MULTISERVER RETRIAL QUEUE WITH SETUP TIME AND ITS APPLICATION TO DATA CENTERS

TUAN PHUNG-DUC∗
Division of Policy and Planning Sciences
Faculty of Engineering, Information and Systems
University of Tsukuba
1-1-1 Tennodai, Tsukuba, Ibaraki 305-8573, Japan

KEN’ICHI KAWANISHI
Division of Electronics and Informatics
Gunma University
1-5-1 Tenjin-cho, Kiryu, Gunma 376-8515, Japan

ABSTRACT. This paper considers a multiserver retrial queue with setup time which is motivated from application in data centers with the ON-OFF policy, where an idle server is immediately turned off. The ON-OFF policy is designed to save energy consumption of idle servers because an idle server still consumes about 60% of its peak consumption processing jobs. Upon arrival, a job is allocated to one of available off-servers and that server is started up. Otherwise, if all the servers are not available upon arrival, the job is blocked and retries in a random time. A server needs some setup time during which the server cannot process a job but consumes energy. We formulate this model using a three-dimensional continuous-time Markov chain obtaining the stability condition via Foster–Lyapunov criteria. Interestingly, the stability condition is different from that of the corresponding non-retrial queue. Furthermore, exploiting the special structure of the Markov chain together with a heuristic technique, we develop an efficient algorithm for computing the stationary distribution. Numerical results reveal that under the ON-OFF policy, allowing retrials is more power-saving than buffering jobs. Furthermore, we obtain a new insight that if the setup time is relatively long, setting an appropriate retrial time could reduce both power consumption and the mean response time of jobs.

1. Introduction. Data centers are the main infrastructure for cloud computing services. In data centers, there are a huge number of computer servers (server farm) which consume a large amount of energy. Therefore, power-saving in data centers is one of the most important issues because it has a big impact on the operational cost and the environment. It is reported that power consumption of an idle server is about 60% of its peak [2]. A natural way for saving power consumption is to turn off physical servers once they become idle. However, this policy, sometimes called as ON-OFF policy, has some side effects. First, arriving jobs are not processed immediately upon arrivals because off-servers are required to restart thus increasing the delay. Second, during the restart period (i.e., setup time hereafter), the server

2010 Mathematics Subject Classification. 60K25, 68M20, 90B22.
Key words and phrases. Data center, power consumption, delay, trade-off, retrials.
The reviewing process of this paper was handled by Yutaka Takahashi and Wuyi Yue.
∗ Corresponding author.
cannot process jobs but consumes energy. In this paper, we consider the applications where one physical server processes one job at a time as in [9].

Investigating the delay-performance and energy consumption trade-off by a queueing model is useful because it can provide some insights that help us to design optimal system saving the power consumption while keeping an acceptable service level for users. Gandhi et al. [9] analyze a multiserver queue with setup times for data centers using the recursive renewal reward approach. The authors investigate the tradeoff between power-consumption and delay-performance. Phung-Duc [22] analyzes the same model and obtain exact closed form expressions of the stationary distribution using the generating function approach and the matrix analytic methods. Furthermore, in [22], the switching cost between ON and OFF is taken into account in the power consumption.

An important feature missing in [9, 22] is the retrial phenomenon of jobs. It is common in computer systems that requests are resubmitted after some random time if they are blocked. For many applications such as Gmail, etc., a request is repeated after some waiting time. It is even more important to take into account the retrial phenomenon in systems with power-saving controls because these policies clearly increase the dropping (blocking) probability and the delay for jobs, thus accelerating retrials. On one hand, allowing retrials may increase the possibility that the server can be in OFF state thus reducing power consumption. On the other hand, a job may suffer from blocking leading to a long delay.

The aim of our paper is to quantify the tradeoff between the power consumption and delay-performance taking into account the presence of retrials. To this end, we consider a multiserver retrial queue with setup time which is a natural retrial version of the multiserver queue in [9, 22]. We assume that the multiserver queue is buffer-less, i.e., no waiting position is available. Instead, an arriving job that cannot find any available servers is blocked and it retries in a random time independently of other jobs in the orbit and the service of servers. During retrial intervals, the job is said to stay in the orbit. We formulate the multiserver retrial queue using a three-dimensional continuous-time Markov chain and establish the stability condition which is the theoretical contribution of our paper. We show that the stability condition of the multiserver retrial queue with setup time is stronger than that of the corresponding non-retrial queue in [9, 22]. From a practical point of view, this result suggests that under power-saving controls such as the ON-OFF policy, more servers are needed in order to stabilize the system with retrials in comparison with that without retrials (with an infinite buffer).

It has been widely known in fairly general settings that the stability condition for a retrial model is the same with that of the corresponding model without retrials [13, 14]. The stability condition of our multiserver retrial queue is a new finding in the sense that it is different from that of the corresponding queue without retrials. In this line, we recently became aware of Shin [24] in which a stability condition is derived for MAP/PH/c/K retrial queues with vacation. It should be noted that vacation model is essentially different from setup model because in the former a vacation is independent of waiting customers while in the latter OFF servers are activated by waiting jobs.

In order to compute the stationary distribution, we treat the Markov chain as a level-dependent quasi birth-and-death process with infinitely many levels. We show that our Markov chain possesses a similar structure to the one of [19], allowing us to compute the stationary distribution efficiently. We also propose a heuristic
technique to determine the truncation point for the level-dependent quasi-birth-and-death process. Finally, we provide some numerical examples to show impacts of retrials and setup time on the power-saving policy in data centers. Furthermore, we show the power consumption and delay performance trade-off and new insights into the performance of data centers. A brief version of our paper was presented in [21].

We shortly summarize the contributions of this paper for clarity. First, we give the stability condition of the multiserver retrial queue with setup time. Second, we develop a heuristic technique to determine the truncation point of the level-dependent quasi-birth-and-death process. Third, we provide performance evaluation of the retrial queue in terms of the power consumption and delay.

The rest of this paper is organized as follows. Section 2 presents the multiserver retrial queue with setup time and its formulation based on a three-dimensional continuous-time Markov chain. Section 3 provides the stability condition of the queueing model. Computational algorithms for the stationary distribution are summarized in Section 4. Section 5 presents performance evaluation via numerical experiments. Concluding remarks are presented in Section 6.

2. System model and Markov chain.

2.1. Queueing model. We consider an M/M/c/c retrial queue with setup time where a server in the queueing system corresponds to one physical server in data centers. We assume that a physical server can process one job at a time as in [9]. In this system, the server is immediately turned off if there is no job to process. An arriving job is allocated to an unoccupied off-server and that the server is started up. The server needs some setup time to be active so as to process the waiting jobs. Upon the completion of a service, the server picks one waiting job (waiting at another setup server) if any and the server in setup for that job is turned off. Thus, in our model a server has one of three states: active (serving a job), off or setup. We assume that the setup time of the server follows the exponential distribution with mean $1/\alpha$ and the service time is exponentially distributed with mean $1/\nu$.

The arrival process of jobs is the Poisson process with rate $\lambda$ and is independent of the setup and service times. Blocked jobs that see all the servers occupied (active or setup) join the orbit from which they retry independently with rate $\mu$. A job in the orbit retries until being served. Regardless of a newly arriving job or a retrial job from the orbit, the process of serving a job is the same. Although our model is buffer-less, jobs still wait for available servers from setup or service completions.

Remark 1. Our model with $c = 1$ is different from the single server retrial queue with setup time in [20, 23] and its variant [4], where the server is not turned off if there are some jobs in the orbit. In contrast, this paper analyzes the case where the servers are not aware of jobs in the orbit. Therefore, the server is turned off immediately upon the completion of a service. As a result, our model with $c = 1$ is identical to the M/G/1 retrial queue with the same arrival rate and retrial rate. Further, the service time of this M/G/1 queue is the sum of the setup time and the service time of our model.

2.2. Continuous-time Markov chain. Let us denote by $S_1(t)$ the number of jobs in the service facility at time $t$. It should be noted that $S_1(t)$ is also the sum of the number of active servers (the number of jobs in service) and the number of servers in setup (the number of jobs waiting for an active and available server) at
time \( t \). Let us denote by \( S_2(t) \) the number of active servers (serving a job) at time \( t \). Furthermore, let \( N(t) \) denote the number of jobs in the orbit at time \( t \). Then, it is easy to see that the stochastic process

\[
\{X(t) = (S_1(t), S_2(t), N(t)); t \geq 0\}
\]

forms a continuous-time Markov chain on the state space \( S \) defined by

\[ S = \{(i, j, k); i = 0, 1, \ldots, c, j = 0, 1, \ldots, i, k \in \mathbb{Z}_+\}, \]

where \( \mathbb{Z}_+ = \{0, 1, \ldots\} \).

**Remark 2.** There are several ways to construct equivalent Markov chains for this system. For example, one may consider the process \((S_1(t) - S_2(t), S_2(t), N(t))\). However, as we will show, our Markov chain possesses a special structure for which the framework in [19] could be adopted to develop an efficient algorithm.

### 2.3. Level-dependent QBD process

We formulate \( \{X(t); t \geq 0\} \) as a level-dependent quasi birth-and-death (QBD) process [16] by taking \( N(t) \) as the level variable and \((S_1(t), S_2(t))\) as the phase variable. Then it is easy to see that the infinitesimal generator \( Q \) is given by

\[
Q = \begin{bmatrix}
Q_0^{(0)} & Q_1^{(0)} & 0 & \cdots \\
Q_2^{(1)} & Q_1^{(1)} & Q_0^{(1)} & \cdots \\
0 & Q_2^{(2)} & Q_1^{(2)} & \cdots \\
\vdots & \vdots & \ddots & \ddots \\
\end{bmatrix},
\]

where \( Q_0^{(k)} \) and \( Q_1^{(k)} \) for \( k = 0, 1, \ldots \), and \( Q_2^{(k)} \) for \( k = 1, 2, \ldots \) are all square matrices of dimension \((c+1)(c+2)/2\) and \( O \) is a zero matrix of appropriate dimension. These matrices have block structures expressed by

\[
Q_0^{(k)} = \begin{bmatrix}
O & O & \cdots & \cdots & O \\
& \ddots & \cdots & \cdots & \vdots \\
& \ddots & \ddots & \cdots & \vdots \\
& \ddots & \ddots & \ddots & \vdots \\
O & O & \cdots & \ddots & A_{c,c}
\end{bmatrix},
\]

\[
Q_1^{(k)} = \begin{bmatrix}
O & N_{0,1}^{(k)} & O & \cdots & O \\
& \ddots & \ddots & \cdots & \vdots \\
& \ddots & \ddots & \ddots & \vdots \\
& \ddots & \ddots & \ddots & \vdots \\
O & \cdots & \cdots & O & N_{c-1,c}^{(k)}
\end{bmatrix},
\]

\[
Q_2^{(k)} = \begin{bmatrix}
L_{0,0}^{(k)} & B_{0,1}^{(k)} & O & \cdots & O \\
D_{1,0}^{(k)} & L_{1,1}^{(k)} & B_{1,2}^{(k)} & \cdots & \vdots \\
& \ddots & \ddots & \ddots & \vdots \\
& \ddots & \ddots & \ddots & \ddots \\
O & \cdots & O & L_{c-1,c}^{(k)} & B_{c-1,c}^{(k)}
\end{bmatrix}.
\]
where $\Lambda_{c,c}, N_{i-1,i}^{(k)} (i = 1, 2, \ldots, c)$, $B_{i-1,i}^{(k)} (i = 1, 2, \ldots, c)$, $D_{i,i-1}^{(k)} (i = 1, 2, \ldots, c)$ and $L_{i,i}^{(k)} (i = 0, 1, \ldots, c)$ are matrices with sizes: $(c+1) \times (c+1), i \times (i+1), i \times (i+1)$, $(i+1) \times i$ and $(i+1) \times (i+1)$, respectively. We have $\Lambda_{c,c} = \lambda I$, and for $i = 1, \ldots, c$

$$N_{i-1,i}^{(k)} = k\mu E_{i-1,i}, \quad E_{i-1,i} = [I \ O],$$

where $I$ is an identity matrix of appropriate dimension. For $i = 1, \ldots, c$,

$$B_{i-1,i}^{(k)} = \begin{bmatrix}
\lambda & 0 & \cdots & 0 \\
0 & \lambda & \cdots & 0 \\
\vdots & \ddots & \ddots & \vdots \\
0 & \cdots & 0 & \lambda
\end{bmatrix} = \Lambda_{i-1,i},$$

$$D_{i,i-1}^{(k)} = \begin{bmatrix}
0 & 0 & \cdots & 0 \\
0 & \nu & \cdots & 0 \\
\vdots & \ddots & \ddots & \vdots \\
0 & \cdots & 0 & (i-1)\nu
\end{bmatrix},$$

We have $L_{0,0}^{(k)} = -q_{0,0} - k\mu$ and for $i = 1, \ldots, c-1$,

$$L_{i,i}^{(k)} = \begin{bmatrix}
-q_{i,0} & i\alpha & 0 & \cdots & 0 \\
0 & -q_{i,1} & (i-1)\alpha & \cdots & 0 \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
0 & \cdots & \cdots & \cdots & \alpha \\
0 & \cdots & \cdots & 0 & -q_{i,i}
\end{bmatrix} - k\mu I,$$

and

$$L_{c,c}^{(k)} = \begin{bmatrix}
-q_{c,0} & c\alpha & 0 & \cdots & 0 \\
0 & -q_{c,1} & (c-1)\alpha & \cdots & 0 \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
0 & \cdots & \cdots & \cdots & \alpha \\
0 & \cdots & \cdots & 0 & -q_{c,c}
\end{bmatrix},$$

where $q_{i,j} = \lambda + j\nu + (i-j)\alpha$ for $i = 0, 1, \ldots, c$ and $j = 0, 1, \ldots, i$. It holds that $\{X(t); t \geq 0\}$ is irreducible.

3. Stability analysis. We consider a sufficient condition for the positive recurrence of $\{X(t); t \geq 0\}$. We define the diagonal matrix $\Delta$ as $\Delta = \text{diag}\{q_{c,0}, q_{c,1}, \ldots, q_{c,c}\}$ and we define square matrices

$$A_0 = \Delta^{-1}\Lambda_{c,c},$$

$$A_1 = I + \Delta^{-1}L_{c,c}^{(k)},$$

$$A_2 = \Delta^{-1}D_{c,c-1}^{(k)}E_{c-1,c}.$$
record the transition probabilities increasing the level by 1, keeping in the same level and decreasing the level by 1, respectively.

Letting \( A = A_0 + A_1 + A_2 \), we then have

\[
A = I + \Delta^{-1}
\begin{bmatrix}
-\alpha & \alpha & 0 & \cdots & 0 \\
0 & -(c-1)\alpha & (c-1)\alpha & \cdots & 0 \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
0 & \cdots & 0 & -\alpha & \alpha \\
0 & \cdots & 0 & c\nu & -c\nu
\end{bmatrix}.
\]

It is clear that \( A \) is the transition probability matrix of a discrete-time Markov chain on the state space \( \{0, 1, \ldots, c\} \). We define \( A(z) \) by

\[
A(z) = A_0 + A_1 z + A_2 z^2, \quad 0 \leq z \leq 1.
\]

It should be noted that \( A \) in our model is reducible. It is clear that \( A(z) \) is also reducible and has the same structure of \( A \). In fact, by a direct calculation, we can see that \( A(z) \) has the block structure given by

\[
A(z) = \begin{bmatrix}
A_{0,0}(z) & A_{0,c-1}(z) \\
O & A_{c-1,c-1}(z)
\end{bmatrix},
\]

where

\[
A_{0,0}(z) = \begin{bmatrix}
a_{0,0}(z) & a_{0,1}(z) & 0 & \cdots & 0 \\
0 & a_{1,1}(z) & a_{1,2}(z) & \cdots & \vdots \\
\vdots & \ddots & \ddots & \ddots & \ddots \\
0 & \cdots & 0 & a_{c-3,c-2}(z) \\
0 & \cdots & \cdots & 0 & a_{c-2,c-2}(z)
\end{bmatrix},
\]

\[
A_{0,c-1}(z) = \begin{bmatrix}
0 & 0 \\
\vdots & \vdots \\
0 & 0 \\
a_{c-2,c-1}(z) & 0
\end{bmatrix},
\]

\[
A_{c-1,c-1}(z) = \begin{bmatrix}
\frac{\lambda + (c-1)\nu z^2}{\lambda + c\nu} & \frac{\alpha z}{(c-1)\nu + \alpha} \\
\frac{c\nu z^2}{\lambda + c\nu} & \frac{\lambda}{\lambda + c\nu}
\end{bmatrix},
\]

and for \( j = 0, 1, \ldots, c-2 \),

\[
a_{j,j}(z) = \frac{\lambda + j\nu z^2}{\lambda + j\nu + (c-j)\alpha}, \quad a_{j,j+1}(z) = \frac{(c-j)\alpha z}{\lambda + j\nu + (c-j)\alpha}.
\]

Because \( A(z) \) is non-negative for \( 0 \leq z \leq 1 \), there exists an eigenvalue \( \chi(z) \) such that \( |\chi(z)| \geq |\lambda_1(z)| \geq \cdots \geq |\lambda_c(z)| \), where \( \lambda_1(z), \ldots, \lambda_c(z) \) are the other eigenvalues of \( A(z) \). Furthermore, we can choose an eigenvector \( v(z) \) corresponding to \( \chi(z) \) such that \( v(z) \geq \mathbf{0} \). In Lemmas 3.1 and 3.2, we prove that under a certain condition (the stability condition presented in Theorem 3.3), there exists some \( z \in (0, 1) \) such that \( v(z) > 0 \), based on which we can construct a Lyapunov function for guaranteeing the stability of our Markov chain.
Remark 3. Neuts [15] mainly treated the case $A(z)$ is irreducible. It was pointed out in Neuts [15] (pages 14–15) that when $A(z)$ is reducible ad hoc arguments are needed, which is our case.

Let $\chi_{c-1}(z)$ be the largest eigenvalue of $A_{c-1,c-1}(z)$ with the right eigenvector $v_{c-1}(z)$. Because $A_{c-1,c-1}(z)$ is irreducible and non-negative, we can choose $v_{c-1}(z) > 0$.

Lemma 3.1. If there exists $z_0 \in (0, 1)$ such that

$$\chi_{c-1}(z) > a_{j,j}(z), \quad z_0 < z < 1,$$

for $j = 0, 1, \ldots, c - 2$, then we have $\chi(z) = \chi_{c-1}(z)$ and $v(z) > 0$ for $z_0 < z < 1$.

Proof. Assume that $\chi_{c-1}(z) > a_{j,j}(z)$ for $z_0 < z < 1$ and $j = 0, 1, \ldots, c - 2$. Then it is obvious that

$$\chi(z) = \chi_{c-1}(z),$$

because $a_{j,j}(z)$ for $j = 0, 1, \ldots, c - 2$ are eigenvalues of $A(z)$. The right eigenvector $v(z)$ of $A(z)$ corresponding to the eigenvalue $\chi(z) = \chi_{c-1}(z)$ is written as

$$v(z) = [v_0(z), \ldots, v_{c-2}(z), v_{c-1}(z)^T]^T,$$

where $v_0(z), \ldots, v_{c-2}(z)$ are all non-negative. If $v_{c-2}(z) = 0$, then it holds that

$$[a_{c-2,c-1}(z), 0]v_{c-1}(z) = 0,$$

due to $A(z)v(z) = \chi_{c-1}(z)v(z)$. Because $a_{c-2,c-1}(z) > 0$ and $v_{c-1}(z) > 0$, we reach a contradiction. Therefore we have $v_{c-2}(z) > 0$. If $v_{c-3}(z) = 0$, $A(z)v(z) = \chi_{c-1}(z)v(z)$ yields $a_{c-3,c-2}(z)v_{c-2}(z) = 0$. Since $a_{c-3,c-2}(z)$ and $v_{c-2}(z)$ are positive, we reach a contradiction again. Therefore we have $v_{c-3}(z) > 0$. Repeating the same argument, we conclude that $v_0(z), \ldots, v_{c-2}(z)$ are all positive. 

Remark 4. It should be noted that $\chi_{c-1}(z)$ is a convex function of $z$ with $\chi_{c-1}(1) = 1$, because $A_{c-1,c-1}(1)$ is an irreducible stochastic matrix. It also holds that $a_{j,j}(z)$ for $j = 0, 1, \ldots, c - 2$ are convex functions of $z$ with $a_{j,j}(1) < 1$. Therefore, we can choose some $z_0 \in (0, 1)$ such that $\chi_{c-1}(z) > a_{j,j}(z)$ for $z_0 < z < 1$ and $j = 0, 1, \ldots, c - 2$.

Let us denote by $\nu_{c-1}$ the invariant probability vector of $A_{c-1,c-1}(1)$, and by $\beta_{c-1}$ the vector given by

$$\beta_{c-1} = A'_{c-1,c-1}(1)e,$$

where $A'_{c-1,c-1}(1)$ is the first derivative of the function of $A_{c-1,c-1}(z)$ with respect to $z$. It is well-known [15] that the equation

$$\chi_{c-1}(z) = z$$

has a solution $\eta \in (0, 1)$, if

$$\nu_{c-1} \beta_{c-1} > 1, \quad \chi_{c-1}(0) > 0.$$ 

Because it holds that $\chi_{c-1}(0) > 0$ in our model, there exists such a solution, if $\nu_{c-1} \beta_{c-1} > 1$. 

Multiserver Retrial Queue with Setup 21
Lemma 3.2. If \( \nu c_{e-1} \beta_{e-1} > 1 \), then it holds that
\[
\chi(z) = \chi_{c-1}(z) < z,
\]
and
\[
A(z)v(z) = \chi(z)v(z) < zv(z),
\]
for \( z \in (\max\{\eta, z_0\}, 1) \), where \( v(z) > 0 \).

Proof. The proof is immediately followed from the convexity of \( \chi_{c-1}(z) \) and Lemma 3.1. \( \square \)

Lemma 3.2 provides a basis to obtain a sufficient condition of the positive recurrence of \( \{X(t); t \geq 0\} \).

Theorem 3.3. Let us define the traffic intensity of our retrial queueing system by
\[
\rho = \lambda/(cv) \quad \text{and}
\]
\[
\rho = \frac{(c - 1)\nu + \alpha}{cv + \alpha} < 1.
\]
If it holds that
\[
\rho < r,
\]
then the stochastic process \( \{X(t); t \geq 0\} \) is positive recurrent.

Proof. The proof is essentially based on the technique by Diamond and Alfa [5, 6], which makes use of the Foster criterion. We use the approach by Tweedie [25] or Statement 8, p. 97 in Falin and Templeton [7], which states that a Markov process with infinitesimal generator \( Q = [q_{s,p}] \) on state space \( S \) is regular and ergodic, if there exist a lower bounded function \( \varphi(\cdot) \) on \( S \), some finite subset \( S_0 \subset S \) and some \( \epsilon > 0 \) such that
\[
y(s) = \sum_{p \in S} q_{s,p} \varphi(p) \leq -\epsilon
\]
for \( s \notin S_0 \), and \( y(s) < \infty \) for \( s \in S_0 \).

For \( z \in (\max\{\eta, z_0\}, 1) \), let \( \varphi^{(k)} \) be a column vector of size \( (c + 1)(c + 2)/2 \) given by
\[
\varphi^{(k)} = z^{-k} \begin{bmatrix}
\begin{bmatrix}
z^c w_0(z) \\
z^{c-1} w_1(z) \\
\vdots \\
z w_{c-1}(z) \\
w_c(z)
\end{bmatrix} + \beta e
\end{bmatrix}, \quad k = 0, 1, \ldots,
\]
where \( b \in (0, 1) \) and \( w_i(z) = E_{i,i+1} \cdots E_{c-1,c} v(z) \) for \( i = 0, 1, \ldots, c - 1 \) is the vector consisting of the first \( i + 1 \) elements of \( v(z) \) and \( w_c(z) = v(z) \). It is clear that each element of \( \varphi^{(k)} \) is lower bounded. Let \( \varphi \) be a column vector composed of \( \varphi^{(k)} \)'s, i.e.
\[
\varphi = [(\varphi^{(0)})^T, (\varphi^{(1)})^T, \ldots, (\varphi^{(k)})^T, \ldots]^T.
\]
Let \( \ell(k) = \{(i, j, k) \in S \mid i = 0, 1, \ldots, c, j = 0, 1, \ldots, i\} \) and \( y^{(k)} \) be a column vector of size \( (c + 1)(c + 2)/2 \) given by the elements of \( \ell(k) \) of \( Q \varphi \). For \( k = 0, 1, \ldots, \) we
have
\[ y^{(k)} = z^{-(k+1)} \begin{pmatrix} z f_0^{(k)}(z) \\ z f_1^{(k)}(z) \\ \vdots \\ z f_{c-1}^{(k)}(z) \\ f_c^{(k)}(z) \end{pmatrix} + b(1-z) \begin{pmatrix} k \mu e \\ k \mu e \\ \vdots \\ k \mu e \\ \Lambda_{c,c} e \end{pmatrix}, \]
where
\[ f_0^{(k)}(z) = B_0^{(k)} w_1(z) + z L_0^{(k)} w_0(z), \]
and for \( i = 1, 2, \ldots, c-1 \)
\[ f_i^{(k)}(z) = B_i^{(k)} w_{i+1}(z) + z L_i^{(k)} w_i(z) + z^2 D_i^{(k)} w_{i-1}(z), \]
and
\[ f_c^{(k)}(z) = (\Lambda_{c,c} + z L_c^{(k)} + z^2 D_{c,c} E_{c-1,c} w_c(z). \]
Due to Lemma 3.2, \( f_c^{(k)}(z) < 0 \) if \( \nu_{c-1} \beta_{c-1} > 1 \) and hence there exists some \( b \in (0,1) \) such that
\[ f_c^{(k)}(z) + b(1-z) \Lambda_{c,c} e < 0. \]
It is clear that there exists an integer \( k_0 \) such that
\[ z^{c-1} f_i^{(k)}(z) - b(1-z) z k \mu e < 0. \]
for all \( k > k_0 \) and \( i = 0, 1, \ldots, c-1 \). Therefore, there exists some \( \epsilon > 0 \) such that
\[ Q \varphi \leq -\epsilon e, \]
except for some finite subset \( S_0 \subset S \), from which we conclude that \( \{X(t); t \geq 0\} \) is ergodic if \( \nu_{c-1} \beta_{c-1} > 1 \).
We can obtain \( \nu_{c-1} \) and \( \beta_{c-1} \) as
\[ \nu_{c-1} = (p_{c-1} \Delta_{c-1} e)^{-1} p_{c-1} \Delta_{c-1}, \]
\[ \beta_{c-1} = e + \Delta_{c-1} \begin{pmatrix} (c-1) \nu - \lambda \\ \nu \lambda - \lambda \end{pmatrix}, \]
where
\[ p_{c-1} = \begin{pmatrix} \nu \lambda + \alpha \\ \nu \lambda + \nu \lambda + \alpha \end{pmatrix}, \]
\[ \Delta_{c-1} = \text{diag}\{\lambda + (c-1) \nu + \lambda + \lambda + \nu\}. \]
It is easy to see that (1) follows from \( \nu_{c-1} \beta_{c-1} > 1. \)

**Remark 5.** Using a similar sample path analysis as in [10], we can also show that (1) holds if the stochastic process \( \{X(t); t \geq 0\} \) is positive recurrent.

**Remark 6.** It is interesting that the stability condition of multiserver queues with setup time is more stringent in the presence of retrials. Indeed, the stability condition of the multiserver queues with setup time and without retrials [9, 22] is \( \rho < 1 \). We note that a similar stability condition is also obtained for MAP/PH/c/c retrial queue with vacation.
Remark 7. The same Lyapunov function can be directly used to derive the sufficient stability condition for $M/M/c + K$ retrial queues with setup time. We confirmed that in the case $K > 0$, the stability condition is $\rho < 1$. The stability conditions of the models with and without buffer are different. It is well known that the stability condition for a retrial queue without setup time and with classical (linear) retrial rate is the same as that of the corresponding model without retrials \cite{13, 14}. This is evident because in the saturated situation where there are many customers in the orbit and all the servers are active. In this situation, the time for a customer to access from the orbit to the server tends to zero and the orbit behaves like the buffer. However, in the model with setup time and without buffer ($M/M/c/c$), when all the servers are active and a job is completed, the server is turned off regardless the presence of jobs in the orbit. Thus, in this sense the service capacity is loosen. In this saturated situation, there likely exists the case that one server is in setup process and the rest of $c - 1$ servers are active. In other words, it seems that one server may lose its service capacity in the saturated situation. This explains why the effect is maximized when $c = 2$. This can be discussed later in formula (3) in detail. However, the effect disappears in the model with buffer because in the saturated situation there are always customers waiting in the buffer and the server is not turned off upon service completions.

4. Algorithm for the stationary distribution. In what follows, we assume that $\{X(t); t \geq 0\}$ is positive recurrent. If we assume that $\{X(t); t \geq 0\}$ is positive recurrent, then we can define $\pi_{i,j}^{(k)} > 0$ for each state $(i, j, k) \in S$ as:

$$
\pi_{i,j}^{(k)} = \lim_{t \to \infty} \Pr\{S_1(t) = i, S_2(t) = j, N(t) = k\}.
$$

For each $k \in \mathbb{Z}_+$, let

$$
\pi^{(k)} = [\pi_{0,0}^{(k)}, \pi_{1,0}^{(k)}, \pi_{1,1}^{(k)}, \ldots, \pi_{c,0}^{(k)}, \pi_{c,1}^{(k)}, \ldots, \pi_{c,c}^{(k)}],
$$

and define

$$
\pi = [\pi^{(0)}, \pi^{(1)}, \ldots, \pi^{(k)}, \ldots].
$$

It holds that $\pi$ is the unique solution of the linear systems given by

$$
\pi Q = 0, \quad \pi e = 1,
$$

where $0$ and $e$ denote a row vector and a column vector of zeros and ones with an appropriate size, respectively.

The matrix analytic method \cite{12} tells us that $\pi^{(k)}$ has a matrix-product form solution given by

$$
\pi^{(k)} = \pi^{(0)} R^{(1)} R^{(2)} \cdots R^{(k)}, \quad k = 1, 2, \ldots,
$$

where $\{R^{(k)}; k = 1, 2, \ldots\}$ are the minimal nonnegative solution to

$$
Q^{(k-1)}_0 + R^{(k)} Q_1^{(k)} + R^{(k)} R^{(k+1)} Q_2^{(k+1)} = O, \quad k = 1, 2, \ldots.
$$

The boundary vector $\pi^{(0)}$ is uniquely determined by

$$
\pi^{(0)} (Q^{(0)}_1 + R^{(1)} Q^{(1)}_2) = 0,
$$

and

$$
\pi^{(0)} \left[ I + \sum_{k=1}^{\infty} R^{(1)} R^{(2)} \cdots R^{(k)} \right] e = 1.
$$
Computing the stationary distribution \( \{ \pi^{(k)}; k = 0, 1, \ldots \} \) in level-dependent quasi birth-and-death processes essentially reduces to computing the rate matrices \( \{ R^{(k)}; k = 1, 2, \ldots \} \). Bright and Taylor [3] propose an algorithm to compute the rate matrices efficiently by extending the logarithmic reduction algorithm [11]. It is reported that a simple algorithm of Phung-Duc et al. [16], based on a matrix continued fraction representation, is easier to implement and less memory-consuming than Bright and Taylor’s algorithm.

**Proposition 1** (Proposition 2.4 in [16]). If we define the matrix sequence \( \{ R^{(k)}_n; n = 0, 1, \ldots \} \) by

\[
R^{(k)}_0 = O, \quad n = 0,
R^{(k)}_n = R^{(k)}_0 \circ \cdots \circ R^{(k+n-1)}(O), \quad n, k = 1, 2, \ldots ,
\]

where, for a square matrix \( X \) of dimension \((c+1)(c+2)/2\),

\[
R^{(k)}(X) = Q^{(k-1)}_0 \left(-Q^{(k)}_1 - XQ^{(k+1)}_2\right)^{-1}, \quad k = 1, 2, \ldots ,
\]

and \( f \circ g(\cdot) = f(g(\cdot)) \), then we have

\[
\lim_{n \to \infty} R^{(k)}_n = R^{(k)}, \quad k = 1, 2, \ldots .
\]

Proposition 1 supports the idea of approximating \( R^{(k)} \) by \( R^{(k)}_n \) for a sufficiently large value of \( n \).

4.1. **A heuristic technique for truncation point.** In order to apply the algorithm in [16], it is necessary to decide where to truncate the level in advance. We propose a heuristic technique to determine the truncation point \( N \). Let us consider a single server M/M/1/1 retrial queue (without setup time) with arrival rate \( \bar{\lambda} \), service rate \( \bar{\nu} \), and retrial rate \( \mu \). We assume the retrial queue is stable. We denote by \( L_1 \) the stationary number of jobs in the orbit for this single server retrial queue. Furthermore, let us consider the number of jobs \( L_2 \) that arrive during an exponentially distributed setup time with mean \( 1/\alpha \). Our heuristic technique is to regard \( L = L_1 + L_2 \) as an approximation of the number of jobs in the orbit of our multiserver retrial queue with setup time. We will choose \( \bar{\lambda} \) and \( \bar{\nu} \) such that the stability condition of the M/M/1/1 queue (without setup time) coincides with our multiserver model with setup time. Instead of multiserver retrial queues, we make use of the single server M/M/1/1 retrial queue because it allows us to obtain an explicit expression of the generating function of the orbit size distribution. We select \( \bar{\lambda} \) and \( \bar{\nu} \) as

\[
\bar{\lambda} = \lambda/c, \quad \bar{\nu} = r\nu.
\]

Under the stability condition of our model \( \rho < r, \bar{\rho} = \bar{\lambda}/\bar{\nu} < 1 \) and thus the generating function \( \Pi(z) \) of \( L_1 \) is given by [7] as:

\[
\Pi(z) = (1 + \bar{\rho} - \bar{\rho}z) \left( \frac{1 - \hat{\rho}}{1 - \rho z} \right)^{1+a},
\]

where \( a = \bar{\lambda}/\mu \). It is easy to see that the generating function \( Q(z) \) of the number of jobs \( L_2 \) arriving during the setup time is given by \( Q(z) = (1 - \theta)/(1 - \theta z) \), where \( \theta = \bar{\lambda}/(\bar{\lambda} + \alpha) \).

The generating function \( H(z) \) of \( L = L_1 + L_2 \) is given by \( H(z) = \Pi(z)Q(z) \), under the assumption that \( L_1 \) and \( L_2 \) are independent. There are two rationales behind
the use of \( L \) for the estimation of the truncation point of our multiserver retrial queue with setup time. First, for a non-retrial multiserver queue with staggered setup, i.e., only one server can be in setup mode at time, the queue length is decomposed to that of the model without setup time and the number of Poisson arrivals during the setup time [18]. Although our model is not staggered setup, we expect that a similar relation, i.e., the orbit size of our multiserver queue with setup time is approximately decomposed into that of the model without setup and the number of Poisson arrivals during the setup time. The second rationale is that it is reasonable to estimate the truncation point of a multiserver retrial queue (without setup time) by a single server retrial model with the same traffic intensity [17]. Since there are two heuristic approximations, our method is rather rough and thus needs to be carefully verified.

Let us denote by \([z^n]f(z)\) the coefficient of \(z^n\) in the series expansion of an arbitrary analytic function \(f(z)\). We use the notation \(g(n) \sim h(n)\) defined by \(\lim_{n \to \infty} g(n)/h(n) = 1\). By investigating the dominant singularity of \(\Pi(z)Q(z)\), we have \([z^n](\Pi(z)Q(z)) \sim m_n(\bar{\rho}, \bar{\theta}, a)\), where

\[
m_n(\bar{\rho}, \bar{\theta}, a) = \begin{cases} 
\Pi(1/\bar{\theta})(1 - \bar{\theta})\bar{\rho}^n, & \bar{\rho} < \bar{\theta}, \\
Q(1/\bar{\rho})\bar{\rho}(1 - \bar{\rho})^{1+a} \frac{n^a}{\Gamma(1+a)}\bar{\rho}^n, & \bar{\rho} > \bar{\theta}, \\
\bar{\rho}(1 - \bar{\rho})^{2+a} \frac{n^{1+a}}{\Gamma(2+a)}\bar{\rho}^n, & \bar{\rho} = \bar{\theta},
\end{cases}
\]

and \(\Gamma(\cdot)\) is the gamma function. Using the asymptotic result, we propose a heuristic technique to determine the truncation point \(N\) by

\[
N = \min\{n \geq 1 \mid m_n(\bar{\rho}, \bar{\theta}, a) < \epsilon_1\},
\]

where \(\epsilon_1 \in (0, 1)\) is an arbitrary small value. It should be noted that the explicit asymptotic expression \(m_n(\bar{\rho}, \bar{\theta}, a)\) of \([z^n](\Pi(z)Q(z))\) gives us the method to compute \(N\) with less effort than those involving matrix computations and thus is practically useful. One of the drawbacks of our heuristic technique is its reliability whose validity will be presented later.

4.2. Approximate stationary distribution. For a given truncation point \(N\), we obtain an approximation \(\hat{R}^{(N)}\) to \(R^{(N)}\) using the algorithm in [16]. Then we can obtain an approximation \(\{\widehat{\pi}^{(k)}; k = 0, 1, \ldots, N\}\) to the stationary distribution \(\{\pi^{(k)}; k = 0, 1, \ldots\}\) by

\[
\widehat{\pi}^{(k)} = \pi^{(k-1)}\hat{R}^{(k)}, \quad k = 1, 2, \ldots, N,
\]

where \(\hat{R}^{(k)} = R^{(k)}(\hat{R}^{(k+1)})\) for \(k = 1, 2, \ldots, N - 1\), and \(\widehat{\pi}^{(0)}\) is the unique solution of the linear systems given by

\[
\widehat{\pi}^{(0)}(Q_{1}^{(0)} + \hat{R}^{(1)}Q_{2}^{(1)}) = 0,
\]

\[
\widehat{\pi}^{(0)}\left[I + \sum_{k=1}^{N} \hat{R}^{(1)} \hat{R}^{(2)} \cdots \hat{R}^{(k)}\right]e = 1.
\]

Because of the special structure of \(Q_{0}^{(k)}\), we observe that \(R^{(k)}(X)\) for a square matrix \(X\) of dimension \((c + 1)(c + 2)/2\) must have the form
$R^{(k)}(X) = \begin{bmatrix}
O & \cdots & O & O \\
\vdots & \ddots & \vdots & \vdots \\
O & \cdots & O & O \\
R^{(k)}_{c,0}(X) & \cdots & R^{(k)}_{c,c-1}(X) & R^{(k)}_{c,c}(X)
\end{bmatrix},$

where $R^{(k)}_{c,i}(X)$ is a $(c+1) \times (i+1)$ matrix for $i = 0, 1, \ldots, c$. Exploiting the property of the sparse structure, it is clear that the approximate stationary distribution of our queueing model can be efficiently computed. In fact, we observe that the structure of the Markov chain of our queueing model is the same as the one of [19] and thus the stationary distribution can be computed by Algorithm 3 in [19].

**Remark 8.** It should be noted that the structure of the underlying Markov chain $\{X(t) = (S_1(t), S_2(t), N(t)); t \geq 0\}$ for the corresponding M/M/c/c + K model is the same as that for the M/M/c/c model. As a result, the algorithm in this section can be directly applied to the M/M/c/c + K model. It should be noted that in the M/M/c/c + K model, $S_1(t)$ is the total number of jobs in the service facility (servers and buffer) and is not necessarily the sum of the number of active servers and the number of servers in setup as in the model without buffer, i.e. M/M/c/c.

### 4.3. Performance measures.

Let us denote by $E[A]$ the expectation of the number of active servers given by

$$E[A] = \sum_{k=0}^{\infty} \sum_{i=0}^{c} \sum_{j=0}^{i} j \pi^{(k)}_{i,j}.$$ 

Because all arriving jobs are eventually served and the expectation of the service time of a server is $1/\nu$, it holds that $E[A] = c\rho$ due to the Little’s law. Let us denote by $E[S]$ the expectation of the number of servers in setup given by

$$E[S] = \sum_{k=0}^{\infty} \sum_{i=0}^{c} \sum_{j=0}^{i} (i-j) \pi^{(k)}_{i,j}.$$ 

It should be noted that we numerically compute $E[S]$ by using $\{\hat{\pi}^{(k)}; k = 0, 1, \ldots, N\}$ instead of $\{\pi^{(k)}; k = 0, 1, \ldots\}$. Assuming that the mean power $E[P]$ is proportional to the number of servers in setup as well as the number of active servers, we propose to evaluate $E[P]$ by

$$E[P] = C_a E[A] + C_s E[S] = C_a c \rho + C_s E[S],$$

where $C_a$ and $C_s$ are the energy per time unit for an active server and a server in setup, respectively. If we assume that a server in setup consumes the same power as in its peak, it holds that $C_s = C_a$.

As a reference model for comparison, we also consider the corresponding M/M/c queue without setup time, where servers become idle if there are no jobs to serve. We refer to the policy of the model as ON-IDLE policy. It is easy to see that the mean power consumption $E[P_{M/M/c}]$ for this model is given by

$$E[P_{M/M/c}] = C_a c \rho + C_i c (1 - \rho),$$

where $C_i$ is the energy per time unit for an idle server. It should be noted that $E[P_{M/M/c}]$ does not depend on the retrial rate $\mu$. 
Furthermore, let $E[W]$ denote the mean response time of a job which is the time interval from arrival to the acceptance of the job to a server including retrial times. $E[W]$ is obtained from the stationary distribution as follows due to Little’s law.

$$E[W] = \sum_{k=0}^{\infty} \sum_{i=0}^{c} \sum_{j=0}^{i} k \pi_{i,j}^{(k)} \frac{l}{\alpha}.$$ 

5. **Numerical examples.** We show some numerical examples to illustrate impacts of retrials and setup time on the performance in terms of the power consumption. For all numerical examples, we set the mean service time as a unit and thus we fix $\nu = 1$. We also fix $C_a = C_s = 1$. If we assume that the power consumption of an idle server is about 60% of its peak [2], it implies that $C_i = 0.6C_s$. The job arrival rate $\lambda$ is adjusted so that we obtain a specific value of $\rho$. It is reported [2] that the utilization of servers is typically between 10 and 50 percents. Unless otherwise stated, we select the value of $\rho = 0.5$ to evaluate the performance in the worst-case scenario.

**Table 1.** Truncation point $N$ and $\hat{\pi}^{(N)}e$ for $c = 30$ and $\mu = 1/10$.

| $c = 30$ | $\alpha = 1/100$ | $\alpha = 1/10$ | $\alpha = 1$ |
|----------|------------------|------------------|-------------|
| $N$      | 1203             | 150              | 59          |
| $\hat{\pi}^{(N)}e$ | $1.0060 \times 10^{-11}$ | $1.6243 \times 10^{-10}$ | $2.2090 \times 10^{-15}$ |

**Table 2.** Truncation point $N$ and $\hat{\pi}^{(N)}e$ for $c = 50$ and $\mu = 1/10$.

| $c = 50$ | $\alpha = 1/100$ | $\alpha = 1/10$ | $\alpha = 1$ |
|----------|------------------|------------------|-------------|
| $N$      | 1203             | 149              | 58          |
| $\hat{\pi}^{(N)}e$ | $4.6265 \times 10^{-11}$ | $2.4851 \times 10^{-10}$ | $1.1490 \times 10^{-16}$ |

5.1. **Validity of heuristic technique.** Before discussing impacts of retrials on the performance measures, we briefly validate our heuristic technique for the truncation point. Tables 1 and 2 show numerical examples of the truncation point $N$ and $\hat{\pi}^{(N)}e$ of the stable multiserver retrial queue with $c = 30, 50$. These values are calculated by setting $\epsilon = 1.0 \times 10^{-12}$ in (2). We choose the retrial rate $\mu = 1/10$. We observe that our heuristic technique gives us sufficiently small value of the stationary probability at the truncation point. It should be noted that the mean setup time to activate an off-server is typically larger than the mean service time of a job. As far as these numerical examples, our heuristic technique is practically useful enough to discuss the performance of data centers. Numerical experiments show that our heuristic technique works well for the case with relatively large retrial rate. Thus, in all of our numerical results, we consider $\mu \geq 1/10$.

5.2. **Effect of retrials on power consumption.** Figure 1 shows the power consumption against the retrial rate. We observe that the power consumption decreases with the decrease in the retrial rate. Thus, the ON-OFF policy works more effectively for long interval of the retrial time. This is because servers are kept off much more time as decreasing the retrial rate. It should be also noted that the multiserver
retrial queue with setup time approximately approaches to the corresponding non-retrial queue as increasing the retrial rate. Therefore Figure 1 suggests that from a power-saving point of view, it is better to reject blocking jobs and allow retrials instead of buffering.

We also observe that the power consumption for $\alpha = 1/10$ is almost insensitive to the retrial rate and is less than that of the $M/M/c$ without setup time. In contrast, the power consumption for $\alpha = 1/100$ is more sensitive to the retrial rate and is higher than that of $M/M/c$ without setup time. This suggests that there exists a threshold $\alpha^*$, which may depend on $\mu$ and possibly on $\rho$ as well, exceeding $\alpha^*$ the power-saving (ON-OFF) policy is more effective than the ON-IDLE policy. Numerical examples of the power consumption against the setup rate in Figure 2 show the evidence supporting that suggestion.

Next we investigate the impact of retrials on the power consumption from the viewpoint of the number of servers. We show the ratio $E[P]/c$ versus the retrial rate in Figure 3. We observe that the ON-OFF policy works effectively as the number of servers increases. This advantage can be viewed as the economies of scale in terms of power-consumption per server.

5.3. Effect of retrials on response time. In this subsection, we investigate the effects of retrials on the mean response time in Figure 4 for $\rho = 0.5$ and in Figure 5 for $\rho = 0.8$. Here the response time of a job is the time from the arrival until the job is accepted to the system. We observe that if the setup time is relatively fast, $\alpha = 1/10$, the mean response time monotonically decreases with the retrial rate as is expected because the response time is likely dominated by the retrial times when the setup time is short enough. This together with the observations in Section 5.2 show the trade-off between power consumption and delay-performance.
Interestingly, we observe in the case $\alpha = 1/100$ that the mean response time first decreases and then increases with the increase in $\mu$. The explanation for the first part (i.e. $\mu$ is small) is the same as for the case $\alpha = 1/10$. On the other hand, the reason for the monotonic increase of the mean response time in $\mu$ could
be explained as follows. If a job frequently retries, the job is likely blocked again because all the servers are either busy or in setup. As a result, the number of retrials increases leading to the increase in the response time. Our result suggests that if the mean setup time is relatively large (about 100 times of the service time), setting an appropriate retrial time could reduce both power consumption and response time.

![Figure 4. Mean response time versus retrial rate for \(c = 30, 50\).](image)

**Table 3.** Truncation point \(N\) and \(\hat{\pi}^{(N)} e\).

| \(\rho\) | \(c = 50\) | \(c = 100\) |
|---|---|---|
| | \(N\) | \(\hat{\pi}^{(N)} e\) | \(N\) | \(\hat{\pi}^{(N)} e\) |
| 0.1 | 39 | 2.3871 \times 10^{-19} | 39 | 1.3689 \times 10^{-25} |
| 0.2 | 66 | 3.3924 \times 10^{-15} | 66 | 1.0369 \times 10^{-17} |
| 0.3 | 92 | 3.7031 \times 10^{-13} | 92 | 3.1474 \times 10^{-14} |
| 0.4 | 118 | 9.5226 \times 10^{-12} | 118 | 4.7215 \times 10^{-12} |
| 0.5 | 144 | 1.4487 \times 10^{-10} | 144 | 2.1293 \times 10^{-10} |
| 0.6 | 170 | 1.7715 \times 10^{-09} | 170 | 5.4344 \times 10^{-09} |
| 0.7 | 197 | 1.7430 \times 10^{-08} | 196 | 1.0141 \times 10^{-07} |
| 0.8 | 228 | 1.0765 \times 10^{-07} | 227 | 1.0942 \times 10^{-06} |
| 0.9 | 349 | 1.5416 \times 10^{-06} | 321 | 7.4458 \times 10^{-07} |

5.4. **Power consumption against traffic intensity.** Figure 6 shows the power consumption versus the traffic intensity. Table 3 shows the truncation point \(N\) and \(\hat{\pi}^{(N)} e\) corresponding to Figure 6 by setting \(\epsilon_1 = 1.0 \times 10^{-12}\) in (2). The numerical values are obtained for \(\alpha = 1/10\) and \(\mu = 1\). For \(\rho \geq 0.6\), we observe that our heuristic technique for the truncation point provides larger values of \(\hat{\pi}^{(N)} e\) than those of \(\rho \leq 0.5\). Even for \(\rho \geq 0.6\), the heuristic technique could provide large values
of truncation point $N$ by adjusting $\epsilon_1$ so that we obtain sufficiently small values of $\tilde{\pi}^{(N)} e$ comparing with those of $\rho \leq 0.5$. However, we observed that selecting smaller $\epsilon_1$, e.g. $\epsilon_1 = 1.0 \times 10^{-15}$, gives us almost the same power consumption computed numerically by setting $\epsilon_1 = 1.0 \times 10^{-12}$. 

**Figure 5.** Mean response time versus retrial rate for $c = 30, 50$.

**Figure 6.** The power consumption versus traffic intensity for $\mu = 1$ and $\alpha = 1/10$. 

Figure 6 suggests that the ON-OFF policy consumes less energy than the ON-IDLE policy up to a moderate value of $\rho$. Furthermore, the ON-OFF policy is more effective as increasing the number of servers. This is also a consequence of the economies of scale.

5.5. **Effect of the buffer on the mean power consumption.** In this section, we consider the M/M/c/c+$K$ retrial queue with setup times. For this model, under the stability condition, the mean number of active servers is also given by $E[A] = cp$ due to the Little’s law. On the other hand, the mean number of servers in setup $E[S]$ is modified as follows.

$$E[S] = \sum_{k=0}^{\infty} \sum_{i=0}^{c+K} \sum_{j=0}^{\min(i,c)} \min(i-j,c-j) \pi_{i,j}^{(k)}.$$ 

We adopt the same truncation point presented in Section 4.1 for the M/M/c/c+$K$ model. Figure 7 shows the effect of the buffer on the mean power consumption for $K = 0, 10, 20$ and 30. We observe that the power consumption for the cases with buffer is greater than that for the case without buffer ($K = 0$). In the case with buffer, jobs have more chance to wait. As a reason, the servers have less change to be in the IDLE state and are more likely to be in setup than the case without buffer.

![Figure 7](image)

**Figure 7.** The power consumption versus retrial rate for $c = 50$ and $\alpha = 1/10$.

6. **Concluding remarks.** In this paper, we have analyzed a multiserver retrial queue with setup time for power-saving data centers. We have analyzed the queueing model by a level-dependent quasi birth-and-death process. Based on the Foster–Lyapunov criteria, we have established a sufficient condition of the stability for the multiserver retrial queue. We have shown that the level-dependent quasi birth-and-death process has the same structure as in [19] allowing us to compute the stationary
distribution efficiently. We have proposed a heuristic technique for determination of the truncation point which has shown to be practically useful. Numerical results have revealed that allowing retrials is less power-consuming than buffering jobs. Furthermore, the longer the retrial interval is, the less data centers consume power. We have also revealed that the power-saving (ON-OFF) policy is more effective compared with the ON-IDLE policy, if the mean setup time is sufficiently short in comparison with the mean service time of a job. Furthermore, we have found a new insight that if the setup time is relatively long, setting an appropriate retrial interval could lead to reducing both power consumption and response time.

Before concluding this paper, we remark on some side effects of retrials in the multiserver queue with setup time. As in Remark 6, retrials of jobs adversely affect the stability of the multiserver queues with setup time. Indeed, let us rewrite the stability condition (1) in Theorem 3.3 as

\[ \rho < 1 - \frac{1}{c + \alpha/\nu}. \]  

(3)

In case \( \alpha/\nu \ll 1 \), (3) implies that we have to reduce the incoming jobs by almost half to stabilize the system with \( c = 2 \). Fortunately, this side effect decreases as the number of servers \( c \) increases. From a managerial point of view, providing servers with shorter setup time is more beneficial for stable operation of data centers under ON-OFF policy in our model.

**Acknowledgments.** The authors would like to thank the two anonymous referees for their constructive and helpful comments which improved the presentation of the paper. T. Phung-Duc and K. Kawanishi were supported in part by JSPS KAKENHI Grant Numbers 26730011 and 26350416, respectively.

**REFERENCES**

[1] J. R. Artalejo and T. Phung-Duc, **Markovian retrial queues with two way communication**, *Journal of Industrial and Management Optimization*, 8 (2012), 781–806.
[2] L. A. Barroso and U. Höltle, **The case for energy-proportional computing**, *Computer*, 40 (2007), 33–37.
[3] L. Bright and P. G. Taylor, **Calculating the equilibrium distribution in level dependent quasi-birth-and-death processes**, *Stochastic Models*, 11 (1995), 497–525.
[4] J. Chang and J. Wang, **Unreliable M/M/1/1 retrial queues with set-up time**, *Quality Technology & Quantitative Management*, (2017), 1–13.
[5] J. E. Diamond and A. S. Alfa, **The MAP/PH/1 retrial queue**, *Stochastic Models*, 14 (1998), 1151–1177.
[6] J. E. Diamond and A. S. Alfa, **Matrix analytic methods for a multiserver retrial queue with buffer**, Top, 7 (1999), 249–266.
[7] G. I. Falin and J. G. C. Templeton, *Retrial Queues*, Chapman & Hall, London, 1997.
[8] A. Gandhi, M. Harchol-Balter and I. Adan, **Server farms with setup costs**, *Performance Evaluation* 67 (2010), 1123–1138.
[9] A. Gandhi, S. Doroudi and M. Harchol-Balter, **Exact analysis of the M/M/k/setup class of Markov chains via recursive renewal reward**, *Queueing Systems*, 77 (2014), 177–209.
[10] Q.-M. He, H. Li and Y. Q. Zhao, **Ergodicity of the BMAP/PH/s/s + K retrial queue with PH-retrial times**, *Queueing Systems*, 35 (2000), 323–347.
[11] G. Latouche and V. Ramaswami, **A logarithmic reduction algorithm for quasi-birth-death process**, *Journal of Applied Probability*, 30 (1993), 650–674.
[12] G. Latouche and V. Ramaswami, *Matrix Analytic Methods in Stochastic Modelling*, ASA-SIAM Series on Statistics and Applied Probability, SIAM, Philadelphia PA, 1999.
[13] E. Morozov, **A multiserver retrial queue: Regenerative stability analysis**, *Queueing Systems*, 56 (2007), 157–168.
[14] E. Morozov and T. Phung-Duc, Stability analysis of a multiclass retrial system with classical retrial policy, *Performance Evaluation*, 112 (2017), 15–26.

[15] M. F. Neuts, *Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach*, Dover Publications, Inc., New York, 1994.

[16] T. Phung-Duc, H. Masuyama, S. Kasahara and Y. Takahashi, A simple algorithm for the rate matrices of level-dependent QBD processes, in *Proceedings of the 5th International Conference on Queueing Theory and Network Applications*, (2010), 46–52.

[17] T. Phung-Duc, H. Masuyama, S. Kasahara and Y. Takahashi, A matrix continued fraction approach to multiserver retrial queues, *Annals of Operations Research*, 202 (2013), 161–183.

[18] T. Phung-Duc, Server farms with batch arrival and staggered setup, in *Proceedings of the Fifth symposium on Information and Communication Technology (SoICT 2014)*, (2014), 240–247.

[19] T. Phung-Duc and K. Kawanishi, An efficient method for performance analysis of blended call centers with redial, *Asia-Pacific Journal of Operational Research*, 31 (2014), 1440008 (33 pages).

[20] T. Phung-Duc, M/M/1/1 retrial queues with setup time, in *Proceedings of the 10th International Conference on Queueing Theory and Network Applications*, (2015), 93–104.

[21] T. Phung-Duc and K. Kawanishi, Impacts of retrials on power-saving policy in data centers, in *Proceedings of the 11th International Conference on Queueing Theory and Network Applications*, ACM, (2016), Article No. 22.

[22] T. Phung-Duc, Exact solutions for M/M/c/Setup queues, *Telecommunication Systems*, 64 (2017), 309–324.

[23] T. Phung-Duc, Single server retrial queues with setup time, *Journal of Industrial and Management Optimization*, 13 (2017), 1329–1345.

[24] Y. W. Shin, Stability of MAP/PH/c/K queue with customer retrials and server vacations, *Bulletin of the Korean Mathematical Society*, 53 (2016), 985–1004.

[25] R. L. Tweedie, Sufficient conditions for regularity, recurrence and ergodicity and Markov processes, *Mathematical Proceedings of the Cambridge Philosophical Society*, 78 (1975), 125–136.

Received February 2017; revised July 2017.

E-mail address: tuan@sk.tsukuba.ac.jp
E-mail address: kawanisi@cs.gunma-u.ac.jp