider an n state discrete-time Markovian arrival process with two types of batch arrivals, one for high priority jobs and the other for low priority jobs. The maximum batch sizes are given by b_1 and b_2 for high priority and low priority arrivals, respectively. The corresponding parameter matrices for the *DBMAP* are given by $\{D_{00}, D_{01}, \ldots\ D_{i_1 i_2}, \ldots\ D_{b_1 b_2}\}$. Each $D_{i_1 i_2}$ is a substochastic matrix with dimension $n \times n$. Suppose that at time $t \geq 0$, the underlying Markov chain of the *DBMAP* process is in state j, $1 \leq j \leq n$; then, at the next time epoch, $t + 1$, with conditional probability $D_{i_1 i_2}(j, j')$, where $0 \leq i_1 \leq b_1, 0 \leq i_2 \leq b_2$, the process changes to state j', $1 \leq j' \leq n$, with a simultaneous batch arrival of i_1 high priority jobs and i_2 low priority jobs. In other words, when the state of the arrival process changes from j to j', the two-dimensional batch size distribution of the triggered arrival is given as follows:

$$[D_{00}(j, j'),\ \ D_{01}(j, j'), \ldots D_{i_1 i_2}(j, j'), \ldots D_{b_1 b_2}(j, j')].$$

The transition probability matrix of the underlying Markov chain is given by $D = \sum_{i_1=0}^{b_1} \sum_{i_2=0}^{b_2} D_{i_1 i_2}$.

Assume that the arrival process is in a stationary state and the initial probability vector of the arrival process is given by

[2] In this paper we let I denote the identity matrix with a proper dimension.

α. We have

$$\begin{cases} \alpha D = \alpha, \\ \alpha e = 1. \end{cases}$$

The arrival rate for the high priority jobs is given by

$$\lambda_h = \alpha(\sum_{i_1=0}^{b_1}\sum_{i_2=0}^{b_2} i_1 D_{i_1 i_2})e;$$

the arrival rate for the low priority jobs is given by

$$\lambda_l = \alpha(\sum_{i_1=0}^{b_1}\sum_{i_2=0}^{b_2} i_2 D_{i_1 i_2})e;$$

and the total arrival rate is given by

$$\lambda = \lambda_h + \lambda_l.$$

The aggregated arrival process that consists of all the high and low priority jobs is a normal *DBMAP* with parameters $\{D_0, D_1, D_2, \ldots D_{b_1+b_2}\}$, where $D_0 = D_{00}$, $D_1 = D_{01} + D_{10}$, $D_2 = D_{02} + D_{11} + D_{20}$, $D_3 = D_{03} + D_{12} + D_{21} + D_{30}$, $\ldots D_{b_1+b_2} = D_{b_1 b_2}$. The arrival rate for this *DBMAP* process is given by $\lambda = \alpha(\sum_{i=1}^{b_1+b_2} i D_i)e$.

3.2. Service disciplines

We consider a queueing system described as follows.

- The arrival process follows the *DBMAP* process with two priority levels, defined by $\{D_{00}, D_{01}, D_{02}, \ldots, D_{b_1 b_2}\}$. All $D_{i_1 i_2}$ matrices are of dimension $n \times n$.
- The service processes for both the high and low priority jobs follow discrete *PH*-type distributions with (β_1, S_1) indicating the high priority jobs and (β_2, S_2) indicating the low priority jobs. β_1 is a row vector with m_1 elements and S_1 is an $m_1 \times m_1$ substochastic matrix; β_2 is row vector with m_2 elements and S_2 is an $m_2 \times m_2$ substochastic matrix.
- There is a *single* priority server in the system and the service discipline can be *preemptive* or *non-preemptive* by nature. In the preemptive case, we consider only the preemptive resume discipline. That is, when the server returns to serve the preempted low priority job, processing will start from the service phase when the job was preempted. Recording the exact preempting phase leads to a relatively large state space. Thus, we take the same approach as in [1] and let β_2^* be the service phase when the preempted low priority job resumes service, where β_2^* satisfies

$$\begin{cases} \beta_2^* e = 1, \\ \beta_2^* = \beta_2^*(S_2 + S_2^0 \beta_2). \end{cases}$$

Here β_2^* can be interpreted as the limiting probability vector of the phase from which the low priority job resumes service. We expect that replacing the exact service preemption phase with β_2^* would not bring a major impact on the performance measures. In the non-preemptive case, there is no need to consider the β_1^* problem because once the low priority job is serviced, there is no interruption until the job is completed.

The above queueing model is called a *DBMAP/PH/*1 priority queue with two levels of priority. It differs from the single arrival model [1] in two ways. Firstly, since the arrival process of the *DBMAP/PH/*1 priority queue allows batch arrivals, the model has much broader application in practice. An wireless multimedia example is given in Section 5.2. Secondly, the solution is also different from that of the single arrival model. For the *DMAP/PH/*1 priority queue, direct computation of the R measure is possible, since R is the solution of a matrix quadratic equation. However, for the *DBMAP/PH/*1 priority queue, direct computation of the R measures can not be done, since they are related by a system of matrix equations, as shown in Section 4.2. We next derive the transition probability matrices under both preemptive and non-preemptive service disciplines.

3.3. The non-preemptive *DBMAP/PH/*1 priority queue

For the non-preemptive *DBMAP/PH/*1 priority queue, we define state space Δ to be $\{(n_1, n_2, i, s, p)\}$[3]. State (n_1, n_2, i, s, p) in Δ has the following conditions.

- The arrival process is in state s, $s = 1, 2, \ldots, n$.
- A high priority job is being served if $i = 1$; or a low priority job is being served if $i = 2$. Here i is an index variable.
- The service process is in phase p, depending on which type of job is in service, as indicated by i. When $i = 1$, $p = 1, 2, \ldots, m_1$; when $i = 2$, $p = 1, 2, \ldots, m_2$.
- There are n_1 high priority jobs in the system, including the one being served, if any.
- In the case of $n_1 = 0$ and $n_2 = 0$, state (n_1, n_2, i, s, p) can be reduced to $(0, 0, s)$, since the whole system is idle and there is no need to record the service phase. In this case the number of high and low priority jobs in the system is zero.

  In the case of $n_1 = 0$ and $n_2 \geq 1$, state (n_1, n_2, i, s, p) can be reduced to $(0, n_2, s, p)$, since only a low priority job can be in service. In this case, n_2 denotes the number of low priority jobs in the system.

  In the case of $n_1 \geq 1$, the server may be busy with a high or low priority job, due to the non-preemptive service discipline. In this case, n_2 denotes the number of low priority

[3] We refer to n_1 and n_2 as *the system numbers* in this paper.

jobs waiting in the queue. Thus if the server is busy serving a low priority job, the total number of low priority jobs in the system is $n_2 + 1$, otherwise the total number of low priority jobs in the system is n_2.

The system can be described by an $M/G/1$-type Markov chain with state space Δ. The transition probability matrix for the non-preemptive case, P_{npm} (the subscript *npm* stands for non-preemptive), is given by

$$P_{npm} = \begin{bmatrix} B_{00} & B_{01} & B_{02} & \cdots & B_{0b_1} & & & & \\ B_{10} & A_0 & A_1 & \cdots & A_{b_1-1} & A_{b_1} & & & \\ & A_{-1} & A_0 & \cdots & A_{b_1-2} & A_{b_1-1} & A_{b_1} & & \\ & & A_{-1} & \cdots & A_{b_1-3} & A_{b_1-2} & A_{b_1-1} & A_{b_1} & \\ & & & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}, \tag{2}$$

with the following sub-matrices[4]:

$$B_{00} = \begin{bmatrix} B_{00}^{00} & B_{00}^{01} & B_{00}^{02} & \cdots & B_{00}^{0b_2} & & & \\ B_{00}^{10} & B_{00}^{0} & B_{00}^{1} & \cdots & B_{00}^{b_2-1} & B_{00}^{b_2} & & \\ & B_{00}^{-1} & B_{00}^{0} & \cdots & B_{00}^{b_2-2} & B_{00}^{b_2-1} & B_{00}^{b_2} & \\ & & B_{00}^{-1} & \cdots & B_{00}^{b_2-3} & B_{00}^{b_2-2} & B_{00}^{b_2-1} & B_{00}^{b_2} \\ & & & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix},$$

where

$$\begin{cases} B_{00}^{00} = D_{00}, \\ B_{00}^{0i_2} = D_{0i_2} \otimes \beta_2, \ i_2 = 1, 2, \ldots, b_2, \\ B_{00}^{10} = D_{00} \otimes S_2^0, \\ B_{00}^{i_2} = D_{0i_2} \otimes S_2 + D_{0(i_2+1)} \otimes S_2^0 \beta_2, \ i_2 = 0, 1, \ldots, b_2 - 1, \\ B_{00}^{b_2} = D_{0b_2} \otimes S_2, \\ B_{00}^{-1} = D_{00} \otimes S_2^0 \beta_2, \end{cases}$$

and

$$B_{0i_1} = \begin{bmatrix} B_{0i_1}^{00} & B_{0i_1}^{01} & B_{0i_1}^{02} & \cdots & B_{0i_1}^{0b_2} & \\ B_{0i_1}^{-1} & B_{0i_1}^{0} & B_{0i_1}^{1} & \cdots & B_{0i_1}^{b_2-1} & \\ & B_{0i_1}^{-1} & B_{0i_1}^{0} & \cdots & B_{0i_1}^{b_2-2} & B_{01}^{b_2-1} \\ & & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix},$$

$i_1 = 1, 2, \ldots, b_1,$

where

$$\begin{cases} B_{0i_1}^{0i_2} = \begin{bmatrix} D_{i_1 i_2} \otimes \beta_1 & 0 \end{bmatrix}, \ i_2 = 0, 1, \ldots, b_2, \\ B_{0i_1}^{-1} = \begin{bmatrix} D_{i_1 0} \otimes S_2^0 \beta_1 & D_{i_1 0} \otimes S_2 \end{bmatrix}, \\ B_{0i_1}^{i_2} = \begin{bmatrix} D_{i_1(i_2+1)} \otimes S_2^0 \beta_1 & D_{i_1(i_2+1)} \otimes S_2 \end{bmatrix}, \\ \qquad i_2 = 0, 1, \ldots, b_2 - 1, \end{cases}$$

and

$$B_{10} = \begin{bmatrix} B_{10}^{00} & B_{10}^{1} & B_{10}^{2} & \cdots & B_{10}^{b_2} & & \\ & B_{10}^{0} & B_{10}^{1} & \cdots & B_{10}^{b_2-1} & B_{10}^{b_2} & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix},$$

where

$$\begin{cases} B_{10}^{00} = \begin{bmatrix} D_{00} \otimes S_1^0 \\ 0 \end{bmatrix}, \\ B_{10}^{i_2} = \begin{bmatrix} D_{0i_2} \otimes S_1^0 \beta_2 \\ 0 \end{bmatrix}, \ i_2 = 0, 1, \ldots, b_2, \end{cases}$$

and

$$A_{i_1} = \begin{bmatrix} A_{i_1}^0 & A_{i_1}^1 & A_{i_1}^2 & \cdots & A_{i_1}^{b_2} & \\ & A_{i_1}^0 & A_{i_1}^1 & \cdots & A_{i_1}^{b_2-1} & A_{i_1}^{b_2} \\ & & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix},$$

$i_1 = 0, 1, \ldots, b_1,$

where

$$\begin{cases} A_{i_1}^{i_2} = \begin{bmatrix} D_{(i_1+1)i_2} \otimes S_1^0 \beta_1 + D_{i_1 i_2} \otimes S_1 & 0 \\ D_{i_1 i_2} \otimes S_2^0 \beta_1 & D_{i_1 i_2} \otimes S_2 \end{bmatrix}, \\ \qquad \text{for } i_1 = 0, 1, \ldots, b_1 - 1, \ i_2 = 0, 1, \ldots, b_2, \\ A_{b_1}^{i_2} = \begin{bmatrix} D_{b_1 i_2} \otimes S_1 & 0 \\ D_{b_1 i_2} \otimes S_2^0 \beta_1 & D_{b_1 i_2} \otimes S_2 \end{bmatrix}, \ i_2 = 0, 1, \ldots, b_2, \end{cases}$$

and

$$A_{-1} = \begin{bmatrix} A_{-1}^0 & A_{-1}^1 & A_{-1}^2 & \cdots & A_{-1}^{b_2} & \\ & A_{-1}^0 & A_{-1}^1 & \cdots & A_{-1}^{b_2-1} & A_{-1}^{b_2} \\ & & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix},$$

where

$$A_{-1}^{i_2} = \begin{bmatrix} D_{0i_2} \otimes S_1^0 \beta_1 & 0 \\ 0 & 0 \end{bmatrix}, \ i_2 = 0, 1, \ldots, b_2.$$

Some elements of the blocks in (2) are briefly explained here. Other elements can be understood in a similar way.

[4] In this paper, the superscript $^{-1}$ denotes an index, while the bold superscript $^{\mathbf{-1}}$ denotes the inverse of a matrix. The symbol $\otimes$ denotes Kronecker product operation.

- $B_{00}^{0i_2}$: This represents the probability of the following event. The original state for both the high and low priority queue is 0. A batch arrival happens (with probability D_{0i_2}), which brings 0 high priority job and i_2 low priority jobs into the system. Thus, the state of the low priority queue is changed to i_2 but the high priority queue remains 0, and the server is initialized (by the initial service vector β_2) to serve a newly arrived low priority job.
- $B_{00}^{i_2}$: This represents the probability of the following two events. In the first event, a batch arrival brings 0 high priority job and i_2 low priority jobs, and the server remains busy with a low priority job. The probability of this event is $D_{0i_2} \otimes S_2$. In the second event, a batch arrival brings 0 high priority job and $i_2 + 1$ low priority jobs. The server completes the low priority job during the current time slot and becomes ready to serve the next low priority job. The probability of this event is $D_{0(i_2+1)} \otimes S_2^0 \beta_2$.
- $A_{i_1}^{i_2}$: This can be understood as the transition probability matrix governing switches in the service between the high and low priority jobs, as follows:

$$\begin{bmatrix} \textit{Prob.}\{\text{Continue with high}\} & \textit{Prob.}\{\text{Switch from high to low}\} \\ \textit{Prob.}\{\text{Switch from low to high}\} & \textit{Prob.}\{\text{Continue with low}\} \end{bmatrix}.$$

3.4. The preemptive *DBMAP/PH/*1 priority queue

For the preemptive *DBMAP/PH/*1 priority queue, we define the state space Δ to be $\{(n_1, n_2, s, p)\}$, where n_1 is the number of high priority jobs in the system; and n_2 is the number of low priority jobs in the system; s denotes the arrival process state, and $s = 1, 2, \ldots, n$; p is the current service phase, depending on which type of job is in service, i.e., when $n_1 = 0$ and $n_2 \geq 1$, p is for the low priority job service phase and $p = 1, 2, \ldots, m_2$; when $n_1 \geq 1$, p is for high priority service phase and $p = 1, 2, \ldots, m_1$. Further, when $n_1 = 0$ and $n_2 = 0$, state (n_1, n_2, s, p) can be reduced to $(0, 0, s)$.

Similarly, the system can be described by an $M/G/1$-type Markov chain with state space Δ. The transition probability matrix for the preemptive case, P_{pm} (the subscript *pm* stands for preemptive), is given by

$$P_{pm} = \begin{bmatrix} B_{00} & B_{01} & B_{02} & \cdots & B_{0b_1} & & & \\ B_{10} & A_0 & A_1 & \cdots & A_{b_1-1} & A_{b_1} & & \\ & A_{-1} & A_0 & \cdots & A_{b_1-2} & A_{b_1-1} & A_{b_1} & \\ & & A_{-1} & \cdots & A_{b_1-3} & A_{b_1-2} & A_{b_1-1} & A_{b_1} \\ & & & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}, \tag{3}$$

with the following sub-matrices:

$$B_{00} = \begin{bmatrix} B_{00}^{00} & B_{00}^{01} & B_{00}^{02} & \cdots & B_{00}^{0b_2} & & & \\ B_{00}^{10} & B_{00}^{0} & B_{00}^{1} & \cdots & B_{00}^{b_2-1} & B_{00}^{b_2} & & \\ & B_{00}^{-1} & B_{00}^{0} & \cdots & B_{00}^{b_2-2} & B_{00}^{b_2-1} & B_{00}^{b_2} & \\ & & B_{00}^{-1} & \cdots & B_{00}^{b_2-3} & B_{00}^{b_2-2} & B_{00}^{b_2-1} & B_{00}^{b_2} \\ & & & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix},$$

where

$$\begin{cases} B_{00}^{00} = D_{00}, \\ B_{00}^{0i_2} = D_{0i_2} \otimes \beta_2, \ i_2 = 1, 2, \ldots, b_2, \\ B_{00}^{10} = D_{00} \otimes S_2^0, \\ B_{00}^{i_2} = D_{0i_2} \otimes S_2 + D_{0(i_2+1)} \otimes S_2^0 \beta_2, i_2 = 0, 1, \ldots, b_2 - 1, \\ B_{00}^{b_2} = D_{0b_2} \otimes S_2, \\ B_{00}^{-1} = D_{00} \otimes S_2^0 \beta_2, \end{cases}$$

and

$$B_{0i_1} = \begin{bmatrix} B_{0i_1}^{00} & B_{0i_1}^{01} & B_{0i_1}^{02} & \cdots & B_{0i_1}^{0b_2} & & \\ B_{0i_1}^{-1} & B_{0i_1}^{0} & B_{0i_1}^{1} & \cdots & B_{0i_1}^{b_2-1} & B_{0i_1}^{b_2} & \\ & B_{0i_1}^{-1} & B_{0i_1}^{0} & \cdots & B_{0i_1}^{b_2-2} & B_{0i_1}^{b_2-1} & B_{0i_1}^{b_2} \\ & & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix},$$

$$i_1 = 1, 2, \ldots, b_1,$$

where

$$\begin{cases} B_{0i_1}^{0i_2} = D_{i_1 i_2} \otimes \beta_1, \ i_2 = 0, 1, \ldots, b_2, \\ B_{0i_1}^{-1} = D_{i_1 0} \otimes S_2^0 \beta_1, \\ B_{0i_1}^{i_2} = D_{i_1(i_2+1)} \otimes S_2^0 \beta_1 + D_{i_1 i_2} \otimes S_2 e \beta_1, i_2 = 0, 1, \ldots, b_2 - 1, \\ B_{0i_1}^{b_2} = D_{i_1 b_2} \otimes S_2 e \beta_1, \end{cases}$$

and

$$B_{10} = \begin{bmatrix} B_{10}^{00} & B_{10}^{01} & B_{10}^{02} & \cdots & B_{10}^{0b_2} & & \\ & B_{10}^{0} & B_{10}^{1} & \cdots & B_{10}^{b_2-1} & B_{10}^{b_2} & \\ & & B_{10}^{0} & \cdots & B_{10}^{b_2-2} & B_{10}^{b_2-1} & B_{10}^{b_2} \\ & & & \ddots & \ddots & \ddots & \ddots \end{bmatrix},$$

where

$$\begin{cases} B_{10}^{00} = D_{00} \otimes S_1^0, \\ B_{10}^{0i_2} = D_{0i_2} \otimes S_1^0 \beta_2, \ i_2 = 1, 2, \ldots, b_2, \\ B_{10}^{i_2} = D_{0i_2} \otimes S_1^0 \beta_2^*, \ i_2 = 0, 1, \ldots, b_2, \end{cases}$$

and

$$A_{i_1} = \begin{bmatrix} A_{i_1}^0 & A_{i_1}^1 & A_{i_1}^2 & \cdots & A_{i_1}^{b_2} & & \\ & A_{i_1}^0 & A_{i_1}^1 & \cdots & A_{i_1}^{b_2-1} & A_{i_1}^{b_2} & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix},$$

$$i_1 = 0, 1, \ldots, b_1,$$

where

$$\begin{cases} A_{i_1}^{i_2} = D_{(i_1+1)i_2} \otimes S_1^0 \beta_1 + D_{i_1 i_2} \otimes S_1, \\ \qquad \text{for } i_1 = 0, 1, \ldots, b_1 - 1,\ i_2 = 0, 1, \ldots, b_2, \\ A_{b_1}^{i_2} = D_{b_1 i_2} \otimes S_1,\ i_2 = 0, 1, \ldots, b_2, \end{cases}$$

and

$$A_{-1} = \begin{bmatrix} A_{-1}^0 & A_{-1}^1 & A_{-1}^2 & \cdots & A_{-1}^{b_2} & & \\ & A_{-1}^0 & A_{-1}^1 & \cdots & A_{-1}^{b_2-1} & A_{-1}^{b_2} & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots \end{bmatrix},$$

where

$$A_{-1}^{i_2} = D_{0 i_2} \otimes S_1^0 \beta_1,\ i_2 = 0, 1, \ldots, b_2.$$

3.5. Stability of the queueing system

The stability condition for the single-arrival *DMAP/PH/*1 priority queue in [1] applies to the batch-arrival *DBMAP/PH/*1 priority queue in this paper as well. The system is stable only if $\lambda_h \bar{t}_h + \lambda_l \bar{t}_l < 1$; or equivalently, the system is stable only if $\rho = \rho_h + \rho_l < 1$, where ρ_h and ρ_l are the traffic intensity for the high and low priority arrivals, respectively.

The above necessary condition is quite intuitive, since otherwise if $\rho \geq 1$, the low priority queue length will add up to infinity. Throughout the rest of the paper, we assume that the queueing system is stable and that the steady state distribution of the system numbers exists.

4. Computational algorithms for the queueing model

In this section, we calculate the R measures for the related $M/G/1$-type Markov chains in Section 4.1. We then introduce the computation algorithms for the stationary distribution of the system numbers in Section 4.2. We analyze the time and storage complexity for the computation algorithms in Section 4.3. Finally, we compute the performance measures for the queueing system in Section 4.4, and give a brief summary in Section 4.5.

4.1. Characteristics of the R measures

We now introduce the definitions for the R measures for a $GI/G/1$-type Markov chain [19]. Consider a discrete-time Markov chain with transition matrix

$$P = \begin{bmatrix} D_0 & D_1 & D_2 & D_3 & \cdots \\ D_{-1} & C_0 & C_1 & C_2 & \cdots \\ D_{-2} & C_{-1} & C_0 & C_1 & \cdots \\ D_{-3} & C_{-2} & C_{-1} & C_0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}, \tag{4}$$

where D_0 is an $m_0 \times m_0$ matrix and C_k is an $m \times m$ matrix. The dimensions of $D_k, k > 0$, can be determined accordingly. Let $C = \sum_{k=-\infty}^{\infty} C_k$, which is assumed to be stochastic and irreducible. P is assumed to be stochastic, irreducible and aperiodic. The structure of (4) has similar row elements, i.e., row $i+1$ can be obtained by shifting row i to the right direction by one block element. Such a transition probability matrix is said to have the *row-repeating property* in this paper. The Markov chain defined in (4) is called a Markov chain of *Toeplitz* type or $GI/G/1$-type. Furthermore, if P is in the upper-Hessenberg form, the Markov chain is called $M/G/1$-type [6, 14]. We notice that both (2) and (3) are $M/G/1$-type.

Let us define

$$\begin{cases} L_0 = \{(0, j);\ j = 0, 1, 2, \ldots, m_0\}, \\ L_i = \{(i, j);\ j = 0, 1, 2, \ldots, m\},\ i \geq 1, \\ L_{\leq i} = \cup_{k=0}^{i} L_k. \end{cases}$$

Then, the state space S of the Markov chain P can be written as

$$S = \cup_{i=0}^{\infty} L_i.$$

For an arbitrary state (i, j), i and j are called *level* and *phase*, respectively[5]. The R measures for a $GI/G/1$-type Markov chain are defined as follows [19].

- For $j > 0$, $R_{0,j}$ is a matrix of size $m_0 \times m$, whose (r, s)th entry is

$$R_{0,j}(r, s) = \text{Expected number of visits to } (j, s) \text{ before hitting } L_{\leq (j-1)}, \text{ given a start at } (0, r).$$

[5] For the *DBMAP/PH/*1 priority queue defined in Section 3.3 and Section 3.4, we have $\Delta = \cup_{i=0}^{\infty} L_i$, and for an arbitrary state $(n_1, n_2, \cdot)$ in Δ, n_1 and n_2 are called *level* and *phase*, respectively.

- Φ_0 is a matrix of size $m_0 \times m_0$, whose (r, s)th entry is

  $\Phi_0(r, s) =$ Probability of hitting $(0, s)$ upon returning to L_0 for the first time, given a start at $(0, r)$.

- Let $i \geq 1$, for $k \geq 1$, R_k is a matrix of size $m \times m$, whose (r, s)th entry is

  $R_k(r, s) =$ Expected number of visits to $(i + k, s)$ before hitting $L_{\leq(i+k-1)}$, given a start at (i, r).

- Let $i \geq 1$, Φ is a matrix of size $m \times m$, whose (r, s)th entry is

  $\Phi(r, s) =$ Probability of hitting (i, s) upon returning to $L_{\leq i}$ for the first time, given a start at (i, r).

Notice that because the transition matrix (4) has the row-repeating property, all the R measures defined above are independent of i. The reason is that the transition probabilities in (4) from level i ($i \geq 1$) to level $i + k$ ($i, k \geq 1$) are independent of i.

We note that for an $M/G/1$-type Markov chain, all the elements below the sub-main diagonal of the transition probability matrix are zeros. According to the $GI/G/1$ results in [19], for the $M/G/1$-type Markov chains (2) and (3) studied in this paper, we have the following relations for $R_{0,j}$, R_k, Φ_0, and Φ:

$$\begin{cases} R_{0,k}(I - \Phi) = D_k + R_{0,k+1}C_{-1}, \\ R_k(I - \Phi) = C_k + R_{k+1}C_{-1}, \\ \Phi_0 = D_0 + R_{0,1}D_{-1}, \\ \Phi = C_0 + R_1 C_{-1}. \end{cases} \tag{5}$$

In addition, as shown in [6, 19], the steady probability vector π for the $GI/G/1$-type Markov chain satisfies

$$\pi_n = \pi_0 R_{0,n} + \sum_{k=1}^{n-1} \pi_k R_{n-k}, \quad n \geq 1,$$

where π_0 is the solution of

$$\pi_0 = \pi_0 \Phi_0,$$

subject to

$$\sum_{k=0}^{\infty} \pi_k e = 1.$$

Since (2) and (3) are banded matrices, with finite number of bands b_1, all the R_k and $R_{0,k}$, for $k > b_1$, are zero matrices. We obtain the following recursive formulas for R_k, $R_{0,k}$, and Φ_0:

$$\begin{cases} R_{0,k} &= (B_{0k} + R_{0,k+1}A_{-1})(I - A_0 - R_1 A_{-1})^{-1}, \\ & \quad \text{for } k = 1, 2, \ldots, b_1 - 1, \\ R_{0,b_1} &= B_{0b_1}(I - A_0 - R_1 A_{-1})^{-1}, \\ R_k &= (A_k + R_{k+1}A_{-1})(I - A_0 - R_1 A_{-1})^{-1}, \\ & \quad \text{for } k = 1, 2, \ldots, b_1 - 1, \\ R_{b_1} &= A_{b_1}(I - A_0 - R_1 A_{-1})^{-1}, \\ \Phi_0 &= B_{00} + R_{0,1}B_{10}\ . \end{cases} \tag{6}$$

Equation (6) can not obtain R measures directly, since all of them have infinite dimensions. The key is to examine the special structures of these R measures. We next illustrate that $R_{0,k}$ has a similar structure as B_{0k} for all k. There are two reasons for this fact. On one hand, note that $R_{0,k}$ represents the expected number of visits to level L_k before entering level $L_{\leq k-1}$, given a start at level L_0. $R_{0,k}$ is a probability measure for the following event:

> The event starts at time t when there is no high priority job in the system, i.e., $n_1 = 0$; at time $t + 1$, a batch arrival brings *at least* k high priority jobs, resulting in $n_1 \geq k$; and during the remaining event period, more high priority arrivals and departures may occur, but the system remains at $n_1 \geq k$ until some time later at $t + \tau$, when $n_1 = k$ and a high priority departure is about to happen, which would end the event period and result in $n_1 = k - 1$.

Since during the first time slot of the above event period, there are high priority arrivals, it is easy to see that during the whole event period, *at most one* low priority job can depart from the system. As a result, $R_{0,k}$ is in upper-Hessenberg form.

On the other hand, because the number of low priority job arrivals that occur after a given time is independent of the number of low priority jobs present in the system at the given time, and this number is also independent of the service process, $R_{0,k}$ has the row-repeating property, except for the first row, which is subject to the boundary condition.

In addition, based on the same argument in [1], we note that all R_k are upper-triangular matrices with the row-repeating property, and Φ_0 is also an upper-Hessenberg matrix with the similar row-repeating structure as B_{00}, since

$\Phi_0 = B_{00} + R_{0,1}B_{10}$. In summary, we have

$$R_{0,k} = \begin{bmatrix} R_{0,k}^{00} & R_{0,k}^{01} & R_{0,k}^{02} & R_{0,k}^{03} & \dots \\ R_{0,k}^{-1} & R_{0,k}^{0} & R_{0,k}^{1} & R_{0,k}^{2} & \dots \\ & R_{0,k}^{-1} & R_{0,k}^{0} & R_{0,k}^{1} & \dots \\ & & R_{0,k}^{-1} & R_{0,k}^{0} & \dots \\ & & & R_{0,k}^{-1} & \ddots \end{bmatrix}, \quad 1 \le k \le b_1,$$

$$R_k = \begin{bmatrix} R_k^0 & R_k^1 & R_k^2 & R_k^3 & \dots \\ & R_k^0 & R_k^1 & R_k^2 & \dots \\ & & R_k^0 & R_k^1 & \dots \\ & & & R_k^0 & \dots \\ & & & & \ddots \end{bmatrix}, \quad 1 \le k \le b_1,$$

$$\Phi_0 = \begin{bmatrix} \Phi_0^{00} & \Phi_0^{01} & \Phi_0^{02} & \Phi_0^{03} & \dots \\ \Phi_0^{10} & \Phi_0^{0} & \Phi_0^{1} & \Phi_0^{2} & \dots \\ & \Phi_0^{-1} & \Phi_0^{0} & \Phi_0^{1} & \dots \\ & & \Phi_0^{-1} & \Phi_0^{0} & \dots \\ & & & \Phi_0^{-1} & \ddots \end{bmatrix}.$$

Several properties for $R_{0,k}$ and R_k are introduced below.

Property I: $R_{0,k}$ and R_k are finite. According to the definition, elements of $R_{0,k}$ and R_k represent conditional expectation measures regarding the queue length. Therefore, all the elements of $R_{0,k}$ and R_k must be finite since the queue under study is stable.

Property II: $\pi_0 R_{0,k} e$ and $\pi_n R_k e$ are finite. This is valid since by virtue of probability interpretation, $\pi_0 R_{0,k} e$ and $\pi_n R_k e$ represent conditional expectations for the low priority queue length. Therefore, $\pi_0 R_{0,k} e$ and $\pi_n R_k e$ must be finite for a stable *DBMAP/PH/*1 priority queue.

Property III: $\lim_{i\to\infty} R_{0,k}^{0i} = 0$, $\lim_{i\to\infty} R_{0,k}^{i} = 0$ and $\lim_{i\to\infty} R_k^i = 0$. This was a conjecture in [1] and here we only use R_1^i as an illustration to prove the correctness of the conjecture. Suppose that this is not the case and $R_1^i \ge \varepsilon_1$ as $i \to \infty$ for some ε_1. Then by referring to (6), we have $R_b^i \to \infty$ based on the fact that $R_{b_1} = A_{b_1}(I - A_0 - R_1 A_{-1})^{-\mathbf{1}}$, since

$$(I - A_0 - R_1 A_{-1})^{-\mathbf{1}} = I + \sum_{r=1}^{\infty} (A_0 + R_1 A_{-1})^r, \quad (7)$$

and all A_k are upper triangular matrices with positive elements on the main diagonal. However, this contradicts the fact that $R_{0,k}$ and R_k are all finite.

4.2. Computation of the stationary distribution of the system numbers

We next focus our discussion on the computation of the stationary distribution of the system numbers, $\pi_n = [\pi_{n0}, \pi_{n1}, \pi_{n2}, \dots]$, $n = 0, 1, 2, \dots$, for the $M/G/1$-type Markov chain. The computation algorithms are identical for both the preemptive and non-preemptive cases.

The special structures of R_k, $R_{0,k}$ and Φ_0 shown in Section 4.1 suggest that it is possible to compute these R measures by iteration, which is required to compute π_{ij} for $i, j = 0, 1, 2, \dots$. In what follows, we compute π_{ij} by two stages as introduced below.

4.2.1. Computation of R_k, $R_{0,k}$ and Φ_0

In stage one, we compute R_k, $R_{0,k}$ and Φ_0 by the following iterative algorithm.

1. Set $R_{0,k} = B_{0k}$, $R_k = A_k$.
2. Update $R_{0,k}$, R_k recursively according to (6) until the differences in the results obtained in two consecutive iterations are element-wise smaller than ε, which is a small threshold value used for precision control.
3. Compute Φ_0 according to (6).

Let $R_{0,k}^{(s)}$ and $R_k^{(s)}$ be the temporary results obtained in the s-th iteration. The two sequences, $\{R_{0,k}^{(s)}, s = 0, 1, 2, \dots, \infty\}$ and $\{R_k^{(s)}, s = 0, 1, 2, \dots, \infty\}$ have the following properties.

Monotonically increasing. $R_{0,k}^{(s)}$ and $R_k^{(s)}$ increase as $s \to \infty$. This can be verified by examining (6). We take $R_k{}^{(s)}$ as an example to reveal this property. The initial value for $R_k^{(0)}$ is given by A_k. After the first iteration, we have $R_k^{(1)} > R_k^{(0)}$ since

$$R_k^{(1)} = (A_k + R_{k+1}^{(0)} A_{-1})(I - A_0 - R_1^{(0)} A_{-1})^{-\mathbf{1}}. \quad (8)$$

Check the right-hand side of (8), the first factor $(A_k + R_{k+1}^{(0)} A_{-1})$ is greater than A_k, and the second factor $(I - A_0 - R_1^{(0)} A_{-1})^{-\mathbf{1}}$ is greater than I by referring to (7). Therefore, the product of the two factors is greater than $R_k^{(0)}$. It is straightforward to argue that $R_k^{(s+1)} > R_k^{(s)}$ for any $s > 0$ by method of mathematical induction on s. Similarly, we have $R_{0,k}^{(s+1)} > R_{0,k}^{(s)}$.

Upper bounded. $R_{0,k}^{(s)}$ and $R_k^{(s)}$ are finite according to Property I of the R measures as introduced in Section 4.1. Therefore, $R_{0,k}^{(s)}$ and $R_k^{(s)}$ must be bounded by some finite values.

Based on the above two properties, we conclude that the proposed iteration algorithm must converge. In other words, $R_{0,k}^{(s)}$ and $R_k^{(s)}$ must converge to some limit value $R_{0,k}^{*}$ and

R_k^*, respectively, as $s \to \infty$. The limit values of $R^*{}_{0,k}$ and R_k^* can be considered as the exact solution for the system of equations (6).

Notice that in step 2 of the iteration algorithm, we must do truncation on $R_{0,k}$ and R_k since it is impossible to store and compute matrices with infinite dimensions. We now discuss how to choose the truncation level K. First, we notice that in real applications we do not compute π_{ij} for an arbitrarily large index. Instead, we have to truncate both the high and low priority queue lengths at some level, which may depend on the maximum queue length that we are interested in. By intuition, we expect that very high dimension elements of the R measures will not contribute too much to the queue length distribution. In this paper we recommend the following truncation method. During the iteration, we refer to the small threshold value ε and choose the truncation level K such that

$$\begin{cases} R_{0,k}^{0K} < \varepsilon, \ R_{0,k}^{K} < \varepsilon, \ R_{k}^{K} < \varepsilon, \ \text{and} \\ R_{0,k}^{0i} \geq \varepsilon, \ R_{0,k}^{i} \geq \varepsilon, \ R_{k}^{i} \geq \varepsilon, \ \text{for } i = 0, 1, \ldots K-1. \end{cases} \tag{9}$$

We notice that K must exist according to Property III of the R measures as introduced in Section 4.1. We discuss the bound of the computation error caused by this truncation method in Section 4.2.2.

We now discuss how to compute $(I - A_0 - R_1 A_{-1})^{-1}$. We notice that all A_i are upper-triangular with the row-repeating property, so $I - A_0 - R_1 A_{-1}$ is also upper-triangular with the row-repeating property, which is an important fact. As a consequence, $(I - A_0 - R_1 A_{-1})^{-1}$ can be computed as shown in [1], although it has infinite dimensions. We omit the details in this paper.

We next discuss how to do multiplications of $R_{0,k}A_{-1}$, $R_{0,1}B_{10}$ and $R_k A_{-1}$. It is quite straightforward to notice that in (6), the result for $R_{0,k}A_{-1}$ (also for $R_{0,1}B_{10}$) has the same structure as that of $R_{0,k}$, and the result for $R_k A_{-1}$ has the same structure of that of R_k, since A_{-1} and B_{10} are upper-triangular with the row-repeating property. Let

$$R_{0,k}A_{-1} = \begin{bmatrix} U^{00} & U^{01} & U^{02} & U^{03} & \ldots \\ U^{-1} & U^{0} & U^{1} & U^{2} & \ldots \\ & U^{-1} & U^{0} & U^{1} & \ldots \\ & & U^{-1} & U^{0} & \ldots \\ & & & U^{-1} & \ddots \end{bmatrix},$$

$$R_k A_{-1} = \begin{bmatrix} V^{0} & V^{1} & V^{2} & V^{3} & \ldots \\ & V^{0} & V^{1} & V^{2} & \ldots \\ & & V^{0} & V^{1} & \ldots \\ & & & V^{0} & \ldots \\ & & & & \ddots \end{bmatrix}.$$

Then we have

$$\begin{cases} U^{0j} = \sum_{n=0}^{min(b_2, j)} R_{0,k}^{0(j-n)} A_{-1}^{n}, \\ U^{j} = \sum_{n=0}^{min(b_2, j+1)} R_{0,k}^{j-n} A_{-1}^{n}, \\ V^{j} = \sum_{n=0}^{min(b_2, j)} R_{k}^{j-n} A_{-1}^{n}. \end{cases} \tag{10}$$

4.2.2. *Computation of* π_{ij}

In stage two, we compute the stationary distribution of the $M/G/1$-type Markov chain. We firstly compute π_0. Note that Φ_0 is an upper-Hessenberg form transition probability matrix with the row-repeating property, which is obviously an $M/G/1$-type. By standard matrix-analytic methods [6], $\pi_0 = \pi_0 \Phi_0$ can be solved. We omit the details here. Based on π_0, we can compute π_n by

$$\begin{cases} \pi_n = \pi_0 R_{0,n} + \sum_{k=1}^{n-1} \pi_k R_{n-k}, \quad 1 \leq n \leq b_1, \\ \pi_n = \sum_{k=1}^{b_1} \pi_{n-k} R_k, \quad b_1 < n < \infty. \end{cases} \tag{11}$$

Multiplication of $\pi_0 R_{0,k}$ and $\pi_{k_1} R_{k_2}$ can be done in the following way. Let $\pi_0 R_{0,k} = [p_0, p_1, p_2, \ldots]$, and let $\pi_{k_1} R_{k_2} = [q_0, q_1, q_2, \ldots]$. Then we have

$$\begin{cases} p_i = \pi_{00} R_{0,k}^{0i} + \sum_{n=-1}^{i-1} \pi_{0(i-n)} R_{0,k}^{n}, \\ q_i = \sum_{n=0}^{i} \pi_{k_1(i-n)} R_{k_2}^{n}. \end{cases}$$

Finally, we normalize π_{ij} and ensure that

$$\sum_{i=0}^{\infty} \sum_{j=0}^{\infty} \pi_{ij} e = 1.$$

Let π_n^* be the theoretical result under the hypothesis that there is no truncation on the R measures and that the limit values $R^*{}_{0,k}$ and $R^*{}_k$ are applied in (11). We can expect that the practical computation result $\pi_n > \pi_n^*$, since π_n is obtained by truncating the probabilities for a high level queue length. Suppose that $R_{0,k}$ and R_k are truncated at level K according to the ε policy recommended in (9), then for $1 \leq$

$n \le b_1$ we have

$$\pi_n e - \pi_n^* e < \left[\pi_0 \Theta_1^K e + \sum_{k=1}^{n-1} \pi_k \Theta_2^K e\right] + \left[\pi_0^* \Theta_1 e + \sum_{k=1}^{n-1} \pi_k^* \Theta_2 e\right], \quad (12)$$

and for $n > b_1$ we have

$$\pi_n e - \pi_n^* e < \left[\sum_{k=1}^{b_1} \pi_{n-k} \Theta_2^K e + \sum_{k=1}^{b_1} \pi_{n-k}^* \Theta_2 e\right], \quad (13)$$

in which

$$\Theta_1 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon & \dots \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon & \dots \\ & \varepsilon & \varepsilon & \varepsilon & \dots \\ & & \varepsilon & \varepsilon & \dots \\ & & & \varepsilon & \ddots \end{bmatrix}, \quad \Theta_2 = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon & \dots \\ & \varepsilon & \varepsilon & \varepsilon & \dots \\ & & \varepsilon & \varepsilon & \dots \\ & & & \varepsilon & \dots \\ & & & & \ddots \end{bmatrix},$$

Θ_1^K and Θ_2^K are finite dimension matrices obtained by truncating Θ_1 and Θ_2, respectively, at level K.

Since Θ_1^K and Θ_2^K only have finite dimensions, we have $\pi_0 \Theta_1^K e \to 0$ and $\sum_{k=1}^{b_1} \pi_k \Theta_2^K e \to 0$ as $\varepsilon \to 0$. Furthermore, according to Property II of the R measures as introduced in Section 4.1, we have $\pi_0^* \Theta_1 e \to 0$ and $\sum_{k=1}^{b_1} \pi_k^* \Theta_2 e \to 0$ as $\varepsilon \to 0$. Therefore, it is evident from (12) and (13) that by choosing sufficiently small ε, an arbitrary precision for π_n can be achieved.

4.3. Analysis of the time and memory complexity

In this subsection, we analyze the time and memory complexity for the computation algorithms developed in Section 4.2. We limit our discussion to the non-preemptive case, since the analysis can also be applied to the preemptive case. The only difference is that the size of the element matrices for the non-preemptive case is $2nm \times 2nm$, while the size of the element matrices for the preemptive case is $nm \times nm$. Therefore, the non-preemptive queue has a higher time and memory cost. We assume that R_k and R_{0k} are truncated at level K and that the algorithm converges after r^* times of iteration.

We take a look at the time complexity for the first step of the algorithm, which is the kernel part of the whole computation. Since Φ_0 is only computed once for the whole algorithm and is easy to consider, we only discuss the computation of R_k and $R_{0,k}$. We also note that the item $(I - A_0 - R_1 A_{-1})^{-1}$ only needs to be computed once for each iteration of the algorithm. In addition, because of the row-repeating property of R_{k+1} and A_{-1}, the result of $R_{k+1}A_{-1}$ has the same structure as that of R_{k+1}. Therefore, we only need to compute the first row for the result of $R_{k+1}A_{-1}$. Similarly, the result of $R_{0,k+1}A_{-1}$ has the same structure with $R_{0,k+1}$, and we only need to compute the first row and the second row for the result of $R_{0,k+1}A_{-1}$. The details can be better understood by examining (10). As a result, for each iteration, we offer the following analysis on the time complexity.

- $R_1 A_{-1}$: $\frac{(K+1)(K+2)}{2}\mathbf{m} + \frac{K(K+1)}{2}\mathbf{a}$, where $\mathbf{m}$ and $\mathbf{a}$ represent the time complexity for the multiplication and addition operations, respectively, of two $2nm \times 2nm$ dimensional matrices.
- $I - A_0 - R_1 A_{-1}$: $\frac{(K+1)(K+2)}{2}\mathbf{m} + \frac{(K+1)(K+4)}{2}\mathbf{a}$.
- $(I - A_0 - R_1 A_{-1})^{-1}$: $(K^2 + 3K + 1)\mathbf{m} + (K^2 + 2K + 2)\mathbf{a} + (K + 1)\mathbf{i}$, where $\mathbf{i}$ represents the time complexity for the inverse of a $2nm \times 2nm$ dimensional matrix. See page 29 of [1] on how to compute the inverse of an upper-triangular matrix.
- $A_k + R_{k+1}A_{-1}$: $\frac{(K+1)(K+2)}{2}\mathbf{m} + \frac{(K+1)(K+2)}{2}\mathbf{a}$.
- $(A_k + R_{k+1}A_{-1})(I - A_0 - R_1 A_{-1})^{-1}$: $(K + 1)(K + 2)\mathbf{m} + (K + 1)^2\mathbf{a}$ (after $(I - A_0 - R_1 A_{-1})^{-1}$ has been computed).
- $R_{0,k+1}A_{-1}$: $(K + 1)(K + 2)\mathbf{m} + K(K + 1)\mathbf{a}$.
- $B_{0k} + R_{0,k+1}A_{-1}$: $(K + 1)(K + 2)\mathbf{m} + (K + 1)(K + 2)\mathbf{a}$.
- $(B_{0k} + R_{0,k+1}A_{-1})(I - A_0 - R_1 A_{-1})^{-1}$: $2(K + 1)(K + 2)\mathbf{m} + 2(K + 1)^2\mathbf{a}$ (after $(I - A_0 - R_1 A_{-1})^{-1}$ has been computed). We observe that the time complexity of $(B_{0k} + R_{0,k+1}A_{-1})(I - A_0 - R_1 A_{-1})^{-1}$ is double the complexity of $(A_k + R_{k+1}A_{-1})(I - A_0 - R_1 A_{-1})^{-1}$.

For some items, the above time complexity is the upper bound for the following three reasons. Firstly, we notice that R_{b_1} takes less time than R_k for $k = 1, 2, \dots, b_1 - 1$; secondly, R_{0,b_1} also takes less time than $R_{0,k}$ for $k = 1, 2, \dots, b_1 - 1$; and thirdly, block elements of B_{0k} and $R_{0,k+1}$ have smaller dimensions than $2nm \times 2nm$, which means that the actual involved computation is less than this estimation. Therefore, the total time complexity for computation of R_k and $R_{0,k}$ is given by

$$r^* b_1 \left((4K^2 + 12K + 7)\mathbf{m} + (4K^2 + 8K + 5)\mathbf{a} + (K + 1)\mathbf{i}\right).$$

Next, we estimate the memory complexity for the computation algorithms. We only consider the required storage for R_k, $R_{0,k}$, Φ_0, and the induced temporary storage during the computation. Again, we assume that the truncation level for R_k, $R_{0,k}$ and Φ_0 is K. Furthermore, in order to simplify the discussion, we assume that all the block elements for R_k, $R_{0,k}$ and $\Phi_0 R_k$ are $2nm \times 2nm$ dimensional matrices, although some boundary elements have smaller sizes. For R_k, we only need to store the first row of the block elements. Therefore, the total storage cost for all R_k is $b_1(K + 1)\mathbf{B}$, where $\mathbf{B}$ is the storage cost of a $2nm \times 2nm$ matrix. For $R_{0,k}$ and Φ_0, we only need to store the first and the second row of the block elements. Therefore, the total storage cost for all $R_{0,k}$ and

Table 1 Temporary matrices used during the computation

	Purpose of the matrix	Storage cost
Temporary matrix 1	Store and compute $(A_k + R_{k+1}A_{-1})$	Same as R_k
Temporary matrix 2	Store and compute $(I - A_0 - R_1 A_{-1})^{-1}$	Same as R_k
Temporary matrix 3	Store and compute $(B_{0,k} + R_{0,k+1}A_{-1})$	Same as $R_{0,k}$

Φ_0 is $2(b_1 + 1)(K + 1)\mathbf{B}$. During the computation, we also use three temporary matrices as explained in Table 1[6]. As a result, the total memory complexity is given by

$$3(b_1 + 2)(K + 1)\mathbf{B}.$$

From the above analysis, we notice that in computation of the R measures, the memory complexity mainly depends on the precision requirement (the K value) as well as the application size (the b_1 value). The time complexity, however, also depends on the nature of the application (the r^* value or the convergence rate of the algorithm). While the time complexity is proportional to K^2, the memory complexity is proportional only to K. The memory complexity is greatly reduced due to the row-repeating property of the R measures.

Finally, we have to emphasize that after all the R measures are obtained, the complexity for computing π_n depends on the truncation level for both the high and low priority queues, as implied by (11). Although in theory the queue length truncation level can be set to an arbitrarily large value, that may require a high time and memory cost. There is always a trade-off between the problem size and the implementation cost. Our experiences from the numerical examples demonstrate that it is feasible to compute realistic *DBMAP/PH/*1 priority queue without prohibitive difficulties.

4.4. Calculation of the performance measures

Before we present the performance calculation, we firstly summarize the terms and notation below:

$P\{idle\}$: probability that the server is idle;

$P\{busy\}$: probability that the server is busy;

$P\{busy_h\}$: probability that the server is busy with a high priority job;

$P\{busy_l\}$: probability that the server is busy with a low priority job;

$P\{N_h = i\}$: probability that there are i high priority jobs in the system;

$P\{N_l = j\}$: probability that there are j low priority jobs in the system;

$P\{Q_h = i\}$: probability that there are i high priority jobs waiting in the queue;

$P\{Q_l = j\}$: probability that there are j low priority jobs waiting in the queue;

$\overline{N}$: average number of jobs in the system;

$\overline{N}_h$: average number of high priority jobs in system;

$\overline{N}_l$: average number of low priority jobs in system;

$\overline{Q}$: average number of jobs in the queue;

$\overline{Q}_h$: average high priority queue length;

$\overline{Q}_l$: average low priority queue length;

$\overline{W}_h$: average waiting time for high priority jobs;

$\overline{W}_l$: average waiting time for low priority jobs.

Based on π_{ij}, for the *preemptive DBMAP/PH/*1 priority queue in Section 3.4, the following performance measures can be derived:

$$P\{idle\} = \pi_{00}e,$$

$$P\{busy\} = 1 - \pi_{00}e,$$

$$P\{busy_h\} = 1 - \sum_{j=0}^{\infty} \pi_{0j}e,$$

$$P\{busy_l\} = \sum_{j=1}^{\infty} \pi_{0j}e,$$

$$P\{N_h = i\} = \sum_{j=0}^{\infty} \pi_{ij}e, \ i \geq 0,$$

$$P\{N_l = j\} = \sum_{i=0}^{\infty} \pi_{ij}e, \ j \geq 0,$$

$$P\{Q_h = 0\} = P\{N_h = 1\} + P\{N_h = 0\},$$

$$P\{Q_h = i\} = P\{N_h = i + 1\}, \ i \geq 1,$$

$$P\{Q_l = 0\} = \pi_{00}e + \pi_{01}e + \sum_{i=1}^{\infty} \pi_{i0}e,$$

$$P\{Q_l = j\} = \pi_{0(j+1)}e + \sum_{i=1}^{\infty} \pi_{ij}e, \ j \geq 1,$$

$$\overline{N}_h = \sum_{i=0}^{\infty} i\, P\{N_h = i\},$$

[6] Different implementations of the algorithms may have different requirements for the temporary storage.

$$\overline{N}_l = \sum_{j=0}^{\infty} j\,P\{N_l = j\},$$
$$\overline{N} = \overline{N}_h + \overline{N}_l,$$
$$\overline{Q}_h = \sum_{i=0}^{\infty} i\,P\{Q_h = i\},$$
$$\overline{Q}_l = \sum_{j=0}^{\infty} j\,P\{Q_l = j\},$$
$$\overline{Q} = \overline{Q}_h + \overline{Q}_l,$$
$$\overline{W} = \overline{Q}/\lambda,$$
$$\overline{W}_h = \overline{Q}_h/\lambda_h,$$
$$\overline{W}_l = \overline{Q}_l/\lambda_l.$$

For the *non-preemptive* priority case in Section 3.3, the same set of performance measures can be derived. Notice that for the non-preemptive queue, state $(n_1, n_2, \cdot)$, $n_1 \geq 1$, may correspond to the case of having a low priority system number of n_2 or $n_2 + 1$, depending on which priority level the server is currently busy with. Therefore, calculation of $P\{N_l = j\}$ is different from the *preemptive* case as follows:

$$P\{N_l = 0\} = \pi_{00}e + \sum_{i=1}^{\infty} \pi_{i0}e_h,$$
$$P\{N_l = j\} = \pi_{0j}e + \sum_{i=1}^{\infty} \pi_{ij}e_h + \sum_{i=1}^{\infty} \pi_{i(j-1)}e_l, \quad j \geq 1,$$

where e_h and e_l are column vectors of dimension $2nm$, $e_h = \langle e, e_0\rangle$, $e_l = \langle e_0, e\rangle$, e is an nm-dimensional column vector of all 1s, and e_0 is an nm-dimensional column vector of all 0s. Here n and m are the dimensions of the arrival and service processes, respectively.

4.5. Section summary

The *DBMAP/PH/*1 priority queueing model defined in this paper is a natural extension of the single arrival *DMAP/PH/*1 priority queue in [1]. In the single arrival model, the underlying Markov chain is quasi-birth-death (*QBD*) type and there is only a single R measure. Furthermore, this R measure is the solution of a matrix quadratic equation $R = A_0 + RA_1 + R^2A_2$ (see page 26 in [1]). This makes it possible to directly compute the elements of R, although R has infinite dimensions. Based on R, the steady state probability vector of the *QBD*-type Markov chain can be obtained.

For the batch arrival queueing model introduced in this paper, the underlying Markov chains are $M/G/1$-type, since (2) and (3) are both in upper-Hessenberg form. Therefore, there exists a set of interrelated R measures, which cannot be explicitly solved. Standard $M/G/1$-type Markov chains may have infinite number of levels, but usually the phase number has to be finite [6, 19]. The Markov chains involved in this paper, however, have infinite number of levels and infinite number of phases, which is the main challenge in solving the *DBMAP/PH/*1 priority queue.

In this section, we showed that the R measures play a key role in the analysis of the involved Markov chain. We revealed that the structure of R_k and R_{0k} are in upper-triangular and upper-Hessenberg form, respectively, and proved that a solution for these R measures exists. Based on the special structure of the R measures, we developed algorithms for computing the stationary distribution of the system numbers and obtained the queue length distribution.

5. Numerical results

In this section, we present numerical results for the computation algorithms. In Section 5.1, we provide a simple non-preemptive *DBMAP/PH/*1 priority queue and demonstrate how to apply the computation algorithms to calculate the queue length distribution. In Section 5.2, we provide another non-preemptive *DBMAP/PH/*1 priority queue with realistic parameters to model wireless multimedia communications. In Section 5.3, we present simulation results for an example preemptive *DBMAP/PH/*1 priority queue with different traffic intensities.

In all the examples we implement the algorithms (without any optimization) on a normal PC with the Maple Linear Algebra software package. The computation is completed within reasonable time. The computation time can be saved by optimizing the algorithm code and by utilizing the more efficient Matlab software package.

5.1. A simple example

In this subsection, we consider a simple non-preemptive *DBMAP/PH/*1 priority queue with the following settings:

$$S_1 = \begin{bmatrix} 0.0 & 0.05 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.05 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.05 \\ 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix},$$
$$S_2 = \begin{bmatrix} 0.0 & 0.05 & 0.0 \\ 0.0 & 0.0 & 0.05 \\ 0.0 & 0.0 & 0.0 \end{bmatrix},$$
$$\beta_1 = [1.0,\ 0.0,\ 0.0,\ 0.0],$$
$$\beta_2 = [1.0,\ 0.0,\ 0.0],$$
$$S_1^0 = [0.95,\ 0.95,\ 0.95,\ 1.0]^{\mathrm{t}},$$
$$S_2^0 = [0.95,\ 0.95,\ 1.0]^{\mathrm{t}},$$
$$D = \begin{bmatrix} 0.1 & 0.9 \\ 0.2 & 0.8 \end{bmatrix},$$

$D_{00} = 3/5D$, $D_{10} = D_{20} = D_{30} = D_{40} = 1/15D$, $D_{01} = D_{02} = 1/15D$ and all other $D_{i_1 i_2} = 0$, $\alpha = [2/11, 9/11]$.

The mean arrival rate for the high priority jobs is $\lambda_h = 0.6666666668$; the mean arrival rate for the low priority jobs is $\lambda_l = 0.2000000000$. The mean service time for the high priority jobs is $\bar{t}_h = 1.052625$; the mean service time for the low priority jobs is $\bar{t}_l = 1.0525$. The traffic intensity is $\rho = \lambda_h \bar{t}_h + \lambda_l \bar{t}_l = 0.91225$.

We apply the computing algorithms for the above non-preemptive *DBMAP /PH/*1 priority queue. We firstly compute $R_{0,k}$, R_k and Φ_0. Shown below is a list of the elements for $R^i_{0,k}e$, $R^i_k e$ and $\Phi^i_0 e$, in which $[\texttt{vector}]^{\texttt{t}}$ with a superscript $\texttt{t}$ indicates the transpose of the $[\texttt{vector}]$. The results demonstrate that $R^i_{0,k}$, R^i_k and Φ^i_0 are monotonically decreasing with index i.

$$\Phi^3_0 e = [0.0172293722644972607, 0.0172293722649843467]^{\texttt{t}},$$

$$\Phi^4_0 e = [0.0132203930775591089, 0.0132203930779325532]^{\texttt{t}},$$

$$\Phi^5_0 e = [0.0067718964634043803, 0.0067718964635953664]^{\texttt{t}},$$

$$R^3_{0,1} e = [0.0195025931377530854, 0.0195025931383034196]^{\texttt{t}},$$

$$R^4_{0,1} e = [0.0151775459130132143, 0.0151775459134413857]^{\texttt{t}},$$

$$R^5_{0,1} e = [0.0080271161708515993, 0.0080271161710779408]^{\texttt{t}},$$

$$R^3_1 e = [0.01133058092, 0.01132962692, 0.01130833870, 0.01090051265, 0.01133058092, 0.01132962692, 0.01130833870, 0.01090051265, 0.01994472184, 0.01992589597, 0.01950259314, 0.01950259314, 0.01994472184, 0.01992589597, 0.01950259314, 0.01950259314]^{\texttt{t}},$$

$$R^4_1 e = [0.008749152227, 0.008748311523, 0.008730811283, 0.008410852562, 0.008749152227, 0.008748311523, 0.008730811283, 0.008410852562, 0.01554337650, 0.01552671376, 0.01517754591, 0.01517754591, 0.01554337650, 0.01552671376, 0.01517754591, 0.01517754591]^{\texttt{t}},$$

$$R^5_1 e = [0.004560695169, 0.004560147956, 0.004549954131, 0.004378279617, 0.004560695169, 0.004560147956, 0.004549954131, 0.004378279617, 0.008242636244, 0.008231717680, 0.008027116171, 0.008027116171, 0.008242636245, 0.008231717680, 0.008027116171, 0.008027116171]^{\texttt{t}},$$

$$R^3_2 e = [0.005058656884, 0.005057806411, 0.005040650502, 0.004751228003, 0.005058656884, 0.005057806412, 0.005040650502, 0.004751228004, 0.01126018458, 0.01124333005, 0.01090051265, 0.01090051265, 0.01126018458, 0.01124333005, 0.01090051265, 0.01090051265]^{\texttt{t}},$$

$$R^4_2 e = [0.003936848553, 0.003936144055, 0.003922704609, 0.003701408279, 0.003936848553, 0.003936144055, 0.003922704609, 0.003701408279, 0.008694236453, 0.008680219463, 0.008410852562, 0.008410852562, 0.008694236454, 0.008680219464, 0.008410852562, 0.008410852562]^{\texttt{t}},$$

$$R^5_2 e = [0.002080874794, 0.002080459269, 0.002073270137, 0.001959981413, 0.002080874794, 0.002080459269, 0.002073270137, 0.001959981413, 0.004531520850, 0.004523197077, 0.004378279617, 0.004378279617, 0.004531520851, 0.004523197078, 0.004378279617, 0.004378279617]^{\texttt{t}}.$$

We obtain π_{ij}, in which π_{00} is

$$\pi_{00} = [0.198450152007130232,\ 0.801549847992869768] \text{ (before normalization)},$$

$$\pi_{00} = [0.018482324985794025,\ 0.074649675014205970] \text{ (after normalization)}.$$

The joint queue length probability mass function is shown in Figure 1. We also determine the marginal high and low priority queue length probability mass functions as shown in Figure 2(a) and Figure 2(b), respectively. We obtain the high

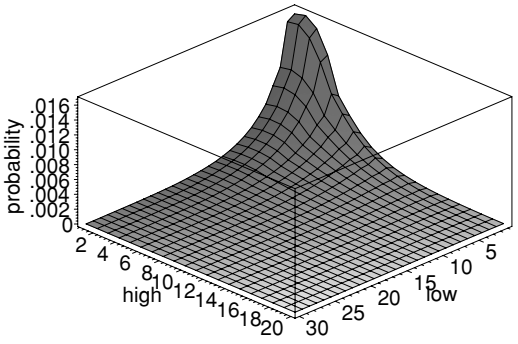

Fig. 1 Joint queue length probability mass function

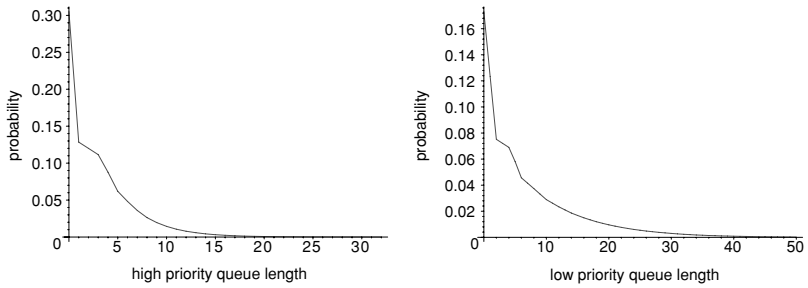

Fig. 2 Marginal queue length probability mass functions

(a) high priority queue (b) low priority queue

priority mean queue length $\overline{Q}_h = 2.5$, and the low priority mean queue length $\overline{Q}_l = 6.6$. By *Little*'s law, we have the high priority queue mean waiting time $\overline{W}_h = 3.75$, and the low priority queue mean waiting time $\overline{W}_l = 33.5$. We obtain the system idle probability $P\{idle\} = \pi_{00}e = 0.09313$.

5.2. A wireless multimedia example

In this subsection, we consider a more realistic non-preemptive *DBMAP/PH/*1 priority example that resembles wireless multimedia communications. We assume that the wireless network can transmit a fixed size data block during one time slot, and each application packet is segmented into a number of data blocks (depending on the actual size of the packet) for the purpose of transmission. We consider a mobile terminal that generates three types of traffic, i.e., voice, video and data. We assume that the voice and video traffic are handled with a higher priority than the data traffic. We further assume that for each transmission of the data block, the average error probability is given by ϵ. In this example, we assume that the data traffic is protected by ARQ, with at most three (re)transmissions. We also assume that the voice and video traffic is not protected by ARQ due to the real-time requirement and the fact that they are more tolerant of loss or error. Therefore, the *PH*-type service parameters (β_1, S_1), (β_2, S_2) for the high and low priority queue are given by

$$S_1 = [0.0]\,,\ S_1^0 = [1.0],\ \beta_1 = [1.0],$$

$$S_2 = \begin{bmatrix} 0.0 & \epsilon & 0.0 \\ 0.0 & 0.0 & \epsilon \\ 0.0 & 0.0 & 0.0 \end{bmatrix},$$

$$S_2^0 = [1-\epsilon,\ \ 1-\epsilon,\ \ 1.0]^{\mathrm{t}},$$

$$\beta_2 = [1.0,\ 0.0,\ 0.0].$$

The mean service time for the high priority traffic is $\bar{t}_h = 1$, and the mean service time for the low priority traffic is $\bar{t}_l = \beta_2(I - S_2)^{-\mathbf{1}}e$.

We assume that the voice and video traffic can be modeled by an ON/OFF source as depicted in Figure 3. In the OFF state the mobile terminal does not generate any voice or video traffic; in the Voice ON state, the mobile terminal generates voice traffic with a fixed batch size of 1; and in the Video ON state, the mobile terminal generates video traffic with a batch size ranging from 1 to 8. In this example, we assume that the transmission block size is 32 bytes. Therefore, the maximum video packet size is 256 bytes. We assume that the voice packet size is fixed and equal to the transmission block size; the video packet size follows a log-normal distribution, which is a valid assumption for typical video conference traffic. As an example, we assume that the probability mass function $v[i]$, $i = 1 \ldots 8$, for the video packet size (in terms of number of transmission blocks) is given by

$$[0.002,\ 0.153,\ 0.427,\ 0.286,\ 0.099,\ 0.025,\ 0.006,\ 0.002].$$

The above voice and video traffic can be modeled by a Markov chain with the following transition probability matrix

$$D = \begin{bmatrix} 1-h_1-h_2 & h_2 & h_1 \\ g_2 & 1-g_2 & 0 \\ g_1 & 0 & 1-g_1 \end{bmatrix}.$$

The steady state probability vector for the above voice and video traffic model is given by

$$\left[\frac{g_1g_2}{h_1g_2+g_1h_2+g_1g_2}, \frac{g_1h_2}{h_1g_2+g_1h_2+g_1g_2}, \frac{h_1g_2}{h_1g_2+g_1h_2+g_1g_2}\right].$$

We further assume that data traffic generated in any state is independent from the voice and video traffic and that the data

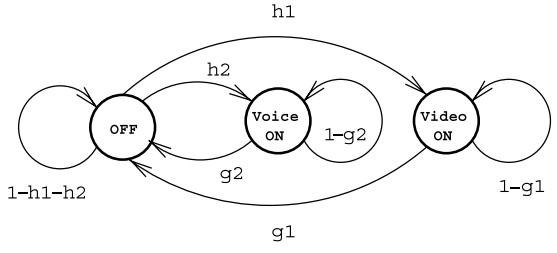

Fig. 3 Voice and video traffic model

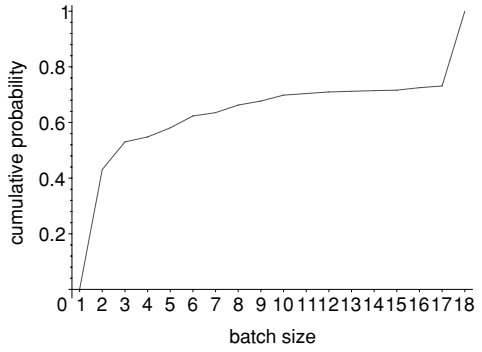

Fig. 4 Data packet batch size distribution

traffic can be modeled by a batch Bernoulli arrival process with arrival probability of p. The batch size of the data packet is assumed to be within a range from 1 to 18. Therefore, the maximum data packet size is 576 bytes, which is a typical configuration for a mobile data terminal with PPP connection to the wireless network. The batch size of the data traffic is assumed to follow a stepwise distribution [4], as shown in Figure 4. As a matter of fact, we assume that the probability mass function $d[i]$, $i = 1 \ldots 18$, for the data packet size is given by

$$[0.0005,\ 0.430,\ 0.099,\ 0.018,\ 0.033,\ 0.043,\ 0.012,\ 0.027,\ 0.014,\ 0.021,\ 0.006,\ 0.006,\ 0.002,\ 0.002,\ 0.002,\ 0.009,\ 0.0065,\ 0.269].$$

Under the above assumptions, we can determine all the parameters for the aggregated traffic arrival process including voice, video and data, which is a *DBMAP* arrival process with priorities as defined in Section 3.1. The $D_{i_1 i_2}$ parameters are listed as follows:

$$D_{00} = \begin{bmatrix} (1-h_1-h_2)(1-p) & 0 & 0 \\ g_2(1-p) & 0 & 0 \\ g_1(1-p) & 0 & 0 \end{bmatrix},$$

$$D_{0i_2} = \begin{bmatrix} (1-h_1-h_2)d[i_2]p & 0 & 0 \\ g_2 d[i_2]p & 0 & 0 \\ g_1 d[i_2]p & 0 & 0 \end{bmatrix},\ i_2 = 1 \ldots 18,$$

$$D_{10} = \begin{bmatrix} 0 & h_2(1-p) & h_1(1-p)v[1] \\ 0 & (1-g_2)(1-p) & 0 \\ 0 & 0 & (1-g_1)(1-p)v[1] \end{bmatrix},$$

$$D_{i_10} = \begin{bmatrix} 0 & 0 & h_1(1-p)v[i_1] \\ 0 & 0 & 0 \\ 0 & 0 & (1-g_1)(1-p)v[i_1] \end{bmatrix},\ i_1 = 2 \ldots 8,$$

$$D_{1i_2} = \begin{bmatrix} 0 & h_2 d[i_2]p & h_1 v[1] d[i_2]p \\ 0 & (1-g2)d[i_2]p & 0 \\ 0 & 0 & (1-g_1)v[1]d[i_2]p \end{bmatrix},$$
$$i_2 = 1 \ldots 18,$$

$$D_{i_1 i_2} = \begin{bmatrix} 0 & 0 & h_1 v[i_1] d[i_2]p \\ 0 & 0 & 0 \\ 0 & 0 & (1-g_1)v[i_1]d[i_2]p \end{bmatrix},$$
$$i_1 = 2 \ldots 8,\ i_2 = 1 \ldots 18.$$

It can be verified that $\sum_{i_1=0}^{8} \sum_{i_2=0}^{18} D_{i_1 i_2} = D$. In this example, we use the following settings for the arrival process and the service process:

$$h_1 = 0.0145,\ g_1 = 0.9,\ h_2 = 0.048,\ g_2 = 0.9,$$
$$p = 0.08,\ \epsilon = 0.05.$$

As a result, the steady probability vector for the arrival process is given by [0.935, 0.05, 0.015]. We have the high priority queue mean arrival rate $\lambda_h = 0.10163$ for the voice and video traffic, and the low priority queue mean arrival rate $\lambda_l = 0.60856$ for the data traffic. The traffic intensity is $\rho = \lambda_h \bar{t}_h + \lambda_l \bar{t}_l = 0.74214$. Therefore, the queue is stable.

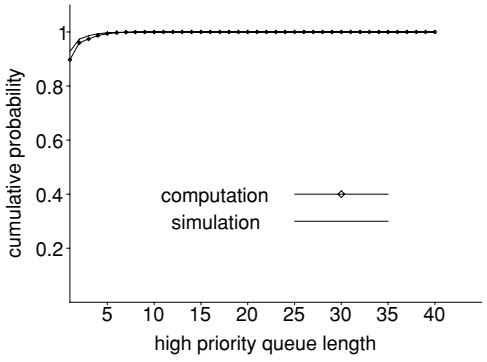

Fig. 5 The high priority queue length distribution function

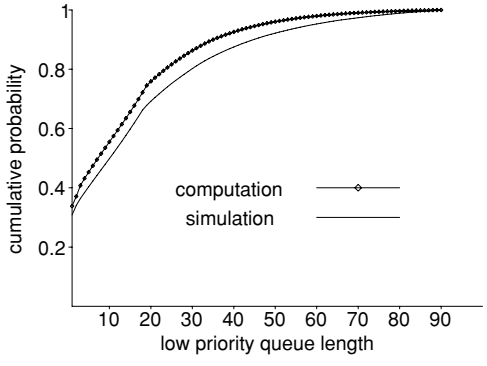

Fig. 6 The low priority queue length distribution function

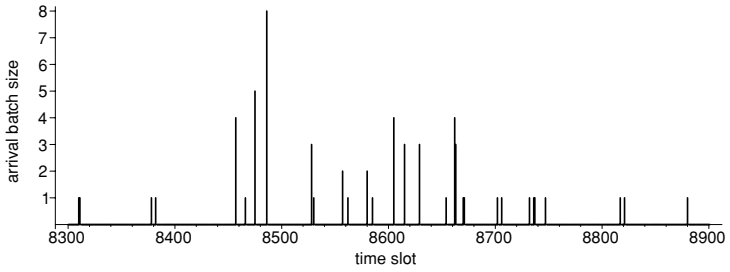

Fig. 7 Arrival patterns for the high priority queue

We apply the computation algorithms developed in Section 4 and obtain the marginal queue length distribution functions as shown in Figure 5 and Figure 6 for the high and low priority queues, respectively. We also simulate the queueing system with the above parameters. The simulated queue length distribution functions are shown together in the above two figures. We observe that the computation results and the simulation results are quite close, especially for the high priority queue. For the low priority queue, discrepancy can arise from truncation on the R measures, as remarked in [1]. We also display part of the simulated arrival patterns in Figure 7 and Figure 8 for the high and low priority queues, respectively. We notice that traffic arrivals for both the high and low priority queues are bursty. In addition, we notice that the high priority arrivals are more correlated, i.e., on average they experience longer OFF periods. The low priority arrival instances are more regular due to the Bernoulli assumption, although the batch size has large variations because they are from data traffic. Finally, we obtain the high priority mean queue length $\overline{Q}_h = 0.11$, and the low priority mean queue length $\overline{Q}_l = 14.6$. By *Little*'s law, we have the high priority queue mean waiting time $\overline{W}_h = 1.08$, and the low priority queue mean waiting time $\overline{W}_l = 23.99$. For this example, the system idle probability $P\{idle\} = \pi_{00}e = 0.25977$.

5.3. A preemptive *DBMAP/PH/*1 priority queue example

In this example, we consider a preemptive *DBMAP/PH/*1 priority queue. The purpose is to examine how large discrepancy could be caused on the resulting queue length distribution when the exact resumption phase is replaced by the limiting resumption phase β_2^*, as introduced in Section 3.2. We use the following parameters for the preemptive *DBMAP/PH/*1 priority queue:

$$S_1 = \begin{bmatrix} 0.0 & 0.05 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.05 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.05 \\ 0.0 & 0.0 & 0.0 & 0.0 \end{bmatrix},$$

$$\beta_1 = [1.0,\ 0.0,\ 0.0,\ 0.0],$$

$$S_2 = \begin{bmatrix} 0.121 & 0.071 & 0.124 & 0.097 & 0.168 & 0.038 & 0.145 & 0.132 \\ 0.160 & 0.093 & 0.140 & 0.081 & 0.123 & 0.103 & 0.078 & 0.018 \\ 0.150 & 0.010 & 0.040 & 0.167 & 0.038 & 0.126 & 0.121 & 0.044 \\ 0.126 & 0.029 & 0.000 & 0.154 & 0.145 & 0.010 & 0.013 & 0.120 \\ 0.166 & 0.042 & 0.065 & 0.030 & 0.147 & 0.007 & 0.015 & 0.023 \\ 0.029 & 0.084 & 0.164 & 0.073 & 0.015 & 0.009 & 0.022 & 0.001 \\ 0.040 & 0.014 & 0.054 & 0.035 & 0.013 & 0.057 & 0.034 & 0.050 \\ 0.007 & 0.023 & 0.041 & 0.036 & 0.049 & 0.005 & 0.001 & 0.034 \end{bmatrix},$$

$$\beta_2 = [1/8,\ 1/8,\ 1/8,\ 1/8,\ 1/8,\ 1/8,\ 1/8,\ 1/8],$$

$$D = \begin{bmatrix} 0.1 & 0.9 \\ 0.2 & 0.8 \end{bmatrix},$$

$D_{00} = (1 - r_1 - r_2)D$, $D_{10} = D_{20} = D_{30} = D_{40} = r_1/4D$, $D_{01} = D_{02} = r_2/2D$ and all other $D_{i_1 i_2} = 0$, $\alpha = [2/11,\ 9/11]$.

Here, the two parameters r_1 and r_2 are utilized to adjust the traffic intensity for the high and low priority queues, respectively. The mean service time $\bar{t}_h$ for the high priority jobs is given by $\bar{t}_h = \beta_1(I - S_1)^{-\mathbf{1}}e = 1.052625$. The mean service time $\bar{t}_l$ for the low priority jobs is given by $\bar{t}_l = \beta_2(I - S_2)^{-\mathbf{1}}e = 2.33$. The limiting probability vector

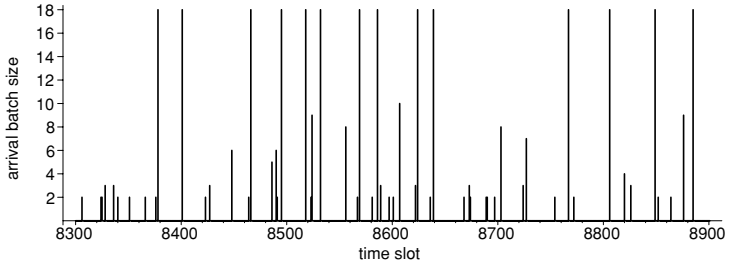

Fig. 8 Arrival patterns for the low priority queue

for the phase preemption is given by

$$\beta_2^* = [0.15971, 0.09903, 0.12966, 0.14091, 0.14925, 0.09734, 0.11118, 0.11291].$$

The average arrival rate for the high and low priority queues are given by $\lambda_h = 2.5r_1$ and $\lambda_l = 1.5r_2$, respectively. We assume that $\lambda_h \bar{t}_h + \lambda_l \bar{t}_l < 1$, or equivalently $2.6315625r_1 + 3.495r_2 < 1$ such that the queue is stable.

We simulate the behavior of the above preemptive *DBMAP/PH/*1 priority queue and observe the cumulative queue length distribution functions (cdfs) for the following two settings of traffic intensity:

$$\{r_1 = 0.20, r_2 = 0.05, \lambda_h = 0.5, \lambda_l = 0.075, \rho_h = 0.5263125, \rho_l = 0.17475\},$$
$$\{r_1 = 0.05, r_2 = 0.20, \lambda_h = 0.125, \lambda_l = 0.30, \rho_h = 0.131578125, \rho_l = 0.699\}.$$

For each setting, we compare the results for the case with the exact preemption phase and for the case with limiting preemption phase β_2^*. The simulation results demonstrate that for the above two preemption rules, the differences in the high priority queue length cdfs are extremely small. This is expected because for a preemptive *DBMAP/PH/*1 priority queue, the preemption rule only affects the behavior of the low priority queue and has no impact on the high priority queue. Therefore, we do not report the simulation results for the high priority queue in this section. For the low priority queue, we observe that for different preemption rules, the differences in the queue length cdfs are very small. The results are shown in Table 2 for $\{r_1 = 0.20,\ r_2 = 0.05\}$, and in Table 3 for $\{r_1 = 0.05,\ r_2 = 0.20\}$. The simulation results, however, do not give a clear indication on whether the traffic intensity of the low priority queue has any impact on the degree of the performance discrepancy. Further study is necessary to investigate this issue in the future.

Table 2 Low priority queue length cdfs for $\{r_1 = 0.20,\ r_2 = 0.05\}$

Queue length	Exact phase	Limiting phase	Queue length	Exact phase	Limiting phase
0	0.676025	0.678363	9	0.999053	0.998664
1	0.814243	0.816150	10	0.999487	0.999250
2	0.906806	0.907062	11	0.999717	0.999526
3	0.949605	0.948945	12	0.999883	0.999701
4	0.973487	0.973033	13	0.999927	0.999799
5	0.985953	0.985604	14	0.999981	0.999887
6	0.992629	0.992410	15	1.0	0.999942
7	0.996146	0.995758	16		0.999979
8	0.998046	0.997733	17		1.0

Table 3 Low priority queue length cdfs for $\{r_1 = 0.05,\ r_2 = 0.20\}$

Queue length	Exact phase	Limiting phase	Queue length	Exact phase	Limiting phase
0	0.330552	0.331740	19	0.992934	0.993506
1	0.470330	0.471671	20	0.994493	0.995081
2	0.577846	0.578842	21	0.995753	0.996284
3	0.664455	0.665371	22	0.996749	0.997118
4	0.733085	0.734002	23	0.997571	0.997806
5	0.787594	0.789151	24	0.998193	0.998372
6	0.831071	0.832922	25	0.998691	0.998846
7	0.865749	0.867868	26	0.999154	0.999195
8	0.893406	0.895800	27	0.999417	0.999491
9	0.914858	0.918094	28	0.999568	0.999671
10	0.932261	0.935583	29	0.999694	0.999816
11	0.946142	0.949702	30	0.999791	0.999899
12	0.957208	0.960935	31	0.999848	0.999942
13	0.966171	0.969558	32	0.999877	0.999959
14	0.973626	0.976357	33	0.999894	0.999971
15	0.979539	0.981681	34	0.999906	0.999979
16	0.984232	0.985759	35	0.999915	0.999987
17	0.987936	0.989082	36	0.999925	0.999998
18	0.990819	0.991518	37	0.999936	1.0

6. Conclusion

In this paper, we defined a discrete-time batch Markovian arrival process (*DBMAP*) with *more than one type of arrival*. We studied the *DBMAP/PH/*1 priority queue, which is a natural extension of the single arrival *DMAP/PH/*1 priority queueing model in [1]. We derived the transition probability matrix for the underlying Markov chain, which is no longer quasi-birth-death (*QBD*) and does not have a matrix-geometric form solution. Instead, the solution to the *DBMAP/PH/*1 priority queue is related to an $M/G/1$-type Markov chain with an infinite number of levels and an infinite number of phases. For such a Markov chain there is no general solution. We developed computational algorithms for the *DBMAP/PH/*1 priority queue and obtained the stationary queue length distributions for both the high and low priority queues. The algorithms are based on matrix-analytic methods and the R measures play a key role in the computation. The structures of R_k and R_{0k} were shown to take upper-triangular and upper-Hessenberg forms, respectively. It is interesting to note that the relationship $R = A_0 + RA_1 + R^2A_2$, or equivalently, $R(I - A_1 - RA_2) = A_0$ for the *DMAP/PH/*1 priority queue in [1], can be considered as a special case of the relationship of all R_k as expressed in (6). Thus, the solution to the *DMAP/PH/*1 priority queue can be regarded as a simplified version of the solution to the *DBMAP/PH/*1 priority queue, when $b_1 = 1$.

The *DBMAP* with a priority arrival process can be particularly useful in modeling multimedia data traffic, especially for compressed video with layered encoding [15]. We

are currently working on modeling MPEG-4 video data encoded in multiple layers by the *DBMAP* process with priorities [17], in which case one layer of the video data traffic is mapped to one priority level. The *DBMAP*/*PH*/1 priority queue has found application in modeling video transmission over wireless networks [18], where the service time of a link layer data burst was shown to follow *PH*-type distribution. The *DBMAP/PH/*1 priority queue can also find application in modeling priority scheduling of traffic in ATM or TDMA networks, since the service time for a single ATM cell or a TDMA time slot is usually assumed to be deterministic, which is a special case of the *PH*-type distribution. Several traditional discrete time queueing systems, such as the *DMAP/PH/*1 priority queue, the single priority *DBMAP/PH/*1 queue, the *DBMAP/D/*1 priority queue, the discrete time $M/M/1$ priority queue, are all special cases of the *DBMAP/PH/*1 priority queue studied in this paper. Future research on calculation of other performance measures such as the waiting time distribution for the *DBMAP*/*PH*/1 priority queue, as well as on applications of the *DBMAP*/*PH*/1 priority queueing model in performance evaluation of computer networks, is needed.

Acknowledgments The authors would like to thank the constructive comments from the editor and the reviewers that improved this paper in both presentation and mathematical development during the review and revision process. The first author of this paper would like to acknowledge the valuable help from Professor Yiqiang Zhao at Carleton University in discussions of the R measures for the $M/G/1$-type Markov chains; the valuable help from Professor Yuhong Dai at the Institute of Computational Mathematics and Scientific/Engineering Computing of the Chinese Academy of Science in discussions of the R truncation and error analysis, and the valuable helps and discussions with A.S. Alfa, W.K. Grassmann and Q.-M. He.

References

1. A.S. Alfa, Matrix-geometric solution of discrete time *MAP/PH/1* priority queue, Naval Research Logistics, 45 (1998) 23–50.
2. C. Blondia and O. Casals, Statistical multiplexing of VBR sources: a matrix-analytic approach, Performance Evaluation, 16 (1992), 5–20.
3. C. Blondia, A discrete time batch Markovian arrival process as B-ISDN traffic model, Belgian Journal of Operations Research, Statistics and Computer Science, 32 (1993), 3–23.
4. K. Claffy, WAN packet size distribution, (1996), http://www.nlanr.net/NA/Learn/packetsizes.html.
5. I. Frigui, A.S. Alfa and X.Y. Xu, Algorithms for computing waiting time distributions under different queue disciplines for the *D-BMAP/PH/1*, Naval Research Logistics, 44 (1997), 559–576.
6. W.K. Grassmann and D.A. Stanford, Matrix analytic methods, in: *Computational probability*, (Kluwer Academic Publishers, Norwell, MA, 1999), 153–203.
7. Q.-M. He, Queues with marked customers, Advances in Applied Probability, 28 (1996), 567–587.
8. Q.-M. He and M.F. Neuts, Markov chains with marked transitions, Stochastic Processes and their Applications, 74 (1998), 37–52.
9. D.M. Lucantoni, K.S. Meier-Hellstern and M.F. Neuts, A single-server queue with server vacations and a class of non-renewal arrival processes, Advances in Applied Probability, 22 (1990), 676–705.
10. D.M. Lucantoni, New results on the single server queue with a batch Markovian arrival process, Stochastic Models, 7 (1991), 1–46.
11. D.M. Lucantoni, The BMAP/G/1 queue: a tutorial, in: *Models and techniques for performance evaluation of computer and communication systems*, (Springer-Verlag, 1993), 330–358.
12. D.G. Miller, Computation of steady-state probabilities for M/M/1 priority queues, Operations Research, 29 (1981), 945–958.
13. M.F. Neuts, *Matrix-geometric solutions in stochastic models - an algorithmic approach*, (The Johns Hopkins University Press, 1981).
14. M.F. Neuts, *Structured stochastic matrices of M/G/1 type and their applications*, (Marcel Dekker Inc., New York, 1989).
15. N. Shacham, Multipoint communication by hierarchically encoded data, in: *IEEE Infocom'02*, Florence, Italy, 3 (1992), 2107– 2114.
16. H. Takagi, *Queueing analysis: a foundation of performance evaluation, Vol 1: vacation and priority systems*, (North-Holland, 1993).
17. J.-A. Zhao, B. Li and I. Ahmad, Traffic model for layered video: an approach on Markovian arrival process, in: *Packet Video 2003*, Universite de Nantes, Nantes, France, (2003).
18. J.-A. Zhao, B. Li, C.-W. Kok and I. Ahmad, MPEG-4 video transmission over wireless networks: a link level performance study, Wireless Networks, 10 (2004), 133–146.
19. Y.Q. Zhao, Censoring technique in studying block-structured Markov chains, in: *The Third International Conference on Matrix Analytic Methods in Stochastic Models*, Leuven, Belgium, (2000).