Multi-Server Queues with s-staggered Setup: A Faster Computational Approach

Thu Le-Anh[1] Tuan Phung-Duc[2]

[1]Graduate School of Science and Technology, University of Tsukuba, Japan
le.anh.thu.tkb_gf@u.tsukuba.ac.jp

[2]Institute of Systems and Information Engineering, University of Tsukuba, Japan
tuan@sk.tsukuba.ac.jp

Abstract In large-scale data centers, achieving high performance while limiting power consumption has become a significant challenge. To address this, we study the M/M/c/Setup queue with an s-staggered setup policy, which limits the number of servers that can enter setup at the same time. Using a generating-function method, we develop an efficient algorithm to evaluate system performance. We show that, for small s, the s-staggered model offers an accurate, low-cost approximation of the standard M/M/c/Setup queue. Unlike general methods such as the matrix-geometric approach, which become infeasible for large-scale systems, our method scales to thousands of servers. Numerical results confirm high accuracy, low computational cost, and demonstrate the energy–performance trade-offs enabled by the s-staggered setup policy.

1. Introduction

Demand for AI-driven services has sharply increased data center power consumption, driven by the need to operate large numbers of servers. As workloads vary over time, a portion of these servers often remains idle. Since idle servers draw approximately 60% of their peak power [2], shutting them down is widely adopted as an effective energy-saving policy. However, when new jobs arrive, these servers must be restarted, which causes delays and increases power consumption during the setup period. To limit these costs, a practical policy is to limit the number of servers that can be set up simultaneously, known as the s-staggered setup policy [4, 11]. A similar idea is used in large-scale storage systems through staggered disk spin-up, which sequences disk activation to avoid power surges and reduce stress on the power supply [5].

Gandhi et al. [4] first proposed the s-staggered setup policy and provided an approximate analysis for queueing systems with an infinite number of servers. Compared with the conventional M/M/c/Setup model [3, 9], this policy restricts the number of servers that can simultaneously be in the setup state to at most s. The policy has been studied with exact analysis for finite-buffer models [10] and for the special case $s = 1$ [1, 8]. However, an exact analysis for multiserver queues with an infinite buffer and general s has not yet been conducted. This motivates our study, in which we aim to improve computational efficiency for the s-staggered setup queueing model with an infinite buffer. In addition, we propose using the results as a fast approximation algorithm for M/M/c/Setup queues without the s-staggered setup policy, which have been analyzed in [3, 9].

Existing studies on the s-staggered setup policy have mainly concentrated on finite-buffer models. In particular, Phung-Duc and Kawanishi [10] analyzed the s-staggered setup policy in an M/M/c/K/Setup queue with abandonment and derived the stationary queue length distribution. They presented a simple recursive algorithm to compute the system's stationary distribution, achieving a computational complexity of $O(cK)$. The same authors [11] further extended this analysis to cases where some servers are allowed to remain idle. They obtained a phase-type expression for the stationary waiting-time distribution by modeling the virtual waiting process

as an absorbing Markov chain.

This paper provides an extensive analysis of the s-staggered setup model with an infinite buffer. By exploiting the special structure of the underlying Markov chain, we employ a generating-function approach to derive the joint queue-length distribution with low computational complexity. In particular, the Markov chain in our model is unidirectional in phase, which allows an effective recursive algorithm. Similar Markov chain structures have been observed in the models presented in [3, 9, 7]. While the algorithms in [10, 11] for finite-buffer models have computational complexity $O(cK)$, our proposed algorithm for the infinite-buffer model has complexity $O(c^2)$. Hence, our approach is particularly useful for approximating the M/M/c/K model when K is large. Furthermore, when a general method such as the matrix-geometric method [6] is employed, the computational complexity is $O(c^6)$. It is worth noting that when $s = c$, the s-staggered setup model becomes the M/M/c/Setup queueing model. An interesting observation is that the performance of the s-staggered setup model converges rapidly as s increases. This property enables us to efficiently approximate the M/M/c/Setup model using relatively small values of s. Although the computational complexity of both models under the generating function approach remains $O(c^2)$, the running time of our approximation algorithm based on the s-staggered setup model is roughly half that of the M/M/c/Setup model, since the required value of s is negligible compared to c.

The rest of the paper is organized as follows. Section 2 describes the model in detail and Section 3 presents the analysis of the model via generating functions. Section 4 is devoted to performance measures and numerical experiments. Finally, concluding remarks are presented in Section 5.

2. MODEL

2.1. Model description

We now describe the M/M/c/Setup model with the s-staggered setup policy. Jobs arrive according to a Poisson process with rate λ, and service times are exponentially distributed with mean $1/\mu$. When a job arrives, and at least one server is idle, that server begins a setup process. The setup time is exponentially distributed with mean $1/\alpha$. The number of servers allowed to be in setup at the same time is limited to s ($s \leq c$). When a service is completed, and no jobs are waiting, the server is immediately turned off; otherwise, it serves the next job in FCFS order. It is important to note that the job might be waiting for the setup of the other server to complete. In such a case, the server currently in setup is turned off if no other jobs are waiting, meaning we are setting up more than needed. If there is at least one job waiting, then the server continues the setup.

2.2. Markov chain and notation

It can be seen that the stability condition for the system is $\lambda < c\mu$, as all servers eventually are active if the number of jobs in the system is large enough. Let $N(t)$ and $L(t)$ denote the number of active servers and the number of jobs in the system at time t, respectively. The two-dimensional process $\{X(t) = (N(t), L(t)); t \geq 0\}$ forms a Markov chain in the state space

$$S = \{(i, j); i = 0, 1, \ldots, c, j = i, i + 1, \ldots\}.$$

Figure 1 shows the state transition diagram of the Markov chain.

Let $\pi_{i,j}$ denote the stationary probability of state (i, j), i.e.,

$$\pi_{i,j} = \lim_{t\to\infty} \mathbb{P}(N(t) = i, L(t) = j), \quad (i, j) \in S.$$

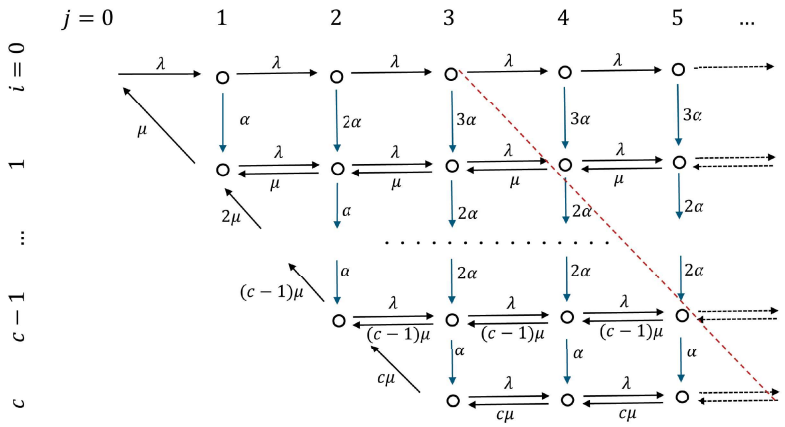


Figure 1: State-transition diagram ($s = 3$).

We define generating functions $\Pi_i(z) = \sum_{j=i}^{\infty} \pi_{i,j} z^{j-i}$, for $i = 0, 1, \ldots, c$. We are interested in finding explicitly the generating functions $\Pi_i(z)$, the factorial moments defined by $\Pi_i^{(n)}(1)$, where $f^{(n)}(x)$ denotes the n-th derivative of $f(x)$, and $\pi_{i,j}$ for $(i, j) \in S$.

Definition 1. *For $x \in \mathbb{R}$, the Pochhammer symbol is defined as follows:*

$$(x)_n = \begin{cases} 1 & n = 0, \\ \prod_{k=1}^{n}(x+k-1) & n \in \mathbb{N} = \{1, 2, 3, \ldots\}. \end{cases}$$

3. GENERATING FUNCTION

To demonstrate our algorithm's computational complexity, Figure 2 shows a geometric representation of the state space, highlighting the non-homogeneous and homogeneous regions where computations are required. This division arises from the flow balance structure of the Markov chain: states in the homogeneous region share identical flow-in and flow-out rates, whereas states in the non-homogeneous region exhibit state-dependent flow structures.

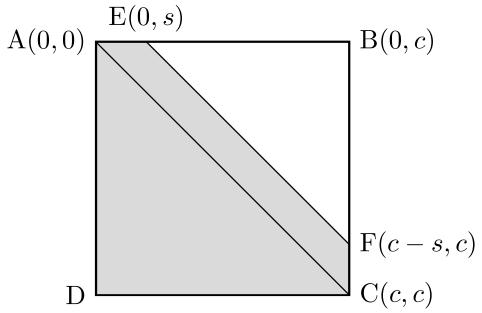


Figure 2: State space structure

In the generating-function–based algorithm for the M/M/c/Setup model in [9], the state probabilities $\pi_{i,j}$ $(0 \le i \le j < c)$ form the non-homogeneous part (area ABC). An equal number of computations is required for the homogeneous part (area ACD), as will be clarified in the following part. Thus, the total number of operations is $2\sum_{i=0}^{c} i = c(c+1) = O(c^2)$.

In the s-staggered model, the non-homogeneous region shrinks from the ABC area to the AEFC area, that is, the non-homogeneous part consists of the state probability $\pi_{i,j}$ $(0 \le i \le j < \min(s+i, c))$, while the homogeneous region remains unchanged. Although the overall complexity stays at $O(c^2)$, the number of operations is roughly halved for small s relative to c.

The balance equations for the case $i = 0$ are given by

$$\lambda\pi_{0,0} = \mu\pi_{1,1}, \quad j = i, \tag{1}$$

$$(\lambda + j\alpha)\pi_{0,j} = \lambda\pi_{0,j-1}, \quad j = 1, 2, \ldots, s-1, \tag{2}$$

$$(\lambda + s\alpha)\pi_{0,j} = \lambda\pi_{0,j-1}, \quad j \geq s. \tag{3}$$

Let $\widehat{\Pi}_0(z) = \sum_{j=s}^{\infty} \pi_{0,j} z^j$. Multiplying (3) by z^j and summing over $j \geq s$, we have

$$\widehat{\Pi}_0(z) = \frac{\lambda\pi_{0,s-1}z^s}{\lambda + s\alpha - \lambda z} = z^s \frac{A_{0,0}}{\widehat{z}_0 - z}, \tag{4}$$

where $A_{0,0} = \pi_{0,s-1}, \widehat{z}_0 = \frac{\lambda + s\alpha}{\lambda}$.

Eq. (2) yields

$$\pi_{0,j} = \pi_{0,0} \prod_{i=0}^{j} \frac{\lambda}{\lambda + j\alpha}, \quad j = 1, 2, \ldots, s-1.$$

From (4), we have

$$\pi_{0,j} = \frac{A_{0,0}}{\widehat{z}_0} \left(\frac{1}{\widehat{z}_0}\right)^{j-s}, \quad \text{for } j \geq s.$$

Remark 1. *At this moment, $\pi_{0,j}$ $(j \geq 1)$ are expressed in terms of $\pi_{0,0}$. Furthermore, $\pi_{1,1}$ is expressed in terms of $\pi_{0,j}$ $(j \geq 1)$ and then in terms of $\pi_{0,0}$ via the balance of the flows in and out the set of states $\{(0, j); j \geq 0\}$, i.e.,*

$$\mu\pi_{1,1} = \sum_{j=1}^{\infty} \min(j, s)\alpha\pi_{0,j} = \sum_{j=1}^{s-1} j\alpha\pi_{0,j} + s\alpha\widehat{\Pi}_0(1). \tag{5}$$

Differentiating (4) n times and substituting $z = 1$ yields the following recursive formulae for the factorial moments.

$$\begin{aligned} \widehat{\Pi}_0^{(n)}(1) =& \frac{n\lambda}{s\alpha}\widehat{\Pi}_0^{(n-1)}(1) + \frac{\lambda}{s\alpha}\pi_{0,s-1}(s-n+1)_n, \\ \Pi_0^{(n)}(1) =& \sum_{j=0}^{s-1} \pi_{0,j}(j-n+1)_n + \widehat{\Pi}_0^{(n)}(1), \quad \text{for } n \in \mathbb{N}. \end{aligned} \tag{6}$$

Now, we consider the cases $i = 1, 2, \ldots, c - s$. The balance equations are as follows.

$$(\lambda + i\mu)\pi_{i,i} = \alpha\pi_{i-1,i} + i\mu\pi_{i,i+1} + (i+1)\mu\pi_{i+1,i+1}, \quad j = i, \tag{7}$$

$$(\lambda + i\mu + (j-i)\alpha)\pi_{i,j} = (j-i+1)\alpha\pi_{i-1,j} + \lambda\pi_{i,j-1} + i\mu\pi_{i,j+1}, \quad i+1 \leq j < s+i, \tag{8}$$

$$(\lambda + i\mu + s\alpha)\pi_{i,j} = \lambda\pi_{i,j-1} + i\mu\pi_{i,j+1} + s\alpha\pi_{i-1,j}, \quad j \geq s+i. \tag{9}$$

Let $\widehat{\Pi}_i(z) = \sum_{j=s+i}^{\infty} \pi_{i,j} z^{j-i}$. Multiplying (9) by z^j and summing over $j \geq s + i$, we have

$$\begin{aligned} \Big(-\lambda z^2 + \big(\lambda + i\mu + \min(s\alpha, (c-i)\alpha)\big)z - i\mu \Big)\widehat{\Pi}_i(z) = s\alpha\widehat{\Pi}_{i-1}(z) - s\alpha\pi_{i-1,s+i-1}z^s \\ + \lambda\pi_{i,s+i-1}z^{s+1} - i\mu\pi_{i,s+i}z^s. \end{aligned} \tag{10}$$

Let $f_i(z) = -\lambda z^2 + \big(\lambda + i\mu + \min(s\alpha, (c-i)\alpha)\big)z - i\mu$. Because $f_i(0) < 0$, $f_i(1) > 0$ and $f_i(\infty) = -\infty$, $f_i(z)$ has two roots z_i and $\hat{z}_i$ such that $0 < z_i < 1 < \widehat{z}_i$.

Substituting $z = z_i$ into (10) yields

$$\pi_{i,s+i} = a_{s+i}^{(i)} + b_{s+i}^{(i)}\pi_{i,s+i-1},$$

where

$$a_{s+i}^{(i)} = \frac{s\alpha}{i\mu}\big(\frac{\widehat{\Pi}_{i-1}(z_i)}{z_i^s} - \pi_{i-1,s+i-1}\big), \quad b_{s+i}^{(i)} = \frac{\lambda z_i}{i\mu}. \tag{11}$$

Lemma 1.

$$\pi_{i,j} = a_j^{(i)} + b_j^{(i)} \pi_{i,j-1}, \quad j = i+1, i+2, \ldots, i+s, \tag{12}$$

where

$$a_j^{(i)} = \frac{i\mu a_{j+1}^{(i)} + (j-i+1)\alpha\pi_{i-1,j}}{\lambda + i\mu + (j-i)\alpha - i\mu b_{j+1}^{(i)}}, \quad b_j^{(i)} = \frac{\lambda}{\lambda + i\mu + (j-i)\alpha - i\mu b_{j+1}^{(i)}},$$

for $j = s+i-1, s+i-2, \ldots, i+1$.

The generating function $\widehat{\Pi}_i(z)$ $(i = 1, 2, \ldots, c-s)$ *is explicitly obtained by*

$$\widehat{\Pi}_i(z) = z^s \sum_{j=0}^{i} \frac{A_{i,j}}{\widehat{z_j} - z}, \tag{13}$$

where

$$A_{i,j} = s\alpha \frac{A_{i-1,j}\widehat{z_j}}{f_i(\widehat{z_j})}, \quad A_{i,i} = \pi_{i,s+i-1} - s\alpha \sum_{j=0}^{i-1} \frac{A_{i-1,j}\widehat{z_j}}{f_i(\widehat{z_j})}.$$

The proof of Lemma 1 is given in Appendix A.1.

Substituting $\widehat{\Pi}_i(z)$ in the form of (13) into (11), we have

$$a_{s+i}^{(i)} = \frac{s\alpha}{i\mu} \left(\sum_{j=0}^{i-1} \frac{A_{i,j}}{\hat{z}_j - z_i} - \pi_{i-1,s+i-1} \right).$$

By rewriting $a_{s+i}^{(i)}$ $(i = 1, \ldots, c-s)$, we eliminate denominator singularities caused by the fact that $z_i^s \to 0$ as $s \to \infty$.

Remark 2. *At this moment,* $\pi_{i,j} (1 \le i \le c-s, j \ge i)$ *is expressed in terms of* $\pi_{0,0}$. *In addition,* $\pi_{i+1,i+1}$ *is expressed in terms of* $\pi_{i,j}$ $(j \ge i)$ *and then in terms of* $\pi_{0,0}$ *via the balance of the flows in and out the set of states* $\{(m,j); 0 \le m \le i, j \ge m\}$, *i.e.,*

$$(i+1)\mu\pi_{i+1,i+1} = \sum_{j=i+1}^{\infty} \min(j-i, s)\alpha\pi_{i,j} = \sum_{j=i+1}^{s+i-1} (j-i)\alpha\pi_{i,j} + s\alpha\widehat{\Pi}_i(1).$$

Taking the derivative of (10) and substituting $z = 1$ yields

$$\begin{aligned}
\widehat{\Pi}_i^{(n)}(1) &= \widehat{\Pi}_{i-1}^{(n)}(1) + \frac{n(\lambda - i\mu - s\alpha)\widehat{\Pi}_i^{(n-1)}(1) + n(n-1)\lambda\widehat{\Pi}_i^{(n-2)}(1)}{s\alpha} \\
&\quad - \pi_{i-1,s+i-1}(s-n+1)_n + \frac{\lambda\pi_{i,i+s-1}(s-n+2)_n - i\mu\pi_{i,s+i}(s-n+1)_n}{s\alpha}, \\
\Pi_i^{(n)}(1) &= \sum_{j=i}^{s+i-1} \pi_{i,j}(j-i-n+1)_n + \widehat{\Pi}_i^{(n)}(1).
\end{aligned} \tag{14}$$

Next, we consider the case $i = c-s+1, c-s+2, \ldots, c-1$. Balance equations are given by

$$(\lambda + i\mu)\pi_{i,i} = \alpha\pi_{i-1,i} + i\mu\pi_{i,i+1} + (i+1)\mu\pi_{i+1,i+1}, \quad j = i, \tag{15}$$

$$(\lambda + i\mu + (j-i)\alpha)\pi_{i,j} = (j-i+1)\alpha\pi_{i-1,j} + \lambda\pi_{i,j-1} + i\mu\pi_{i,j+1}, \quad i+1 \le j < c-1, \tag{16}$$

$$(\lambda + i\mu + (c-i)\alpha)\pi_{i,j} = (c-i+1)\alpha\pi_{i-1,j} + \lambda\pi_{i,j-1} + i\mu\pi_{i,j+1}, \quad j \ge c. \tag{17}$$

We define $\widehat{\Pi}_i(z) = \sum_{j=c}^{\infty} \pi_{i,j} z^{j-i}$, for $c - s < i < c$.

Multiplying (17) by z^{j-i} and summing over $j \geq c$, we obtain

$$(-\lambda z^2 + (\lambda + i\mu + (c-i)\alpha)z - i\mu)\widehat{\Pi}_i(z) = (c-i+1)\alpha\widehat{\Pi}_{i-1}(z) + \lambda\pi_{i,c-1}z^{c-i+1} - i\mu\pi_{i,c}z^{c-i}. \tag{18}$$

Similar to the previous cases, substituting $z = z_i$ into (18) yields

$$\pi_{i,c} = a_c^{(i)} + b_c^{(i)}\pi_{i,c-1},$$

where

$$a_c^{(i)} = \frac{(c-i+1)\alpha\widehat{\Pi}_{i-1}(z_i)}{i\mu z_i^{c-i}}, \quad b_c^{(i)} = \frac{\lambda z_i}{i\mu}. \tag{19}$$

Then, we derive a recursive scheme to determine $\pi_{i,j}$ for $c-s+1 \leq i \leq c-1, \quad i+1 \leq j \leq c$, as expressed by the following lemma.

Lemma 2.

$$\pi_{i,j} = a_j^{(i)} + b_j^{(i)}\pi_{i,j-1}, \quad j = i+1, i+2, \ldots, c, \tag{20}$$

where $a_j^{(i)}$ and $b_j^{(i)}$ are given in Lemma 1.

The generating function $\widehat{\Pi}_i(z)$ $(c-s+1 \leq i \leq c-1)$ is explicitly obtained by

$$\widehat{\Pi}_i(z) = z^{c-i}\sum_{j=0}^{i}\frac{A_{i,j}}{\widehat{z}_j - z}, \tag{21}$$

where

$$A_{i,j} = (c-i+1)\alpha\frac{A_{i-1,j}\widehat{z}_j}{f_i(\widehat{z}_j)}, \quad A_{i,i} = \pi_{i,c-1} - (c-i+1)\alpha\sum_{j=0}^{i-1}\frac{A_{i-1,j}\widehat{z}_j}{f_i(\widehat{z}_j)}.$$

Proof. The proof is given in Appendix A.2. □

Substituting $\widehat{\Pi}_i(z)$ in the form of (21) into (19), we have

$$a_c^{(i)} = \frac{(c-i+1)\alpha}{i\mu}\sum_{j=0}^{i-1}\frac{A_{i,j}}{\hat{z}_j - z_i}.$$

Remark 3. *At this moment, $\pi_{i,j}(c-s+1 \leq i \leq c-1, j \geq i)$ is expressed in terms of $\pi_{0,0}$. In addition, $\pi_{i+1,i+1}(c-s+1 \leq i \leq c-1)$ is expressed in terms of $\pi_{i,j}$ $(j = i+1, i+2, \ldots)$ and then in terms of $\pi_{0,0}$.*

Finally, we consider the case $i = c$. Balance equations are given by

$$(\lambda + c\mu)\pi_{c,c} = \alpha\pi_{c-1,c} + (c\mu)\pi_{c,c+1}, \tag{22}$$

$$(\lambda + c\mu)\pi_{c,j} = \lambda\pi_{c,j-1} + \alpha\pi_{c-1,j} + c\mu\pi_{c,j+1}, \quad j \geq c+1. \tag{23}$$

Let $\widehat{\Pi}_c(z) = \sum_{j=c}^{\infty}\pi_{c,j}z^{j-c}$, we have $\Pi_c(z) = \widehat{\Pi}_c(z)$. Multiplying (23) by z^{j-c} and summing over $j \geq c$ yields

$$(-\lambda z^2 + (\lambda + c\mu)z - c\mu)\widehat{\Pi}_c(z) = \alpha\widehat{\Pi}_{c-1}(z) - c\mu\pi_{c,c}. \tag{24}$$

Let $f_c(z) = -\lambda z^2 + (\lambda + c\mu)z - c\mu = (z-1)(c\mu - \lambda z)$. Then

$$\widehat{\Pi}_c(z) = \frac{\alpha(\widehat{\Pi}_{c-1}(z) - \widehat{\Pi}_{c-1}(1))}{z-1} \frac{1}{c\mu - \lambda z}. \tag{25}$$

Applying l'Hopital's rule, we obtain

$$\widehat{\Pi}_c(1) = \frac{\alpha \widehat{\Pi}'_{c-1}(1)}{c\mu - \lambda}.$$

Substituting $\widehat{\Pi}_{c-1}(z)$ in the form of (21) with $i = c-1$ into (24), we obtain

$$\widehat{\Pi}_c(z) = \sum_{j=0}^{c} \frac{A_{c,j}}{\widehat{z}_j - z}, \tag{26}$$

where

$$\widehat{z}_c = \frac{c\mu}{\lambda}, \quad A_{c,j} = \alpha \frac{A_{c-1,j}\widehat{z}_j}{f_c(\widehat{z}_j)}, \quad j = 0, 1, \ldots, c-1, \quad A_{c,c} = -\alpha \sum_{j=0}^{c-1} \frac{A_{c-1,j}\widehat{z}_j}{f_c(\widehat{z}_j)}.$$

Differentiating (24) n times and rearranging the results, and then applying l'Hopital's rule yields

$$\Pi_c^{(n)}(1) = \frac{\alpha \widehat{\Pi}_{c-1}^{(n+1)}(1) + n(n+1)\lambda \Pi_c^{(n-1)}(1)}{(n+1)(c\mu - \lambda)}. \tag{27}$$

It can be noted that $\Pi_{c-1}^{(n+1)}(1)$ and $\Pi_c^{(0)}(1) = \Pi_c(1)$ are already known.

At this moment, $\pi_{i,j}$ $(j \le c)$ and the generating functions $\widehat{\Pi}_i(z)$ $(i = 0, 1, \ldots, c)$ are expressed in terms of $\pi_{0,0}$ which is uniquely determined using the following normalization condition.

$$\Pi_0(1) + \Pi_1(1) + \ldots + \Pi_c(1) = 1.$$

Remark 4. *Explicit results for the factorial moments and the probabilities $\pi_{i,j}$ $(j \ge i + s)$ can be derived either through the recursive equations (6), (14), and (27), or by using the quantities $A_{i,j}$ $(0 \le i \le j \le c)$ and $\hat{z}_i$ $(i = 0, 1, \ldots, c)$ since expressions for the generating functions are given in (4), (13) and (21), (26). The computational complexity of the homogeneous part, which corresponds to the number of coefficients $A_{i,j}$, is $\sum_{i=0}^{c} i = c(c+1)/2$. Thus, it requires the same order of computations as the non-homogeneous part in the generating-function algorithm of [9], represented by the ACD region in Figure 2.*

4. PERFORMANCE MEASURES AND NUMERICAL EXAMPLES

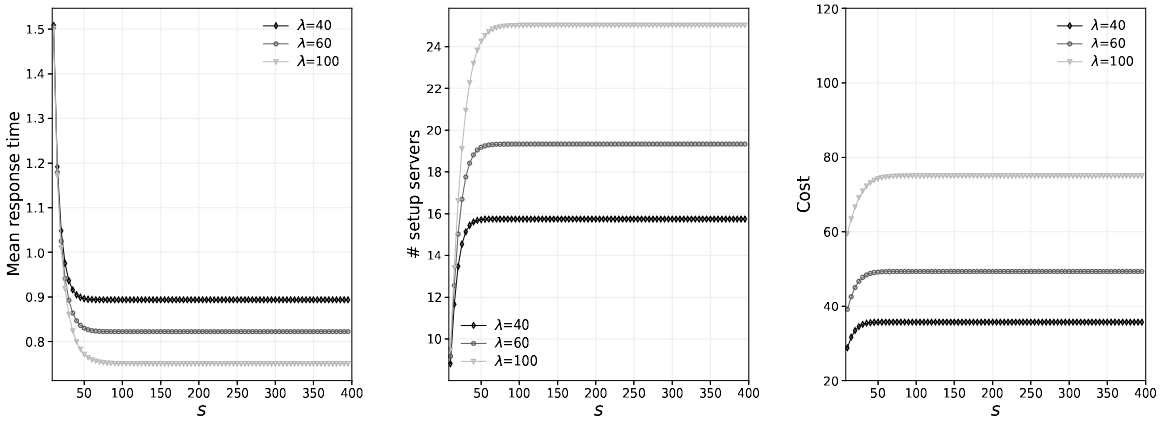
4.1. Performance measures

Let L be the number of jobs in the system, B be the number of busy servers, and M be the number of setup servers. Then, the mean values $\mathrm{E}[L]$, $\mathrm{E}[B]$ and $\mathrm{E}[M]$ are given by

$$\mathrm{E}[L] = \sum_{(i,j)\in S} j\pi_{i,j} = \sum_{i=0}^{c} \Pi_i'(1),$$

$$\mathrm{E}[B] = \sum_{(i,j)\in S} i\pi_{i,j} = \sum_{i=0}^{c} i\Pi_i(1),$$

$$\mathrm{E}[M] = \sum_{(i,j)\in S} \min(s, j-i, c-i)\pi_{i,j} = \sum_{i=0}^{c} \Big(\sum_{j=i}^{s+i-1} \min(j-i, c-i)\pi_{i,j} + \min(s, c-i)\Pi_i(1) \Big).$$



(a) Mean response time (b) Mean number of setup servers (c) Power consumption

Figure 3: Performance measures versus the maximum setup server s.

From Little's law, the average response time is given by

$$\mathrm{E}[R] = \frac{1}{\lambda}\mathrm{E}[L].$$

The power consumption is given by

$$Cost = C_b\mathrm{E}[B] + C_m\mathrm{E}[M],$$

where C_b and C_m are the cost per unit time for a busy server and a setup server, respectively.

4.2. Numerical examples

In this section, we present performance measures obtained from our analysis to illustrate the efficiency of the proposed algorithm for the s-staggered setup model. All numerical experiments are conducted for an extensive system with $c = 400$ servers. The mean service time is set to $1/\mu = 1$s, the mean setup time to $1/\alpha = 10$s, and we take $C_b = C_m = 1$.

Figure 3 illustrates the mean response time, the mean number of setup servers, and the power consumption cost as functions of the maximum number of setup servers s under different workload levels. As shown in Figure 3, the mean response time decreases as s increases, whereas both the mean number of setup servers and the power consumption cost increase with s. Intuitively, allowing more servers to enter setup in parallel improves response time but also increases power consumption. Consequently, a fundamental trade-off arises between system performance and energy usage. Furthermore, Figure 3 shows that the performance measures change rapidly for small values of s, and then remain unchanged as s approaches c. It should be noted that when $s = c$, the s-staggered setup model becomes equivalent to M/M/c/Setup model. This observation suggests that the M/M/c/Setup model can be accurately approximated by an s-staggered setup model with a relatively small value of s.

Next, we use the algorithm developed in Section 3 for the s-staggered setup model to approximate the M/M/c/Setup model. We compare the computational complexity of our approximation with algorithms for the M/M/c/Setup model based on the generating-function approach and the matrix-geometric method. All running times are measured on a PC equipped with an Intel Core i5-8400 2.81 GHz CPU and 8 GB of RAM, running Windows 10 Pro and Python 3.10.9.

Table 1 reports the running times required to compute performance measures using three methods: (1) our approximation, which selects the smallest value of s that achieves an error below 0.1%; (2) the generating-function (GF) method; and (3) the matrix-geometric (MG) method.

Table 1: Running times for the M/M/c/Setup model (sec.). The s values correspond to the smallest s achieving approximation error $< 0.1\%$.

Number of servers	**Approximation (s-staggered)**	**Generating-function method [9]**	**Matrix-geometric method**
100	0.02 ($s = 18$)	0.03	7.67
200	0.06 ($s = 25$)	0.11	332.16
500	0.36 ($s = 37$)	0.66	memory exceeded
1000	1.22 ($s = 51$)	2.23	memory exceeded
2000	5.32 ($s = 95$)	10.87	memory exceeded
5000	31.39 ($s = 153$)	58.98	memory exceeded
10000	122.71 ($s = 324$)	248.63	memory exceeded

The results show that both our approximation and the GF method consistently outperform the MG method, particularly as the number of servers increases. For instance, when $c = 200$, the MG method is roughly 3000 times slower than the GF method and about 4600 times slower than our approximation. Under our computing environment, the MG method becomes impractical for large-scale systems, especially beyond 200 servers. In contrast, both our approximation and the GF method remain computationally efficient even for systems with up to 10,000 servers. Moreover, our approximation performs almost identically to the GF method while requiring only about half of the running time.

5. CONCLUSIONS

This paper presented a comprehensive study of the M/M/c/Setup queueing model under the s-staggered setup policy, motivated by the need to improve energy efficiency in large-scale data centers. Using the generating-function approach, we developed an efficient algorithm to compute key performance measures for the s-staggered setup model. We further showed that, by choosing a suitably small value of s, the s-staggered model provides an accurate and computationally efficient approximation of the M/M/c/Setup model. Our approximation scales well to large systems, whereas the matrix-geometric method becomes impractical when the number of servers exceeds a few hundred. Moreover, our approach consistently outperforms the generating-function method for the M/M/c/Setup model, requiring roughly half the running time while maintaining very high accuracy. Numerical experiments demonstrated that the s-staggered setup policy offers valuable insights into the trade-offs between performance and energy consumption in power-saving server farms.

A. Proofs of Section 3

A.1. Proof of Lemma 1

Proof. We use induction for the proof of this lemma. It is easy to see that (12) is true for $j = s + i$. Assuming that (12) holds in the case $j + 1$, i.e., $\pi_{i,j+1} = a^{(i)}_{j+1} + b^{(i)}_{j+1}\pi_{i,j}$, for some $j < s + i$. Substituting this expression into (8) and rearranging the result, we derive (12).

Next, we prove (13) using mathematical induction. Substituting

$$\widehat{\Pi}_{i-1}(z) = z^s \sum_{j=0}^{i-1} \frac{A_{i-1,j}}{\hat{z}_j - z}$$

into (10) and rearranging the result, we obtain (13). The following formula is used to decompose

$\widehat{\Pi}_i(z)$ into a simple form:

$$\frac{1}{(a-z)(b-z)} = \frac{1}{b-a}\left(\frac{1}{a-z} - \frac{1}{b-z}\right), \quad a \neq b.$$

□

A.2. Proof of Lemma 2

Proof. With arguments similar to those used for Lemma 1, Lemma 2 can be proved by induction. Subtituting

$$\widehat{\Pi}_{i-1}(z) = z^{c-i+1} \sum_{j=0}^{i-1} \frac{A_{i-1,j}}{\hat{z}_j - z}$$

into (18) and rearranging the result, we obtain (26). □

References

[1] Artalejo, J. R., Economou, A., and Lopez-Herrero, M. J. Analysis of a multiserver queue with setup times. *Queueing Systems 51*, 1 (2005), 53–76.

[2] Barroso, L. A., and Hölzle, U. The case for energy-proportional computing. *Computer 40*, 12 (2007), 33–37.

[3] Gandhi, A., Doroudi, S., Harchol-Balter, M., and Scheller-Wolf, A. Exact analysis of the M/M/k/setup class of markov chains via recursive renewal reward. In *Proceedings of the ACM SIGMETRICS/International Conference on Measurement and Modeling of Computer Systems* (2013), pp. 153–166.

[4] Gandhi, A., Harchol-Balter, M., and Adan, I. Server farms with setup costs. *Performance Evaluation 67*, 11 (2010), 1123–1138.

[5] Intel Corporation. Serial ATA staggered spin-up. White Paper, Sept. 2004.

[6] Le-Anh, T., and Phung-Duc, T. Analysis of multi-server queueing systems with setup times and power-saving modes. In *proceedings of the 17th EAI International Conference on Performance Evaluation Methodologies and Tools, Milan, Italy, December 12-13* (2024), pp. 271–278.

[7] Le-Anh, T., and Phung-Duc, T. A fast algorithm for multiserver queueing systems with setup times and power-saving modes. In *Proceedings of 36th International Teletraffic Congress (ITC-36), Trondheim, Norway, June 02-05* (2025), IEEE, pp. 1–9.

[8] Phung-Duc, T. Server farms with batch arrival and staggered setup. In *Proceedings of the 5th Symposium on Information and Communication Technology* (2014), pp. 240–247.

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

[10] Phung-Duc, T., and Kawanishi, K. Energy-aware data centers with s-staggered setup and abandonment. In *Analytical and Stochastic Modelling Techniques and Applications: 23rd International Conference, ASMTA 2016, Cardiff, UK, August 24-26, 2016, Proceedings 23* (2016), Springer, pp. 269–283.

[11] Phung-Duc, T., and Kawanishi, K. Delay performance of data-center queue with setup policy and abandonment. *Annals of Operations Research 293*, 1 (2020), 269–293.

Acknowledgments

This work was supported by the Research Institute for Mathematical Sciences, an International Joint Usage/Research Center located in Kyoto University.