Contents lists available at ScienceDirect

Journal of King Saud University - Science

journal homepage: www.sciencedirect.com

Full Length Article

Stochastic modelling and meta heuristic optimization of a batch production system

Harini R., Indhira K. *

Department of Mathematics, School of Advanced Sciences, Vellore Institute of Technology, Vellore 632 014, Tamil Nadu, India

ARTICLE INFO

Keywords:
Batch production
Quorum queues
Stand-by server
Cost optimization

ABSTRACT

Batch production is a modern strategy for manufacturing, especially for products unsuitable for continuous assembly lines. This study proposes a method for analysing one such production facility. Here, we characterize a batch production system as a unique sort of quorum feedback retrial queue deployed under Bernoulli working vacation and outfitted with a secondary, stand-by server. The system's dynamics are thoroughly examined using the supplementary variable technique, providing insights into its behaviour and performance. Key metrics are determined, and the impact of specific parameters is demonstrated through numerical examples. Furthermore, this research aims to achieve the optimal system cost by employing a variety of optimization approaches. Thus, optimizing costs and identifying key metrics are crucial for enhancing efficiency and profitability in batch production systems. This process ensures that resources are used effectively, service quality is maintained, and overall operational costs are reduced, leading to a more sustainable and profitable production system which helps to make informed decisions.

1. Introduction

For many years, batch production has been a key element of manufacturing. It is a low-cost manufacturing technology that enables manufacturers to produce different types of products in discrete batches, making it ideal for industries that require frequent product changes or variations. It is crucial in modern manufacturing as it offers flexibility and efficiency, allowing for the production of diverse products in discrete batches. Unlike continuous assembly line production, batch production is ideal for managing varying demand, reducing setup times, and minimizing waste. This approach is particularly advantageous for producing customized or seasonal products, ensuring optimal resource utilization and cost-effectiveness. Overall, batch production provides manufacturers with the agility to respond to market changes and customer needs more effectively, while maintaining high standards of quality and efficiency. Further, a significant part of our lives is spent waiting in line, or "queuing", where we present ourselves to a server in order to have our requests met. A queueing system (QS) can be thought of as any instance in which a client must wait in line. There are many queueing models emerged in the last few decades which were helpful in designing many real life systems. In this regard, we characterize a batch production system (BPS) as a novel bulk feedback quorum retrial queue (RQ). In this approach, the server will only start manufacturing if the predetermined quorum size, r, is met. However, if there are more products waiting than r, just the first r will be serviced. Each product is served by a single server in batches of exactly size r. In addition, the server will undergo working vacation (WV) if there are no products in the system. Further, if there is breakdown due to machine malfunction or any other reason, the server will be immediately sent for repair. During that period, a secondary server namely the stand-by server will come in rescue to ensure continuous service.

1.1. Literature survey

To comprehend the research gaps present, a thorough review of the prior studies is presented as follows:

Several experts have investigated client arrivals in bulks. Moreover, due to its multidisciplinary nature, most research focuses on batch service queues operating under various regulations. In their groundbreaking work, Chaudhry and Templeton (1983) devised a QS called quorum queue, in which a server prioritizes and serves requests in batches of fixed size $r \leq 1$. By employing embedded Markov chain methods, Tadj and Ke (2008)

* Corresponding author.
E-mail address: kindhira@vit.ac.in (Indhira K.).

https://doi.org/10.1016/j.jksus.2024.103308
Received 22 November 2023; Received in revised form 30 May 2024; Accepted 12 June 2024
Available online 20 June 2024

addressed a hysteretic bulk quorum queue. Tadj et al. (2012) also scrutinized a dual-phase unreliable bulk quorum queue. Taking into account a variety of arrival scenarios, Kashifalghitaa and Al-Obaidi (2023) recently examined a quorum queue with an N-policy.

If a client is dissatisfied with the service they received, they can request that the service be redone until they are satisfied. Lately, an asymptotic analysis of a RQ with feedback was carried out by Nazarov et al. (2023). In addition, Laxmi et al. (2023) scrutinized a feedback batch arrival QS with second discretionary service.

Further, distinct systems face varying kinds of vacations based on the circumstances. Servi and Finn (2002) were among the first to introduce an $M/M/1$ QS subjected under WV. An batch arrival unreliable RQ with negative consumers under Bernoulli working vacation (BWV) was addressed by Rajadurai et al. (2016). Further, an analysis of a bulk arrival pre-emptive priority RQ subjected to delayed repairs under a modified Bernoulli vacation schedule was done by Yuvarani and Saravanarajan (2017). Lately, a transient analysis of a two classes of bulk arrivals, retrial clients with non-pre-emptive priority service and WVs was studied by Ayyappan and Thamizhselvi (2018)

While developing any system, preventative steps should be incorporated. Therefore, one of the most crucial topics in this study is server breakdowns and their repair. Thus, many of the explored systems have thought about this issue, including Bagyam et al. (2013) etc,. Recently, Bharathidass et al. (2018) analysed a batch service queue with server breakdowns and repairs. A finite Markovian QS subjected to breakdown was also recently analysed by Das et al. (2022).

There is not much of work on RQ that addresses cost optimization and evaluates appropriate control parameters for the queueing model using many optimization techniques. Lately, Vaishnawi et al. (2022) analysed the accuracy of a $Geo^{[X]}/Geo/1$ recurrent model in discrete time to induce the lowest possible cost for the system. Moreover, refer to articles (Kumar and Jain, 2023; Deora et al., 2021; Jain and Kumar, 2022) for a review of published works on cost optimization.

1.2. Research novelty

Based on the literature review section (Section 1.1), no work on quorum RQ with BWV has been conducted. Thus, with the objective of filling this research gap, this study addresses a BPS via bulk quorum feedback RQ with BWV, random breakdown, and a stand-by server, which is evaluated by the supplementary variable technique (SVT). Further, the proposed framework has been analysed using six distinct optimization methods.

1.3. Rationale for choosing a BPS and its benefits

- **Flexibility:** Allows production of varied products without extensive changeovers.
- **Efficiency:** Optimizes resource use by grouping similar tasks.
- **Cost Efficiency:** Leads to economies of scale and efficient resource use.
- **Customization:** Easily accommodates custom orders and product variations.
- **Risk Mitigation:** Smaller batches reduce the risk of large-scale production defects.
- **Resource Allocation:** Improves resource allocation by dedicating specific resources to particular batches.

1.4. Structure of the work

The rest of the work is outlined as follows: Section 2 presents an analysis of the proposed framework. The system's stability criterion, associated equations, and steady state (SS) outcomes are determined in Section 3. Several essential system metrics are covered in Section 4. The system's behaviour is addressed via a few numerical examples in Section 5. Section 6 delves into the detailed assessment of cost optimization via distinct approaches. In Section 7, the suggested model is discussed briefly. Overall study is summarized in Section 8.

2. Description of the model

Arrival process: A compound "Poisson process" with rate ϵ enables products to congregate in bulk. Presuming a general distribution for $\mathcal{W}_k, k = 1, 2, 3, \dots$, then $\mathcal{W}_k$ represents the no. of products linked alongside kth arrival batch. $Pr[\mathcal{W}_k = n] = \mathcal{B}_n, n = 1, 2, 3, \dots$ and $\mathcal{W}(\check{z})$ indicate the PGF of $\mathcal{W}$.

Retrial process: In the event if there is no waiting area and a bulk of products arrive while the server is still available, one of them starts manufacturing while the others enrol into orbit. Given probability $1 - b$, incoming products either leave the service sector or, with the probability b, they enter an orbit of obstructed products if the server is busy, under vacation, or under breakdown. We presume the RQ follows $FCFS$ restriction, with the products retrial times distributed equally with a random distribution $\mathcal{G}(\tilde{x})$ in accordance with Laplace Stieltjes Transform (LST) $\mathcal{G}^*(\hat{t})$.

Regular service process: The server waits inactively until the queue length reaches an integer threshold, r, before taking any action. However, if there are at least r products waiting, the server will begin manufacturing them in batches of precisely r. Consider the service time period to be $\mathcal{Q}$ which follows a general distribution with a probability distribution function (PDF) $\mathcal{Q}_b(\tilde{x})$ with a corresponding LST $\mathcal{Q}_b^*(\hat{t})$.

Bernoulli Working Vacation process: When an orbit becomes accessible, the server initiates a WV with parameter η which follows an exponential distribution. In the event that any new users sign up while the server is on vacation, the service will continue albeit at a slower pace. The server will terminate the vacation and return to the normal busy period if any orbiting products with noticeably longer completion times complete their service within the vacation time. Consequently, this will cut into your vacation time. In case there are no products within system when the vacation finishes, the server will either enter the system and wait passively for a new batch of products with probability p (single WV) and service them in ordinary mode as they arrive, or with probability $\bar{p}$ it will leave for next WV. A arbitrary variable $\mathcal{Q}_v$ with a distribution function $\mathcal{Q}_v(\tilde{x})$ of LST $\mathcal{Q}_v^*(\hat{t})$ describes the time spent in service throughout the WV period.

Feedback Rule: Upon the termination of service for every products, unsatisfied products have a probability of s $(0 \le s \le 1)$ of reverting to the orbit to acquire one more ordinary service, or a probability of $1 - s$ of quitting the system.

Random Breakdown process: A momentary server downtime could occur at any time while the server remains functional owing to an interruption in the service channel. The server breakdowns occur at a pace of δ associated with exogenous Poisson processes. The server that malfunctions is sent out right away for repair. The product, who had just begun receiving service before the server went down, is waiting for the rest of the service to be provided. The server's repair time distribution is taken to be randomly distributed with PDF $\mathcal{S}(\hat{y})$ and LST $\mathcal{S}^*(\hat{t})$.

Stand-by server: When the primary server goes down, the stand-by server takes over and begins manufacturing the products immediately. The stand-by service times obey an exponential distribution with a rate $\alpha > 0$.

It is presumed that there is no connection between any of the system's stochastic processes.

2.1. Interpretation of the proposed model into BPS

The proposed framework introduces a novel application for batch production systems, enhancing production facility operations.

Products arrive in batches, allowing efficient processing of multiple items simultaneously, reducing setup times, and increasing throughput. If the processing system is busy, batches wait in an orbit or may balk to avoid delays. Batches in orbit retry processing later if the server remains busy or the queue limit is reached, managing workload efficiently. To handle random equipment breakdowns, a standby server activates during failures. The secondary stand-by server acts as a backup, remaining inactive to conserve resources but swiftly activating during server failures or peak demand. Its role improves system reliability by minimizing downtime and disruptions, while also optimizing resource utilization for consistent and efficient production. During periods of low demand or when the system is not operating at full capacity, servers may enter a working vacation mode where they operate at a reduced capacity. Finally, the system undergoes feedback mechanisms where the queue lengths and processing times, as well as adjusting system parameters such as batch sizes, processing rates, server availability, and system reliability, are continuously monitored. Thus, the production process significantly influences the modelling, analysis, and outcomes of the study leading to more effective and resilient production management strategies.

3. Steady state evaluation

We establish the hazard rate functions for system states within this section. To elucidate the framework, we defined distinct server states based on products availability. Prior to formulating the state space differential equations for various system states, we verified the model's ergodicity. Additionally, by incorporating supplementary variables, we derived the SS differential equations and their corresponding boundary conditions using the SVT. Ultimately, employing the generating function method, we obtained the steady-state solution of the proposed model, thereby determining the PGF of the number of products in the system and the orbit.

3.1. The steady state equations

In SS, we take into consideration that $\mathcal{G}(0)=0$, $\mathcal{G}(\infty)=1$, $\mathcal{Q}_b(0)=0$, $\mathcal{Q}_b(\infty)=1$, and $\mathcal{Q}_v(0)=0$, $\mathcal{Q}_v(\infty)=1$, are continuous at $\tilde{x}=0$ and $\mathcal{S}(0)=0$, $\mathcal{S}(\infty)=1$, are continuous at $\hat{y}=0$. Hence, the functions $\theta(\tilde{x})$, $\upsilon_b(\tilde{x})$, $\upsilon_v(\tilde{x})$, and $\nu(\hat{y})$ are the "conditional completion rates (hazard rates)" for retrial, regular service, on BWV, and under repair, respectively.

$$\theta(\tilde{x})d\tilde{x}=\frac{d\mathcal{G}(\tilde{x})}{1-\mathcal{G}(\tilde{x})};\ \upsilon_b(\tilde{x})d\tilde{x}=\frac{d\mathcal{Q}_b(\tilde{x})}{1-\mathcal{Q}_b(\tilde{x})};\ \upsilon_v(\tilde{x})d\tilde{x}=\frac{d\mathcal{Q}_v(\tilde{x})}{1-\mathcal{Q}_v(\tilde{x})};\ \nu(\hat{y})d\hat{y}=\frac{d\mathcal{S}(\hat{y})}{1-\mathcal{S}(\hat{y})}$$

$\mathcal{G}^0(\check{\epsilon})$, $\mathcal{Q}_b^0(\check{\epsilon})$, $\mathcal{Q}_v^0(\check{\epsilon})$ and $\mathcal{S}^0(\check{\epsilon})$ indicate the elapsed retrial period, service times, BWV, and repair period during time $\check{\epsilon}$, correspondingly. Furthermore, create the random variable,

$$\Theta(\check{\epsilon})=\begin{cases}0, & \text{the server being available and in WV period}\\ 1, & \text{the availability and normal operation of the server}\\ 2, & \text{the server being occupied and in normal service period at time } \check{\epsilon}\\ 3, & \text{the server being occupied and in WV period at time } \check{\epsilon}\\ 4, & \text{the server is under repair at time } \check{\epsilon}.\end{cases}$$

Furthermore, we emphasize how the bivariate Markov process can be utilized to describe the system's state at a given time $\check{\epsilon}$ $\{\Theta(\check{\epsilon}),\ \Xi(\check{\epsilon});\ \check{\epsilon}\geq 0\ \}$, where $\Theta(\check{\epsilon})$ indicates the server state $(0,1,2,3,4)$ based on whether the server is inactive, engaged, on BWV or under repair. For all products in the orbit at time $\check{\epsilon}$, $\Xi(\check{\epsilon})$ is the number that represents them. If $\Theta(\check{\epsilon})=1$ and $\Xi(\check{\epsilon})>0$, then $\mathcal{G}^0(\check{\epsilon})$ corresponds to the elapsed retrial time. If $\Theta(\check{\epsilon})=2$ and $\Xi(\check{\epsilon})\geq 0$, then $\mathcal{Q}_b^0(\check{\epsilon})$ is equivalent to the elapsed time of the product being manufactured regularly. If $\Theta(\check{\epsilon})=3$ and $\Xi(\check{\epsilon})\geq 0$, then $\mathcal{Q}_v^0(\check{\epsilon})$ proportional to the elapsed time of the product spent manufacturing in slow service period. The elapsed time of the server getting fixed corresponds to $\mathcal{S}^0(\check{\epsilon})$ if $\Theta(\check{\epsilon})=4$ and $\Xi(\check{\epsilon})\geq 0$.

SVT, a technique for analysing non-Markovian processes $\{\Xi(\check{\epsilon});\ \geq 0\ \}$, is crucial for studying complex systems like BPS. By incorporating supplementary variables, SVT transforms the BPS into a Markovian process, enabling analysis using generating function methods. This approach enhances precision in calculating performance metrics and offers a deeper understanding of system behaviour. SVT's flexibility allows for comprehensive analysis of the BPS under different scenarios, facilitating informed decision-making.

Theorem 3.1. *The embedded Markov chain (MC) $\{D_n; n\epsilon N\}$ is ergodic iff $\omega<R$, where $\omega=\left[s+\mathcal{X}_1(1-\mathcal{G}^*(\epsilon))-\left(\epsilon b\mathcal{X}_1(1+\delta\mu^{(1)})+\delta\alpha\mu^{(1)}\right)\beta_b^{(1)}\right]$*

Proof. It is proved on the lines of the proof given by Kumar and Madheswari (2003) in Theorem 1 □

Let $\{\check{\epsilon}_n; n=1,2,\dots\}$ be the set of epochs during which a regular service period, BWV period, or repair period ends. As a result, an MC is generated and added in the retry QS utilizing a series of random vectors $D_n=\{\Theta(\check{\epsilon}_n+),\Xi(\check{\epsilon}_n+)\}$. Theorem 3.1 asserts that $D_n; n\in N$ is ergodic iff $\left[s+\mathcal{X}_1(1-\mathcal{G}^*(\epsilon))-\left(\epsilon b\mathcal{X}_1(1+\delta\mu^{(1)})+\delta\alpha\mu^{(1)}\right)\beta_b^{(1)}\right]<1$, consequently our system must be stable. The probabilities, $\Lambda_0(\check{\epsilon})=P\{\Theta(\check{\epsilon})=0,\Xi(\check{\epsilon})=0\}$ and $\Psi_0(\check{\epsilon})=P\{\Theta(\check{\epsilon})=1,\Xi(\check{\epsilon})=0\}$; and probability densities for the method $\{\Xi(\check{\epsilon}),\check{\epsilon}\geq 0\}$, are stated below,

$$\begin{aligned}\Psi_n(\tilde{x},\check{\epsilon})d\tilde{x}=&P\{\Theta(\check{\epsilon})=0,\Xi(\check{\epsilon})=n,\tilde{x}\leq\mathcal{G}_i^0(\check{\epsilon})<\tilde{x}+d\tilde{x}\},\ \text{for } \check{\epsilon},\tilde{x}\geq 0\ \text{ and }\ n\geq 1\\ \Gamma_{b,n}(\tilde{x},\check{\epsilon})d\tilde{x}=&P\{\Theta(\check{\epsilon})=1,\Xi(\check{\epsilon})=n,\tilde{x}\leq\mathcal{Q}_b^0(\check{\epsilon})<\tilde{x}+d\tilde{x}\},\ \text{for } \check{\epsilon},\tilde{x}\geq 0,n\geq 0\\ \Lambda_{v,n}(\tilde{x},\check{\epsilon})d\tilde{x}=&P\{\Theta(\check{\epsilon})=2,\Xi(\check{\epsilon})=n,\tilde{x}\leq\mathcal{Q}_v^0(\check{\epsilon})<\tilde{x}+d\tilde{x}\},\ \text{for } \check{\epsilon},\tilde{x}\geq 0,n\geq 0\\ \Omega_n(\tilde{x},\hat{y},\check{\epsilon})d\tilde{x}=&P\{\Theta(\check{\epsilon})=3,\Xi(\check{\epsilon})=n,\hat{y}\leq\mathcal{S}^0(\check{\epsilon})<\hat{y}+d\hat{y}/\mathcal{Q}_b^0(\check{\epsilon})=\tilde{x}\},\\ &\text{for }\ \check{\epsilon}\geq 0,(\tilde{x},\hat{y})\geq 0,n\geq 0\end{aligned}$$

We believe the follow-up meets the stability criteria, and as a result we get $\Psi_0 = \lim_{\check{\varepsilon}\to\infty} \Psi_0(\check{\varepsilon})$ and $\Lambda_0 = \lim_{\check{\varepsilon}\to\infty} \Lambda_0(\check{\varepsilon})$. We can therefore designate limiting probabilities for $\tilde{x}, \hat{y} > 0,\ n \geq 0$

$$\Psi_n(\tilde{x}) = \lim_{\check{\varepsilon}\to\infty} \Psi_n(\tilde{x}, \check{\varepsilon});\ \Gamma_{b,n}(\tilde{x}) = \lim_{\check{\varepsilon}\to\infty} \Gamma_{b,n}(\tilde{x}, \check{\varepsilon});$$
$$\Lambda_{v,n}(\tilde{x}) = \lim_{\check{\varepsilon}\to\infty} \Lambda_{v,n}(\tilde{x}, \check{\varepsilon});\ \Omega_n(\tilde{x}, \hat{y}) = \lim_{\check{\varepsilon}\to\infty} \Omega_n(\tilde{x}, \hat{y}, \check{\varepsilon}).$$

SVT yields the subsequent equations, that regulate the behaviour of the system dynamics.

$$\epsilon\Psi_0 = \eta p \Lambda_0 \tag{3.1}$$

$$(\epsilon+\eta)\Lambda_0 = \eta\bar{p}\Lambda_0 + \int_0^\infty \Gamma_{b,0}(\tilde{x})v_b(\tilde{x})d\tilde{x} + \int_0^\infty \Lambda_{v,0}(\tilde{x})v_v(\tilde{x})d\tilde{x} + \int_0^\infty \Omega_0(\tilde{x},\hat{y})v(\hat{y})d\hat{y} \tag{3.2}$$

$$\frac{d\Psi_n(\tilde{x})}{d\tilde{x}} = -(\epsilon+\theta(\tilde{x}))\Psi_n(\tilde{x});\ \ n \geq 1 \tag{3.3}$$

$$\frac{d\Gamma_{b,n}(\tilde{x})}{d\tilde{x}} + (\epsilon+\delta+v_b(\tilde{x}))\Gamma_{b,n}(\tilde{x}) = \epsilon(1-b)\Gamma_{b,n}(\tilde{x}) + \epsilon b\sum_{k=1}^{n} b_k\Gamma_{b,n-k}(\tilde{x}) + \int_0^\infty \Omega_n(\tilde{x},\hat{y})v(\hat{y})d\hat{y};\ \ n \geq 0 \tag{3.4}$$

$$\frac{d\Lambda_{v,n}(\tilde{x})}{d\tilde{x}} + (\epsilon+\eta+v_v(\tilde{x}))\Gamma_{b,n}(\tilde{x}) = \epsilon(1-b)\Lambda_{v,n}(\tilde{x}) + \epsilon b\sum_{k=1}^{n} b_k\Lambda_{v,n-k}(\tilde{x});\ \ n \geq 0 \tag{3.5}$$

$$\frac{d\Omega_n(\tilde{x},\hat{y})}{d\hat{y}} + (\epsilon+\alpha+v(\hat{y}))\Omega_n(\tilde{x},\hat{y}) = \epsilon(1-b)\Omega_n(\tilde{x},\hat{y}) + \epsilon b\sum_{k=1}^{n} b_k\Omega_{n-k}(\tilde{x},\hat{y}) + \alpha\Omega_n(\tilde{x},\hat{y});\ \ n \geq 0 \tag{3.6}$$

At $\tilde{x} = 0,\ \ \hat{y} = 0$ the SS boundary conditions are as follows:

$$\Psi_n(0) = s\int_0^\infty \Gamma_{b,n-1}(\tilde{x})v_b(\tilde{x})d\tilde{x} + \bar{s}\int_0^\infty \Gamma_{b,n}(\tilde{x})v_b(\tilde{x})d\tilde{x} + s\int_0^\infty \Lambda_{v,n-1}(\tilde{x})v_v(\tilde{x})d\tilde{x} + \bar{s}\int_0^\infty \Lambda_{v,n}(\tilde{x})v_v(\tilde{x})d\tilde{x};\ \ n \geq 1 \tag{3.7}$$

$$\Gamma_{b,n}(0) = \int_0^\infty \Psi_{n+R}(\tilde{x})\theta(\tilde{x})d\tilde{x} + \epsilon\sum_{k=1}^{n} b_k\int_0^\infty \Psi_{n-k+R}(\tilde{x})d\tilde{x} + \epsilon b_{k+R}\Psi_0 + \eta\int_0^\infty \Lambda_{v,n+R}d\tilde{x};\ \ n \geq 0 \tag{3.8}$$

$$\Lambda_{v,n,}(0) = \begin{cases} \epsilon\Lambda_0;\ \ n = 0 \\ 0;\ \ n \geq 1 \end{cases} \tag{3.9}$$

$$\Omega_n(\tilde{x}, 0) = \delta\Gamma_b(\tilde{x}); n \geq 0 \tag{3.10}$$

The normalizing condition is

$$\Psi_0 + \Lambda_0 + \sum_{n=1}^{\infty}\int_0^\infty \Psi_n(\tilde{x})d\tilde{x} + \sum_{n=0}^{\infty}\left[\int_0^\infty \Gamma_{b,n}(\tilde{x})d\tilde{x} + \int_0^\infty \Lambda_{v,n}(\tilde{x})d\tilde{x} + \int_0^\infty\int_0^\infty \Omega_n(\tilde{x},\hat{y})d\hat{y}\right] = 1 \tag{3.11}$$

3.2. The steady state solution

The generating function technique assists to generate the SS solution to fit the RQ system. Furthermore, the generating functions for $|\ \check{z}\ | < 1$ are stated beneath in order to solve the equations above:

$$b(\check{z}) = \sum_{n=1}^{\infty} \mathcal{B}_n\check{z}^n;\ \Psi(\tilde{x},\check{z}) = \sum_{n=1}^{\infty}\Psi_n(\tilde{x})\check{z}^n;\ \Psi(0,\check{z}) = \sum_{n=1}^{\infty}\Psi_n(0)\check{z}^n;$$
$$\Gamma_b(\tilde{x},\check{z}) = \sum_{n=0}^{\infty}\Gamma_{b,n}(\tilde{x})\check{z}^n;\ \Gamma_b(0,\check{z}) = \sum_{n=0}^{\infty}\Gamma_{b,0}(0)\check{z}^n;\ \Lambda_v(\tilde{x},\check{z}) = \sum_{n=0}^{\infty}\Lambda_{v,n}(\tilde{x})\check{z}^n;$$
$$\Lambda_v(0,\check{z}) = \sum_{n=0}^{\infty}\Lambda_{v,0}(0)\check{z}^n;\ \Omega(\tilde{x},\hat{y},\check{z}) = \sum_{n=0}^{\infty}\Omega_n(\tilde{x},\hat{y})\check{z}^n;\ \Omega(\tilde{x},0,\check{z}) = \sum_{n=0}^{\infty}\Omega_0(\tilde{x},0)\check{z}^n$$

Multiply the SS equations and boundary conditions from (3.3) to (3.10) by $\check{z}^n$ and sum on n where $(n = 0, 1, 2, \ldots)$.

$$\frac{\partial}{\partial\tilde{x}}\Psi(\tilde{x},\check{z}) + (\epsilon+\theta(\tilde{x}))\Psi(\tilde{x},\check{z}) = 0 \tag{3.12}$$

$$\frac{\partial}{\partial\tilde{x}}\Gamma_b(\tilde{x},\check{z}) + [\epsilon b(1-b(\check{z})) + \delta + v_b(\tilde{x})]\Gamma_b(\tilde{x},\check{z}) = \int_0^\infty \Omega(\tilde{x},\hat{y},\check{z})v(\hat{y})d\hat{y} \tag{3.13}$$

$$\frac{\partial}{\partial\tilde{x}}\Lambda_v(\tilde{x},\check{z}) + [\epsilon b(1-b(\check{z})) + \eta + v_v(\tilde{x})]\Lambda_v(\tilde{x},\check{z}) = 0 \tag{3.14}$$

$$\frac{\partial}{\partial \hat{y}}\Omega(\tilde{x},\hat{y},\check{z})+[\epsilon b(1-b(\check{z}))+\alpha(1-\frac{1}{\check{z}})+\nu(\hat{y})]\Omega(\tilde{x},\hat{y},\check{z})=0 \tag{3.15}$$

$$\Psi(0,\check{z})=(s\check{z}+\bar{s})\int_0^\infty \Gamma_b(\tilde{x},\check{z})\upsilon_b(\tilde{x})d\tilde{x}+(s\check{z}+\bar{s})\int_0^\infty \Lambda_v(\tilde{x},\check{z})\upsilon_v(\tilde{x})d\tilde{x}-(\epsilon+p\eta)\Lambda_0 \tag{3.16}$$

$$\Gamma_b(0,\check{z})=\frac{1}{\check{z}^R}\int_0^\infty \Psi(\tilde{x},\check{z})\theta(\tilde{x})d\tilde{x}+\frac{b(\check{z})\epsilon\Psi_0}{\check{z}^R}+\frac{\epsilon b(\check{z})}{\check{z}^R}\int_0^\infty \Psi(\tilde{x},\check{z})d\tilde{x}+\frac{\eta}{\check{z}^R}\int_0^\infty \Lambda_v(\tilde{x},\check{z})d\tilde{x} \tag{3.17}$$

$$\Lambda_v(0,\check{z})=\epsilon\Lambda_0 \tag{3.18}$$

$$\Omega(\tilde{x},0,\check{z})=\delta\Gamma_b(\tilde{x},\check{z}) \tag{3.19}$$

Theorem 3.2. *Underneath the stability condition $\omega < R$, the stationary distribution of the no. of products in the system when the server is vacant, is engaged, on BWV and during repair are provided by,*

$$\Psi(\check{z})=\frac{\Lambda_0[1-\mathcal{G}^*(\epsilon)]\left\{[s\check{z}+\bar{s}]\mathcal{Q}_b^*(C_b(\check{z}))\big[b(\check{z})\eta p+\epsilon\mathcal{L}(\check{z})\big]+\check{z}^R\big[\epsilon[(s\check{z}+\bar{s})\mathcal{Q}_v^*(C_v(\check{z}))-1]-\eta p\big]\right\}}{\epsilon\left[\check{z}^R-[s\check{z}+\bar{s}]\mathcal{Q}_b^*(C_b(\check{z}))\big[\mathcal{G}^*(\epsilon)+b(\check{z}(1-\mathcal{G}^*(\epsilon)))\big]\right]} \tag{3.20}$$

$$\Gamma_b(\check{z})=\frac{\Lambda_0[1-\mathcal{Q}_b^*(C_b(\check{z}))]\left\{\eta p b(\check{z})+\epsilon\mathcal{L}(\check{z})+\big[\mathcal{G}^*(\epsilon)+b(\check{z}(1-\mathcal{G}^*(\epsilon)))\big]\big[\epsilon[(s\check{z}+\bar{s})\mathcal{Q}_v^*(C_v(\check{z}))-1]-\eta p\big]\right\}}{C_b(\check{z})\left[\check{z}^R-[s\check{z}+\bar{s}]\mathcal{Q}_b^*(C_b(\check{z}))\big[\mathcal{G}^*(\epsilon)+b(\check{z}(1-\mathcal{G}^*(\epsilon)))\big]\right]} \tag{3.21}$$

$$\Lambda_v(\check{z})=\frac{\epsilon\Lambda_0\mathcal{L}(\check{z})}{\eta} \tag{3.22}$$

$$\Omega(\check{z})=\frac{\delta\Lambda_0[1-\mathcal{Q}_b^*(C_b(\check{z}))][1-\mathcal{S}^*(\mathcal{A}(\check{z}))]\left\{\eta p b(\check{z})+\epsilon\mathcal{L}(\check{z})+\big[\mathcal{G}^*(\epsilon)+b(\check{z}(1-\mathcal{G}^*(\epsilon)))\big]\big[\epsilon[(s\check{z}+\bar{s})\mathcal{Q}_v^*(C_v(\check{z}))-1]-\eta p\big]\right\}}{\mathcal{A}(\check{z})C_b(\check{z})\left[\check{z}^R-[s\check{z}+\bar{s}]\mathcal{Q}_b^*(C_b(\check{z}))\big[\mathcal{G}^*(\epsilon)+b(\check{z}(1-\mathcal{G}^*(\epsilon)))\big]\right]} \tag{3.23}$$

where,

$$\Lambda_0=\frac{R-\left[s+\mathcal{X}_1(1-\mathcal{G}^*(\epsilon))-\left(\epsilon b\mathcal{X}_1(1+\delta\mu^{(1)})+\delta\alpha\mu^{(1)}\right)\beta_b^{(1)}\right]}{\left\{\begin{array}{l}\big[1+\frac{\epsilon}{\eta}(1-\mathcal{Q}^*(\eta))\big]\rho+\beta_b^{(1)}\left[\epsilon(1-s\mathcal{Q}^*(\eta))+\mathcal{X}_1(1-\mathcal{G}^*(\epsilon))+\epsilon\big[\frac{\epsilon b(1-\mathcal{Q}^*(\eta))}{\eta}+s\big]+\eta p[\mathcal{X}_1+1]\right]\\ \quad[1+\delta\mu^{(1)}]+(1-\mathcal{G}^*(\epsilon))\left[s+R[1-s\mathcal{Q}^*(\eta)]-[1-\mathcal{Q}^*(\eta)]\big[\epsilon b\mathcal{X}_1(1+\delta\mu^{(1)})\beta_b^{(1)}+s+\frac{\epsilon b\mathcal{X}_1}{\eta}\big]\right]\\ \qquad-\frac{\epsilon p}{\eta}\mathcal{G}^*(\epsilon)\big[R+s+\epsilon b\mathcal{X}_1(1+\delta\mu^{(1)})\beta_b^{(1)}\big]\end{array}\right\}} \tag{3.24}$$

$$\Psi_0=\frac{\eta p}{\epsilon}\times\Lambda_0 \tag{3.25}$$

and

$$\mathcal{E}(\check{z})=\epsilon b(1-b(\check{z}));\ C_b(\check{z})=\mathcal{E}(\check{z})+\delta[1-\mathcal{S}^*(\mathcal{A}(\check{z}))];$$
$$C_v(\check{z})=\eta+\mathcal{E}(\check{z});\ \mathcal{A}(\check{z})=\mathcal{E}(\check{z})+\alpha(1-\frac{1}{\check{z}});$$
$$\rho=R-\left[s+\mathcal{X}_1(1-\mathcal{G}^*(\epsilon))-\left(\epsilon b\mathcal{X}_1(1+\delta\mu^{(1)})+\delta\alpha\mu^{(1)}\right)\beta_b^{(1)}\right];$$
$$\mathcal{L}(\check{z})=\frac{\eta[1-\mathcal{Q}_v^*(C_v(\check{z}))]}{\eta+\mathcal{E}(\check{z})}$$

Proof. By formulating the below PGF and further by integrating them with respect to $\tilde{x}$ and $\hat{y}$

$$\Psi(\check{z})=\int_0^\infty \Psi(\tilde{x},\check{z})d\tilde{x},\ \Gamma_b(\check{z})=\int_0^\infty \Gamma_b(\tilde{x},\check{z})d\tilde{x},\ \Lambda_v(\check{z})=\int_0^\infty \Lambda_v(\tilde{x},\check{z})d\tilde{x},$$

$$\Omega(\tilde{x},\check{z})=\int_0^\infty \Omega(\tilde{x},\hat{y},\check{z})d\hat{y},\ \Omega(\check{z})=\int_0^\infty \Omega(\tilde{x},\check{z})d\tilde{x}$$

we get the above mentioned equations.

The two unknowns, Ψ_0 and Λ_0, which represent the likelihood that the server will be idle both in the ordinary busy phase and in the BWV phase if there are no products in the orbit, may be determined by employing the normalizing condition. Thus, via the use of $\check{z}=1$ in the Eqs. (3.20)–(3.23) and the rule of L-Hospitals when desired, we obtain,

$$\Psi_0+\Lambda_0+\Psi(1)+\Gamma_b(1)+\Lambda_v(1)+\Omega(1)=1.\quad\square$$

Theorem 3.3. *Pursuant to the stability criteria $\omega < 1$. the PGF of the no. of products in the system as well the orbit at a stationary point of period are estimated as follows,*

$$K_e(\check{z}) = \frac{Nr(\check{z})}{Dr(\check{z})} \tag{3.26}$$

$$\begin{aligned} Nr(\check{z}) =& \Lambda_0 \Bigg\{ \Bigg\{ \Big[\eta p + \frac{\epsilon}{\eta}[\eta \check{z}\mathcal{L}(\check{z})]\Big] \Bigg[\check{z}^R - [s\check{z}+\bar{s}]\mathcal{Q}_b^*(C_b(\check{z}))\big[\mathcal{G}^*(\epsilon) + b(\check{z}(1-\mathcal{G}^*(\epsilon)))\big]\Bigg] + \frac{[1-\mathcal{G}^*(\epsilon)]}{\epsilon} \\ & \Bigg[(s\check{z}+\bar{s})\mathcal{Q}_b^*(C_b(\check{z}))\big[b(\check{z}\eta p + \epsilon\mathcal{L}(\check{z}))\big] + \check{z}^R\big[\epsilon[(s\check{z}+\bar{s})\mathcal{Q}_v^*(C_v(\check{z}))-1]-\eta p\big]\Bigg]\Bigg\}\mathcal{A}(\check{z})C_b(\check{z}) + \check{z} \\ & \Bigg[\eta p b(\check{z}) + \epsilon\mathcal{L}(\check{z}) + \big[\mathcal{G}^*(\epsilon)+b(\check{z}(1-\mathcal{G}^*(\epsilon)))\big]\big[\epsilon[(s\check{z}+\bar{s})\mathcal{Q}_v^*(C_v(\check{z}))-1]-\eta p\big]\Bigg]\big[1-\mathcal{Q}_b^*(C_b(\check{z}))\big] \\ & \big[\mathcal{A}(\check{z}) + \delta(1-\mathcal{S}^*(\mathcal{A}(\check{z})))\big]\Bigg\} \end{aligned}$$

$$Dr(\check{z}) = \mathcal{A}(\check{z})\mathcal{Q}_b^*(C_b(\check{z}))\Bigg[\check{z}^R - [s\check{z}+\bar{s}]\mathcal{Q}_b^*(C_b(\check{z}))\big[\mathcal{G}^*(\epsilon)+b(\check{z}(1-\mathcal{G}^*(\epsilon)))\big]\Bigg]$$

$$R_e(\check{z}) = \frac{Nr_e(\check{z})}{Dr(\check{z})} \tag{3.27}$$

$$\begin{aligned} Nr_e(\check{z}) = & \Lambda_0 \Bigg\{ \Bigg\{ \Big[\eta p + \frac{\epsilon}{\eta}[\eta\mathcal{L}(\check{z})]\Big] \Bigg[\check{z}^R - [s\check{z}+\bar{s}]\mathcal{Q}_b^*(C_b(\check{z}))\big[\mathcal{G}^*(\epsilon) + b(\check{z}(1-\mathcal{G}^*(\epsilon)))\big]\Bigg] + \frac{[1-\mathcal{G}^*(\epsilon)]}{\epsilon} \\ & \Bigg[(s\check{z}+\bar{s})\mathcal{Q}_b^*(C_b(\check{z}))\big[b(\check{z}\eta p + \epsilon\mathcal{L}(\check{z}))\big] + \check{z}^R\big[\epsilon[(s\check{z}+\bar{s})\mathcal{Q}_v^*(C_v(\check{z}))-1]-\eta p\big]\Bigg]\Bigg\}\mathcal{A}(\check{z})C_b(\check{z}) \\ & + \Bigg[\eta p b(\check{z}) + \epsilon\mathcal{L}(\check{z}) + \big[\mathcal{G}^*(\epsilon)+b(\check{z}(1-\mathcal{G}^*(\epsilon)))\big]\big[\epsilon[(s\check{z}+\bar{s})\mathcal{Q}_v^*(C_v(\check{z}))-1]-\eta p\big]\Bigg]\big[1-\mathcal{Q}_b^*(C_b(\check{z}))\big] \\ & \big[\mathcal{A}(\check{z}) + \delta(1-\mathcal{S}^*(\mathcal{A}(\check{z})))\big]\Bigg\} \end{aligned}$$

where Λ_0 is represented by Eq. (3.24)

Proof. To determine the PGF for the no. of products in the system ($K_e(\check{z})$) and in the orbit ($R_e(\check{z})$), the subsequent equations are incorporated.

$$K_e(\check{z}) = \Psi_0 + \Lambda_0 + \Psi(\check{z}) + \check{z}\{\Gamma_b(\check{z}) + \Lambda_v(\check{z}) + \Omega(\check{z})\}$$

and

$$R_e(\check{z}) = \Psi_0 + \Lambda_0 + \Psi(\check{z}) + \Gamma_b(\check{z}) + \Lambda_v(\check{z}) + \Omega(\check{z})$$

Once the Eqs. (3.20) to (3.23) are substituted in the previous findings, the Eqs. (3.26) and (3.27) can potentially be estimated right away. □

4. System performance measures

4.1. System state probabilities

Note that the SS probability when the server is accessible but unoccupied within system is given in Eqs. (3.24) and (3.25). Thus, the probabilities of different server states are estimated from (3.20) to (3.23) which is provided beneath.

1. The SS probability that the server will be idle during the retry period can be represented as Ψ.

$$\Psi = \lim_{\check{z}\to 1}\Psi(\check{z}) = \frac{\Lambda_0[1-\mathcal{G}^*(\epsilon)]\left\{ \begin{aligned} & R[s\mathcal{Q}_v^*(\eta)-1]\epsilon b\mathcal{X}_1(1+\mu^{(1)})\beta_b^{(1)}\big[(1-\mathcal{Q}_v^*(\eta)) + \tfrac{\eta p}{\epsilon}\big] \\ & + \big[(1-\mathcal{Q}_v^*(\eta))[s+\tfrac{\epsilon b\mathcal{X}_1}{\eta}]+s\big] + \tfrac{\eta p}{\epsilon}[s+\mathcal{X}_1] - R \end{aligned}\right\}}{R - \Big[s + \mathcal{X}_1(1-\mathcal{G}^*(\epsilon)) - \Big(\epsilon b\mathcal{X}_1(1+\delta\mu^{(1)}) + \delta\alpha\mu^{(1)}\Big)\beta_b^{(1)}\Big]}$$

2. Let Γ_b represent the SS probability that the server is busy manufacturing

$$\Gamma_b = \lim_{\check{z}\to 1}\Gamma_b(\check{z}) = \frac{\Lambda_0\beta_b^{(1)}\Big\{\big[\epsilon[1-s\mathcal{Q}_v^*(\eta)]\big]\mathcal{X}_1(1-\mathcal{G}^*(\epsilon)) + \epsilon\big[\frac{\epsilon b\mathcal{X}_1(1-\mathcal{Q}_v^*(\eta))}{\eta}+s\big] + \eta p\mathcal{X}_1\mathcal{G}^*(\epsilon)\Big\}}{R - \Big[s + \mathcal{X}_1(1-\mathcal{G}^*(\epsilon)) - \Big(\epsilon b\mathcal{X}_1(1+\delta\mu^{(1)}) + \delta\alpha\mu^{(1)}\Big)\beta_b^{(1)}\Big]}$$

3. Let Λ_v represent the SS probability when the server is experiencing BWV

$$\Lambda_v = \lim_{\check{z}\to 1}\Lambda_v(\check{z}) = \frac{\Lambda_0\epsilon[1-\mathcal{Q}_v^*(\eta)]}{\eta}$$

Table 1
The effect of BWV (*eta*) on L_q, W_q, Ω, Γ_b, Ψ.

BWV rate (η)	L_q	W_q	Ω	Γ_b	Ψ
3.3	0.6365	3.1827	0.0633	0.0032	0.0134
3.5	0.5495	2.7475	0.0977	0.0049	0.0132
3.7	0.4917	2.4587	0.1359	0.0068	0.0128
3.9	0.4520	2.2599	0.1779	0.0089	0.0124
4.1	0.4243	2.1215	0.2237	0.0112	0.0119

4. Let Ω represent the SS probability that the server is under repair

$$\Omega = \lim_{\check{z}\to 1} \Omega(\check{z}) = \frac{\Lambda_0 \delta \beta_b^{(1)} \mu^{(1)} \left\{ \left[\epsilon[1 - s\mathcal{Q}_v^*(\eta)]\right]\mathcal{X}_1(1-\mathcal{G}^*(\epsilon)) + \epsilon\left[\frac{\epsilon b \mathcal{X}_1(1-\mathcal{Q}_v^*(\eta))}{\eta} + s\right] + \eta p \mathcal{X}_1 \mathcal{G}^*(\epsilon) \right\}}{R - \left[s + \mathcal{X}_1(1-\mathcal{G}^*(\epsilon)) - \left(\epsilon b \mathcal{X}_1(1+\delta\mu^{(1)}) + \delta\alpha\mu^{(1)}\right)\beta_b^{(1)}\right]}$$

4.2. Average size of the proposed system and its orbit

Upon entering an SS, the system

(i) The average amount of products in the system (L_s) is computed by differentiating (3.26) with reference to $\check{z}$ and yielding $\check{z} = 1$ considering stability conditions.

$$L_s = K_e'(1) = \lim_{\check{z}\to 1} \frac{d}{d\check{z}} K_e(\check{z}) = \Lambda_0 \left[\frac{Nr_s''''(1)Dr_q'''(1) - Dr_q''''(1)Nr_q'''(1)}{4(Dr_q'''(1))^2}\right] \tag{4.28}$$

(ii) The average amount of products in orbit (L_q) can be estimated by differentiating (3.27) with respect to $\check{z}$ and yielding $\check{z} = 1$ considering stability conditions.

$$L_q = R_e'(1) = \lim_{\check{z}\to 1} \frac{d}{d\check{z}} R_e(\check{z}) = \Lambda_0 \left[\frac{Nr_q''''(1)Dr_q'''(1) - Dr_q''''(1)Nr_q'''(1)}{4(Dr_q'''(1))^2}\right] \tag{4.29}$$

(iii) The length of time a user can anticipate spending in the system (W_s) and in the queue (W_q) can both be predicted with the assist of Little's law. Therefore, we have $W_s = \frac{L_s}{\epsilon \mathcal{X}_1}$ and $W_q = \frac{L_q}{\epsilon \mathcal{X}_1}$.

5. Numerical examples

We will leverage MATLAB in this part to illustrate various possibilities for the dynamic adaptability of the system. To ascertain whether numerical measurements satisfy the stability requirements, a random selection process is employed. Tables 1 through 3 exhibit the implications of a few system parameters.

Table 1 displays that as the BWV rate (η) rises, Ω and Γ_b also mounts, whereas L_q, W_q and Ψ declines for the value of $\epsilon = 1$, $b = 0.1$, $s_1 = 0.1$, $\alpha = 6$, $p = 0.2$, $s = 0.1$, $\delta = 7$, $v = 0.1$, $v_v = 5$ and $\theta = 7$. The increase in the service (Γ_b) and repair rate (Ω) suggests that the BPS can process batches more quickly and efficiently. This leads to smoother production flow and reduced waiting times for jobs in the queue which leads to optimized production schedules and higher throughput. Further, the decrease in L_q and W_q indicates that jobs spend less time waiting for processing. Though the facility runs at a slower rate, it experiences reduced downtime as batches move through the production process more quickly and efficiently. In addition, the decrease in retrial rate Ψ suggests improved system stability.

Table 2 displays that as the stand-by (α) mounts, Γ_b and Ω also mount; on the other hand, Ψ_0, W_s and W_q reduce for the value of $\epsilon = 1$, $\eta = 4.1$, $b = 0.1$, $s_1 = 0.1$, $p = 0.2$, $s = 0.1$, $\delta = 7$, $v = 0.1$, $v_b = 2$ and $\theta = 7$. The decrease in W_s and W_q suggests that jobs spend less time in the queue and waiting for service since the stand-by server plays a vital role during main server breakdown. This leads to faster batch processing, reducing overall lead times, and improving production efficiency. Then, Ψ_0 decrease indicates that the server is more consistently utilized. With fewer idle periods, resources are used more efficiently, resulting in reduced production bottlenecks. Further, the service (Γ_b) and repair rate (Ω) always rise, which suggests improved reliability and availability of production resources leading to more consistent production output.

Table 3 displays that as repair rate (v) on mounts, L_s, L_q, Ψ_0, Λ_0 and Ψ declines for the value of $\epsilon = 1$, $\eta = 4.1$, $\alpha = 6$, $b = 0.1$, $s_1 = 0.1$, $p = 0.2$, $s = 0.1$, $\delta = 7$, $v_b = 2$ and $\theta = 7$. Here, the decrease in L_s and L_q, suggests that jobs spend less time waiting for processing since the server gets repaired in a short period of time, and moreover, there is also the availability of a standby server. This leads to smoother production flow. Since, the server is being utilized continuously, the probability that the server is inactive during normal service (Ψ_0) and working vacation periods (Λ_0) reduces. Thus, with fewer idle periods, production resources are utilized well. The decrease in the retrial rate (Ψ) suggests improved system stability, as fewer jobs are re-entering the system for processing. This indicates that the facility can handle variations in workload more effectively.

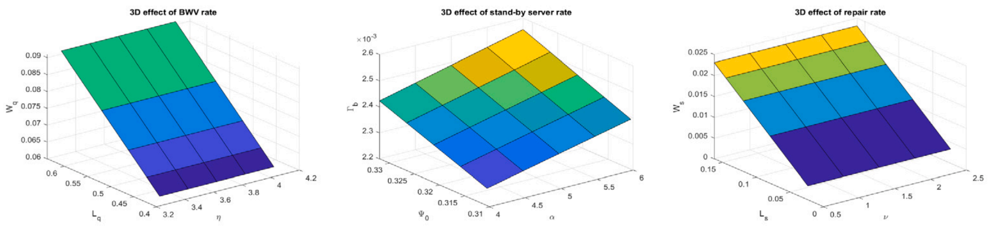
Fig. 1(a) address the impact of a few system parameters via $3D$ graphs. Fig. 1(a) depicts that as BWV rate (η) mounts, L_q and W_q diminish. Here, the reduction in queue length and waiting time indicates that jobs are processed more swiftly and spend less time waiting in the system. This enhances production efficiency by minimizing delays and reducing the overall time of production process. Fig. 1(b) addresses that as stand-by server rate (α) rises, Γ_b elevates; however, Ψ_0 declines. The increase in the service rate and the decrease in server inactivity suggest better utilization of the production servers. This ensures that the servers are more frequently in use, reducing idle times and maximizing productivity. With a higher stand-by server rate, the production system can handle higher loads, thereby improving the overall reliability and robustness of the process. In Fig. 1(c) as the repair rate (v) mounts, L_s and L_q decline. A higher repair rate means that equipment and machinery are repaired and brought back into operation more quickly which reduces downtime, leading to a smoother production with fewer interruptions. The reduction in queue length and system size indicates that jobs move through the production system more efficiently, thus improving the overall flow within the production facility.

Table 2
The effect of stand-by server rate (α) on W_s, W_q, Ψ_0, Γ_b.

Stand-by server rate (α)	W_s	W_q	Ψ_0	Γ_b	Ω
4.0	0.4697	0.9579	0.3319	0.1591	0.1114
4.5	0.3382	0.5560	0.3265	0.1597	0.1118
5.0	0.2758	0.4157	0.3211	0.1602	0.1122
5.5	0.2393	0.3442	0.3158	0.1608	0.1126
6.0	0.2152	0.3006	0.3105	0.1614	0.1130

Table 3
The impact of repair rate (ν) on L_s, L_q, Ψ_0, Λ_0, Ψ.

Repair rate (ν)	L_s	L_q	Ψ_0	Λ_0	Ψ
0.5	0.1618	0.1883	0.7842	0.9564	1.3652
1.0	0.1446	0.1729	0.1839	0.2243	0.4228
1.5	0.1188	0.1469	0.0992	0.1209	0.2553
2.0	0.0793	0.1018	0.0670	0.0817	0.1835
2.5	0.0235	0.0309	0.0504	0.0614	0.1434



(a) L_q, W_q verses BWV η (b) Ψ_0, Γ_b verses stand-by server rate α (c) L_s, W_s verses repair rate ν

Fig. 1. The effect of various system parameters depicted via 3D graphs.

6. Cost analysis

Cost analysis generally involves evaluating the expenses associated with a project or operation to ensure optimal resource allocation and cost efficiency. This approach is employed here to estimate the optimal parameters for the service rates, of the system's normal (τ_b) and slow service mode (τ_v). The expected cost function is presumed to exhibit a linear cost structure in accordance with cost elements associated to various system functions.

The subsequent cost components have been entailed in the expected total cost function TC (τ_b, τ_v,) for per unit of time (PUT):

$\mathcal{M}_h$	-	Each consumer's holding cost PUT spent within the system
$\mathcal{M}_b$	-	Cost PUT when the server is normally active
$\mathcal{M}_v$	-	Cost PUT when the server is in BWV
$\mathcal{M}_f$	-	Cost PUT when the server is under repair
$\mathcal{M}_1$	-	Cost per consumer served during server's normal service mode
$\mathcal{M}_2$	-	Cost per consumer served during server's lower service mode

$$TC = \mathcal{M}_h L_q + \mathcal{M}_b \Gamma_b + \mathcal{M}_v \Lambda_v + \mathcal{M}_f \Omega + \mathcal{M}_1 \tau_b + \mathcal{M}_2 \tau_v \tag{6.30}$$

The significant non-linearity of the cost function outlined by (6.30) makes analytical optimization of it challenging. Therefore, we employ a heuristic approach to optimize overall cost.

6.1. Cost optimization

Cost optimization focuses on strategically minimizing expenses while maintaining or enhancing value delivery, aiming for the most efficient allocation of resources to achieve desired outcomes. It involves continuous refinement of processes, technologies, and strategies to achieve maximum cost-effectiveness. The main objective of this chapter is to find the ideal service rates τ_b^* and τ_v^* for the busy state of the server and WV mode, accordingly, by minimizing the TC function. Mathematically, the cost-minimization problem is stated as

$$TC(\tau_b^*, \tau_v^*) = \underset{\tau_b^*, \tau_v^*}{Minimize} TC(\tau_b, \tau_v)$$

The ranges of the cost elements are listed in Table 4.

6.2. Artificial Bee Colony Optimization (ABC)

The swarm-based ABC strategy, which was introduced by Karaboga and Basturk (2007), is a method for solving optimization problems that mimics the astute behaviour of honey bees when they are foraging. It optimizes solutions by employing three types of bees: employed, onlooker, and scout bees. It iteratively updates candidate solutions to converge towards the optimal cost. Algorithm 1 describes the steps of the ABC technique.

Table 4
Different cost sets for cost analysis.

Cost sets	$\mathcal{M}_h$	$\mathcal{M}_b$	$\mathcal{M}_v$	$\mathcal{M}_f$	$\mathcal{M}_1$	$\mathcal{M}_2$
Set 1	15	95	20	45	15	30
Set 2	13	100	15	55	10	20
Set 3	17	90	25	65	12	35

Algorithm 1 Pseudo Code for ABC Algorithm

```
INPUT: Input function =$TC(\tau_b, \tau_v)$,
OUTPUT: $TC(\tau_b^*, \tau_v^*)$ denotes the value of the proposed cost function
Initialize the population of solutions (food sources)
Evaluate the fitness of each solution
Repeat until stopping criterion is met:
for each employed bee do
    Select a solution to modify
    Generate a new solution by modifying the selected solution
    Evaluate the fitness of the new solution
    If the new solution is better, replace the old solution with the new one
end for
for each onlooker bee do
    Select a solution based on the fitness probability
    Generate a new solution by modifying the selected solution
    Evaluate the fitness of the new solution
    If the new solution is better, replace the old solution with the new one
end for
for each solution do
    If a solution has not improved for a certain number of iterations, replace it with a new random solution
end for
Memorize the best solution found so far
Return the best solution
```

6.3. *Differential Evolution (DE)*

DE is an optimization algorithm for tackling non-linear optimization problems using a stochastic population-based approach. It optimizes a given issue by searching for better solutions until one is identified that meets certain specified criteria. It iteratively improves candidate solutions by generating trial vectors from the mutation and crossover of existing solutions. Through this process, DE explores the solution space efficiently. Storn and Price (1996) introduced the algorithm to the world. Algorithm 2 describes each stage of the DE method.

Algorithm 2 Pseudo Code for DE Algorithm

```
INPUT: Input function = $TC(\tau_b, \tau_v)$, no. of iterations, crossover, mutation
OUTPUT: $TC(\tau_b^*, \tau_v^*)$ denotes the value of the proposed cost function
Initiate population $\mathcal{W}$
for each member i in the population $\mathcal{W}$ do
    Select three candidates $b_1$, $b_2$ and $b_3$ such that, $1 \geq b_1, b_2, b_3 \leq$ N
    Generate a arbitrary integer i $\in$ (1,N)
    while iteration < MaxIteration do
        for each factor i do
            Estimate the fitness of every member
            Generate mutant vectors utilizing mutation approach
            Generate trial vectors using recombining noisy vectors with parent vectors
            Estimate trial vectors with their fitness values
        end for
        Choose winning vectors as members in the new generation
        iteration ++
    end while
end for
Return the best values
```

6.4. *Genetic Algorithm (GA)*

Bremermann, Holland (Bremermann, 1958; Holland, 1975), and associates invented the GA in the 1960s and 1970s. It evolves a population of candidate solutions through selection, crossover, and mutation operations. By mimicking natural selection, GA efficiently explores the solution space, gradually improving solutions over multiple generations to find the optimal cost. Recently, the usage of a GA is outlined entirely in Arqub's (Arqub and Al-Smadi, 2020) study. The pseudocode for the GA algorithm's steps is presented by Algorithm 3.

During the 1960s and 1970s, Bremermann, Holland (Bremermann, 1958; Holland, 1975) and their colleagues developed the GA.

Algorithm 3 Pseudo Code for GA Algorithm

```
INPUT: Input function =$TC(\tau_b, \tau_v)$,
OUTPUT: $TC(\tau_b^*, \tau_v^*)$ denotes the value of the cost function
Initialize population with random individuals
Evaluate fitness of each individual in the population
while termination condition not met do
    Select parents from the population based on fitness
    Perform crossover on parents to create offspring
    Perform mutation on offspring
    Evaluate fitness of offspring
    Select individuals to form the new population
end while
Return the best individual from the population
```

6.5. *Grey Wolf Optimizer (GWO)*

One of the more prominent recent swarm intelligence approach is called GWO inspired by the social hierarchy and hunting behaviour of grey wolves introduced by Mirjalili et al. (2014). It optimizes solutions by simulating the hunting process of alpha, beta, and delta wolves. Through collaboration and competition, GWO efficiently explores and exploits the solution space to find the optimal cost. Algorithm 4 describes the steps of the GWO approach.

Algorithm 4 Pseudo Code for GWO Algorithm

```
INPUT: Input function =$TC(\tau_b, \tau_v), a, B, D$
OUTPUT: $TC(\tau_b^*, \tau_v^*)$ denotes the value of the proposed cost function
Initiate the grey wolf population $\mathcal{Y}_i$, i= 1,n
Initiate a, B, D
Estimate the fitness of every search factor
$\mathcal{Y}_\alpha$ = the best search factor
$\mathcal{Y}_\beta$ = the second best search factor
$\mathcal{Y}_\delta$ = the third best search factor
while t < maximum no. of iteration do
    for every search factor do
        Arbitrarily initiate $c_1$ and $c_2$
        Update the place of the present search factor
        Update a, B and D
    end for
    Estimate the fitness of every search factors
    Update $\mathcal{Y}_\alpha$, $\mathcal{Y}_\beta$ and $\mathcal{Y}_\delta$
    t=t+1
end while
return $\mathcal{Y}_\alpha$
```

6.6. *Particle Swarm Optimization (PSO)*

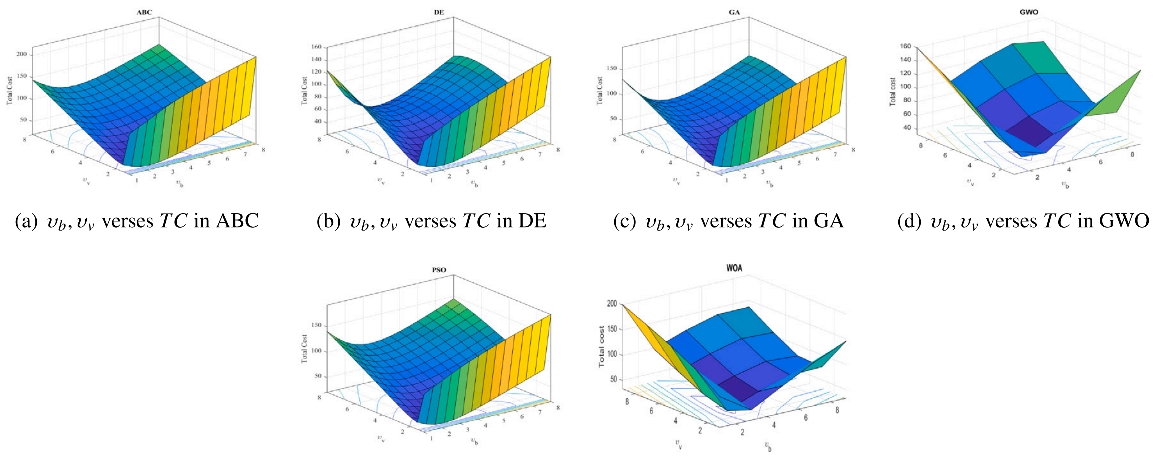
In 1995, Kennedy and Eberhart (1995) introduced the world to PSO, modelled after the social behaviour of bird flocks or fish schools. It iteratively updates candidate solutions based on the movement of particles towards the best solution found so far. PSO leverages swarm intelligence to find the optimal cost through collective movement. Upadhyaya (2020) and Malik et al. (2021) elaborated on this method. The PSO method's stages are explained in Algorithm 5.

6.7. *Whale Optimization Algorithm (WOA)*

An optimization algorithm that takes its cues the bubble-net hunting tactic employed by humpback whales. In the context of continuous optimization problems, Mirjalili and Lewis (2016) presented this swarm intelligence approach. It optimizes solutions through a combination of exploration and exploitation phases. By leveraging the movement of prey and the leader-following behaviour of whales, WOA efficiently explores the solution space to find the optimal cost. Moreover, recently Gharehchopogh and Gholizadeh (2019) gave a comprehensive survey on WOA and its applications. Algorithm 6 provides the pseudocode for the WOA algorithm's order of procedures.

Algorithm 5 Pseudo Code for PSO Algorithm

```
INPUT: Input function =$TC(\tau_b, \tau_v)$,
OUTPUT: $TC(\tau_b^*, \tau_v^*)$ represents the value of the proposed cost function
Initialize population of particles with random positions and velocities
Initialize global best position and fitness to infinity
while termination criteria not met do
    for each particle do
        Update velocity based on current position, personal best, and global best
        Update position based on velocity
        Evaluate fitness of current position
        if current position is better than personal best then
            Update personal best position and fitness
        else if current position is better than global best then
            Update global best position and fitness
        end if
    end for
end while
Return global best position and fitness
```



(a) υ_b, υ_v verses TC in ABC (b) υ_b, υ_v verses TC in DE (c) υ_b, υ_v verses TC in GA (d) υ_b, υ_v verses TC in GWO

(e) υ_b, υ_v verses TC in PSO (f) υ_b, υ_v verses TC in WOA

Fig. 2. Optimality of the cost function.

6.8. Convergence in ABC, DE, GA, GWO, PSO and WOA

While implementing optimization techniques such as ABC, DE, GA, GWO, PSO, or WOA, it is essential to understand whether a particle strays and how it navigates to achieve better results. In light of this, examining the convergence of costs is essential in addressing these issues. The cost function's convergence is depicted in Fig. 1. In ABC, DE, GA, GWO, PSO, and WOA, it is apparent that particles converge towards the best solution after a series of iterations. In all six ways, particles converge to the lowest possible cost after a few iterations. Nonetheless, the cost analysis reveals that GWO reaches convergence more quickly. Furthermore, Fig. 2 illustrates the cost function's optimality. This allows experts to look at the system as a whole, which will lessen their financial constraints (see Fig. 3).

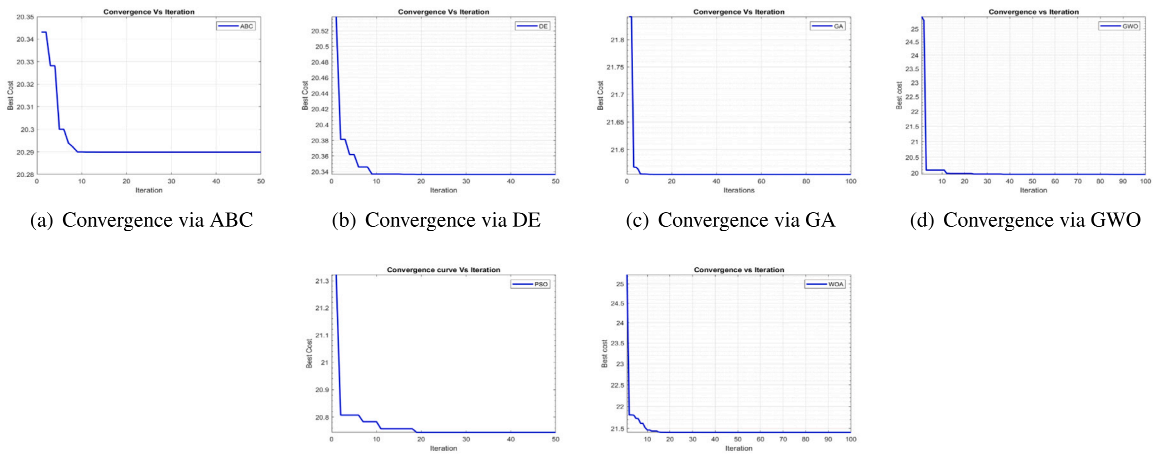
There are many methods beyond ABC, GA, DE, PSO, WOA, and GWO that can be utilized for cost optimization. Recent algorithms such as the Genghis Khan Shark optimizer, geyser-inspired algorithm, prairie dog optimization, dwarf mongoose optimization, Gazelle optimization, lungs performance-based optimization, multi-objective snow ablation optimization and Sinh cosh optimizer provide unique strategies for tackling complex optimization problems, highlighting the diverse range of tools available for optimizing costs in various systems. These sources could enrich the analysis and provide valuable comparisons with existing approaches, enhancing the comprehensiveness and relevance of the research.

7. Discussion

- Research gaps addressed:
  - Notably, no prior research has specifically investigated quorum retrial queues in the context of BPS.
  - The unique combination of batch arrivals and batch service under BWV conditions along with a standby server has not been explored.
  - This study pioneers the use of SVT to analyse such systems, filling a critical gap in the literature.
  - This research also breaks new ground by undertaking cost optimization using six different approaches. Prior studies have not explored cost optimization to this extent.

Algorithm 6 Pseudo Code for WOA Algorithm

```
INPUT: Input function =$TC(\tau_b, \tau_v)$,
OUTPUT: $TC(\tau_b^*, \tau_v^*)$ denotes the value of the proposed cost function
Initialize a new population of n whales $\mathcal{N}_i(1 = 1, 2, \dots n)$
Estimate the fitness function value
Arbitrarily choose the search agent $\mathcal{N}^*$
while b=1 and b< maximum iterations do
    for every search agent do
        Update c, C, D, m and q
        if q <0.5 then
            if |C|<1 then
                Enhance the existing search agent's position
            else if |C|≥1 then
                Again enhance the existing search agent's position
            end if
        else if q≥0.5 then
            Again enhance the existing search position
        end if
    end for
    Ensure that no search agent goes outside the search area or else fix it.
    Estimate the fitness function value
    Update $\mathcal{N}^*$ if it is better
    b=b+1
end while
return $\mathcal{N}^*$
```



(a) Convergence via ABC (b) Convergence via DE (c) Convergence via GA (d) Convergence via GWO

(e) Convergence via PSO (f) Convergence via WOA

Fig. 3. Convergence of the cost function.

- Authors Contribution:
  - The authors have made a significant contribution by modelling the batch production system as a novel quorum RQ. This innovative approach provides a new perspective on handling batch production processes.
  - By incorporating SVT to conduct a thorough analysis of the system, they were able to delve deeply into the system's behaviour and performance, offering precise insights into its dynamics.
  - The authors performed cost optimization using six different techniques. This extensive approach to cost analysis ensures that the proposed model is not only theoretically robust but also practically viable, providing substantial economic benefits to BPS.
- Applicability to different types of manufacturing environments
  - Ideal for environments where customization or seasonal production is critical, enabling efficient handling of varying product demands.
  - Applicable to any manufacturing environment looking to reduce costs through optimized scheduling, resource allocation, and advanced optimization techniques.
  - The findings can be scaled and customized to suit both small-scale and large-scale production facilities, ensuring broad applicability across different manufacturing contexts.

Therefore, this study significantly advances the understanding of BPS by addressing previously unexplored aspects

8. Conclusion

This research investigates a batch production system by utilizing a unique bulk quorum RQ. We have employed the SVT and generating function method to derive analytical formulas for several important performance indices. A wide of optimization approaches is also incorporated in the cost estimation process. By implementing the proposed strategies, firms can enhance production efficiency and reduce costs through optimized scheduling and resource allocation thereby having a significant impact on manufacturing firms. The use of advanced optimization techniques ensures robust and reliable production processes, minimizing downtime and improving overall productivity. Additionally, the insights gained from analysing system behaviour with the SVT enable better decision-making and strategic planning, leading to more efficient operations and cost-effective production. Overall, this research equips manufacturing firms with the tools and insights needed to optimize their batch production processes, ultimately leading to increased productivity, reduced costs, and improved competitiveness in the marketplace.

Future research could expand on this study by examining it in the context of other types of vacations, such as hybrid vacations, coxian vacations, and also with heterogeneous services. Moreover, these systems were not studied using the matrix geometric technique, which is an interesting direction for further research. A further intriguing direction for future research is the machine learning techniques, such as reinforcement learning and deep learning, to improve analysis and optimization accuracy. Additionally, exploring hybrid optimization algorithms could provide more robust and adaptable solutions for complex manufacturing environments.

Funding

We would like to express our sincere gratitude to Vellore Institute of Technology, Vellore, India for their generous financial support, which made this research possible. Thanking again for their unwavering support and encouragement.

CRediT authorship contribution statement

Harini R.: Conceptualization, Methodology, Writing – original draft. **Indhira K.:** Investigation, Supervision, Validation, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Arqub, OA., Al-Smadi, M., 2020. Fuzzy conformable fractional differential equations: novel extended approach and new numerical solutions. Soft Comput. 24 (16), 12501–12522. http://dx.doi.org/10.1007/s00500-020-04687-0.

Ayyappan, G., Thamizhselvi, P., 2018. Transient analysis of $m^X{}_1$, $m^X{}_2/g_1$, $g_2/1$ retrial queueing system with priority services, working vacations and vacation interruption, emergency vacation, negative arrival and delayed repair. Int. J. Appl. Comput. Math. 4, 1–35. http://dx.doi.org/10.1007/s40819-018-0509-7.

Bagyam, JE., Chandrika, KU., Rani, KP., 2013. Bulk arrival two phase retrial queueing system with impatient customers, orbital search, active breakdowns and delayed repair. Int. J. Comput. Appl. 73 (11).

Bharathidass, S., Arivukkarasu, V., Ganesan, V., 2018. Bulk service queue with server breakdown and repairs. Int. J. Stat. Appl. Math. 3 (1), 136–142.

Bremermann, HJ., 1958. The Evolution of Intelligence: The Nervous System as a Model of its Environment. University of Washington, Department of Mathematics.

Chaudhry, ML., Templeton, JG., 1983. A first course in bulk queues. (No Title).

Das, RR., Devi, VN., Rathore, A., Chandan, K., 2022. Analysis of Markovian queueing system with server failures, N-policy and second optional service. Int. J. Nonlinear Anal. Appl. 13 (1), 3073–3083. http://dx.doi.org/10.22075/IJNAA.2022.6048.

Deora, P., Kumari, U., Sharma, DC., 2021. Cost analysis and optimization of machine repair model with working vacation and feedback–policy. Int. J. Appl. Comput. Math. 7, 1–4. http://dx.doi.org/10.1007/s40819-021-01185-1.

Gharehchopogh, FS., Gholizadeh, H., 2019. A comprehensive survey: Whale Optimization Algorithm and its applications. Swarm Evol. Comput. 48, 1–24. http://dx.doi.org/10.1016/j.swevo.2019.03.004.

Holland, J., 1975. Adaptation in Natural and Artificial Systems. vol. 7, univ. of mich. Press, Ann Arbor, pp. 390–401.

Jain, M., Kumar, A., 2022. Unreliable server M [X]/G/1 queue with working vacation and multi-phase repair with delay in verification. Int. J. Appl. Comput. Math. 8 (4), 210. http://dx.doi.org/10.1007/s40819-022-01383-5.

Karaboga, D., Basturk, B., 2007. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J. Glob. Optim. 39, 459–471. http://dx.doi.org/10.1007/s10898-007-9149-x.

Kashifalghitaa, HH., Al-Obaidi, AH., 2023. Some arrival cases in quorum queues under N-policy. Util. Math. 120, 437–454.

Kennedy, J., Eberhart, R., 1995. Particle swarm optimization. In: Proceedings of ICNN'95-International Conference on Neural Networks. Vol. 4, IEEE, pp. 1942–1948.

Kumar, A., Jain, M., 2023. Cost Optimization of an Unreliable server queue with two stage service process under hybrid vacation policy. Math. Comput. Simul. 204, 259–281. http://dx.doi.org/10.1016/j.matcom.2022.08.007.

Kumar, BK., Madheswari, SP., 2003. M x/G/1 retrial queue with multiple vacations and starting failures. Opsearch 40, 115–137. http://dx.doi.org/10.1007/BF03398688.

Laxmi, PV., Qrewi, HA., George, AA., 2023. Analysis of batch arrival general service queue with balking, feedback and second optional service. Contemp. Math. 15, 1109–1124. http://dx.doi.org/10.37256/cm.4420232688.

Malik, G., Upadhyaya, S., Sharma, R., 2021. Cost inspection of a Geo/G/1 retrial model using particle swarm optimization and Genetic algorithm. Ain Shams Eng. J. 12 (2), 2241–2254. http://dx.doi.org/10.1016/j.asej.2020.11.012.

Mirjalili, S., Lewis, A., 2016. The whale optimization algorithm. Adv. Eng. Softw. 95, 51–67. http://dx.doi.org/10.1016/j.advengsoft.2016.01.008.

Mirjalili, Seyedali, Mirjalili, Seyed Mohammad, Lewis, Andrew, 2014. Grey wolf optimizer. Adv. Eng. Softw. 69, 46–61. http://dx.doi.org/10.1016/j.advengsoft.2013.12.007.

Nazarov, AA., Rozhkova, SV., Titarenko, EY., 2023. Asymptotic diffusion analysis of the retrial queuing system with feedback and batch Poisson arrival. Discrete Contin. Models Appl. Comput. Sci. 31 (3), 205–217.

Rajadurai, P., Chandrasekaran, VM., Saravanarajan, MC., 2016. Analysis of an M [X]/G/1 unreliable retrial G-queue with orbital search and feedback under Bernoulli vacation schedule. Opsearch 53, 197–223. http://dx.doi.org/10.1007/s12597-015-0226-5.

Servi, LD., Finn, SG., 2002. M/m/1 queues with working vacations (m/m/1/wv). Perform. Eval. 50 (1), 41–52. http://dx.doi.org/10.1016/S0166-5316(02)00057-3.

Storn, R., Price, K., 1996. Minimizing the real functions of the ICEC'96 contest by differential evolution. In: Proceedings of IEEE International Conference on Evolutionary Computation. IEEE, pp. 842–844. http://dx.doi.org/10.1109/ICEC.1996.542711.

Tadj, L., Choudhury, G., Rekab, K., 2012. A two-phase quorum queueing system with Bernoulli vacation schedule, setup, and N-policy for an unreliable server with delaying repair. Int. J. Serv. Oper. Manag. 12 (2), 139–164. http://dx.doi.org/10.1504/IJSOM.2012.047103.

Tadj, L., Ke, JC., 2008. A hysteretic bulk quorum queue with a choice of service and optional re-service. Qual. Technol. Quant. Manage. 5 (2), 161–178. http://dx.doi.org/10.1080/16843703.2008.11673394.

Upadhyaya, S., 2020. Cost optimisation of a discrete-time retrial queue with Bernoulli feedback and starting failure. Int. J. Ind. Syst. Eng. 36 (2), 165–196. http://dx.doi.org/10.1504/IJISE.2020.110245.

Vaishnawi, M., Upadhyaya, S., Kulshrestha, R., 2022. Optimal cost analysis for discrete-time recurrent queue with Bernoulli feedback and emergency vacation. Int. J. Appl. Comput. Math. 8 (5), 254. http://dx.doi.org/10.1007/s40819-022-01445-8.

Yuvarani, S., Saravanarajan, MC., 2017. An analysis on Mx/G/1 preemptive priority retrial queue with delayed repairs under modified Bernoulli vacation schedule. Int. J. Pure Appl. Math. 117 (12), 73–82.