Algorithmic approach to Markovian multi-server retrial queues with vacations

Yang Woo Shin

Department of Statistics, Changwon National University, Changwon, Gyeongnam 641-773, Republic of Korea

**Abstract**

There are many practical situations that have both features of customer retrials and server vacations. The vacation policy is characterized by the vacation startup rules and vacation termination rules. The queueing system with retrials and vacations has been described in variety of ways according to the vacation rules and retrial policies such as constant retrial policy and linear retrial policy and analyzed for each specific model. In this paper, we model the Markovian multi-server queues with customers’ retrials and vacations with level dependent quasi-birth-and-death (LDQBD) process in a unified manner and present an algorithmic approach to compute the performance measures of the system.

© 2014 Elsevier Inc. All rights reserved.

1. Introduction

Queueing system with repeated customers, called retrial queue consists of an orbit with infinite capacity and a service facility that has finite servers and waiting space of finite capacity. An arriving customer enters the service facility if the service facility is not full, otherwise the customer joins orbit and repeats its request after random amount of time, called retrial time until the customer gets into the service facility. Retrial queue has been widely used for modeling and analyzing many practical problems arising in computer and communication systems. Exact analytical results have been focused on some special cases, e.g. see [12,23,25]. Many approximations and algorithmic approaches have been presented for the multi-server retrial queues. The detailed overviews of the related references with retrial queues can be found in the monographs Artalejo and Gómez-Corral [4], Falin and Templeton [12] and references therein.

Queueing systems in which some of servers may not be unavailable have been modeled as a vacation queue. The time period that the servers do not provide their service is considered as a vacation time. The vacation queue reflects the situation that servers’ working on supplementary jobs and have been studied extensively for modeling and analyzing the practical problems such as computer and communications systems, manufacturing systems and call centers with multitask employees. Many efforts have been devoted to derive the stationary distribution of the system and decomposition property in single server vacation queues, e.g. see Takagi [30] and references therein. The interests on vacation queues with multiple servers have been increased for the last decade. Many Markovian multi-server vacation queues are analyzed by matrix geometric method e.g. Chakravarthy [8] and one can refer the recent monograph Tian and Zhang [31] and references therein for detailed overview of the multi-server vacation queues.

There are many practical situations that have both features of customer retrials and server vacations. For example, consider a situation arising in call center with multiple agents that answer the customer calls. If an agent is available upon arrival of a customer call (inbound call), the call is served immediately. If all the agents are busy with other calls, the arriving
customer puts on hold and wait in buffer until an agent becomes available. If all the lines (buffer) are seized with other calls, the customer will hang-up and retry to access the call center after random amount of time. Management would like to increase the efficiency of agents and wants some of idle agents to work secondary job like outbound calls. This type of call center with multi-task agents can be modeled by the queueing systems with repeated customers and server vacations.

Retrial queues and vacation queues have been studied separately for last several decades. Recently, the interests on the queues with vacations and retrials are increasing rapidly. However, the analysis focused on the system with single-server and/or constant retrial policy that the only one customers in orbit can retry at once e.g. see [2,3,6,10,14]. The \( M/G/1 \) retrial queue with linear retrial policy and Bernoulli vacation schedule is considered by Choudhury [9] and a matrix geometric solution for the \( M/M/c \) retrial queue with constant retrial rate and Bernoulli vacation schedule is given by Kummar et al. [15].

The vacation policy is characterized by the vacation startup rules and vacation termination rules. The queueing system with retrials and vacations have been described in variety of ways according to the vacation rules and retrial policies such as constant retrial policy and linear retrial policy and analyzed for each specific model. In this paper, we model the Markovian multi-server queues with customers retrials and vacations with level dependent quasi-birth-and-death (LDQBD) process in a unified manner and present an algorithmic approach to compute the performance measures of the system. The call center with after-call work introduced by Phung-Duc and Kawanishi [20,22] and the call center with outgoing calls in Phung-Duc et al. [21] also can be formulated using the LDQBD framework as in this paper.

In Section 2, we show that multi-server retrial queue with vacations under various vacation policies can be modeled by LDQBD process with special structure of block matrix components. Algorithm for the stationary distribution of the LDQBD process frequently arising in modeling the queueing systems with retrials and vacations is presented in Section 3. Some numerical results and concluding remarks are given in Section 4.
distribution of a customer in orbit is of exponential with rate $\gamma$ and is independent of others. Thus, the retrial rate of customers is $\gamma_n = n\gamma$ when there are $n$ customers in orbit. The service time distribution of a customer is of exponential with rate $\mu$. We adopt the following vacation policy called $(a, b)$-vacation policy that is introduced by Xu and Zhang [32]. If any $a (1 \leq a \leq c)$ or more servers are idle at a service completion, that is, the number of customers at the service facility is less than or equal to $a^* = c - a$ upon a service completion, then $b (b \leq a)$ servers among idle servers take a vacation and the remaining $b^* = c - b$ servers are available. If $a = b = c$, then this policy is synchronous vacation policy. The vacation time distribution is assumed to be of phase type $PH(\delta, V)$, where $\delta = (\delta_1, \ldots, \delta_w)$ with $\delta e = 1$ and $V = (v_{ij})$ is a nonsingular $w \times w$ matrix with $v_{ii} = -\delta_i < 0$, $1 \leq i \leq w$. Let $V^T = -Ve = (v^T_1, \ldots, v^T_w)$ and $m_e = (\delta(U)^{-1}e$ the mean vacation time. For detailed description of the $PH$-distribution, see Neuts [18, Chapter 2]. We consider single vacation policy under which the servers take only one vacation and after the vacation the servers either serves the waiting customer in service facility if any or stays idle in service facility.

Let $X(t)$ be the number of customers in orbit and $Y(t)$ be the number of customers in service facility at time $t$. Denote by $J_a(t)$ the phase of arrival process and let $J_i(t)$ be the server state at time $t$ defined by $J_i(t) = \begin{cases} 0, & \text{c servers are available} \\ j, & \text{the phase of vacation time is of } j, \quad 1 \leq j \leq w. \end{cases}$

The stochastic process $Z = \{Z(t), t \geq 0\}$ with $Z(t) = (X(t), Y(t), J_a(t), J_i(t))$ is a continuous time Markov chain on the state space $S = \cup_{n=0}^{\infty}n$, where $n = \{(n, k, i, j) : 0 \leq k \leq K, 1 \leq i \leq l, 0 \leq j \leq w\}$. $n \geq 0$ and the generator of the Markov chain $Z$ is of the form (2.1). The block matrix components of $A(n)$ and $C(n)$ of the generator $Q$ of $Z$ are given as

$$A_k^{(n)} = D \otimes I_{w+1}, \quad C_k^{(n)} = \gamma_n I_{w+1}, \quad 0 \leq k \leq K - 1,$$

where $I_m$ is the identity matrix of size $m$ and $A \otimes B$ is the Kronecker product of $A$ and $B$. The upper and lower diagonal blocks of $B^{(n)}$ are given by

$$B_{k,k}^{(n)} = D \otimes I_{w+1}, \quad 0 \leq k \leq K - 1$$

and

$$B_{k,k-1}^{(n)} = \begin{cases} I_l \otimes M_0(\mu), & 1 \leq k \leq a^* + 1 \\ I_l \otimes M_1(\mu), & a^* + 1 < k \leq K, \end{cases}$$

where $M_0(\mu) = \begin{pmatrix} 0 & \min(k, c)\mu \delta \\ 0 & \min(k, b^*)\mu I_w \end{pmatrix}$, $M_1(\mu) = \begin{pmatrix} \min(k, c)\mu & 0 \\ 0 & \min(k, b^*)\mu I_w \end{pmatrix}$.

The diagonal blocks of $B^{(n)}$ are given by

$$B_k^{(n)} = C \otimes I_{w+1} + I_l \otimes V^* - \Delta_k^{(n)}, \quad 0 \leq k \leq K,$$

where $\Delta_k^{(n)}$ is the diagonal matrix that makes $Qe$, that is,

$$\Delta_k^{(n)} = \begin{pmatrix} \Delta_k^{(n)}(B_{01}^{(n)} + \gamma_n I_{w+1}) & 0 \\ 0 & \Delta_k^{(n)} \end{pmatrix}, \quad \Delta_k^{(n)} = \begin{pmatrix} \Delta_k^{(n)}(B_{11}^{(n)} + \gamma_n I_{w+1}) & 0 \\ 0 & 0 \end{pmatrix}, \quad 0 \leq k \leq K - 1,$$

and $\Delta_k^{(n)}$ is the diagonal matrix whose diagonal vector is $x$, and $V^* = \begin{pmatrix} 0 \\ v^T \\ 0 \end{pmatrix}$.

2. MAP/M/c/K retrial queue with PH-vacation time under $(a, b)$-vacation policy and multiple vacation. Consider the multiple vacation policy under which the servers keep taking vacation until the vacation termination rule is satisfied. That is, if the number of customers in service facility is less than or equal to $a^* = c - a$ upon returning of servers from vacation, then the servers take another vacation. By $X(t)$, $Y(t)$, $J_a(t)$ and $J_i(t)$ denote the same as those in the single vacation system and let $Z_M$ be the stochastic process corresponding to $Z$ of the system with single vacation. Under the multiple vacation policy, $Z_M$ cannot take the state $(n, k, i, 0)$ for $k \leq a^*$ and the state space of $Z_M$ is $S_M = \cup_{n=0}^{\infty}n$, where $n = \{(n, k, i, j) : 0 \leq k \leq a^*, 1 \leq i \leq l, 1 \leq j \leq w\}$. $n \geq 0$ and the state space of $Z_M$ is of the form (2.1) and the matrix components of $A(n)$ and $C(n)$ are $A_k^{(n)} = D \otimes I_{w+1}, \quad C_k^{(n)} = \gamma_n I_{w+1}, \quad 0 \leq k \leq K - 1$ and the upper blocks of $B^{(n)}$ are

$$B_{k,k}^{(n)} = D \otimes I_{w+1}, \quad 0 \leq k \leq a^* - 1$$

and

$$B_{k,k-1}^{(n)} = \begin{cases} D \otimes I_{w+1}, & k = a^* \\ D \otimes I_{w+1}, & a^* + 1 < k \leq K - 1, \end{cases}$$
where $\mathbf{0} : I_w$ is the $w \times (w + 1)$ matrix whose first column is zero vector and remaining $w$ columns are the same as those of identity matrix $I_w$. The lower diagonal blocks $B_{kk-1}^{(n)}$ of $B^{(n)}$ are as follows:

$$
B_{kk-1}^{(n)} = \begin{cases}
\min(k,b^*)\mu I_{w}, & 1 \leq k \leq a' \\
I_k \otimes \left( \begin{array}{c}
\min(k,c)\mu I_{w} \\
\min(k,b^*)\mu I_{w}
\end{array} \right), & k = a' + 1 \\
I_k \otimes \left( \begin{array}{c}
\min(k,c)\mu 0 \\
0 \min(k,b^*)\mu I_{w}
\end{array} \right), & a' + 2 \leq k \leq K.
\end{cases}
$$

The diagonal blocks of $B^{(n)}$ are given by

$$
B_{kk}^{(n)} = \begin{cases}
C \otimes I_w + I_k \otimes (V + V^0 \delta) - \Delta_k^{(n)}, & 0 \leq k \leq a' \\
C \otimes I_{w+1} + I_k \otimes V^* - \Delta_k, & a'+1 \leq k \leq K,
\end{cases}
$$

where $\Delta_k^{(n)}$ is the diagonal matrix that makes $Qe = 0$ which are given by

$$
\Delta_k^{(n)} = \begin{cases}
\Delta \left[ B_{01}^{(n)} e + \gamma_u I_{w}e \right], & k = 0, \\
\Delta \left[ B_{kk-1}^{(n)} e + B_{kk+1}^{(n)} e + \gamma_u I_{w}e \right], & 1 \leq k \leq a', \\
\Delta \left[ B_{kk-1}^{(n)} e + B_{kk+1}^{(n)} e + \gamma_u I_{w+1}e \right], & a' + 1 \leq k \leq K - 1, \\
\Delta \left[ B_{KK-1} e + A_k^{(n)} e \right], & k = K.
\end{cases}
$$

3. $M/M/c/K$ retrial queue with asynchronous single vacation. Consider the $M/M/c/K$ retrial queue in which there are $c$ parallel servers and an waiting space of size $K - c$. We assume that the interarrival time of customers from outside, service time of each serve, and retrial time of each customer in orbit are all exponential whose rates are $\lambda$, $\mu$ and $\gamma$, respectively. The servers take a vacation under the asynchronous vacation (AS) policy under which any of $c$ servers in service facility starts a vacation independently if this server finds no waiting customer in the system at his or her service completion instant. Assume the vacation time is exponential with rate $\gamma$ and single vacation policy. Let $X(t)$ be the number of customers in orbit and $Y(t)$ be the number of customers in service facility at time $t$ and denote by $f(t)$ the number of servers in vacation at time $t$. The stochastic process $Z_{AS} = \{Z(t), t \geq 0\}$ with $Z(t) = (X(t), Y(t), f(t))$ is a continuous time Markov chain on the state space $S_{AS} = \{(n,k,j) : n \geq 0, 0 \leq k \leq K, 0 \leq j \leq c\}$ and the generator of $Z$ is of the form as (2.1). The block matrix components of $A^{(n)}$ and $C^{(n)}$ of the generator of $Z_{AS}$ are given as $A_k^{(n)} = \lambda I_{c+1}$, $C_k^{(n)} = \gamma_u I_{c+1}$, $0 \leq k \leq K - 1$ and $B_{kk+1}^{(n)} = \lambda I_{c+1}$, $0 \leq k \leq K - 1$. For $1 \leq k \leq c$,

$$
[B_{kk-1}^{(n)}]_{ij} = \begin{cases}
k\mu, & 0 \leq i \leq c - k, \quad j = i + 1 \\
(c - i)\mu, & c - k + 1 \leq i \leq c, \quad j = i, \\
0, & \text{otherwise}
\end{cases}
$$

and $B_{kk-1}^{(n)} = \Delta(\min(k,c - j)\mu, j = 0, \ldots, c)$, $c + 1 \leq k \leq K$. The diagonal blocks of $B^{(n)}$ are given by

$$
[B_{kk}^{(n)}]_{ij} = \begin{cases}
iv, & j = i - 1, \quad 0 \leq i \leq c, \\
-\Delta_k^{(n)}, & j = i, \quad 0 \leq i \leq c, \\
0, & \text{otherwise}
\end{cases}
$$

where $\Delta_k^{(n)}$ is the positive number that makes $Qe = 0$.

4. $M/M/c/K$ retrial queue with asynchronous multiple vacations. Denote by $X(t)$, $Y(t)$ and $f(t)$, the same as those in the system with single vacation policy and let $Z_{AM} = \{Z(t), t \geq 0\}$ with $Z(t) = (X(t), Y(t), f(t))$. In this case, $Y(t) \geq c - f(t)$ and the state space of $Z_{AM}$ is

$$
S_{AM} = \{(n,k,j) : n \geq 0, 0 \leq k \leq K, \max(0,c - k) \leq j \leq c\}.
$$

It can be easily seen that the generator of $Z_{AM}$ is of the form (2.1) with matrix components as follows: $A_k^{(n)} = \lambda I_{c+1}$, $C_k^{(n)} = \gamma_u I_{c+1}$, $0 \leq k \leq K - 1$. The block matrix components of $B^{(n)}$ are given as follows:

$$
B_{kk+1}^{(n)} = \lambda I_{\min(c,k)+1}, \quad 0 \leq k \leq K - 1,
$$

$$
B_{kk-1}^{(n)} = \Delta((c - i)\mu, i = 0, \ldots, c), \quad c + 1 \leq k \leq K
$$

and for $1 \leq k \leq c$,
\[
[B^{(n)}_{k,i}] = \begin{cases} 
\min(k, c-i) \mu, & i = c - k, \ j = i + 1 \\
\min(k, c-i) \mu, & c - k < i \leq c, \ j = i, \\
0, & \text{otherwise}
\end{cases}
\]

and for \(0 \leq k \leq K,\)
\[
[B^{(n)}_{k,i}] = \begin{cases} 
(c - i) \nu, & j = i - 1, \ \max(0, c - k) \leq i \leq c, \\
-\Delta^{(n)}_k, & j = i, \ \max(0, c - k) \leq i \leq c, \\
0, & \text{otherwise}
\end{cases}
\]

where \(\Delta^{(n)}_k\) is the positive number that makes \(Qe = 0.\)

### 3. Algorithm for stationary distribution

Let \(X\) be an LDQBD process with generator \(Q\) of the form (2.1) and assume that \(X\) is irreducible and positive recurrent. Let \(x = (x^{(0)}, x^{(1)}, x^{(2)}, \ldots)\) with \(x^{(n)} = (x_0^{(n)}, \ldots, x_c^{(n)}), \ n \geq 0\) be the stationary distribution of \(Q.\) It is well known (e.g. see [7,16]) that the stationary distribution \(x\) is given by
\[
x^{(n)} = x^{(0)} \left( \prod_{k=1}^n R^{(k)} \right), \ n = 0, 1, \ldots,
\]
where the matrices \(\{R^{(n)}, n \geq 0\}\) are the minimal nonnegative solutions to the systems of equations
\[
A^{(n-1)} + R^{(0)} B^{(n)} + R^{(n)} R^{(n+1)} C^{(n+1)} = 0, \ n \geq 1
\]
and \(x^{(0)}\) is the unique solution of the equation
\[
x^{(0)} B^{(0)} R = 0,
\]
with the normalizing condition
\[
x^{(0)} \left[ e + \sum_{n=1}^\infty \left( \prod_{k=1}^n R^{(k)} \right) e \right] = 1.
\]

It follows from the special structure of the matrix \(A^{(n)}\) that
\[
R^{(n)} = A^{(n-1)} \left[ - \left( B^{(0)} + R^{(n+1)} C^{(n+1)} \right) \right]^{-1}
\]
has the following formula
\[
R^{(n)} = \left( R_0^{(n)} R_1^{(n)} \ldots R_K^{(n)} \right), \ n \geq 1.
\]
where \(O\) is the zero matrix of appropriate size and \(R_k^{(n)}\) is the matrix of size \(m_k \times m_k, \ k = 0, 1, \ldots, K.\) Thus the \(k\)th block \(x_k^{(n)}\) of \(x^{(n)}\) is given by for \(n \geq 1,\)
\[
x_k^{(n)} = x_k^{(0)} p_k^{(n)}, \ \ k = 0, 1, \ldots, K,
\]
where \(p_k^{(1)} = R_k^{(1)}\) and for \(n \geq 2,\)
\[
p_k^{(n)} = \left( \prod_{i=1}^{n-1} R_k^{(i)} \right) R_k^{(n)}, \ \ k = 0, 1, \ldots, K
\]
and the normalizing condition (3.3) becomes
\[
\sum_{k=0}^K x_k^{(0)} e + x_0^{(0)} \left( e + \sum_{n=1}^\infty \sum_{j=0}^K p_j^{(n)} e \right) = 1.
\]
been presented, e.g. heuristic methods using asymptotic formulae for simple models [20,22] and the method based on trial and error [7,19,28] that increase the level until a satisfactory accuracy is obtained. Since the method of determining truncation point may depend on the specific system, we focus on computing the rate matrices given a truncation level.

One of the main difficulties in matrix analytic method is to derive rate matrix for which it requires to make an inversion of many matrices of large size especially for the system with retrials. The LDQBD process (2.1) with matrix components \( \mathbf{A}^{(n)}, \mathbf{B}^{(n)} \) and \( \mathbf{C}^{(n)} \) of special formula are considered by [26,28] for scalar entries and [21] for the case of matrix form entries. The approach given below for computing rate matrices is based on the method in [28] and is similar to that of [21]. The difference between [21] and our approach is the way to compute the rate matrix at the truncation level, the former use an iteration method and the latter directly invert the matrices using Algorithm A1. Now we derive an algorithm for \( \mathbf{x}^{(n)}, n = 0, 1, \ldots \)

**Computation of \( \mathbf{R}^{(n)} \).** Write the matrix \( \mathbf{B}^{(n)}|\mathbf{R}| = \mathbf{B}^{(n)} + \mathbf{R}^{(n+1)}\mathbf{C}^{(n+1)} \) in the block form as

\[
\mathbf{B}^{(n)}|\mathbf{R}| = \begin{pmatrix} \mathbf{B}_{00}^{(n)} & \mathbf{B}_{01}^{(n)} \\ \mathbf{B}_{10}^{(n)} & \mathbf{B}_{11}^{(n)} \end{pmatrix},
\]

where

\[
\mathbf{B}_{00}^{(n)} = \begin{pmatrix} \mathbf{B}_{00}^{(0)} & \mathbf{B}_{01}^{(0)} \\ \mathbf{B}_{10}^{(0)} & \mathbf{B}_{12}^{(0)} \\ \cdots & \cdots & \cdots \\ \mathbf{B}_{n-K-2}^{(n)} & \mathbf{B}_{n-K-2,K-2}^{(n)} & \mathbf{B}_{n-K-2}^{(n)} \\ \mathbf{B}_{n-K-1}^{(n)} & \mathbf{B}_{n-K-1,K-2}^{(n)} & \mathbf{B}_{n-K-1}^{(n)} \\ \mathbf{B}_{n-K}^{(n)} & \mathbf{B}_{n-K,K-2}^{(n)} & \mathbf{B}_{n-K}^{(n)} \end{pmatrix},
\]

and the block matrix component of \( \mathbf{B}_{00}^{(n)} \) is

\[
[\mathbf{B}_{00}^{(n)}]_{j} = \begin{cases} O_{m_{0},m_{0}}, & j = 0 \\ \mathbf{B}_{j-1}^{(n+1)}C_{j-1}^{(n+1)}, & 1 \leq j \leq K - 2 \\ \mathbf{R}_{K,j-1}^{(n+1)} + \mathbf{B}_{K,K-1}^{(n+1)}C_{K-2}^{(n+1)}, & j = K - 1 \end{cases}
\]

and

\[
\mathbf{B}_{11}^{(n)} = \mathbf{B}_{K,K}^{(n)} + \mathbf{R}_{K-1}^{(n+1)}C_{K-1}^{(n+1)}.
\]

Write the inverse matrix of \( \mathbf{B}^{(n)}|\mathbf{R}| \) in the block form as

\[
(\mathbf{B}^{(n)}|\mathbf{R}|)^{-1} = \begin{pmatrix} \mathbf{D}_{00}^{(n)} & \mathbf{D}_{01}^{(n)} \\ \mathbf{D}_{10}^{(n)} & \mathbf{D}_{11}^{(n)} \end{pmatrix}.
\]

and let \( \mathbf{X}_{ij}^{(n)} \) be the \((i,j)\)-block matrix component of the matrix \((-\mathbf{B}_{00}^{(n)})^{-1} = (\mathbf{X}_{ij}^{(n)}) \). Then it can be seen from Horn and Johnson [13, page 18] that

\[
\mathbf{D}_{11}^{(n)} = (\mathbf{B}_{11}^{(n)} + \mathbf{B}_{10}^{(n)}(-\mathbf{B}_{00}^{(n)})^{-1}\mathbf{B}_{01}^{(n)})^{-1}
\]

and for \( j = 0, 1, \ldots, K - 1 \),

\[
[\mathbf{D}_{11}^{(n)}]_{j} = \left([\mathbf{D}_{11}^{(n)}]_{j} - \mathbf{R}_{j}^{(n+1)}\mathbf{X}_{ij}^{(n)}\right)^{-1} = \mathbf{D}_{11}^{(n)}\left(\mathbf{B}_{K,j-1}^{(n)}\mathbf{X}_{K,j-1}^{(n)} + \sum_{i=0}^{K-2} \mathbf{R}_{i}^{(n+1)}\mathbf{X}_{i+1,j}^{(n)}\right)
\]

The \( j \)th block matrix component \( \mathbf{R}_{j}^{(n)} \) of \( \mathbf{R}^{(n)} \) is given by

\[
\mathbf{R}_{j}^{(n)} = \begin{cases} \mathbf{A}_{j}^{(n)}(-\mathbf{D}_{10}^{(n)}), & j = 0, 1, \ldots, K - 1 \\ \mathbf{A}_{K}^{(n)}(-\mathbf{D}_{11}^{(n)}), & j = K \end{cases}
\]

Now we present an algorithm for calculating \( \mathbf{x}^{(n)} \). Let \( \mathbf{r} = e + \sum_{i=0}^{K-1} \mathbf{r}_{i}^{(n)} \) and let \( (\mathbf{e}, \mathbf{r}) \) be the column vector of size \( \sum_{k=0}^{K} m_{k} \) whose first \( \sum_{k=0}^{K-1} m_{k} \) components are all 1 and the last \( m_{k} \) components are the same as \( \mathbf{r} \). Let \( \mathbf{B}^{(0)}|\mathbf{R}| \) be the matrix obtained by replacing the last column of \( \mathbf{B}^{(0)}|\mathbf{R}| \) with \((\mathbf{e}, \mathbf{r}) \). It follows from (3.2) and (3.7) that

\[
\mathbf{x}^{(0)} = (0, \ldots, 0, 1)\left(\mathbf{B}^{(0)}|\mathbf{R}|\right)^{-1}.
\]
In the following, we derive an algorithm for computing (3.11). Let $E_k$ be the $m_k \times m_k$ matrix whose $m_k$th column vector is $e$ and others are all zero. Noting that $\hat{B}^{0\mid R}$ is of the form

$$\hat{B}^{0\mid R} = \begin{pmatrix} B^{0}_{00} & B^{0}_{01} \\ B^{0}_{10} & B^{0}_{11} \end{pmatrix},$$

where

$$\hat{B}^{0}_{01} = \begin{pmatrix} E_0 \\ E_1 \\ \vdots \\ E_{K-1} \\ \tilde{B}^{0}_{K-1K} \end{pmatrix},$$

and $\tilde{B}^{0}_{K-1K}$ is obtained by replacing the last column of $B^{0}_{K-1K}$ with $e$ and $\hat{B}^{0}_{11}$ is obtained by replacing the last column of $B^{0}_{11}$ with $r = (r_1, \ldots, r_m)$, that is,

$$[\hat{B}^{0}_{K-1K}]_j = \begin{cases} \left[ B^{0}_{K-1K} \right]_j & 1 \leq i \leq m, \ 1 \leq j \leq m - 1 \\ 1, & 1 \leq i \leq m, \ j = m \end{cases}$$

and

$$[\hat{B}^{0}_{11}]_j = \begin{cases} \left[ B^{0}_{11} \right]_j & 1 \leq i \leq m, \ 1 \leq j \leq m - 1 \\ r_i, & 1 \leq i \leq m, \ j = m \end{cases}$$

It follows from (3.11) that we need only the last row of $(\hat{B}^{0\mid R})^{-1}$. Writing

$$\left( \hat{B}^{0\mid R} \right)^{-1} = \begin{pmatrix} \hat{D}^{0}_{00} & \hat{D}^{0}_{01} \\ \hat{D}^{0}_{10} & \hat{D}^{0}_{11} \end{pmatrix},$$

and $(-B^{00})^{-1} = \left( X^{i}_{ij} \right)_{0 \leq i \leq K - 1}$ in the block matrix form, the last blocks $\hat{D}^{0}_{11}$ and $\hat{D}^{0}_{10}$ are given as follows:

$$\hat{D}^{0}_{11} = \left( \hat{D}^{0}_{11} + B^{0}_{10}(-B^{00})^{-1}\hat{B}^{0}_{01} \right)^{-1},$$

(3.12)

where $B^{0}_{10}(-B^{00})^{-1}\hat{B}^{0}_{01}$ is given by

$$B^{0}_{10}(-B^{00})^{-1}\hat{B}^{0}_{01} = \sum_{j=0}^{K-2} \sum_{i=0}^{K-2} R^{(1)}_{i} c_{i}^{(1)} X^{(0)}_{i+1,j} E_j + B^{0}_{K-1} \sum_{j=0}^{K-2} X^{(0)}_{K-1,j} E_j + \sum_{j=0}^{K-2} R^{(1)}_{j} c_{j}^{(1)} X^{(0)}_{j+1,K-1} \hat{B}^{0}_{K-1K} + B^{0}_{K-1} X^{(0)}_{K-1,K-1} \hat{B}^{0}_{K-1K}$$

and the $j$th block matrix of $\hat{D}^{0}_{10}$ is

$$\left[ \hat{D}^{0}_{10} \right]_j = \left[ \hat{D}^{0}_{11} B^{0}_{10} (-B^{00})^{-1} \right]_j = \hat{D}^{0}_{11} \left( B^{0}_{K-1,K-1} X^{0}_{K-1,j} + \sum_{i=0}^{K-2} R^{(1)}_{i} c_{i}^{(1)} X^{0}_{i+1,j} \right), \quad j = 0, 1, \ldots, K - 1.$$

(3.13)

The $k$th block vector $x^{(0)}_{k}$ of $x^{(0)}$ is

$$x^{(0)}_{k} = \begin{cases} e_m & k = 0, 1, \ldots, K - 1 \\ e_{m}(-\hat{D}^{0}_{11}) & k = K, \end{cases}$$

(3.14)

where $e_{n} = (0, \ldots, 0, 1)$ is an $m_k$-dimensional vector whose $m_k$th component is 1 and others are all zero.

Summarizing the results above, we have the algorithm for $x$ as follows.

**Algorithm for $x$.**

**Step 0:** Choose an initial truncation level $N$.

**Step 1:** Let $B = B^{(N)} + A^{(N)}$ and compute

$$R^{(N)}_j = A^{(N-1)}(-B^{(N)})^{-1} \hat{x}^{(N)}_{kj}, \quad j = 0, 1, \ldots, K,$$

where $(-B^{(N)})^{-1} = \left( x^{(N)}_{ij} \right)$ can be obtained by using Algorithm A1 in Appendix A.1.

**Step 2:** For $n = N - 1, N - 2, \ldots, 2, 1, R^{(n)}_j$ using (3.10), where $(-B^{(n)})^{-1} = \left( x^{(n)}_{ij} \right)$ can be obtained by using Algorithm A1 in Appendix A.1.
Step 3: Compute \( x^{(0)} \) using (3.14).
Step 4: Compute \( x^{(n)} \) using (3.6), that is, \( x^{(n)}_{k} = x^{(0)}_{k} p^{(n)}_{k} \), \( k = 0, 1, \ldots, K \), \( 1 \leq n \leq N \).
Step 5: Stopping Criterion. If \( x^{(n)} e < \epsilon \), then stop iteration. Otherwise, increase the truncation level \( N \) and go to step 1.

4. Numerical examples

In this section, the algorithm presented in the previous section is implemented for two systems described in Section 2.

1. MAP/M/c/K retrial queue with PH-vacation time under \((a, b)\)-vacation policy and single vacation. We assume \( a > 2 \) and the stability condition \( \rho = \frac{\mu}{\sigma} < 1 \) of the Markov chain \( Z \) (see Appendix A.2). Let \( x^{(n)}(i,j) = P(X = n, Y = k, J_n = i, J_i = j), (n, k, i, j) \in S \) be the stationary distribution of \( Z \) and set \( x^{(n)} = \{x^{(n)}_{1}, \ldots, x^{(n)}_{K}\} \) with \( x^{(n)}_{k} = x^{(n)}_{k}(i,j) \), \( 1 \leq i \leq l, 0 \leq j \leq w \). Once the stationary distribution \( x = (x^{(n)}_{n}, n \geq 0) \) with truncation level \( N \) is obtained, the marginal distributions \( x_n = P(X = n) \) and \( y_k = P(Y = k) \) of \( X(t) \) and \( Y(t) \) in stationary state are given by

\[
x_n = \sum_{k=0}^{K} x^{(n)}_k e, \quad n = 0, 1, 2, \ldots, N, \\
y_k = \sum_{n=0}^{N} x^{(n)}_k e, \quad k = 0, 1, 2, \ldots, K,
\]

and performance measures such as the mean number \( L_0 = E[X] \) of customers in orbit and the mean number \( L_1 = E[Y] \) of customers in service facility, the blocking probability \( P_B = P(Y = K) \) and the probability \( P_V = P(J_i \geq 1) \) that the servers are in vacation state can be calculated as follows:

\[
L_0 = \sum_{n=0}^{N} nx_n, \quad L_1 = \sum_{k=0}^{K} ky_k, \quad P_B = y_k, \quad P_V = \sum_{n=0}^{N} \sum_{k=0}^{K} \sum_{l=1}^{w} \sum_{j=1}^{w} x^{(n)}_{k}(i,j).
\]

Table 1 lists the numerical results for MAP/M/20/K retrial queue with vacations under \((a, b)\) with \( a = 10 \), \( b = 6 \) vacation policy and single vacation. To show the effectiveness and feasibility of algorithm to the case of large \( K \) and/or highly congested system, we fix the service rate \( \mu = 1.0 \) and mean vacation time \( m_v = 1.5 \) and consider the following combinations \( \rho = 0.8, 0.9, 0.95, \gamma = 0.1, 0.5, 1.0, 5.0, 10.0, \text{ and } K = c, 1.5c \). We choose the tolerance \( \epsilon = 10^{-3} \) for stopping criterion. For given traffic intensity \( \rho \), we use MAP(C,D) with mean \( m_a = 1.0/(c/\mu \rho) \) as an arrival process, where

\[
C = \begin{pmatrix}
-\lambda_1 & 0 & 0 \\
0 & -\lambda_1 & 0 \\
0 & 0 & -\lambda_2
\end{pmatrix}, \quad D = \begin{pmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & p & 0 \cdot (1-p)\lambda_2
\end{pmatrix},
\]

with \( p = 0.751282 \) and \( \lambda_1 = -2.88098/m_a, \lambda_2 = 2.09007/m_a \) which is used for fitting the first three moments of the Weibull distribution with mean \( m_v \) and squared coefficient of variation 0.5 [29]. For the vacation time with mean \( m_v \), we choose PH(\( \delta, V \)) with \( \delta = (d, 1-d, 0, 0) \) with \( d = 0.1167466452506409 \) and

<table>
<thead>
<tr>
<th>( \rho )</th>
<th>( \gamma )</th>
<th>( K = 20 )</th>
<th>( K = 30 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( L_0 )</td>
<td>( L_1 )</td>
<td>( P_B )</td>
<td>( P_V )</td>
</tr>
<tr>
<td>0.8</td>
<td>0.1</td>
<td>16.70</td>
<td>16.26</td>
</tr>
<tr>
<td>0.5</td>
<td>4.096</td>
<td>16.25</td>
<td>0.1216</td>
</tr>
<tr>
<td>1.0</td>
<td>2.535</td>
<td>16.24</td>
<td>0.1318</td>
</tr>
<tr>
<td>5.0</td>
<td>1.271</td>
<td>16.26</td>
<td>0.1728</td>
</tr>
<tr>
<td>10.0</td>
<td>1.094</td>
<td>16.27</td>
<td>0.1930</td>
</tr>
<tr>
<td>0.9</td>
<td>0.1</td>
<td>62.48</td>
<td>18.08</td>
</tr>
<tr>
<td>0.5</td>
<td>15.92</td>
<td>18.09</td>
<td>0.3055</td>
</tr>
<tr>
<td>1.0</td>
<td>10.08</td>
<td>18.09</td>
<td>0.3272</td>
</tr>
<tr>
<td>5.0</td>
<td>5.278</td>
<td>18.12</td>
<td>0.4074</td>
</tr>
<tr>
<td>10.0</td>
<td>4.861</td>
<td>18.13</td>
<td>0.4443</td>
</tr>
<tr>
<td>0.95</td>
<td>0.1</td>
<td>163.2</td>
<td>19.01</td>
</tr>
<tr>
<td>0.5</td>
<td>42.35</td>
<td>19.02</td>
<td>0.5039</td>
</tr>
<tr>
<td>1.0</td>
<td>27.08</td>
<td>19.02</td>
<td>0.5284</td>
</tr>
<tr>
<td>5.0</td>
<td>14.45</td>
<td>19.05</td>
<td>0.6156</td>
</tr>
<tr>
<td>10.0</td>
<td>12.73</td>
<td>19.06</td>
<td>0.6542</td>
</tr>
</tbody>
</table>
\[ V = \frac{1}{m_v} \begin{pmatrix} -v_1 & v_1 & 0 & 0 \\ 0 & -v_2 & v_2 & 0 \\ 0 & 0 & -v_2 & v_2 \\ 0 & 0 & 0 & -v_2 \end{pmatrix}, \]

where \( v_1 = 0.9501281620787609 \), \( v_2 = 3.420263623178192 \) which is used for fitting the first three moments of the log-normal distribution with mean \( m_v \) and squared coefficient of variation 0.5 [29]. In case of \( K = 30 \), we just invert the matrices of size \( (w+1) = 15 \) in order to compute the rate matrix \( R^{(n)} \) instead of directly inverting the matrix \( B^{(n)}[R] \) of size \( 31 \times 3 \times 3 = 465 \).

2. \( M/M/c/K \) retrial queue with asynchronous single vacation. We assume the stability condition \( \rho = \frac{m_v}{c} < 1 \) which can be proved by following the similar argument of the previous example. Let \( X^{(i)}(i,j) = P(X=n,Y=k,J=j) \). \( (n,k,j) \in S_{AS} \) be the stationary distribution of \( Z_{AS} \) and set \( X^{(n)} = (X^{(n)}_1, \ldots, X^{(n)}_c) \) with \( X^{(n)}_k = (X^{(n)}_k(j), 0 \leq j \leq c) \). Once the stationary distribution \( x = (x^n, n \geq 0) \) with truncation level \( N \) is obtained, the performance measures such as the mean number \( L_0 = E[X] \) of customers in orbit and the mean number \( L_1 = E[Y] \) of customers in service facility, the blocking probability \( P_b = P(Y = K) \) and the mean number \( E[J] \) of servers in vacation are obtained by the formulae

\[ L_0 = \sum_{n=0}^{N} nx_n, \quad L_1 = \sum_{k=0}^{K} ky_k, \quad P_b = y_K, \quad E[J] = \sum_{n=0}^{N} \sum_{k=0}^{K} \sum_{j=1}^{c} jx_2^{(n)}(j), \]

where \( x_n = P(X=n) \) and \( y_k = P(Y=k) \) are the stationary distributions of \( X(t) \) and \( Y(t) \), respectively in stationary state.

In Table 2, some numerical results for \( M/M/20/30 \) retrial queue with exponential vacation time under asynchronous and single vacation policy are presented. In this case the algorithm requires to invert the matrices of size \( c+1 = 21 \) in order to compute the rate matrix \( R^{(n)} \) instead of inverting the matrix \( B^{(n)}[R] \) of size \( 31 \times 21 = 651 \).

Table 2
\( M/M/20/30 \) retrial queue with exponential vacation time under asynchronous and single vacation policy with \( \mu = 1.0 \). \( m_v = 1.5 \).

<table>
<thead>
<tr>
<th>( \gamma )</th>
<th>( \rho )</th>
<th>( L_0 )</th>
<th>( L_1 )</th>
<th>( P_b )</th>
<th>( E[J] )</th>
<th>( N )</th>
<th>( \rho )</th>
<th>( L_0 )</th>
<th>( L_1 )</th>
<th>( P_b )</th>
<th>( E[J] )</th>
<th>( N )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.7</td>
<td>1.014</td>
<td>16.65</td>
<td>0.0070</td>
<td>5.838</td>
<td>30</td>
<td>0.8</td>
<td>5.002</td>
<td>20.01</td>
<td>0.0295</td>
<td>3.942</td>
<td>55</td>
</tr>
<tr>
<td>0.5</td>
<td>0.258</td>
<td>16.67</td>
<td>0.0082</td>
<td>5.836</td>
<td>25</td>
<td>1.378</td>
<td>20.05</td>
<td>0.0358</td>
<td>3.939</td>
<td>45</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>0.167</td>
<td>16.67</td>
<td>0.0094</td>
<td>5.835</td>
<td>25</td>
<td>0.959</td>
<td>20.09</td>
<td>0.0429</td>
<td>3.938</td>
<td>40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.0</td>
<td>0.095</td>
<td>16.71</td>
<td>0.0147</td>
<td>5.835</td>
<td>25</td>
<td>0.647</td>
<td>20.24</td>
<td>0.0711</td>
<td>3.937</td>
<td>40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.0</td>
<td>0.085</td>
<td>16.72</td>
<td>0.0172</td>
<td>5.834</td>
<td>25</td>
<td>0.605</td>
<td>20.28</td>
<td>0.0838</td>
<td>3.937</td>
<td>40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.1</td>
<td>0.9</td>
<td>20.20</td>
<td>23.74</td>
<td>0.0974</td>
<td>1.986</td>
<td>115</td>
<td>0.95</td>
<td>44.55</td>
<td>25.87</td>
<td>0.1814</td>
<td>0.996</td>
<td>210</td>
</tr>
<tr>
<td>0.5</td>
<td>6.367</td>
<td>23.87</td>
<td>0.1255</td>
<td>1.983</td>
<td>85</td>
<td>16.79</td>
<td>26.15</td>
<td>0.2480</td>
<td>0.994</td>
<td>165</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>4.921</td>
<td>24.02</td>
<td>0.1556</td>
<td>1.982</td>
<td>85</td>
<td>14.19</td>
<td>26.38</td>
<td>0.3087</td>
<td>0.993</td>
<td>160</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5.0</td>
<td>3.995</td>
<td>24.40</td>
<td>0.2573</td>
<td>1.981</td>
<td>85</td>
<td>12.79</td>
<td>26.88</td>
<td>0.4747</td>
<td>0.993</td>
<td>160</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.0</td>
<td>3.889</td>
<td>24.49</td>
<td>0.2967</td>
<td>1.981</td>
<td>85</td>
<td>12.66</td>
<td>26.99</td>
<td>0.5291</td>
<td>0.993</td>
<td>160</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Acknowledgements

This paper was supported by Basic Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (Grant Number 2012R1A1B3004158).

Appendix A

A.1. Inversion of the transient LDQBD generator with finite states

Let

\[ \tilde{Q} = \begin{pmatrix} B_0 & A_0 & A_1 \\ C_1 & B_1 & A_1 \\ \vdots & \vdots & \vdots \\ C_{n-1} & B_{n-1} & A_{n-1} \\ C_n & A_n \end{pmatrix}. \]
Here $B_k \ (1 \leq k \leq n)$ is the square matrix of order $m_k$ whose diagonal elements are nonnegative. The entries of the matrices $A_k \ (0 \leq k \leq n-1)$ and $C_k \ (1 \leq k \leq n)$ are nonnegative. We assume that the inverse matrix $Q^{-1}$ exists and let $X(i,j), \ 0 \leq i, j \leq n$ be the $(i,j)$ block matrix component of $(-Q)^{-1} = (X(i,j))$. The following algorithm for $(-Q)^{-1} = (X(i,j))$ is given by Shin [27].

Algorithm A1

1. Compute $R_k$ and $G_k, \ 0 \leq k \leq n$.
   (1) $S_n = (-B_n)^{-1}; \ R_n = A_{n-1}S_n; \ G_n = S_nC_n$
   (2) For $k = n-1, n-2, \ldots, 1$,

   $$S_k = [-(B_k + R_{k+1}C_{k+1})]^{-1}; \ R_k = A_{k-1}S_k; \ G_k = S_kC_k.$$ 

   (3) $S_0 = [-(B_0 + R_1C_1)]^{-1}; \ R_0 = S_0; \ G_0 = S_0$.

2. Compute $X(i,j), \ 0 \leq i, j \leq n$.
   For $k = 0, 1, 2, \ldots, n$,

   $$X(k,k) = \begin{cases} R_0, & k = 0, \\ (I + X(k-1)A_{k-1})S_k, & k > 1. \end{cases}$$

   For $j = k + 1, k + 2, \ldots, n$,

   $$X(k,j) = X(k,j-1)R_j; \ X(j,k) = G_jX(j-1,k).$$

A2. Ergodic condition of MAP/M/c/K retrial queue with PH-vacation time under $(a,b)$-vacation policy and single vacation

Let $Z' = \{(Z'(t), t \geq 0)\}$ with $Z'(t) = (X(t), Y(t), J_y(t), J_c(t))$ and denote the state space of $Z'$ by $S' = \cup_{m=0}^{\infty} \mathbb{N}$, where $\mathbb{N} = \{(n,k,j,i): 0 \leq k \leq K, \ 0 \leq j \leq w, \ 1 \leq i \leq l, \ n \geq 0\}$. The matrix components $A^{(n)}, B^{(n)}$ and $C^{(n)}$ of the generator $Q'$ of $Z'$ corresponding to $A^{(n)}, B^{(n)}$ and $C^{(n)}$ of $Q$, respectively are of the form:

$$A_{k}^{(n)} = I_{w+1} \otimes D, \quad C_{k}^{(n)} = \gamma_m I_{l(w+1)}, \quad B_{k}^{(n)} = I_{w+1} \otimes D,$$

$$B_{k,k-1}^{(n)} = \begin{cases} \begin{pmatrix} \mu_k l_w & 0 \\ \mu_k \delta \otimes I_l & 0 \\ \mu_k l_w & 0 \\ 0 & \mu_k l_l \\ \end{pmatrix}, & 1 \leq k \leq a^* + 1 \\ \begin{pmatrix} \mu_k l_w & 0 \\ \mu_k \delta \otimes I_l & 0 \\ \mu_k l_w & 0 \\ 0 & \mu_k l_l \\ \end{pmatrix}, & a^* + 1 < k \leq K, \end{cases}$$

$$B_{kk}^{(n)} = \begin{pmatrix} V \otimes (C - \mu_k l_l) & V^0 \otimes I_l & 0 \\ V \otimes (C - \mu_k l_l) & V^0 \otimes I_l & 0 \\ 0 & C - \mu_k l_l & \end{pmatrix}, \quad 0 \leq k \leq K.$$

where $\mu_k = \min(k, c\mu), \quad \mu_k = \min(k, b^*\mu)$. Following the same procedure of the proof in Diamond and Alfa [11, Proposition 2.1], it can be seen that if the quasi-birth-and-death (QBD) process with generator of the form

$$Q_0' = \begin{pmatrix} B & A \\ C & B & A \\ & & \ddots \end{pmatrix}$$

with

$$A = \begin{pmatrix} I_w \otimes D \\ 0 \\ \end{pmatrix}, \quad B = \begin{pmatrix} V \otimes (C - b^*\mu l_l) & V^0 \otimes I_l \\ 0 & C - c\mu l_l \\ \end{pmatrix}, \quad C = \begin{pmatrix} b^* \mu l_w & 0 \\ 0 & c\mu l_l \end{pmatrix}$$

is positive recurrent, then $Z'$ is positive recurrent. Let $R$ be the minimal nonnegative solution of the matrix equation

$$A + RB + RC^2 = 0.$$ 

It can be seen from the formula

$$A + B + C = \begin{pmatrix} H_w & H_{w0} \\ 0 & H_0 \end{pmatrix}$$

that $R$ is of the form

$$R = \begin{pmatrix} H_w & H_{w0} \\ 0 & H_0 \end{pmatrix}.$$
\[
\mathcal{R} = \begin{pmatrix}
R_V & R_0 \\
0 & R_0
\end{pmatrix}.
\]

where \( R_V \) and \( R_0 \) are for the states corresponding to the server vacation and to that no servers are in vacation, respectively. Following the arguments Neuts [18, Section 1.4], we can see that \( Q^1_0 \) is positive recurrent if and only if the spectral radius of \( R \) \( \text{sp}(R) < 1 \). Note that \( \text{sp}(R) < 1 \) if and only if \( \text{sp}(R_V) < 1 \) and \( \text{sp}(R_0) < 1 \). Since \( \text{sp}(R_V) < 1 \), see Neuts [18, Corollary 1.3.1]. Thus \( Q^1_0 \) is positive recurrent if and only if \( \text{sp}(R_0) < 1 \). We have from (4.2) that \( R_0 \) is a minimal non-negative solution of the matrix equation
\[
D + R_0(C - c\mu I) + R_0^2 c\mu I = 0.
\]

It can be easily seen that \( R_0 \) is the same as the rate matrix arising in ordinary MAP/M/c queue. Thus \( \rho < 1 \) is a sufficient condition for \( Q^1_0 \) to be positive recurrent.

References