*axioms*

MDPI

*Article*

# Analysis of Bulk Queueing Model with Load Balancing and Vacation

**Subramani Palani Niranjan [1], Suthanthiraraj Devi Latha [1,2,*], Sorin Vlase [3,4,*] and Maria Luminita Scutaru [3]**

[1] Department of Mathematics, Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Chennai 600062, India; drniranjansp@veltech.edu.in
[2] Department of Mathematics, Vel Tech Ranga Sanku Arts College, Chennai 600062, India
[3] Department of Mechanical Engineering, Transilvania University of Brasov, Romania, B-dul Eroilor, 29, 500036 Brașov, Romania; lscutaru@unitbv.ro
[4] Romanian Acad, Inst Solid Mech, Str C-tin Mille 15, 010141 Bucharest, Romania
* Correspondence: devilatha451@velsrscollege.com (S.D.L.); svlase@unitbv.ro (S.V.); Tel.: +40-722-643020 (S.V.)

**Abstract:** Data center architecture plays an important role in effective server management network systems. Load balancing is one such data architecture used to efficiently distribute network traffic to the server. In this paper, we incorporated the load-balancing technique used in cloud computing with power business intelligence (BI) and cloud load based on the queueing theoretic approach. This model examines a bulk arrival and batch service queueing system, incorporating server overloading and underloading based on the queue length. In a batch service system, customers are served in groups following a general bulk service rule with the server operating between the minimum value '***a***' and the maximum value '*b*'. But in certain situations, maintaining the same extreme values of the server is difficult, and it needs to be changed according to the service request. In this paper, server load balancing is introduced for a batch service queueing model, which is the capacity of the server that can be adjusted, either increased or decreased, based upon the service request by the customer. On service completion, if the service request is not enough to start any of the services, the server will be assigned to perform a secondary job (vacation). After vacation completion based upon the service request, the server will start regular service, overload or underload. Cloud computing using power BI can be analyzed based on server load balancing. The function that determines the probability of the queue size at any given time is derived for the specified queueing model using the supplementary variable technique with the remaining time as the supplementary variable. Additionally, various system characteristics are calculated and illustrated with suitable numerical examples.

**Keywords:** overloading; underloading; load balancing; batch service; supplementary variable technique

**MSC:** 60K30; 68M20



Academic Editor: Jiujun Zhang

Received: 23 November 2024
Revised: 23 December 2024
Accepted: 25 December 2024
Published: 30 December 2024

**Citation:** Niranjan, S.P.; Latha, S.D.; Vlase, S.; Scutaru, M.L. Analysis of Bulk Queueing Model with Load Balancing and Vacation. *Axioms* **2025**, *14*, 18. https://doi.org/10.3390/axioms14010018



## 1. Introduction

The efficient management of servers in network systems is one of the challenging issues for improving service quality. Many mathematical techniques are implemented to study these issues and provide possible solutions. Server load balancing is an effective technique for efficiently managing system traffic. The modeling and analysis of symmetric bulk arrival queueing systems has attracted a lot of scientific attention. Takagi [1] gave a detailed explanation of symmetric bulk queueing systems both with and without server vacations.

Numerous writers have also examined various combinations of queueing schemes and server vacations. Doshi [2] first carried out an extensive analysis of queueing systems with vacations, highlighting load balancing as one of their applications. Using the supplemental variable technique, Lee et al. [3] examined a $M^X/G/1$ queueing system with *N* policy and multiple vacations. A queueing model with batch arrivals and a threshold was covered by Lee et al. [4]. Numerous academics have examined the steady-state analysis of a queueing system with non-Markovian symmetric bulk arrivals, batch service, vacation periods, and certain extensions. The paper by Krishna Reddy et al. [5] explores the operational dynamics of a queueing system characterized by symmetric bulk arrivals, an *N*-policy for managing queue length, multiple vacation periods for servers, and setup times between tasks. Using the supplementary variable technique, the authors analyze various aspects of system performance, including queue length distributions, server utilization rates, and overall system efficiency. Their approach not only contributes to theoretical advances in queueing theory but also provides practical insights into optimizing system parameters to enhance operational efficiency and minimize delays. A cost model for a symmetric bulk arrival and batch service queueing system without service interruption was also looked into by Jeyakumar and Senthilnathan [6]. In order to incorporate closedown times into their investigation, the author deduced the probability-generating function for the service process in their model. Arumuganathan and Jeyakumar [7] extended the above work with closedown times. The optimal cost analysis and certain performance characteristics of the $M^x/G(a,b)/1$ queueing system with interrupted vacation have been derived by Haridass and Arumuganathan [8]. Siddiqui et al. [9] present the QPSL queuing model as a notable advancement in load balancing within cloud computing. This approach improves the efficacy and efficiency of resource allocation in cloud environments by utilizing queuing theory, most especially the M/M/k queue with a FIFO discipline. The study highlights the model's potential to optimize performance, reduce latency, and improve overall system reliability through sophisticated load-balancing mechanisms. The analysis of queueing systems with breakdown is essential to deal with many real-time systems. Using a recursive technique, Praveen Kumar Agarwal et al. [10] investigated the best N-policy for a finite queue with server failures and state-dependent service rates.

A queueing system with batch services and symmetric bulk arrivals was presented by Niranjan et al. [11], adding queue size-dependent vacations for an unreliable server. An analysis of a symmetric bulk arrival queueing system with extra services and numerous vacation periods was conducted by Ayappan and Deepika [12]. Stochastic Petri nets should be used to evaluate the queuing models' performance, especially in distributed and multiprocessor systems, according to Siddiqui et al. [13]. By leveraging the strengths of SPNs, such as their visual and expressive power combined with stochastic modeling capabilities, the authors provide a compelling case for their adoption in performance evaluation studies.

Banerjee et al. [14] analyze rank-based routing policies in queuing systems, introducing MSBLB and MJSQ policies that reduce communication costs while ensuring effective load balancing supported by rigorous proof. Otten [15] examines the supply chain system with inventory-dependent routing, offering the explicit solutions and algorithms developed to optimize the production and inventory management, advancing supply chain efficiency. Van der Boor et al. [16] propose a hyper-scalable scheme with universal throughput bounds, optimizing distributed systems with limited communication. Liu and Ying [17] study load balancing in systems with constrained local queues, focusing on performance under heavy traffic in data centers. Ahmed et al. [18] introduce QLLB, a queue-length-based load-balancing scheme for data centers, addressing congestion and network asymmetries. Hellemans et al. [19] examine load-balancing policies for distributed systems with variable

job sizes, highlighting latency reduction. Van der Boor et al. [20] review scalable adaptations of the JSQ policy, balancing delay performance with reduced communication overhead. Delasay and Akan [21] analyze a two-phase queuing model to optimize delay costs for tasks with different priority classes. Santos et al. [22] discuss the importance of pre-emptive evaluation and the optimization of IoT-based monitoring systems in smart buildings. The research offers a solid foundation for evaluating and enhancing system performance by using a queuing network-based message exchange architecture and carrying out an extensive sensitivity analysis using Design of Experiments (DoE). The findings can significantly aid in designing efficient, reliable, and responsive smart building infrastructures, ensuring better QoS and minimizing risks associated with system failures.

Rodrigues et al. [23] use an M/M/c/K queuing model to optimize IoP-based ambient assisted living systems, improving aged care with reliable and cost-effective health monitoring solutions. Katayama and Tachibana [24] propose a task allocation strategy for MEC and cloud servers, reducing latency and optimizing resource use in 5G and future network environments. Zhan et al. [25] explore how service environments and omni-channel sales affect customer behavior, helping merchants optimize service capacity and profits. Alnowibet et al. [26] introduce a stochastic inventory model considering lead times and impatient customers, improving real-world inventory management under random demand.

Existing queuing models overlook dynamically adjusting server capacity based on queue length, motivating the development of a batch-service queuing system with load balancing and server vacations to address real-time network congestion. Jin et al. [27] analyzed the nonlinear dynamics of TCP/AQM networks, identifying critical delay values for Hopf bifurcation and stability, using $\tau$-decomposition and central manifold theory. Numerical examples validate the framework for delay-stable intervals and bifurcation stability. Yen et al. [28] study an N-policy M/G/1 queue with working breakdowns, deriving steady-state results and optimizing thresholds and service rates for system stability. Numerical results and sensitivity analysis validate its practical application. Kothandaraman and Kandaiyan [29] examined a Markovian queue with heterogeneous, intermittently available servers under a hybrid vacation policy, using the matrix geometric method to derive stability conditions and performance indicators. Numerical examples highlight parameter impacts. Kempa and Paprocka [30] present a discrete-time queue model for a bottleneck production system with energy-saving features, analyzing performance and optimizing throughput using the Drum–Buffer–Rope concept. Simulations show its effectiveness in aligning arrival rates with capacity. Chydzinski and Adamczyk [31] investigated response time in queues with active/passive management and complex arrivals, deriving distributions and key properties using integral equations. Numerical results reveal insights into response time behavior across scenarios. Many authors have extensively researched bulk arrivals and batch services, exploring various parameters and their implications [32–35]. Niranjan and Devi Latha presented a novel two-phase bulk service queueing model incorporating active Bernoulli feedback, server vacations, breakdowns, and setup time. The model explores the queue size's probability-generating function and analyzes performance metrics with numerical examples [36]. Gautam et al. [37] explore optimizing power saving and QoS in LTE-A networks through a new DRX mechanism that switches between power-active and power-saving states based on user traffic. It uses an $M^{[X]}/G/1$ vacation queue model with N-policy to analyze performance and energy metrics. Numerical results identify optimal N values and DRX cycles to minimize power use. The study provides practical guidelines for balancing energy efficiency and network performance.

The main contributions of this study include the following:

1. Overloading and underloading of the server are addressed to reduce customer waiting time and minimize the system's total average cost.

2. An application in cloud computing using power BI is provided to demonstrate the effectiveness of the load-balancing approach.

The paper is organized as follows: Section 1 introduces the topic and provides a review of the relevant literature and discusses the application of the proposed model. Section 2 outlines the model description and methodology for the queuing system under consideration and defines the notations of the stochastic model, governing equations, and queue size distribution. Section 3 evaluates various performance measures and examines special cases. Section 4 develops the cost model for the system and includes numerical illustrations. Finally, Section 5 concludes the paper with suggestions for future research.

Cloud computing is the process of using and gaining access to a variety of online services, including apps, computer power, and data storage. Organizations and individuals can purchase and utilize hardware and software as needed from cloud service providers, eliminating the need to own and maintain physical assets.

Cloud computing is used predominately nowadays, as each and every process depends more on power BI. The application starts with the transfer of data files from the client to the server with a virtual prediction range of 500–900 GB scheduled as regular service. And if the service is 100–400 GB, it starts its process as an underloading service to access customer software in power BI, as it is used to prevent damage to the files given by the clients. When there are various needs to implement security and DAX (Double Application Extension) expressions that are more than 1000 GB, this is qualified as overloading, because there is a sudden spike in traffic and it consumes more power comparatively; however, the quality of service is not affected by this. The next process consists of cloud load where the expected data are marked with the expected time and space consumption in the burst of cloud load, which creates space, prevents the data from being formed as layers, and corrects errors using the Losant principle. In addition, there is a responsibility assigned to the class that has the necessary information to fulfill it. Thus, the process has been completed successfully. If the data are less than 100 GB, they are classified as vacation data and can be affected due to processes like input/output errors or file errors. To rectify the missed data, the software finds the relevant and required data and fixes the missing data, but there is a time sequence delay. Hence, the application for the cloud computing using power BI and cloud load can be modeled as a bulk arrival and batch service queueing model with load balancing and vacation.

## 2. Methods and Models

### *2.1. Model Description*

Customers are demanding service in batches afforded to the Poisson process with rate $\boldsymbol{\xi}$. In a bulk service system, customers are served in groups based on a general bulk service rule [38] with a lower bound of '$a$' and upper bound '$b$' of the server. Maintaining the constant threshold values of the server is not always possible in real-time systems; it has to be changed according to the number of service request. So, in this paper, the server threshold values for batch service may be increased or decreased (overloading or underloading) according to the number of service request. A regular batch service will be initiated only if at least '$\boldsymbol{a}$' customers are waiting for service. Upon service completion or vacation completion, if the queue length denoted by $\boldsymbol{\varepsilon}$ satifies $\boldsymbol{a} \leq \boldsymbol{\varepsilon} \leq \boldsymbol{b}$, then the service will be provided for $\boldsymbol{min}(\varepsilon, b)$ customers. Conversely, if the server finds more than '$N$' ($N > b$) customers waiting, it selects '$N$' customers for service by increasing the server's maximum capacity to '$N$'. Sometimes, the queue length will be less than the minimum server capacity '$\boldsymbol{a}$'; during that time, the minimum threshold value of the server may be reduced to '$c$'$(c < a)$ and take all '$c$' customers waiting for service. The primary benefit of the offered system is that the server capacity may be increased or decreased

dynamically based on the queue size. The server goes on vacation if the queue length falls below '*c*'. After vacation completion, depending upon the queue length, the server will provide a regular symmetric bulk service or batch service with overloading or batch service with underloading. The above systematic procedure can be implemented in cloud computing (power BI) to solve congestion problems. To estimate various performance metrics of the server, which tends to minimize the overall cost of the cloud system, a diagram demonstration of the offered system is depicted below.

Notations

Let $Z$ be the random variable representing the group size of the arrival and $Z(Y)$ be the probability generating function (PGF) of $Z$ (Table 1).

$\xi$—Poisson arrival rate.
$g_k$—The probability that a batch arrives '$k$' customers.
$C_q(t)$—At time $t$, the number of customers awaiting service.
$C_s(t)$—The number of customers currently receiving service at time $t$.

**Table 1.** Notations.

| | Cumulative Distribution Function | Probability Density Function | Laplace–StieltjesTransform | Remaining Service Time |
|---|---|---|---|---|
| Regular batch service | $B(z)$ | $b(z)$ | $\widetilde{B}(\alpha)$ | $B^0(z)$ |
| Batch service with overloading | $B_1(z)$ | $b_1(z)$ | $\widetilde{B}_1(\alpha)$ | ${B_1}^0(z)$ |
| Batch service with underloading | $B_2(z)$ | $b_2(z)$ | $\widetilde{B}_2(\alpha)$ | ${B_2}^0(z)$ |
| Vacation | $V(z)$ | $v(z)$ | $\widetilde{V}(\alpha)$ | $V^0(z)$ |

$$\psi(t) = \begin{cases} 0, \textit{ when the server is busy with regular batch service} \\ 1, \textit{ when the server is busy and doing batch service with overloading} \\ 2, \textit{ when the server is busy and doing service with underloading} \\ 3, \textit{ when the server is on vacation} \end{cases}$$

$$A_{ij}(z,t)\mathrm{dt} = P\left\{ C_s(t) = i, C_q(t) = j, \quad z \le B^0(t) \le z + dt, \psi(t) = 0 \right\} \text{ a} \le \text{i} \le \text{b; j} \ge 1;$$

is the probability that at time $t$, the server is busy with i customers under regular batch service and $j$ customers in the queue, and the remaining service time of a customer under service is between $z$ and $z + dt$.

$$A^1_{Nj}(z,t)\mathrm{dt} = P\left\{ C_s(t) = N, \ \ C_q(t) = j, z \le {B_1}^0(t) \le z + dt, \psi(t) = 1 \right\} \text{ j} \ge 1; N > b,$$

is the probability that at time $t$, the server is busy with '$N$' customers under batch service with overloading and $j$ customers in the queue, and the remaining service time of a customer under service is between $z$ and $z + dt$.

$$A^2_{cj}(z,t)\mathrm{dt} = P\left\{ C_s(t) = c, \ C_q(t) = j \ \ z \le {B_2}^0(t) \le z + dt, \psi(t) = 2 \right\}; j \ge 1;$$

is the probability that at time $t$, the server is busy with '$c$' customers under batch service with underloading and j customers in the queue, and the remaining service time of a customer under service is between z and $z + dt$.

$$S_n(z,t)dt = P\Big\{\ C_q(t) = n,\ z \le V^0(t) \le z + dt, \psi(t) = 3\Big\};\ \ 0 \le n \le c-1;$$

is the probability that at time $t$, the server is on vacation, the queue length is $n$ and the remaining vacation time of a customer at an arbitrary time is between $z$ and $z + dt$.

### 2.2. Steady-State Queue Size Distribution

The supplementary variable technique is used to derive the steady-state system equations [39].

$$\begin{aligned} -\tfrac{d}{dz}A_{i0}(z) = -\xi A_{i0}\ (z) + \textstyle\sum_{m=a}^{b} A_{mi}(0)b(z) + A^1_{Ni}(0)b(z) \\ +A^2_{ci}(0)b(z) + S_i(0)b(z);\ c\ \le\ i\ \le\ \ b. \end{aligned} \tag{1}$$

Equation (1) denotes the different probabilities for the server in a regular batch service with 'i' customers in service and '0' customer in the queue in the remaining service time $z - \Delta t$ at time $t + \Delta t$. In the RHS, the first term indicates that there is no arrival at that time, and the second term indicates that when regular batch service is completed, if '*i*' customers are in the queue, then 'i' customers go for regular batch service and '0' customers are waiting in the queue. The next term indicates that when overloading batch service is completed, if '*i*' customers are in the queue, then '*i*' customers go for a regular batch service, and '0' customers will be waiting in the queue. The fourth term indicates that when underloading batch service is completed, if '*i*' customers are in the queue, then '*i*' customers go for a regular batch service, and '0' customers will be waiting in the queue. The last term indicates after vacation completion, if '*i*' customers are in the queue, then 'i' customers go for regular batch service, and '0' customers will be waiting in the queue. Similarly, Equations (2)–(7) can be expressed with the help of a schematic representation of the queueing model shown in Figure 1.

$$-\frac{d}{dz}A_{ij}(z) = -\xi A_{ij}(z) + \textstyle\sum_{k=1}^{j} A_{i\ j-k}(z)\ \xi g_k;\quad c\ \le\ i\ \le\ b-1;\quad j\ \ge\ 1; \tag{2}$$

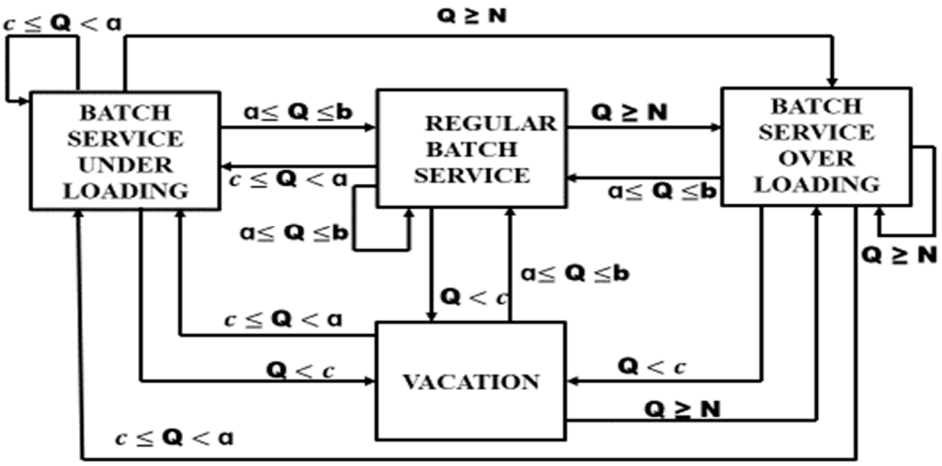


**Figure 1.** Schematic representation of the queueing model. Q—queue size.

$$\begin{aligned} -\tfrac{d}{dz}A_{bj}(z) = -\xi A_{bj}(z) + \textstyle\sum_{k=1}^{j} A_{b\ j-k}(z)\ \xi g_k + A^1_{Nb+j}(0)\ b(z) + A^2_{cb+j}(0)\ b(z) \\ +\textstyle\sum_{m=c}^{b} A_{mb+j}(0)b(z) + S_{b+j}(0)b(z);\ j \ge 1; \end{aligned} \tag{3}$$

$$\begin{aligned} -\tfrac{d}{dz}\boldsymbol{A^1_{Nj}(z)} = -\boldsymbol{\xi A^1_{Nj}(z)} + \textstyle\sum_{k=1}^{j} \boldsymbol{A^1_{Nj-k}(z)\xi g_k} + \boldsymbol{A^1_{NN+j}(0)b_1(z)} \\ +\textstyle\sum_{m=a}^{b} A_{mN+j}(0)b_1(z) + A^2_{c\ N+j}(0)b_1(z) + S_{N+j}(0)b_1(z);\ j \ge\ 0; \end{aligned} \tag{4}$$

$$\begin{gathered} -\tfrac{d}{dz}\boldsymbol{A_{cj}^{2}}(z) = -\boldsymbol{\xi A_{cj}^{2}}(z) + \textstyle\sum_{k=1}^{j} \boldsymbol{A_{cj-k}^{2}}(z)\boldsymbol{\xi g_k} + \boldsymbol{A_{cc+j}^{2}}(\mathbf{0})\boldsymbol{b_2}(z) \\ +\textstyle\sum_{m=a}^{b} A_{m\ c+j}(0)b_2(z) + A_{N\ c+j}^{1}(0)b_2(z) + S_{c+j}(0)b_2(z); j \geq\ 0; \end{gathered} \tag{5}$$

$$-\frac{\boldsymbol{d}}{\boldsymbol{dz}}\boldsymbol{S_0}(z) = -\boldsymbol{\xi S_0}(z) + \textstyle\sum_{\boldsymbol{m=c}}^{\boldsymbol{b}} \boldsymbol{A_{m0}}(\mathbf{0})\boldsymbol{V}(z) + \boldsymbol{A_{N0}^{1}}\boldsymbol{v}(z) + \boldsymbol{A_{c0}^{2}}\boldsymbol{v}(z) \tag{6}$$

$$\begin{gathered} -\tfrac{d}{dz}\boldsymbol{S_n}(z) = -\boldsymbol{\xi S_n}(z) + \textstyle\sum_{\boldsymbol{m=c}}^{\boldsymbol{b}} \boldsymbol{A_{mn}}(\mathbf{0})\boldsymbol{v}(z) + \boldsymbol{A_{Nn}^{1}}(\mathbf{0})\boldsymbol{v}(z) + \boldsymbol{A_{cn}^{2}}(\mathbf{0})\boldsymbol{v}(z) \\ +\textstyle\sum_{k=\mathbf{1}}^{n} S_{n-k}(z)\ \xi g_k\ ,\ \mathbf{1}\ \leq\ n\ \leq\ c-\mathbf{1}. \end{gathered} \tag{7}$$

The Laplace–Stieltjes transforms of $A_{in}(y)$, $S_n(y)$, $A_{Nj}^{1}(y)$ and $A_{cj}^{2}(y)$ are defined as

$$\widetilde{A}_{in}(\alpha) = \int_0^\infty \mathrm{e}^{-\alpha z}\mathrm{A_{in}}(z)dz;\ \widetilde{S}_n(\alpha) = \int_0^\infty \mathrm{e}^{-\alpha z}\mathrm{S_n}(z)dz;$$

$$A_{Nj}^{1}(\alpha) = \int_0^\infty \mathrm{e}^{-\alpha z}A_{Nj}^{1}(z)dz;\ A_{cj}^{2}(\alpha) = \int_0^\infty \mathrm{e}^{-\alpha z}A_{cj}^{2}(z)dz.$$

Applying Laplace–Stieltjes transform to Equations (1)–(7),

$$\begin{gathered} \alpha\widetilde{A}_{i0}(\alpha) - A_{i0}(0) = \xi\widetilde{A}_{i0}(\alpha) + \textstyle\sum_{m=a}^{b} A_{mi}(0)\widetilde{B}(\alpha) + A_{Ni}^{1}(0)\widetilde{B}(\alpha) \\ +A_{ci}^{2}(0)\widetilde{B}(\alpha) + S_i(0)\widetilde{B}(\alpha) a \leq\ i\ \leq\ \ b \end{gathered} \tag{8}$$

$$\alpha\widetilde{A}_{ij}(\alpha) - A_{ij}(0) = \xi\widetilde{A}_{ij}(\alpha) + \textstyle\sum_{k=1}^{j} \widetilde{A}_{i\ j-k}(\alpha)\ \xi g_k; a \leq\ i\ \leq\ \ b-1; \tag{9}$$

$$\begin{gathered} \alpha\widetilde{A}_{bj}(\alpha) - A_{bj}(0) = \xi\widetilde{A}_{bj}(\alpha) + \textstyle\sum_{k=1}^{j} \widetilde{A}_{b\ j-k}(\alpha)\ \xi g_k + A_{Nb+j}^{1}(0)\widetilde{B}(\alpha) \\ +A_{cb+j}^{2}(0)\widetilde{\mathrm{B}}(\alpha) + \textstyle\sum_{m=c}^{b} A_{mb+j}(0)\widetilde{\mathrm{B}}(\alpha) + S_{b+j}(0)\widetilde{\mathrm{B}}(\alpha);\ j \geq\ 1; \end{gathered} \tag{10}$$

$$\begin{gathered} \alpha\widetilde{A}_{Nj}^{1}(\alpha) - A_{Nj}^{1}(0) = \xi\widetilde{A}_{Nj}^{1}(\alpha) + \textstyle\sum_{k=1}^{j} \widetilde{A}_{Nj-k}^{1}(\alpha)\ \xi g_k + A_{NN+j}^{1}(0)\widetilde{B}_1(\alpha) \\ +\textstyle\sum_{m=a}^{b} A_{mN+j}(0)\widetilde{B}_1(\alpha) + A_{c\ N+j}^{2}(0)\widetilde{B}_1(\alpha) + S_{N+j}(0)\widetilde{B}_1(\alpha);\ j \geq\ 0; \end{gathered} \tag{11}$$

$$\begin{gathered} \alpha\widetilde{A}_{cj}^{2}(\alpha) - A_{cj}^{2}(0) = \xi\widetilde{A}_{cj}^{2}(\alpha) + \textstyle\sum_{k=1}^{j} \widetilde{A}_{c\ j-k}^{2}(\alpha)\ \xi g_k + A_{c\ c+j}^{2}(0)\widetilde{B}_3(\alpha) \\ +\textstyle\sum_{m=a}^{b} A_{m\ c+j}(0)\widetilde{B}_3(\alpha) + A_{N\ c+j}^{1}(0)\widetilde{B}_3(\alpha) + S_{c+j}(0)\widetilde{B}_3(\alpha);\ j \geq\ 0; \end{gathered} \tag{12}$$

$$\alpha\widetilde{S}_0(\alpha) - S_0(0) = \xi\widetilde{S}_0(\alpha) + \textstyle\sum_{m=c}^{b} A_{m0}(0)\widetilde{V}(\alpha) + A_{N0}^{1}\widetilde{V}(\alpha) + A_{c0}^{2}\widetilde{V}(\alpha); \tag{13}$$

$$\begin{gathered} \alpha\widetilde{S}_n(\alpha) - S_n(0) = \xi\widetilde{S}_n(\alpha)\textstyle\sum_{m=c}^{b} A_{mn}(0)\widetilde{V}(\alpha) + A_{Nn}^{1}(0)\widetilde{V}(\alpha) + A_{cn}^{2}(0)\widetilde{V}(\alpha) \\ +\textstyle\sum_{k=1}^{n} \widetilde{S}_{n-k}(\alpha)\ \xi g_k;\ 1\ \leq\ n\ \leq\ a-1. \end{gathered} \tag{14}$$

The following probability-generating functions are defined to obtain the PGF of the queue size at any given time.

$$\widetilde{A}_c^{2}(y,\alpha) = \textstyle\sum_{j=0}^{\infty} \widetilde{A}_{cj}^{2}(\alpha)y^j;\ A_c^2(y,0) = \sum_{j=0}^{\infty} A_{cj}^{2}(0)y^j;$$

$$\widetilde{\mathrm{S}}(y,\alpha) = \textstyle\sum_{\mathrm{n}=0}^{\mathrm{a}-1} \widetilde{S}_n(\alpha)\mathrm{y^n};\ \ S(y,\ 0) = \sum_{n=0}^{a-1} S_n(0)y^n;$$

$$Let A_I = \textstyle\sum_{i=a}^{b-1}\sum_{m=a}^{b} A_{mi}(0);\ A_N^1(0) = A_N^1\ ;\ A_c^2(0) = A_c^2\ ;$$

$$S_i(0) = S_i;\ \beta_i = A_c^2 + A_i + S_i + A_N^1.$$

By multiplying Equations (8)–(14) by suitable powers of $y^n$ and summing over $n$ then by using the PGF of queue size at an arbitrary time epoch, we obtain

$$(\alpha - \xi + \xi\mathrm{Z(y)})\widetilde{A}_i(y,\alpha) = \mathrm{A_i}(y,0) - \widetilde{B}(\alpha)\Big(A_c^2 + A_i + S_i + A_N^1\Big); \tag{15}$$

$$y^{b}(\alpha-\xi+\xi Z(y))\widetilde{A}_b(y,\alpha)=y^{b}A_b(y,0)-\widetilde{B}(\alpha)\sum_{j=b}^{N-1}\left(A_c^2+A_i+S_i+A_N^1\right)y^j; \quad (16)$$

$$(\alpha-\xi+\xi Z(y))\widetilde{A}_N^1(y,\alpha)y^N=A_N^1(y,0)y^N-\widetilde{B}_1(\alpha)h_1, \quad (17)$$

where $h_1=\begin{matrix}\left(\sum_{m=a}^{b}A_m(y,0)-\sum_{j=b}^{N-1}\left(A_iy^j\right)\right)+\left(S(y,0)-\sum_{j=b}^{N-1}\left(S_iy^j\right)\right)\\ \left(A_N^1(y,0)-\sum_{j=b}^{N-1}\left(A_J^1y^j\right)\right)\\ \left(A_C^2(y,0)-\sum_{j=b}^{N-1}\left(A_J^2y^j\right)\right)\end{matrix}$.

$$(\alpha-\xi+\xi Z(y))\widetilde{A}_c^2(y,\alpha)y^c=A_c^2(y,0)y^c-\widetilde{B}_2(\alpha)\sum_{m=0}^{c-1}\left(A_c^2+A_i+A_N^1+S_n\right)y^n; \quad (18)$$

$$(\alpha-\xi+\xi Z(y))\widetilde{S}(y,\alpha)=S(y,0)-\widetilde{S}(\alpha)\sum_{n=0}^{c-1}\left(A_c^2+A_i+A_N^1+S_n\right)y^n. \quad (19)$$

*2.3. PGF of Queue Size at an Arbitrary Time*

The function that generates the probability of queue size at an arbitrary time epoch can be obtained by using the below given equation

$$P(y)=\sum_{m=a}^{b-1}\widetilde{A}_m(y,0)+\widetilde{A}_b(y,0)+\widetilde{A}_N^1(y,0)+\widetilde{A}_c^2(y,0)+\widetilde{S}(y,0). \quad (20)$$

Substituting $\alpha=\xi-\xi Z(y)$ in Equations (15)–(19), after performing some algebraic manipulations, the PGF of queue size is defined below, and we obtain

$$P(y)=\frac{\begin{matrix}\left(\widetilde{B}(\xi-\xi Z(y))-1\right)\left(\sum_{j=0}^{N-1}\beta_jy^j+\sum_{i=a}^{b-1}\beta_i\right)\\ +\left(\widetilde{S}(\xi-\xi Z(y))-1\right)\sum_{j=0}^{C-1}\beta_jz^j\end{matrix}}{\left(y^N-\widetilde{B}_1(\xi-\xi Z(y))\right)(-\xi+\xi Z(y))}+\frac{\left(\widetilde{B}_1(\xi-\xi Z(y))-1\right)f_1+\left(\widetilde{B}_2(\xi-\xi Z(y))-1\right)\sum_{j=0}^{c}\beta_jy^j}{\left(y^N-\widetilde{B}_1(\xi-\xi Z(y))\right)(-\xi+\xi Z(y))}, \quad (21)$$

where $f_1=\begin{matrix}\left(\sum_{m=a}^{b}A_m(y,0)-\sum_{j=b}^{N-1}\left(A_iy^j\right)\right)+\left(S(y,0)-\sum_{j=b}^{N-1}\left(S_iy^j\right)\right)\\ -\sum_{j=b}^{N-1}\left(A_J^1y^j\right)\\ \left(A_C^2(y,0)-\sum_{j=b}^{N-1}\left(A_J^2y^j\right)\right)\end{matrix}$.

## 3. Results: Steady-State Condition

To ensure the existence of a steady state for the model under consideration, the condition $\rho < 1$ must be satisfied, where $\rho$ is defined as $\rho = N-\xi E(B_1)E(Z)$. The PGF P(y) must satisfy P(1) = 1. By applying L'Hospital's rule and evaluating $\lim_{y\to 1}P(y)=1$, and equating the expression to 1, the condition $\xi E(Z)(N-\xi E(B_1)E(Z))>0$ is obtained (Algorithm 1).

**Algorithm 1. Algorithm to Find Unknown Probabilities**

Step 1: Use Equation (21) to initiate the procedure.
Step 2: Consider the unknowns $S_i$ and $\beta_j$ exists in P(Y).
Step 3: Apply Rouche's theorem of complex variables to find the vanishing points at $|Y| = 1$.
Step 4: '*N*' equations and '*N*' unknowns obtained in step 3 are solved using MATLAB 2020.

**Lemma 3.1.** *Let $\beta_i$ be the probability that 'i' customers arrive during the vacation. The probability generating function of $\beta_i$ is given by $\sum_{i=0}^{\infty} \beta_i\, y^i = \widetilde{S}(\xi - \xi Z(y))$.*

**Proof:** Conditioning on the actual vacation length, number of arrivals and the group size, we obtain

$$\beta_i = \int_0^{\infty} \left(\sum_{m=0}^{i} \frac{(e^{-\xi t}\ )(\xi t)^m g_i{}^m}{m!}\right) dv(t)$$

where $g_i{}^m$ is the m-fold convolution of $g_i$ with itself (i.e., the total of m arrivals make '*i*' customers).

Multiplying the above equation by $z^i$ and taking the summation from $i = 0$ to $\infty$, we obtain

$$\begin{aligned}\sum_{i=0}^{\infty} \beta_i y^i &= \int_0^{\infty} e^{-\xi t} \left(\sum_{m=0}^{\infty} \frac{(\xi t)^m}{m!} \sum_{i=m}^{\infty} g_i{}^m\, y^i\right) dv(t)\\ &= \int_0^{\infty} e^{-\xi t} \left(\sum_{m=0}^{\infty} \frac{(\xi t)^m \quad (Z(y))^{\ m}}{m!}\right) dv(t)\\ &= \widetilde{S}(\xi - \xi Z(y))\end{aligned}$$

Hence, the Lemma. □

**Theorem 3.1.** *The unknown constants $s_i$ are expressed in terms of $\beta_i$ as, $s_i = \sum_{i=0}^{n} \beta_{n-i}\omega_i$, $n = 0, 1, 2 \ldots a - 1$, where $\omega_i$ is the probability that 'i' customers arrive during the underloading or overloading service period.*

**Proof:** Substituting $\alpha = \xi - \xi\, Z(y)$ in Equation (19), we have

$$S(y,0) = \widetilde{S}(\xi - \xi Z(y)) \sum_{n=0}^{c-1} \left(A_c^2\ + A_i + A_N^1\ + S_n\right) y^n$$

By using the above lemma

$$\sum_{n=0}^{a-1} s_n y^n = \left(\sum_{n=0}^{\infty} \omega_n y^n\right)\ \left(\sum_{n=0}^{a-1} \beta_n y^n\right)$$

$$\sum_{n=0}^{a-1} s_n y^n = \sum_{n=0}^{a-1} \quad \left(\sum_{i=0}^{n} \beta_{n-i}\omega_i\right) y^n \tag{22}$$

Equating the coefficient of $y^n$; n = 0, 1, 2, 3 . . . a − 1, on both sides of Equation (22), we obtain $s_i = \sum_{i=0}^{n} \beta_{n-i}\omega_i$. □

## 4. Discussion

### *4.1. Efficiency Metrics of Cloud Resource Utilization*

In a queueing system, it is common practice to evaluate the average number of inputs to be processed in buffer and the average waiting time for inputs to be processed in buffer. This section derives efficiency metrics of cloud resource utilization from the steady-state probability distribution function, which are essential for determining the aggregate mean cost of the system.

#### 4.1.1. Average Number of Inputs to Be Processed in Buffer

The average number of inputs to be processed in buffer $L_{queue}$ at an arbitrary time epoch is obtained by differentiating $P(y)$ at $y = 1$ and is given by

$$L_{queue} = \lim_{y \to 1} P'(y)$$

4.1.2. Average Waiting Time for Inputs to Be Processed in Buffer

The average waiting time for inputs to be processed in buffer $W_{queue}$ can be readily determined using Little's formula

$$W_{queue} = \frac{L_{queue}}{\xi E(Z)}$$

4.1.3. Average Processing Period of the Cloud Server

$$L_{busy} = \frac{\sum_{i=a}^{N-1}\left(A_i^2 + A_i + A_i^1\right) + \left(1 - \sum_{i=a}^{N-1}\left(A_i^2 + A_i + A_i^1\right) + (1 - E(B_2)\right)}{\sum_{i=0}^{a-1}\left(A_i^2 + A_i + A_i^1\right)}$$

$$L_{busy} = \frac{\mathrm{E(S)}}{\sum_{\mathrm{i=0}}^{\mathrm{a-1}}\left(\mathrm{P^1{}_i} + \mathrm{R_i}\right)}.$$

4.1.4. Average Sleep Time of the Cloud Server

The time interval between the start of the busy period and the vacation epoch is known as the cloud server's average sleep time.

Let I be the random variable for the 'idle period'.

$\alpha_{\mathrm{j}}$, the probability that the system state visits '$j$' during an idle period is given by $j$ = 0, 1, 2. . . $a - 1$.

$$\text{Let } I_{j=}\begin{cases} 1 \text{ if the state '}j\text{' during an idle period} \\ 0 \text{ otherwise} \end{cases}$$

Based on the size of the queue at the service completion epoch, we have $\alpha_0 = \pi_0$ :

$$\alpha_{\mathrm{j}} = P\left(I_j = 1\right) = \pi_{\mathrm{j}} + \sum_{\mathrm{k=0}}^{\mathrm{c-1}} \pi_{\mathrm{k}} P\left(I_{j-k}{}^1 = 1\right); j = 1, 2, \ldots c-1.$$

Thus, the average sleep time of the cloud server is obtained:

$$L_{idle} = \frac{E(S)}{\sum_{n=0}^{a-1}\sum_{i=0}^{n}\alpha_i\left(A_{n-i}^2 + A_{n-i} + A_{n-i}^1\right)}.$$

*4.2. Special Case: Hyper-Exponential Regular Bulk Service Time*

When the regular service time follows hyper-exponential distribution with probability density function, then $b(z) = cde^{-dz} + (1-c)fe^{-fz}$ where $d$ and $f$ are parameters, and

$$\overset{\sim}{B}(\xi - \xi\mathrm{Z(y)}) = \frac{dc}{d + (\xi - \xi\mathrm{Z(y)})} + \frac{f(1-c)}{f + (\xi - \xi\mathrm{Z(y)})}.$$

The PGF of the queue size for hyper-exponential service time is derived by substituting the expression for $\overset{\sim}{B}(\xi - \xi\mathrm{Z(y)})$ in Equation (21).

*4.3. Cost Model*

Optimization techniques play a crucial role in minimizing the aggregate mean cost of a queueing system in many real-world situations. The rate investigation typically involves various constraints, such as switch-on cost, in-service cost, carrying cost and any potential remuneration costs. The primary objective in managing the system is to minimize the aggregate mean cost. The cost analysis of the proposed queueing system is presented in this section. Considering the following assumptions:

$\gamma_h$ : carrying cost per customer;

$\gamma_0$ : For each unit of time spent on service;

$\gamma_s$ : *cost per cycle of switching on*;

$\gamma_r$ : remuneration expense per cycle as aresult of vacation.

Given that the cycle's duration is the product of the cloud server's average sleep time and processing period:

$$\begin{aligned}\text{Aggregate Mean Cost} =&(\text{Average sleep time of the cloud server})\\ &+(\text{Average processing period of the cloud server});\end{aligned}$$

$$\text{Aggregate Mean Cost} = \frac{E(S)}{\sum_{n=0}^{a-1}\sum_{i=0}^{n}\alpha_i\left(A_{n-i}^{2}+A_{n-i}+A_{n-i}^{1}\right)}+\frac{E(S)}{\sum_{I=0}^{a-1}(P^{1}{}_{i}+R_i)}.$$

$$\text{Aggregate Mean Cost} = [\gamma_s - \ \gamma_r E(I)]\frac{1}{E(T_c)}+\gamma_h E(Q)+\gamma_0\rho;$$

where $\rho = \frac{(\xi E(B_1)E(Z))}{N}$.

Determining the optimal value c* that minimizes the aggregate mean cost is challenging to achieve through direct analytical methods. The definition of the optimal policy for a threshold $c^*$ to minimize the aggregate mean cost using the simple value direct search method is outlined below.

Stage 1: Determine the value of upper bound server capacity '$N$';
Stage 2: Select the value "c" that will fulfill the subsequent relation.

$$\text{AMC}(c^*) \ \leq \text{AMC(c)}, \quad 1 \leq \text{c} \leq N;$$

Stage 3: The value $c^*$ is optimal, since it minimizes the aggregate mean cost.

The best value of $c^*$ that minimizes the aggregate mean cost function is obtained by following the preceding technique. In the following part, a numerical example will be provided to verify this solution.

### *4.4. Numerical Illustration*

The theoretical findings derived for the proposed model are supported by numerical validation using the following assumptions and symbols (Table 2).

**Table 2.** Parameters and symbols.

| Parameter | Distribution | Symbols |
|---|---|---|
| Arrival rate | Poisson distribution | $\xi$ |
| Regular batch service time | 4-Erlang distribution | $\mu$ |
| Overloading batch service time | 3-Erlang distribution | $\mu_1$ |
| Underloading batch service time | 2-Erlang distribution | $\mu_2$ |
| Vacation time | Exponential distribution | $\varphi$ |

The group size distribution of the arrival is geometric with a mean of 3.

| | |
|---|---|
| Lower bound server capacity | $a$ |
| Upper bound server capacity | $b$ |
| Boundary value | $N$ |
| Switch-on cost | ₹6 |
| Carrying cost per customer | ₹3 |
| In service cost per unit time | ₹4 |
| Remuneration per unit time due to vacation | ₹2 |

### *4.5. Effects of Efficiency Metrics of Cloud Resource Utilization*

The impacts of efficiency metrics of cloud resource utilization for fixed boundary values are presented. From Figure 2, it is clear that if the service rate increases, $L_{queue}$ and $L_{busy}$ decrease whereas $L_{idle}$ increases. From Table 3, it is evident that when the service rate is fixed for underloading and overloading but varies for regular batch service, as $\xi$ increases, $L_{queue}$ and $L_{busy}$ increase whereas $L_{idle}$ decreases. In Table 4 and Figure 3, it is evident that when the service rate is fixed for regular batch service and underloading but varies for the overloading service, as $\xi$ increases, $L_{queue}$ and $L_{busy}$ increase, whereas $L_{idle}$ decreases. From Table 5, it is evident that when the service rate is fixed for overloading and regular batch service but varies for underloading, as $\xi$ increases, $L_{queue}$ and $L_{busy}$ increase, whereas $L_{idle}$ decreases. Tables 3–5 present the impact of efficiency metrics on cloud resource utilization under varying arrival and service rates.

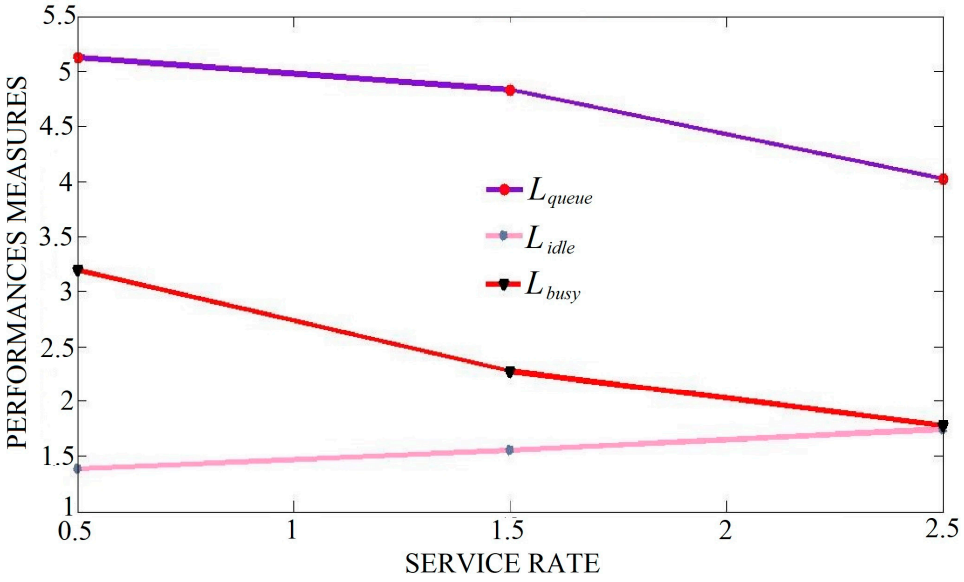


**Figure 2.** Service rate vs. efficiency metrics of cloud resource utilization.

**Table 3.** Arrival rate vs. efficiency metrics of cloud resource utilization ($\mu_1 = 3,\ \mu_2 = 4,\ \varphi = 5$).

| $\xi$ | $\mu$ = 0.5 | | | $\mu$ = 1.5 | | | $\mu$ = 2.5 | | |
|---|---|---|---|---|---|---|---|---|---|
| | $L_{queue}$ | $L_{busy}$ | $L_{idle}$ | $L_{queue}$ | $L_{busy}$ | $L_{idle}$ | $L_{queue}$ | $L_{busy}$ | $L_{idle}$ |
| 1 | 5.0262 | 3.1932 | 1.9349 | 4.5935 | 2.5985 | 1.8412 | 4.1923 | 1.9134 | 1.7239 |
| 2 | 5.9271 | 3.9757 | 1.1352 | 5.0349 | 3.2431 | 1.4357 | 5.0921 | 2.5932 | 1.6128 |
| 3 | 6.8314 | 4.5932 | 0.9652 | 6.2145 | 4.9562 | 1.0932 | 6.3245 | 3.2398 | 1.3217 |
| 4 | 8.6783 | 5.3326 | 0.5923 | 7.3542 | 5.8235 | 0.7312 | 7.1256 | 4.4532 | 0.9234 |
| 5 | 9.1296 | 7.5561 | 0.1475 | 8.4563 | 6.9329 | 0.4956 | 8.3672 | 5.9431 | 0.5498 |

**Table 4.** Arrival rate vs. efficiency metrics of cloud resource utilization ($\mu = 4,\ \mu_2 = 5,\ \varphi = 6$).

| $\xi$ | $\mu_1$ = 1.5 | | | $\mu_1$ = 2.5 | | | $\mu_1$ = 3.5 | | |
|---|---|---|---|---|---|---|---|---|---|
| | $L_{queue}$ | $L_{busy}$ | $L_{idle}$ | $L_{queue}$ | $L_{busy}$ | $L_{idle}$ | $L_{queue}$ | $L_{busy}$ | $L_{idle}$ |
| 6 | 7.6212 | 4.3291 | 2.3229 | 5.5325 | 3.4574 | 2.7523 | 5.1259 | 2.5632 | 2.2423 |
| 6.2 | 8.7132 | 5.5785 | 1.9542 | 6.2382 | 4.5627 | 2.0286 | 5.9241 | 3.4574 | 1.3426 |
| 6.4 | 9.9823 | 7.7643 | 1.3326 | 7.4143 | 6.3387 | 1.5762 | 6.6229 | 4.9231 | 1.0572 |
| 6.8 | 10.7378 | 8.3265 | 0.9523 | 9.1732 | 7.6124 | 1.1421 | 7.2283 | 5.4213 | 0.8763 |
| 7 | 11.8126 | 9.2411 | 0.4875 | 10.3132 | 8.8332 | 0.7327 | 8.8621 | 6.1257 | 0.1287 |


**Figure 3.** Arrival rate vs. efficiency metrics of cloud resource utilization.

**Table 5.** Arrival rate vs. efficiency metrics of cloud resource utilization ($\mu_1 = 3,\ \mu = 4,\ \varphi = 5$).

| $\xi$ | $\mu_2$ = 2.5 | | | $\mu_2$ = 3.5 | | | $\mu_2$ = 4.5 | | |
|---|---|---|---|---|---|---|---|---|---|
| | $L_{queue}$ | $L_{busy}$ | $L_{idle}$ | $L_{queue}$ | $L_{busy}$ | $L_{idle}$ | $L_{queue}$ | $L_{busy}$ | $L_{idle}$ |
| 9.1 | 4.6293 | 2.5932 | 2.5829 | 3.4328 | 2.3427 | 2.4523 | 5.6259 | 1.7752 | 2.1423 |
| 9.3 | 5.1247 | 3.6757 | 2.2542 | 4.5985 | 3.6727 | 1.9286 | 6.4241 | 2.4274 | 1.5426 |
| 9.5 | 6.4238 | 4.7932 | 1.8326 | 5.1253 | 4.5238 | 1.2762 | 7.1529 | 3.7423 | 1.0572 |
| 9.7 | 7.9378 | 5.2326 | 1.4523 | 6.9732 | 5.1763 | 0.8421 | 8.3283 | 4.5139 | 0.9763 |
| 9.9 | 9.8126 | 6.8561 | 0.9875 | 8.3547 | 6.8332 | 0.4327 | 9.0621 | 5.8237 | 0.0287 |

Consider a situation where the service times follow a hyper-exponential distribution under the following assumptions: $\mu = 5\ \mu_1 = 6,\ \mu_2 = 7,\ \varphi = 3$. From Table 6, it is clear that for regular batch service, underloading and overloading with hyper-exponentially distributed service times, as $\xi$ increases, both $L_{queue}$ and $L_{busy}$ grow, while $L_{idle}$ decreases.

**Table 6.** Hyper-exponential service time vs. efficiency metrics of cloud resource utilization. Service time follows hyper-exponential distribution $\mu = 5$ $\mu_1 = 6$, $\mu_2 = 7$, $\varphi = 3$.

| $\xi$ | $L_{queue}$ | $L_{busy}$ | $L_{idle}$ |
|---|---|---|---|
| 2.1 | 5.3478 | 2.0749 | 1.4267 |
| 2.3 | 6.4523 | 2.9342 | 1.0542 |
| 3.5 | 7.9256 | 3.8432 | 0.9586 |
| 4.7 | 8.6431 | 4.5624 | 0.6752 |
| 5.9 | 9.2567 | 5.6572 | 0.4175 |

Table 7 and Figure 4 illustrates the impact of batch size 'c' on the aggregate mean cost for b = 15. To achieve the minimal aggregate mean cost, the lower bound server capacity should be set at c = 7. Furthermore, c should be optimized based on varying arrival, service, and vacation rates. For the best results, c = 7 and N == 20 should be maintained. Figure 5 demonstrates that increasing the threshold value '*a*' and $L_{queue}$ leads to a reduction in the aggregate mean cost. These results are essential for optimizing average system costs using power BI in cloud computing. The efficiency metrics for cloud resource utilization are presented below.

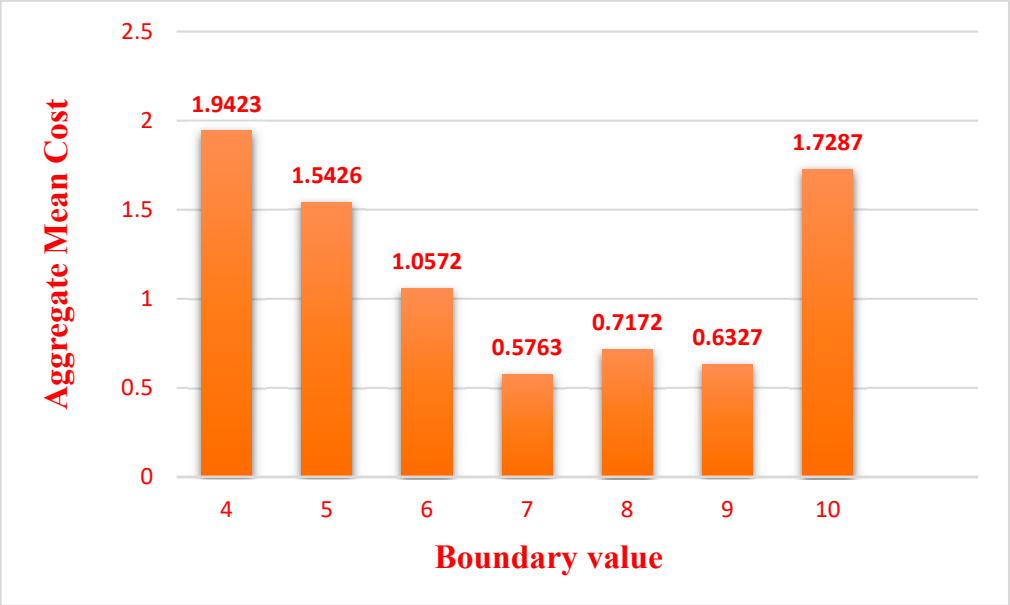


**Figure 4.** Boundary value '*c*' vs. aggregate mean cost.

**Table 7.** Boundary value '*c*' vs. aggregate mean cost, $\xi = 2$, $b = 15$, N $= 20$, $\mu_1 = 7$, $\mu_2 = 6$, $\varphi = 5$, $a = 11$.

| $c$ | $L_{queue}$ | $L_{busy}$ | $L_{idle}$ | AMC |
|---|---|---|---|---|
| 4 | 4.6293 | 1.7752 | 2.5632 | 1.9423 |
| 5 | 5.1247 | 2.4274 | 3.4574 | 1.5426 |
| 6 | 6.4238 | 3.7423 | 4.9231 | 1.0572 |
| 7 | 7.9378 | 4.5139 | 5.4213 | 0.5763 * |
| 8 | 8.7132 | 7.7643 | 6.1257 | 0.7172 |
| 9 | 9.9823 | 8.3265 | 7.5392 | 0.6327 |
| 10 | 10.7378 | 9.2411 | 8.7982 | 1.7287 |

*—Denotes the lowest value of the aggregate mean cost.

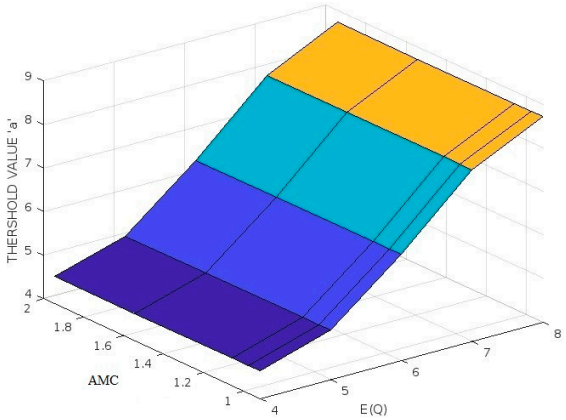


**Figure 5.** Boundary value '*a*' vs. aggregate mean cost, $\xi = 2$, $b = 5$, $N = 8$, $\mu_1 = 7$, $\mu_2 = 6$, $\varphi = 5$, $c = 3$.

### *4.6. Effects of Erlang Service Time and Hyper-Exponential Service Time over the Efficiency Metrics*

Consider a situation where the service times follow a hyper-exponential distribution with $\mu = 5$, $\mu_1 = 6$, $\mu_2 = 7$, $\varphi = 3$. Table 6 shows that for regular batch service across underloading and overloading conditions, $L_{queue}$ and $L_{busy}$ increase as $\xi$ grows, while $L_{idle}$ decreases. In contrast, when service times follow an Erlang distribution, the efficiency metrics of cloud resource utilization are lower compared to those observed with the hyper-exponential distribution.

## 5. Conclusions

An approach to batch and bulk arrival service queuing that includes load balancing and vacation time is examined in this study. This queueing system differs from others in that it adds load balancing to a $M^X/G(a,b)/1$ queueing model that includes vacation. Through the use of supplementary variables, the PGF of the queue size at any given time is determined. Various efficiency metrics of cloud resource utilization are calculated, supplemented by appropriate numerical examples. The results obtained are highly useful for optimizing the total average system cost using power BI in the context of cloud computing.

**Author Contributions:** Conceptualization, S.P.N. and S.D.L.; methodology, S.P.N. and S.D.L.; software, S.P.N. and S.D.L.; validation, S.P.N., S.D.L., S.V. and M.L.S.; formal analysis, S.P.N., S.D.L., S.V. and M.L.S.; investigation, S.P.N. and S.D.L.; resources, S.P.N., S.D.L., S.V. and M.L.S.; data curation, S.P.N. and S.D.L.; writing—original draft preparation, S.P.N.; writing—review and editing, S.P.N., S.V. and M.L.S.; visualization, S.P.N., S.D.L., S.V. and M.L.S.; supervision, S.P.N., S.D.L., S.V. and M.L.S.; project administration, S.P.N. and S.D.L.; funding acquisition, S.P.N., S.D.L., S.V. and M.L.S. All authors have read and agreed to the published version of the manuscript.

**Funding:** The APC was funded by the Transilvania University of Brasov, HBS 2017/2024.

**Data Availability Statement:** The original contributions presented in the study are included in the article; further inquiries can be directed to the corresponding authors.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Takagi, H. *Queueing Analysis: A Foundation of Performance Evaluation*; Distributors for the U.S. and Canada, Elsevier Science Pub. Co.: Amsterdam, The Netherlands; New York, NY, USA, 1991.
2. Doshi, B.T. Queueing systems with vacations—A survey. *Queueing Syst.* **1986**, *1*, 29–66. [CrossRef]
3. Lee, H.W.; Lee, S.S.; Park, J.O.; Chae, K.C. Analysis of the Mx/G/1 queue by N-policy and multiple vacations. *J. Appl. Probab.* **1994**, *31*, 476–496. [CrossRef]
4. Lee, H.; Lee, S.; Chae, K. A fixed-size batch service queue with vacations. *J. Appl. Math. Stoch. Anal.* **1996**, *9*, 205–219. [CrossRef]

5. Krishna Reddy, G.V.; Nadarajan, R.; Arumuganathan, R. Analysis of a bulk queue with N-policy multiple vacations and setup times. *Comput. Oper. Res.* **1998**, *25*, 957–967. [CrossRef]
6. Jeyakumar, J.; Senthilnathan, B. A study on the behaviour of the server breakdown without interruption in a Mx/G(a, b)/1 queueing system with multiple vacations and closedown time. *Appl. Math. Comput.* **2012**, *219*, 2618–2633. [CrossRef]
7. Arumuganathan, R.; Jeyakumar, S. Steady state analysis of a bulk queue with multiple vacations, setup times with N-policy and closedown times. *Appl. Math. Model.* **2005**, *29*, 972–986. [CrossRef]
8. Haridass, M.; Arumuganathan, R. Analysis of a MX/G(a,b)/1 queueing system with vacation interruption. *RAIRO-Oper. Res.* **2012**, *46*, 305–334. [CrossRef]
9. Siddiqui, S.; Darbari, M.; Yagyasen, D. An QPSL Queuing Model for Load Balancing in Cloud Computing. *Int. J. E-Collab. (IJeC)* **2020**, *16*, 33–48. [CrossRef]
10. Agrawal, P.; Jain, M.; Singh, A. Optimal N-Policy for Finite Queue with Server Breakdown and State-Dependent Rate. 2017. Available online: https://www.semanticscholar.org/paper/Optimal-N-policy-for-Finite-Queue-with-Server-and-%22-Agrawal-Jain/784cd6d4343269915464c47eeb604ddc2ab4ff8c (accessed on 25 May 2024).
11. Niranjan, S.P.; Chandrasekaran, V.M.; Indhira, K. Two-Level Control Policy of an Unreliable Queueing System with Queue Size-Dependent Vacation and Vacation Disruption. In *Trends in Mathematics, Proceedings of the International Conference on Advances in Mathematical Sciences, Vellore, India, 1 December 2017*; Birkhäuser: Cham, Switzerland, 2018; Volume I, pp. 373–382. [CrossRef]
12. Govindan, A.; Deepa, T. Analysis of batch arrival bulk service queue with additional optional service multiple vacation and setup time. *Int. J. Math. Oper. Res.* **2019**, *15*, 1. [CrossRef]
13. Siddiqui, D.-S. Modelling and Simulation of Queuing Models Through the concept of Petri Nets ADCAIJ: Advances in Distributed Computing. *Adv. Distrib. Comput. Artif. Intell. J.* **2020**, *9*, 17–28.
14. Banerjee, S.; Budhiraja, A.; Estevez, B. Load Balancing in Parallel Queues and Rank-based Diffusions. *arXiv* **2024**, arXiv:2302.10317. [CrossRef]
15. Otten, S. Load balancing in a network of queueing-inventory systems. *Ann. Oper. Res.* **2023**, *331*, 807–837. [CrossRef]
16. van der Boor, M.; Borst, S.; van Leeuwaarden, J. Optimal hyper-scalable load balancing with a strict queue limit. *Perform. Eval.* **2021**, *149–150*, 102217. [CrossRef]
17. Liu, X.; Ying, L. Universal Scaling of Distributed Queues Under Load Balancing in the Super-Halfin-Whitt Regime. *IEEE/ACM Trans. Netw.* **2022**, *30*, 190–201. [CrossRef]
18. Ahmed, H.; Arshad, M.J.; Muhammad, S.; Ahmad, S.; Zahid, A.H. Queue length-based load balancing in data center networks. *Int. J. Commun. Syst.* **2020**, *33*, e4472. [CrossRef]
19. Hellemans, T.; Kielanski, G.; Van Houdt, B. Performance of Load Balancers with Bounded Maximum Queue Length in Case of Non-Exponential Job Sizes. *IEEE/ACM Trans. Netw.* **2023**, *31*, 1626–1641. [CrossRef]
20. der Boor, M.V.; Borst, S.C.; Van Leeuwaarden, J.S.H.; Mukherjee, D. Scalable Load Balancing in Networked Systems: A Survey of Recent Advances. *SIAM Rev.* **2022**, *64*, 554–622. [CrossRef]
21. Delasay, M.; Akan, M. Efficient Allocation of Load-Balancing and Differentiation Tasks in Tandem Queue Services; SSRN, 2024; p. 4830834. Available online: https://ssrn.com/abstract=4830834 (accessed on 13 December 2024).
22. Santos, B.; Soares, A.; Nguyen, T.-A.; Min, D.-K.; Lee, J.-W.; Silva, F.-A. IoT Sensor Networks in Smart Buildings: A Performance Assessment Using Queuing Models. *Sensors* **2021**, *21*, 5660. [CrossRef]
23. Rodrigues, L.; Rodrigues, J.J.P.C.; de Serra, S.B.; Silva, F.A. A Queueing-Based Model Performance Evaluation for Internet of People Supported by Fog Computing. *Future Internet* **2022**, *14*, 23. [CrossRef]
24. Katayama, Y.; Tachibana, T. Optimal Task Allocation Algorithm Based on Queueing Theory for Future Internet Application in Mobile Edge Computing Platform. *Sensors* **2022**, *22*, 4825. [CrossRef]
25. Zhan, W.; Jiang, M.; Wang, X. Optimal Capacity Decision-Making of Omnichannel Catering Merchants Considering the Service Environment Based on Queuing Theory. *Systems* **2022**, *10*, 144. [CrossRef]
26. Alnowibet, K.A.; Alrasheedi, A.F.; Alqahtani, F.S. Queuing Models for Analyzing the Steady-State Distribution of Stochastic Inventory Systems with Random Lead Time and Impatient Customers. *Processes* **2022**, *10*, 624. [CrossRef]
27. Jin, H.-L.; Di, T.-L.; Yu, H.; Zhang, R.R. On the τ Decomposition Method for the Stability and Bifurcation of the TCP/AQM Networks versus Time Delay. *Symmetry* **2022**, *14*, 463. [CrossRef]
28. Yen, T.-C.; Wang, K.-H.; Chen, J.-Y. Optimization Analysis of the N Policy M/G/1 Queue with Working Breakdowns. *Symmetry* **2020**, *12*, 583. [CrossRef]
29. Kothandaraman, D.; Kandaiyan, I. Analysis of a Heterogeneous Queuing Model with Intermittently Obtainable Servers under a Hybrid Vacation Schedule. *Symmetry* **2023**, *15*, 1304. [CrossRef]
30. Kempa, W.M.; Paprocka, I. A Discrete-Time Queueing Model of a Bottleneck with an Energy-Saving Mechanism Based on Setup and Shutdown Times. *Symmetry* **2024**, *16*, 63. [CrossRef]
31. Chydzinski, A.; Adamczyk, B. Response Time of Queueing Mechanisms. *Symmetry* **2024**, *16*, 271. [CrossRef]

32. Niranjan, S.P.; Devi Latha, S.; Mahdal, M.; Karthik, K. Multiple Control Policy in Unreliable Two-Phase Bulk Queueing System with Active Bernoulli Feedback and Vacation. *Mathematics* **2024**, *12*, 75. [CrossRef]
33. Cost Optimization in Sintering Process on the Basis of Bulk Queueing System with Diverse Services Modes and Vacation. Available online: https://www.mdpi.com/2227-7390/12/22/3535 (accessed on 13 December 2024).
34. Niranjan, S.P. Managerial decision analysis of bulk arrival queuing system with state dependent breakdown and vacation. *Int. J. Adv. Oper. Manag.* **2020**, *12*, 351–376. [CrossRef]
35. Niranjan, S.P.; Chandrasekaran, V.M.; Indhira, K. Queue size dependent service in bulk arrival queueing system with server loss and vacation break-off. *Int. J. Knowl. Manag. Tour. Hosp.* **2017**, *1*, 176–207. [CrossRef]
36. Niranjan, S.P.; Latha, S.D. Analyzing the Two-Phase Heterogeneous and Batch Service Queuing System with Breakdown in Two-Phases, Feedback, and Vacation. *Baghdad Sci. J.* **2024**, *21*, 2701. [CrossRef]
37. Gautam, A.; Choudhury, G.; Dharmaraja, S. Performance analysis of DRX mechanism using batch arrival vacation queueing system with N-policy in LTE-A networks. *Ann. Telecommun.* **2020**, *75*, 353–367. [CrossRef]
38. Neuts, M.F. A General Class of Bulk Queues with Poisson Input. *Ann. Math. Stat.* **1967**, *38*, 759–770. [CrossRef]
39. Cox, D.R. The analysis of non-Markovian stochastic processes by the inclusion of supplementary variables. *Math. Proc. Camb. Phil. Soc.* **1955**, *51*, 433–441. [CrossRef]