Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 31 No. 7s (2024)

Efficient Bulk Arrival Management in Multi-Service Retrail Queues with Orbital search Leveraging Rabbit MQ

Vivesini.D[*,1], Poongodi.T[2], Ebenesar Anna Bagyam[3]

[*,1] Department of Mathematics, Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, Tamil Nadu - 601 043, viveshpriya@gmail.com@gmail.com

[2]Department of Mathematics, Avinashilingam Institute for Home Science and Higher Education for Women, Coimbatore, Tamil Nadu - 601 043, poongodi.math@gmail.com

[3]Department of Mathematics, Karpagam Academy of Higher Education,Coimbatore, Tamilnadu- 641004, ebenesar.j@gmail.com

Article History:

Received: *04-06-2024*

Revised: *03-07-2024*

Accepted: *30-07-2024*

Abstract

This research paper introduces a novel approach to managing bulk arrivals in a two-phase multi-service retrial queue system, leveraging the efficiency of the Rabbit MQ architecture. The proposed system integrates the concept of orbital search to optimize the queue's performance and resource allocation. Unlike conventional queueing systems, this approach effectively handles large influxes of requests and ensures optimal service delivery across multiple services. In this innovative system, the Rabbit MQ messaging framework is employed as the backbone, providing robust message queuing capabilities. The two-phase retrial mechanism facilitates the efficient handling of failed requests, minimizing resource wastage and maximizing system throughput. Furthermore, the incorporation of orbital search techniques enhances the system's ability to prioritize and route incoming requests, reducing queueing delays and optimizing resource allocation.

This research contributes to the field of queueing theory and system optimization by presenting a comprehensive solution for bulk arrival scenarios. The combination of two-phase retrial and orbital search techniques within the RabbitMQ architecture offers a scalable and efficient approach to managing multi-service queues, making it particularly valuable for modern applications with dynamic workloads and stringent performance requirements.

Keywords : Two-phase queue, multi-service retrial, orbital search, Rabbit MQ, Particle swarm optimization algorithm.

1. Introduction

In today's dynamic world of information technology and communication networks, the efficient management of incoming requests and services is of paramount importance. To address this challenge, we present an innovative system: the Bulk Arrival Two-Phase Multi-Service Retrial Queue with Orbital Search, powered by RabbitMQ.

This cutting-edge system combines several key elements to optimize the handling of incoming requests and services. Firstly, the "Bulk Arrival" feature enables the system to process a large influx of requests efficiently. Secondly, the "Two-Phase" mechanism divides the processing into distinct stages, enhancing overall performance and response times. Moreover, the "Multi-Service" capability allows the system to cater to a variety of service types simultaneously, making it adaptable to diverse

business needs. To ensure reliability and scalability, the system harnesses Rabbit MQ, a robust message broker, for seamless communication and task distribution.

One of the standout features of this system is the "Orbital Search" algorithm, which intelligently locates and prioritizes pending tasks for optimal resource allocation. This advanced search mechanism significantly improves the efficiency of service provisioning, reducing latency and enhancing customer satisfaction.

we will delve deeper into the components, functionalities, and benefits of the Bulk Arrival Two-Phase Multi-Service Retrial Queue with Orbital Search Using Rabbit MQ, exploring how it revolutionizes the way we manage and process service requests in today's fast-paced digital landscape.

Sharvari and Sowmiya Nag (2019) are analysed the critical role of distributed messaging systems in big data streaming, cloud-native applications, and micro-services. It highlights the growing demand for scalable, fault-tolerant, and low-latency messaging platforms in real-time critical applications. The review acknowledges the proliferation of modern messaging systems, leading to industry challenges in selecting the most suitable system for specific applications. It focuses on Apache Kafka, Rabbit MQ, and NATS Streaming, providing insights into their features, use cases, and differences, aiming to aid informed decision-making and inspire future research and development in this domain

Jarvela and Lindmark (2019) are analysed the implementation of a scalable and highly available Rabbit MQ cluster on AWS and Azure. It also examines an alternative AWS solution via Cloud Formation. The study assesses performance using Rabbit MQ PerfTest, analyzing throughput and price-performance ratios across different cloud instances. The research delves into how performance varies between instance families and cloud platforms, offering insights into the broader infrastructure performance disparities between AWS and Microsoft Azure.

Johansson (2018) was analysed the pressing need for today's applications to continually adapt and adjust in response to fluctuating demand, driven by the ever-increasing volume of data. Selecting the appropriate communication method is paramount in this dynamic environment, as a solution that may have worked effectively for an extended period can suddenly become a bottleneck, leading to inefficiencies. Alternatively, specialized communication systems have been developed explicitly for efficiently distributing messages to available parties. The implementation of a message-oriented middleware, such as Rabbit MQ and Active MQ, offers asynchronous communication, fostering loose coupling among applications. Consequently, this approach optimizes resource utilization and enhances overall system performance.

Hedadli et al.(2023) have analysed the M/M/1 queuing system with service interruptions, where customers take vacations and join an orbit. The analysis considers a scenario where the server, upon becoming free, can search for customers in the orbit. Access to the server can occur from the queue by primary customers or from the orbit by secondary customers who have passed through the server at least once. This situation involves either repeated customer attempts until server availability or immediate server searches after completing each service if the queue is empty. To tackle the model's complexity, the study employs the matrix analytic method, providing approximate limiting

probabilities. Various performance measures are computed, and these findings are reinforced through numerical examples and simulations, shedding light on how certain parameters affect system characteristics.

Nila and Sumitha (2019) are analysed a comprehensive analysis is conducted on a single server batch arrival retrial queueing model. This model takes into account several critical factors, including the presence of starting failures, customer impatience, and a multi-optional second service phase. The server's service process is divided into two phases: an essential first phase and a multi-optional second phase. After completing each service, the server actively searches for customers in the orbit, if any exist, or remains idle. The paper successfully derives essential system performance measures for this intricate model, shedding light on its operational efficiency and effectiveness.

Sangeetha and Chandrika (2022) are analysed sophisticated Single Server Batch Arrival Retrial G-Queue with Orbital Search model, where positive customers arrive in batches and negative customers arrive singly. It features essential and optional service phases, each with L sequential stages offering heterogeneous services. Customers can choose between departure, feedback to the orbit, or optional services at each stage. Server vacations occur during system emptiness, and the arrival of a negative customer affects server activity. The study employs the supplementary variable technique to analyse probability functions, determining key metrics like system size, orbit size, server availability, and failure frequency, supported by numerical findings.

Rani et al .(2017) have analysed bulk Arrival General Service Retrial Queue with a server that offers both essential and optional phases of service. They take into account the complexities of customers potentially balking or renege at specific times, along with the consideration of accidental server breakdowns. The repair process is initiated after a random delay period, and upon recovery, the server resumes service for the interrupted customer or waits for their return. The study establishes necessary and sufficient conditions for system stability and employs supplementary variable techniques to derive steady-state distributions for server states and orbit customer counts. Numerical examples are also provided, shedding light on the parameter effects on various performance metrics.

Li et al .(2019) have analysed an M/M/1 retrial queue with working vacation, orbit search, and balking. It employs the matrix-analytic method to establish stability conditions, stationary probability distribution, and performance measures. Furthermore, it introduces conditional stochastic decomposition for queue length in the orbit during server busy times and illustrates the impact of model parameters via numerical examples.

Wichner et al.(2009) have analysed multi-server finite-source retrial queue system, introducing the elements of balking and impatience for customers with limited service facility capacity. Customers arriving can either join a FIFO queue or enter an orbit, and they may abandon the buffer and join the orbit after a random time. Servers search for customers in the orbit when the buffer is empty. All random variables in the model are assumed to be exponentially distributed and independent. The study aims to analyze how balking, impatience, and buffer size impact steady-state performance measures, particularly focusing on mean response time, using the MOSEL-2 tool for modeling and calculating stationary characteristics.Sangeetha and Chandrika (2022) are analysed sophisticated Single Server Batch Arrival Retrial G-Queue with Orbital Search model, where positive customers

arrive in batches and negative customers arrive singly. It features essential and optional service phases, each with L sequential stages offering heterogeneous services. Customers can choose between departure, feedback to the orbit, or optional services at each stage. Server vacations occur during system emptiness, and the arrival of a negative customer affects server activity. The study employs the supplementary variable technique to analyse probability functions, determining key metrics like system size, orbit size, server availability, and failure frequency, supported by numerical findings.

Rani et al .(2017) have analysed bulk Arrival General Service Retrial Queue with a server that offers both essential and optional phases of service. They take into account the complexities of customers potentially balking or renege at specific times, along with the consideration of accidental server breakdowns. The repair process is initiated after a random delay period, and upon recovery, the server resumes service for the interrupted customer or waits for their return. The study establishes necessary and sufficient conditions for system stability and employs supplementary variable techniques to derive steady-state distributions for server states and orbit customer counts. Numerical examples are also provided, shedding light on the parameter effects on various performance metrics.

Li et al .(2019) have analysed an M/M/1 retrial queue with working vacation, orbit search, and balking. It employs the matrix-analytic method to establish stability conditions, stationary probability distribution, and performance measures. Furthermore, it introduces conditional stochastic decomposition for queue length in the orbit during server busy times and illustrates the impact of model parameters via numerical examples.

Wichner et al.(2009) have analysed multi-server finite-source retrial queue system, introducing the elements of balking and impatience for customers with limited service facility capacity. Customers arriving can either join a FIFO queue or enter an orbit, and they may abandon the buffer and join the orbit after a random time. Servers search for customers in the orbit when the buffer is empty. All random variables in the model are assumed to be exponentially distributed and independent. The study aims to analyze how balking, impatience, and buffer size impact steady-state performance measures, particularly focusing on mean response time, using the MOSEL-2 tool for modeling and calculating stationary characteristics.

Choudhury et al.(2015) have analysed M/G/1 retrial queue with dual service phases, incorporating random server breakdowns and Bernoulli vacation scheduling. The study emphasizes a FCFS retrial policy for queuing primary customers during server unavailability. Key findings include stability conditions, system size distribution, probability generating functions for server state and orbit size, and a focus on system reliability. Numerical examples illustrate practical applications of these performance metrics, enhancing system management insights.

Rajadurai et al.(2015) have analyzed into a detailed analysis of a complex batch arrival feedback retrial queue system featuring a two-phase service model, Bernoulli vacation scheduling, and orbit search. This intriguing system handles batch arrivals, with the behavior of the server contingent on its status. The server's post-service actions include vacation or waiting for the next customer. Breakdowns can occur during service, leading to channel failure for a brief period. Notably, repair doesn't commence immediately after a breakdown, introducing a repair delay. After vacation, the

server searches for orbiting customers, serving them promptly or remaining idle. The study employs the supplementary variable technique to derive probability generating functions for customers in the system and orbit, subsequently analyzing mean customer numbers and special cases. The review concludes by numerically highlighting how various parameters impact system performance.

Rajadurai et al.(2016) a batch arrival retrial queue with feedback, operating under a Bernoulli vacation schedule, and dealing with the challenge of server breakdown due to negative customer arrivals. It examines how positive customer batches interact with the server, addressing scenarios like customer balk and rejoining. The paper also delves into the server's vacation and repair routines, analyzing system size probabilities using the supplementary variable method. It provides insights into system performance, reliability, and stochastic decomposition, along with practical numerical examples and cost optimization analysis.

Nila and Sumitha (2021) are analysed $M^x/G/1$ retrial queueing model with an unreliable server, encompassing two distinct types of essential services. Notably, customers entering the system may exhibit balking and reneging behavior at specific time points. Upon a failure, the server promptly begins repairs and then resumes service for customers who were interrupted. After receiving the first essential service (FES), customers have the option to either enter the orbit as feedback customers or move on to the second essential service (SES). Once the SES is completed, the server probabilistically seeks out customers in the orbit.The study employs the Supplementary Variable Technique to analyze the system, deriving probability generating functions (PGFs) for system size, orbit size, and diverse performance metrics.

Kirupa and chandhirka (2018) are anaylsed a single server retrial queueing system that deals with two distinct customer types. This system introduces a dynamic element, as the server offers M variable service modes. If the server is available when a batch of customers arrives, one customer receives service right away, while the rest enter a waiting orbit.However, when the server is occupied, a customer from the incoming batch interrupts the ongoing service to initiate their own. Through a rigorous mathematical approach, the authors provide a deeper understanding of system dynamics, highlighting the impact of different parameters on system performance in numerical simulations.

Yang and Wu (2019) are analysed the realm of queueing theory, this paper offers a comprehensive exploration of an M/M/1 retrial queue incorporating the complexity of working vacations and server starting failures. The framework for this queuing model is elegantly formulated using the quasi-birth-death (QBD) process, laying the foundation for in-depth analysis. To evaluate the system's performance, the paper devises an array of performance metrics, shedding light on its efficiency and effectiveness. Crucially, a cost model is constructed, underscoring the economic implications of the system. The primary aim is to pinpoint the optimal service rates during normal operation and vacation periods, strategically minimizing the anticipated cost per unit time.In tackling this optimization conundrum, the canonical particle swarm optimization (CPSO) algorithm is deftly employed. Its application in this context demonstrates a practical approach to finding solutions that balance operational efficiency and cost-effectiveness.

Zhang et al.(2017) have analysed a single server retrial queue with a state-dependent service policy. It utilizes Poisson arrivals, exponential service times, and inter-retrial times. When the customer queue meets a specific threshold, the service rate switches from low to high. The study derives stationary distributions and performance measures through partial generating functions. Importantly, it reveals that this state-dependent policy can revert to a traditional retrial queue under certain conditions. By introducing a reward-cost function and using the Canonical Particle Swarm Optimization algorithm, it establishes that implementing this policy can yield greater benefits for managers compared to the classic model.

Wang et al. (2021) have analysed a retrial queue system with N-policy and breakdowns, focusing on cost control and information guidance. The system features no waiting space in front of the server and utilizes a virtual waiting list. Customers either receive immediate service or enter the virtual waiting list. The system activates when the current vacation ends and at least N customers are waiting. Two customer joining cases are explored: non-cooperative individuals seeking personal optimization and a cooperative social planner considering the entire system's profit. Equilibrium joining strategies are determined for both cases. The study employs an improved particle swarm optimization (PSO) algorithm for analysis due to the complexity of analytical characterization, supported by numerous numerical experiments to visualize parameter influences.

The proposed system has chosen the Rabbit Messaging Queue as an application, which has been solved theoretically using the supplementary variable technique and experimentally using the MATLAB Software.

2 Proposed System:

RabbitMQ is a robust and widely-used open-source message broker software that facilitates communication between different parts of a distributed system. It was originally developed by Rabbit Technologies and is now maintained by the RabbitMQ community.At its core, RabbitMQ enables the exchange of messages between various components of a system, allowing them to communicate asynchronously and efficiently. This messaging middleware is particularly valuable in scenarios where different parts of an application or multiple applications need to send and receive data without direct, synchronous connections.

Message Queues: RabbitMQ uses queues to store and manage messages. Producers publish messages to these queues, and consumers retrieve and process them. This decouples producers and consumers, allowing them to operate independently.

Message Routing: RabbitMQ supports various exchange types, including direct, fanout, topic, and headers exchanges. These exchanges determine how messages are routed to queues, providing flexibility in message routing based on criteria such as routing keys or message headers.

Message Acknowledgment: RabbitMQ ensures message reliability by allowing consumers to acknowledge receipt and processing of messages. This acknowledgment mechanism helps prevent message loss and ensures that messages are not processed more than once.

Message Persistence: Messages can be made durable in RabbitMQ, meaning they survive broker restarts. This ensures that critical data is not lost in case of system failures.

Publish-Subscribe: RabbitMQ's fanout exchange type enables a publish-subscribe pattern, where multiple consumers can receive the same message simultaneously, making it useful for broadcasting messages to multiple recipients.

Scalability: RabbitMQ can be configured to work in clusters, providing horizontal scalability and high availability. This ensures system resilience and load distribution.

Support for Multiple Protocols: RabbitMQ supports various messaging protocols, including Advanced Message Queuing Protocol (AMQP), MQTT, and STOMP, making it versatile and adaptable to different use cases.

Plugins and Extensions: RabbitMQ offers a wide range of plugins and extensions that can enhance its functionality, such as management and monitoring tools, authentication mechanisms, and message transformation capabilities.

Community and Ecosystem: RabbitMQ benefits from a strong and active community of users and developers, which means ongoing support, documentation, and a wealth of resources are readily available.

In summary, RabbitMQ plays a crucial role in modern distributed systems by enabling reliable, asynchronous communication between different components. Its flexibility, scalability, and support for various messaging patterns make it a valuable tool for building robust and efficient applications

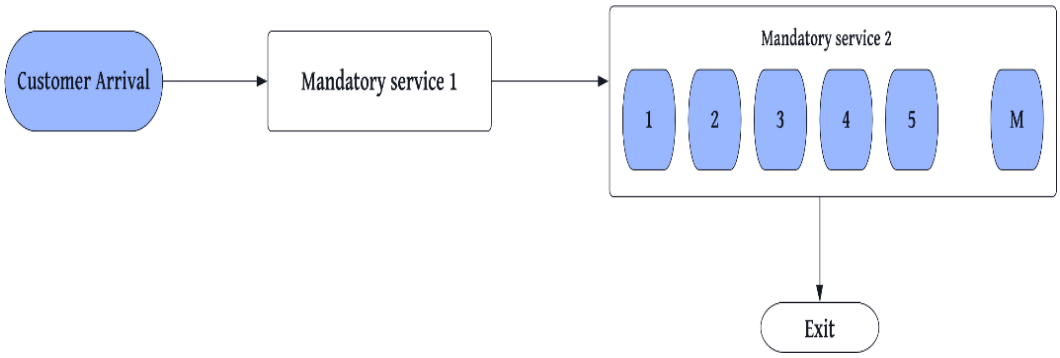


Figure 1 Structural Outline of Rabbit MQ

3. Queueing Description of Rabbit MQ

Consider a single server retrial queueing system with two phase service. Customer arrives in batches according to Poisson process with rate α. Let Y be the batch size which is a random variable with

P(Y=K)= C_k K=1,2,3… $\sum_{k=1}^{\infty} c_k = 1$ and c(z) is the probability generating function (PGF) with first two moments B_1 and B_2.

Blocked customers enter the retrial group termed as orbit. The retrial time is generally distributed with distribution function A(s), density function a(s), Laplace Stieltje's transform $A^*(\alpha)$ and the and conditional completion rate

$$\tau(x) = \frac{a(x)}{1 - A(x)}$$

The arriving customer takes service immediately when the server is in ideal state. The first service will serve for the entire arriving customer and it is generally distributed with distribution function $B_1(x)$, density function $b_1(x)$, Laplace transform $B_1^*(x)$, two moments b_1, b_2. As soon as the first service is completed, the customer will automatically perform the compulsory multi-service with probability (σ_m). After completing the multi-Service the server searches for customers in the orbit (if any) with probability $(1 - \theta)$, or remains idle with probability θ. The compulsory second phase service time follows an arbitrary distribution with distribution function $B_m(x)$, density function $b_m(x)$,Laplace transform $B_m^*(x)$, two moments $\bar{c}_1$ and $\bar{c}_2$ and conditional complete rate are

$$\varepsilon_1 = \frac{b_1(x)}{1-B_1(x)} \qquad \varepsilon_m = \frac{b_m(x)}{1-B_m(x)}$$

After the second service, the customer will leave the entire system.

3.1 Steady- state distributions

The system of equations that governs the model under steady state, by supplementary variable method are

$$\alpha R_0^F = \int_0^\infty \sum_{m=1}^{k} R_{m,0}^Q(x)\varepsilon_m(x)dx \tag{1}$$

$$\frac{dR_n^F(x)}{dx} = -(\alpha+\tau(x))R_n^F(x) \quad n \geq 1 \tag{2}$$

$$\frac{dR_0^E(x)}{dx} = -(\alpha+\varepsilon_1(x))R_0^E(x) \tag{3}$$

$$\frac{dR_n^E(x)}{dx} = -(\alpha+\varepsilon_1(x))R_n^E(x) + \sum_{k=1}^{n} c_k \alpha R_{n-1}^E(x) \ , \ n \geq 1 \tag{4}$$

$$\frac{dR_{m,0}^Q(x)}{dx} = -(\alpha+\varepsilon_m(x))R_{m,0}^Q(x) \qquad \text{m = 1,2,...k} \tag{5}$$

$$\frac{dR_{m,n}^Q(x)}{dx} = -(\alpha+\varepsilon_m(x))R_{m,n}^Q(x) + \sum_{k=1}^{n} c_k \alpha R_{m,n-1}^Q \ , \ n \geq 1, \ \text{m=1,2…k} \tag{6}$$

Boundary Conditions

$$R_0^F(0) = (1-\theta)\int_0^\infty \sum_{m=1}^{k} R_{m,n}^Q(x)\varepsilon_m(x)dx, \qquad n \geq 1 \tag{7}$$

$$R_0^E(0) = c_1\alpha R_0^F + \int_0^\infty R_1^F(x)\tau(x)dx \qquad \text{m=1,2...k} \tag{8}$$

$$R_n^E(0) = \sum_{k=1}^{n} c_k \alpha \int_0^\infty R_n^F(x) + \int_0^\infty R_{n+1}^E(x)\tau(x)dx + \alpha\int_0^\infty R_{n+1}^F(x)dx \qquad n \geq 1 \tag{9}$$

$$R_{m,n}^Q(0) = \sigma_m \int_0^\infty R_n^E(x)\varepsilon_1(x)dx \qquad n \geq 0 \qquad \text{m =1,2...k} \tag{10}$$

$$R_{m,n}^Q(0) = \int_0^\infty R_n^E \varepsilon_1 dx + \theta\int_0^\infty R_{m,n}^Q \varepsilon_m dx \tag{11}$$

The normalizing condition is

$$R_0^F + \sum_{n=1}^{\infty}\int_0^{\infty} R_n^F(x)dx + \sum_{n=0}^{\infty}\int_0^{\infty} R_n^E(x)dx + \sum_{n=0}^{\infty}\int_0^{\infty} R_{m,n}^Q(x)dx = 1$$

3.2 Steady state Solution

We use probability generating function (pgf) corresponding to different states of the server to solve the set of supplementary variable method so as to obtain the steady state solution of retrial queuing model. We define the probability generating function corresponding to the various state as

$$R^F(x,z) = \sum_{n=1}^{\infty} R_n^F(x)z^n$$

$$R^E(x,z) = \sum_{n=0}^{\infty} R_n^E(x)z^n$$

$$R_m^Q(x,z) = \sum_{n=0}^{\infty} R_{m,n}^Q(x)z^n$$

Solving partial differential equations (2) to (6) yield

$$R^F(x,z) = R^F(0,z)e^{-\alpha x}\left(1-A(x)\right) \tag{12}$$

$$R^E(x,z) = R^E(0,z)e^{-\alpha(1-c(z)x}\left(1-B_1(x)\right) \tag{13}$$

$$R_m^Q(x,z) = R_m^Q(0,z)e^{-\alpha(1-c(z)x}\left(1-B_m(x)\right) \tag{14}$$

Multiplying the steady state equation and steady state boundary condition (7) to (10) by z^n and summing over n, n=0,1,2,.. we get,

$$R^F(0,z) = (1-\theta)\sum_{m=1}^{k} R_m^Q(0,z)B_m^*(\alpha(1-c(z)) - \alpha R_0^F \tag{15}$$

$$R^E(0,z) = \frac{\alpha R_0^F}{z}c(z) + \frac{R^F(0,z)}{z}\left[(A^*(\alpha)(1-c(z)+c(z)\right] + \tag{16}$$

$$R_m^Q(0,z) = \sum_{m=1}^{k} \sigma_m R^E(0,z)B_1^*(b\alpha(1-c(z))) \tag{17}$$

Substituting the expression $R^F(0,z)$ given by equation (15) in (16) equation we get

$$R^E(0,z) = \frac{\alpha R_0^F}{z}C(z) + \frac{(1-\theta)\sum_{m=1}^{k} R_m^Q(0,z)B_m^*(\alpha(1-c(z))) - \alpha R_0^F}{z}\left[A^*(\alpha)(1-c(z)+c(z)\right] \tag{18}$$

Using equation $R_m^Q(0,z)$ given in (17) in (18) equation and simplifying we get

$$R^E(0,z) = \frac{\alpha R_0^F\left[(A^*(\alpha)(1-c(z))\right]}{D(z)} \tag{19}$$

where,

$$D(z) = (1-\theta)\sum_{m=1}^{k}\sigma_m B_1^*(\alpha(1-c(z)) + \theta\sum_{m=1}^{k} \sigma_m B_1^*(\alpha(1-c(z))B_m^*(\alpha(1-c(z)))\left[A^*(\alpha)(1-c(z)) + c(z)\right] - z$$

Substituting the expression $R^E(0,z)$ given by equation (19) in (17) equation we get

$$R_m^Q(0,z)=\frac{\sum_{m=1}^{k}\sigma_m\alpha R_0^F B_1^*(\alpha(1-c(z)))\left[A^*(\alpha)(1-c(z))\right]}{D(z)} \tag{20}$$

Substituting $R_m^Q(0,z)$ equation given by (20) in (15) equation

$$R^F(0,z)=\frac{\alpha R_0^F}{D(z)}\left[\begin{array}{l} z+\sum_{m=1}^{k}\sigma_m\left(A^*(\alpha)(1-c(z))\right)B_1^*(\alpha(1-c(z))B_m^*(\alpha(1-c(z))-(1-\theta)\sum_{m=1}^{k}\sigma_m B_1^*(\alpha(1-c(z))+ \\ \theta\sum_{m=1}^{k}\sigma_m B_1^*(\alpha(1-c(z))B_m^*(\alpha(1-c(z))c(z) \end{array}\right]$$

(21)

Substituting $R^F(0,z)$ equation given by (21) in (12) equation

$$R^F(x,z)=\frac{\alpha R_0^F}{D(z)}\left[\begin{array}{l} z+\sum_{m=1}^{k}\sigma_m\left(A^*(\alpha)(1-c(z))\right)B_1^*(\alpha(1-c(z))B_m^*(\alpha(1-c(z))-(1-\theta)\sum_{m=1}^{k}\sigma_m B_1^*(\alpha(1-c(z))+ \\ \theta\sum_{m=1}^{k}\sigma_m B_1^*(\alpha(1-c(z))B_m^*(\alpha(1-c(z))c(z) \end{array}\right] e^{-\alpha}(1-A(x))$$

Substituting the expression of $R^E(0,z)$, $R_m^Q(0,z)$ and given by equation(19) and (20) in (13) (14) equation we get

$$R^E(x,z) = \frac{\alpha R_0^F\left[\left(A^*(\alpha)(1-c(z))\right]\right.}{D(z)}\ e^{-\alpha(1-c(z))x}\left[1-B_1(x)\right]$$

$$R_m^Q(x,z)=\frac{\sum_{m=1}^{k}\sigma_m\alpha R_0^F B_1^*(\alpha(1-c(z)))\left[A^*(\alpha)(1-c(z))\right]}{D(z)}\quad e^{-\alpha(1-c(z))x}\left[1-B_m(x)\right]$$

3.3 Steady state Analysis

The Probability Generating Function of the orbit size when the server is in idle is

$$R^F(z)=\int_0^{\infty}R^F(x,z)dx.$$

$$R^F(z) = \frac{R_0^F}{D(z)}\left[\begin{array}{l} z+\sum_{m=1}^{k}\sigma_m\left(A^*(\alpha)(1-c(z))\right)B_1^*(\alpha(1-c(z))B_m^*(\alpha(1-c(z))-(1-\theta)\sum_{m=1}^{k}\sigma_m B_1^*(\alpha(1-c(z))+ \\ \theta\sum_{m=1}^{k}\sigma_m B_1^*(\alpha(1-c(z))B_m^*(\alpha(1-c(z))c(z) \end{array}\right]\left[1-A^*(\alpha)\right]$$

The probability generation function of the orbit size when the server is busy is

$$R^E(z)=\int_0^{\infty}R^E(x,z)dx.$$

$$R^E(z)=\frac{\alpha R_0^F\left[\left(A^*(\alpha)(1-c(z))\right]\right.}{D(z)}\left[1-B_1^*(\alpha(1-c(z))\right]$$

The probability generation function of the orbit size when the server in multi-service is

$$R_m^Q(z) = \int_0^\infty R_m^Q(x,z)dx.$$

$$R_m^Q(z) = \frac{\sum_{m=1}^{k} \sigma_m \alpha R_0^F B_1^*(\alpha(1-c(z)))\left[A^*(\alpha)(1-c(z))\right]}{D(z)} \left[1 - B_m(x)(\alpha(1-c(z)))\right]$$

Using normalizing condition

$$R_0^F + R^F(1) + R^E(1) + \sum_{m=1}^{k} R_M^Q(1) = 1$$

we get

$$R_0^F = \frac{D'(1)}{\sum_{m=1}^{k} \sigma_m \bar{c}_1 \lambda \bar{c}_1 \mu_2 \left[\theta - A^*(\alpha)\right] + \sum_{m=1}^{k} \sigma_m \lambda \bar{c}_1 \mu_1 \bar{c}_1 A^*(\alpha)(1-\bar{c}_1) + \theta \sum_{m=1}^{k} \sigma_m \bar{c}_1 \lambda \bar{c}_1 \mu_2 + \theta \sum_{m=1}^{k} \sigma_m \bar{c}_1 \lambda \bar{c}_1 \mu_1 - A^*(\alpha) \lambda \bar{c}_1 \mu_1 \bar{c}}$$

4 PERFORMANCE MEASURES

This section presents the derivation of performance measures of proposed model.

❖ **The steady state probability that the server is idle during the retrial time is**

$$R^F(1) = \frac{R_0^F\left[1 + \sum_{m=1}^{k} \sigma_m \left(\lambda \bar{c}_2 \mu_2 + \lambda \bar{c}_1 \mu_1\right)\right]\left[A^*(\alpha)(1-c(z)\right] - (1-\theta)\sum_{m=1}^{k} \sigma_m \lambda \bar{c}_1 \mu_1 + \theta \sum_{m=1}^{k} \sigma_m \left(\lambda \bar{c}_2 \mu_2 + \lambda \bar{c}_1 \mu_1\right) + \theta \sum_{m=1}^{k} \sigma_m \bar{c}_1}{D'(1)}$$

❖ **The steady state probability that the server is busy in the first phase service is**

$$R^E(1) = \frac{R_0^F\left[(A^*(\alpha) - 1)\right]\left[\lambda \vec{c}_1 \mu_1\right]}{D'(1)}$$

❖ **The steady state probability that the server is busy in second phase of multi-service is**

$$R_m^Q(1) = \frac{R_0^F\left[(A^*(\alpha))\right]\left[\lambda \vec{c}_2 \mu_2\right]}{bD'(1)}$$

❖ **The probability generating function of the number of customers in the queue is given by:**

$$P_q(z) = R_0^F + R^F(z) + R^E(z) + R_m^Q(z)$$

$$N(z) = \theta \sum_{m=1}^{k} \sigma_m B_1^*(\alpha(1-c(z)) B_m^*(\alpha(1-c(z))c(z) + \left[\sum_{m=1}^{k} \sigma_m B_1^*(\alpha(1-c(z)) B_m^*(\alpha(1-c(z))A^*(\alpha) + \theta \sum_{m=1}^{k} \sigma_m B_1^*(\alpha(1-c(z))\right]\left[A^*(\alpha)\right] +$$

$$\theta \sum_{m=1}^{k} \sigma_m B_1^*(\alpha(1-c(z))c(z) + A^*(\alpha)\left(1 - B_1^*(\alpha(1-c(z))\right)$$

❖ **The mean number of customers in the queue :**

$$L_q = \frac{D'N'' - N''D''}{2D'^2}$$

where,

$$D'(1) = (1-\theta)\sum_{m=1}^{k}\sigma_m\lambda\bar{c}_1\mu_1 + \theta(\lambda\bar{c}_1\mu_2 + \lambda\bar{c}_1\mu_1)(\bar{c}_1(1-A^*(\alpha)) - 1$$

$$D''(1) = (1-\theta)\sum_{m=1}^{k}\sigma_m\left(\lambda\bar{c}_1\mu_1 + \lambda^2\bar{c}_2^2\mu_2 + \lambda\bar{c}_1\mu_2 + \lambda^2\bar{c}_2^2\mu_1\right) + \theta\left(\left(\lambda\bar{c}_1\mu_1 + \lambda^2\bar{c}_2^2\mu_2 + \lambda\bar{c}_1\mu_2 + \lambda^2\bar{c}_2^2\mu_1\right)\right)(\bar{c}_1(1-A^*(\alpha)) - 1$$

$$N'(1) = 2\theta\sum_{m=1}^{k}\sigma_m\left[\lambda\bar{c}_1\mu_2 + \lambda\bar{c}_1\mu_1\right] + \left[\theta\sum_{m=1}^{k}\sigma_m\bar{c}_1 + \sum_{m=1}^{k}\sigma_m A^*(\alpha)(1-\bar{c}_1) + \sum_{m=1}^{k}\sigma_m\lambda\bar{c}_1\mu_2\right]\left[A^*(\alpha)\right] + A^*(\alpha)(1-\lambda\bar{c}_1\mu_1)$$

$$N''(1) = 2\theta\sum_{m=1}^{k}\sigma_m\left[\lambda^2\bar{c}_1^2\mu_2 + \lambda\bar{c}_2\mu_1 + \lambda^2\bar{c}_1^2\mu_2 + \lambda\bar{c}_1\mu_2\right] + \theta\sum_{m=1}^{k}\sigma_m\left[\lambda\bar{c}_2\mu_2 + \lambda\bar{c}_1\mu_1\right]\bar{c}_1 + \theta\sum_{m=1}^{k}\sigma_m\bar{c}_1 + 2\sum_{m=1}^{k}\sigma_m A^*(\alpha) + 2\sum_{m=1}^{k}\sigma_m\left[\lambda\bar{c}_2\mu_2 + \lambda\bar{c}_1\mu_1\right]$$
$$+ \sum_{m=1}^{k}\sigma_m A^*(\alpha)(1-\bar{c}_2) + \theta\sum_{m=1}^{k}\sigma_m A^*(\alpha)\left[\lambda^2\bar{c}_1^2\mu_2 + \lambda\bar{c}_2\mu_1\right]\left[A^*(\alpha)\right] + A^*(\alpha)(1-\lambda^2\bar{c}_1^2\mu_2 + \lambda\bar{c}_1\mu_2)$$

❖ **The probability generating function of the number of customers in the system is given by**

$$P_s(z) = R_0^F + R^F(z) + zR^E(z) + zR_m^Q(z)$$

$$T(z) = \left[\begin{array}{l}\left[\begin{array}{l}\theta\sum_{m=1}^{k}\sigma_m B_1^*(\alpha(1-c(z))B_m^*(\alpha(1-c(z))A^*(\alpha) + \theta\sum_{m=1}^{k}\sigma_m B_1^*(\alpha(1-c(z))B_m^*(\alpha(1-c(z))c(z) + \sum_{m=1}^{k}\sigma_m B_1^*(\alpha(1-c(z))B_m^*(\alpha(1-c(z)) - \\ \sum_{m=1}^{k}\sigma_m B_1^*(\alpha(1-c(z))B_m^*(\alpha(1-c(z))A^*(\alpha)c(z)\end{array}\right]\left[1-A^*\right]\theta\sum_{m=1}^{k}\sigma_m B_1^*(\alpha(1-c(z))B_m^*(\alpha(1-c(z))c(z) \\ + z\left[A^*(\alpha)(1-\right]B_1^*(\alpha(1-c(z)) + \sum_{m=1}^{k}\sigma_m B_1^*(\alpha(1-c(z)) - \sum_{m=1}^{k}\sigma_m B_1^*(\alpha(1-c(z))B_m^*(\alpha(1-c(z))\end{array}\right]$$

❖ **The mean number of customers in the system**

$$L_s = \frac{D'T'' - T'D''}{2D'^2}$$

where,

$$T'(1)=\theta\sum_{m=1}^{k}\sigma_m\left[\lambda\bar{c}_2\mu_2+\lambda\bar{c}_1\mu_1\right]A^*(\alpha)+[2\left[\theta+\sum_{m=1}^{k}\sigma_m\left[\lambda\bar{c}_2\mu_2+\lambda\bar{c}_1\mu_1\right]\right]+\theta\sum_{m=1}^{k}\sigma_m\bar{c}_1-\theta\sum_{m=1}^{k}\sigma_m A^*(\alpha)\bar{c}_1](1-A^*(\alpha))$$

$$-A^*(\alpha)\lambda\bar{c}_1\mu+\sum_{m=1}^{k}\sigma_m\lambda\bar{c}_1\mu_1 rA^*(\alpha)-\sum_{m=1}^{k}\sigma_m A^*(\alpha)\left[\lambda\bar{c}_2\mu_2+\lambda\bar{c}_1\mu_1\right]$$

$$T''(1)=\theta\sum_{m=1}^{k}\sigma_m\left[\lambda^2\bar{c}_1^2\mu_2+\lambda\bar{c}_2\mu_1+\lambda^2\bar{c}_2^2\mu_2+\lambda\bar{c}_1\mu_2\right]A^*(\alpha)+\left[\begin{array}{l}2\left(\theta\sum_{m=1}^{k}\sigma_m\left[\lambda^2\bar{c}_1^2\mu_2+\lambda\bar{c}_2\mu_1+\lambda^2\bar{c}_2^2\mu_2+\lambda\bar{c}_1\mu_2\right]A^*(\alpha)\right)+\\ 2\theta\left(\sum_{m=1}^{k}\sigma_m\left[\lambda\bar{c}_1\mu_1+\lambda\bar{c}_1\mu_2\right]\bar{F}_1\right)+\theta\left(\sum_{m=1}^{k}\sigma_m\bar{c}_1\right)+\left(\sum_{m=1}^{k}\sigma_m A^*(\alpha)\left[\lambda\bar{c}_1\mu_1+\lambda\bar{c}_1\mu_2\right]\bar{F}_1\right)-\left(\sum_{m=1}^{k}\sigma_m\bar{c}_1\right)\end{array}\right]\left[1-A^*(\alpha)\right]-\lambda\bar{c}_1\mu A^*(\alpha)-$$

$$2\left(\sum_{m=1}^{k}\sigma_m A^*(\alpha)\left[\lambda\bar{c}_1\mu_1+\lambda\bar{c}_1\mu_2\right]\right)+A^*(\alpha)\left(\lambda^2\bar{c}_1^2\mu_2+\lambda\bar{c}_2\mu_1\right)+\sum_{m=1}^{k}\sigma_m\left[\lambda^2\bar{c}_2^2\mu_2+\lambda\bar{c}_1\mu_2\right]A^*(\alpha)$$

❖ **The busy period**, is one of the significant performance measures in the retrial situation. The system busy period B defines the period that starts at an epoch when an arriving customer finds an empty system and ends at the next departure epoch at which the system is empty again.

$$\mathrm{E(B)}=\frac{1}{\alpha}\left(\mathrm{R}_0^{\mathrm{F}^{-1}}-1\right)$$

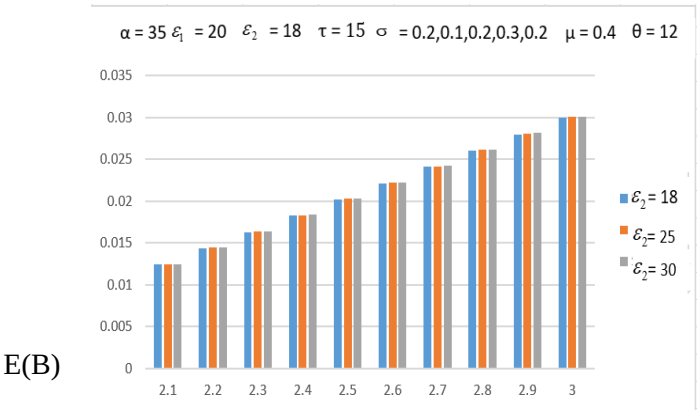
$$\sum_{m=1}^{k}\sigma_m\bar{c}_1\lambda\bar{c}_1\mu_2\left[\theta-A^*(\alpha)\right]+\sum_{m=1}^{k}\sigma_m\lambda\bar{c}_1\mu_1\bar{c}_1A^*(\alpha)(1-\bar{c}_1)+\theta\sum_{m=1}^{k}\sigma_m\bar{c}_1\lambda\bar{c}_1\mu_2+\theta\sum_{m=1}^{k}\sigma_m\bar{c}_1\lambda\bar{c}_1\mu_1-A^*(\alpha)\lambda\bar{c}_1\mu_1\bar{c}$$

$$=\frac{}{D'(1)}-$$

5. Numerical illustration

In this section, we present some numerical examples to study the effect of the parameters on the main system performance. The system parameters are arrival rate (α), retrial rate (τ), first phase mandatory service rate (ε_1) and the probability of customer is opting for multi-phase mandatory service (σ_m), balking rate (b), reneging rate (r) and batch size follows geometric distribution with mean 1/μ where μ ∈ (0, 1] on the following measures.

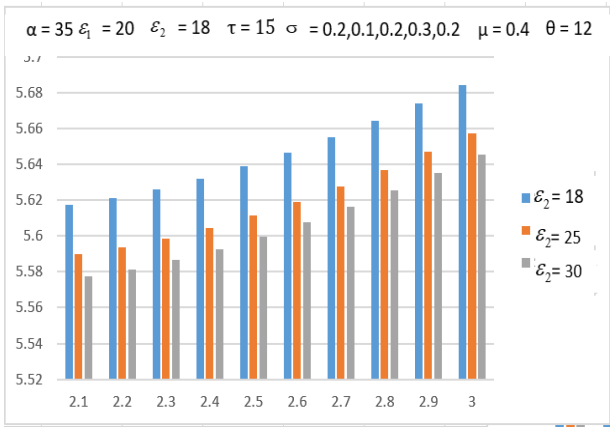
- L_s expected number of customers in the system.
- L_q expected number of customers in the queue.
- E(B) expected number of busy period



E(B)

ε1

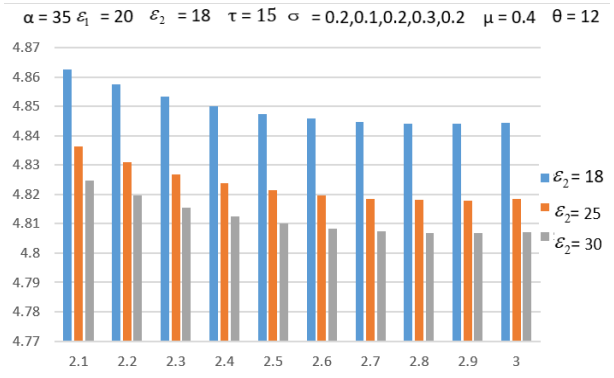
Figure 2: Effect of ε_1 and ε_2 on E(B)



L_s

ε1

Figure 3: Effect of ε_1 and ε_2 on L_s



L_q

ε1

Figure 4: Effect of ε_1 and ε_2 on L_q

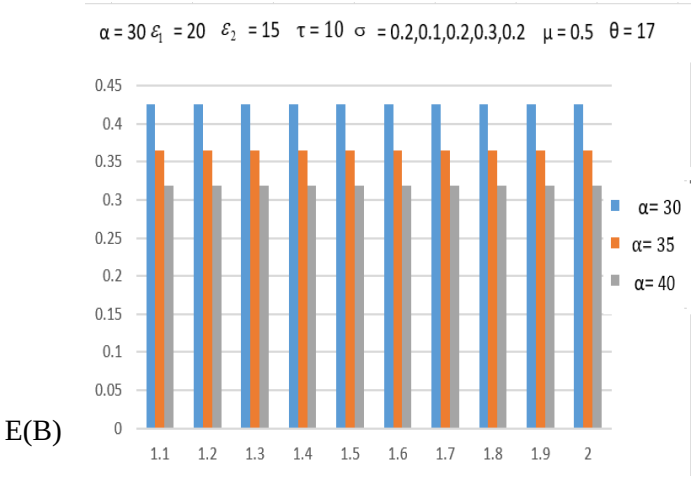


Figure 5: Effect of τ and α on E(B)

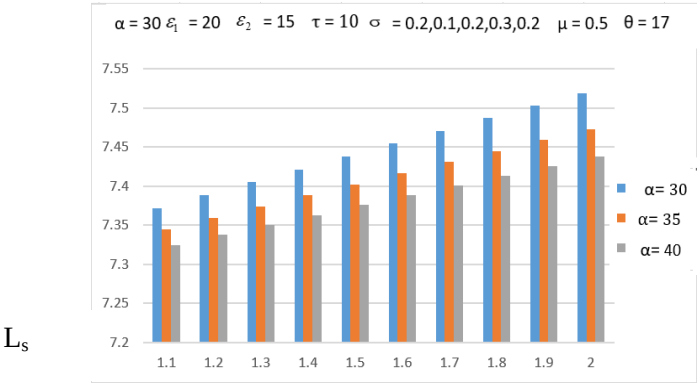


Figure 6: Effect of τ and α on L_s

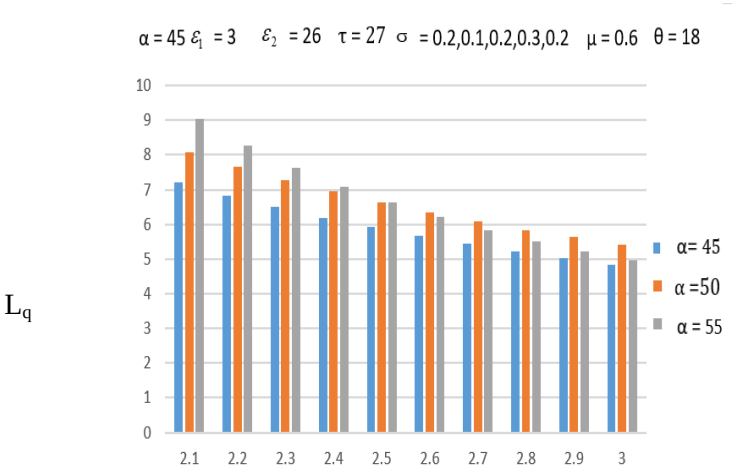


Figure 7: Effect of θ and α on L_q

Figure 2 verifies Second phase of multi-service rate increases, busy period of system to be constant. Figure 3 verifies Second phase of multi-service rate increases, length of the system will be decreases. Figure 4 verifies Second phase of service rate increase, length of queue size will be decreases. Figure 5 verifies arrival rate increases, busy period of system will be decreases. Figure 6 verifies arrival rate increases, length of the system will be decreases. Figure 7 arrival rate increases, length of queue size also increase.

6 Cost Analysis

We contract an expected cost function per unit for the M/M/1 Queue with two phase multi service retrial Queue with Orbital Search. Our objectives is to determine the optimal values for service rate $(\varepsilon_{1,} \varepsilon_m)$ say $(\varepsilon_1^*, \varepsilon_m^*)$ in order to minimize the proposed cost function.

Let us define the cost elements as follows:

C_h : holding cost per unit time for each customer present in the system.

P_e : Probability that the server is idle during retrial time.

P_f : Probability that the server is busy in first phase of service.

P_r : Probability of that the server is busy in second phase of multi- service.

P_b : Probability of busy period.

Using the above defined cost element, we develop the cost function as follow

$$Tc\ (\varepsilon_{1,} \varepsilon_m) = C_h L_s + P_e R^F\ (1) + P_f R^E(1) + P_r R_m{}^Q(1) + P_b E(B).$$

Cost minimization problem of the conceived model can be mathematically described as an uncontrained problem as follow

$$Tc\ (\varepsilon_1^*, \varepsilon_m^*) = \min Tc\ (\varepsilon_{1,} \varepsilon_m)$$

7 Optimal Analysis:

The most prevalent optimisation issue for service systems is highly nonlinear and subject to a number of intricate restrictions. Even for a single purpose, numerous cost factors are frequently in conflict, making it difficult to always find the best answer. Some classical optimization techniques are not suitable for such problems. The alternative efficient algorithm for finding the near optimal solution of a such problem. In this paper particle swarm algorithm is used to find the optimal solution for our model.

Particle swarm optimization (PSO) is a heuristic algorithm used in various optimization problems. PSO is inspired by social behaviour of animals, such as flocks of birds or schools of fish, where individuals communicate with each other to find optimal solutions. In PSO, the potential solutions are represented by particles, which move through the search space and adjust their positions based on their own best-known location and the best-known location of the swarm. This way, PSO effectively explores the search space and finds the global optimum. The algorithm can be further optimized by adjusting the swarm size, the maximum velocity of the particles, and the coefficients of the position update equations. PSO has been successfully applied to various problems, such as function optimization, neural network training, and image processing.

Particle Swarm Optimization: Pseudo code

Input: $\varepsilon_{1,}\ \varepsilon_m$ service rate, learning parameter.

Output: Approximate the solution vector $[\varepsilon_1^*, \varepsilon_m^*]$ and compute the values of cost function $Tc[\varepsilon_{1,}\ \varepsilon_m]$

Step 1: Initialization: Find location ε_m of m particle.

Step 2: Find G^* (g-best) from (min) form $[Tc\ (\varepsilon_{1),}Tc(\varepsilon_2) \ldots Tc(\varepsilon_m)]$

Step 3: while (t < max generation) or (stop criterion)

for loop over all n particle and all dimension.

Step 4: Find new location for i th particle $\varepsilon_1^t + \varepsilon_m^{t+1}$

Step 5: Evaluate objective function at new location ε_m^{t+1}

Step 6: Find the current best (p-best) for each particle ε end for.

Step 7: update global for G^*

t- t+1

end while

Step 8: output find results $\varepsilon_1^*, \varepsilon_m^*$

7.2 Numerical Analysis:

In this section we present numerical analysis on the parameter optimization of queueing system under consideration. We consider the parameters C_h= 8, P_e = 9, P_f = 4, P_r = 9, P_b = 8.

Table 1: Optimal value of ε_1^* and ε_m^* with minimal excepted cost Tc.

$(\varepsilon_{1,}\ \varepsilon_m)$	ε_1^*	ε_m^*	Tc
(36,20)	60.7515	67.4451	8632.042
(36,25)	67.1602	61.8861	8731.438
(36,30)	70.4021	62.5321	8871.046
(40.20)	64.3990	58.4771	9334.613
(45,20)	79.7400	64.3047	9373.348
(50,20)	56.4393	57.6626	9420.614

Table 1 refers to varied values of system parameters for proposed algorithm. In table 1 we are find the total execpted cost of first phase of service and multi phase of service for Rabbit MQ model. If the messages are increase for their service in two phases, the total excepted cost increase.

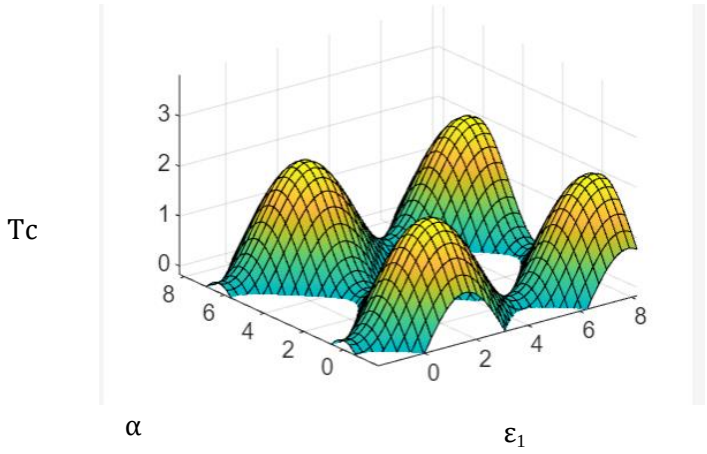


Figure 8 Total excepted cost with parameters $\varepsilon_{2,}$ and α

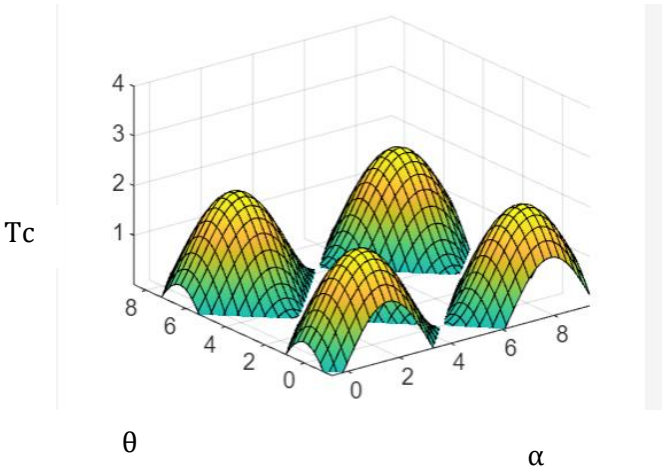


Figure 9 Total excepted cost with parameters α and θ

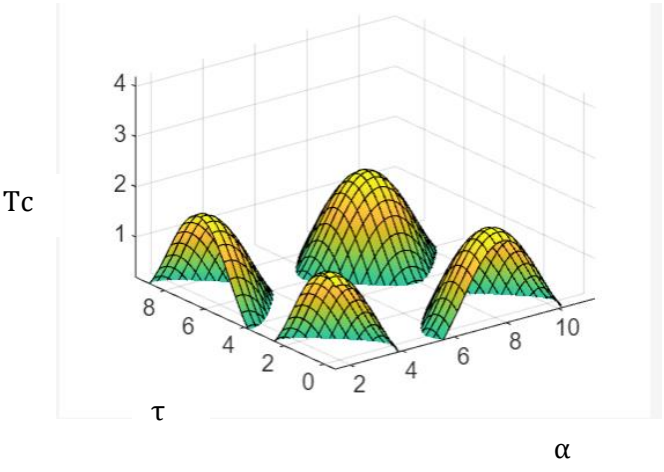


Figure 10 Total excepted cost with parameters α and τ

8 Conclusion

RabbitMQ is a widely used message broker that plays a crucial role in cloud computing for communication between distributed systems. In this context, two-phase retrial queue and multiphase

retrial queue with orbital search are important paradigms for message queueing systems. The probability generating function of the number of customers in the system is found using the supplementary variable technique Performance metrics and special case are analysed. Furthermore, the use of MATLAB and PSO algorithm provides an effective way to optimize the performance of these message queueing systems. PSO algorithm can be used to optimize the parameters of the message queueing system to achieve optimal performance.

Reference:

[1] Choudhury, G., Tadj, L., & Deka, M. (2015). An unreliable server retrial queue with two phases of service and general retrial times under Bernoulli vacation schedule. *Quality Technology & Quantitative Management*, *12*(4), 437-464. https://www.tandfonline.com/doi/abs/10.1080/16843703.2015.11673430

[2] Hedadji, C., Arrar, N., & Djellab, N. (2023). APPROXIMATION OF M/M/1 QUEUE WITH INTERRUPTION SERVICE, RETRIALS AND ORBITAL SEARCH. *Advanced Mathematical Models & Applications*, *8*(2). http://jomardpublishing.com/UploadFiles/Files/journals/AMMAV1N1/v8n2/Hedadli_et_al.pdf

[3] Järvelä, A., & Lindmark, S. (2019). Evaluation and comparison of a RabbitMQ broker solution on Amazon Web Services and Microsoft Azure. https://www.diva-portal.org/smash/get/diva2:1331518/FULLTEXT01.pdf

[4] Johansson, T. (2018). Message-Oriented Middleware as a Queue Management Solution to Improve Job Handling within an E-Commerce System. https://www.diva-portal.org/smash/get/diva2:1251225/FULLTEXT01.pdf

[5] Kirupa, K., & Chandrika, K. U. (2018). Unreliable batch arrival retrial G-queue with fluctuating modes of service, preemptive priority and orbital search. *Int J Math Trends Technol (Special Issue)*, 45-53. http://ijmttjournal.org/special-issue/ICRMIT/ICRMIT-P108.pdf

[6] Li, J. T., Li, T., & An, M. (2019). An M/M/1 retrial queue with working vacation, orbit search and balking. *Engineering Letters*, *27*(1), 97-102. https://www.engineeringletters.com/issues_v27/issue_1/EL_27_1_12.pdf

[7] Nila, M., & Sumitha, D. A FEED BACK MX/G/1 RETRIAL QUEUE WITH IMPATIENT CUSTOMERS, ACTIVE BREAKDOWN AND ORBITAL SEARCH. https://www.research-publication.com/amsj/uploads/papers/vol-10/iss-03/AMSJ-2021-N03-28.pdf

[8] Nila, M., & Sumitha, D. Batch Arrival Retrial Queueing Model with Starting Failures, Customer Impatience, Multi Optional Second Phase and Orbital Search. https://d1wqtxts1xzle7.cloudfront.net/60865390/batch-arrival-retrial-queueing-model-with-starting-failures-IJERTV8IS09023320191010-111141-nr9r3c-lib

[9] Rajadurai, P., Chandrasekaran, V. M., & Saravanarajan, M. C. (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. https://www.researchgate.net/profile/P-Rajadurai/publication/282572601_Analysis_of_an_MXG1_unreliable_retrial_G-queue_with_orbital_search_and_feedback_under_Bernoulli_vacation_schedule/links/56126f0b08aec422d117b16d/Analysis-of-an-MX-G-1-unreliable-retrial-G-queue-with-orbital-search-and-feedback-under-Bernoulli-vacation-schedule.pdf [3] Kalaichelvi, A. (2007), Second Order Fuzzy Topological Spaces – I, *ActaCienciaIndica*, 33(**3**), 819-826.

[10] Rajadurai, P., Indhira, K., Saravanarajan, M. C., & Chandrasekaran, V. M. (2015). Analysis of an M [X]/G/1 Feedback Retrial Queue with Two Phase Service, Bernoulli Vacation, Delaying Repair and Orbit Search. *Advances in Physics Theories and Applications*, 40. https://www.researchgate.net/profile/Chandrasekaran-M/publication/273145114_Analysis_of_an_MXG1_feedback_retrial_queue_with_two_phase_service_Bernoulli_vacation_delaying_repair_and_orbit_search/links/5aebe23c0f7e9b01d3e06a69/Analysis-of-an-MX-G-1-feedback-retrial-queue-with-two-phase-service-Bernoulli-vacation-delaying-repair-and-orbit-search.pdf

[11] Rani, K. P., Chandrika, K. U., & Bagyam, J. E. A.(2017) BULK ARRIVAL RETRIAL QUEUE WITH NON PERSISTENT CUSTOMERS, ACTIVE BREAKDOWN, DELAYED REPAIR AND ORBITAL SEARCH. https://www.researchgate.net/profile/Ebenesar-Anna-Bagyam/publication/324064822_BULK_ARRIVAL_RETRIAL_QUEUE_WITH_NON_PERSISTENT_CUSTOMERS_ACTIVE_BREAKDOWN_DELAYED_REPAIR_AND_ORBITAL_SEARCH/links/5abb9eb10f7e9b822c6d4

088/BULK-ARRIVAL-RETRIAL-QUEUE-WITH-NON-PERSISTENT-CUSTOMERS-ACTIVE-BREAKDOWN-DELAYED-REPAIR-AND-ORBITAL-SEARCH.pdf

[12] Sangeetha, N., & Chandrika, K. U. (2022, November). MX/G/1 retrial G-queue with multistage and multi-optional services, feedback, randomized J vacation and orbital search. In *AIP Conference Proceedings* (Vol. 2516, No. 1). AIP Publishing. https://web.archive.org/web/20221204064753id_/https://aip.scitation.org/doi/pdf/10.1063/5.0109259

[13] Sharvari, T., & Sowmya Nag, K. (2019). A study on modern messaging systems-kafka, rabbitmq and nats streaming. *CoRR abs/1912.03715*. https://arxiv.org/abs/1912.03715

[14] Wang, Z., Liu, L., Shao, Y., & Zhao, Y. Q. (2021). Joining strategies under two kinds of games for a multiple vacations retrial queue with N-policy and breakdowns. *AIMS Math*, *6*, 9075-9099. http://www.aimspress.com/aimspress-data/math/2021/8/PDF/math-06-08-527.pdf

[15] Wüchner, P., Sztrik, J., & de Meer, H. (2009). Finite-source M/M/S retrial queue with search for balking and impatient customers from the orbit. *Computer networks*, *53*(8), 1264-1273. https://www.sciencedirect.com/science/article/abs/pii/S1389128609000516

[16] Yang, D. Y., & Wu, C. H. (2019). Performance analysis and optimization of a retrial queue with working vacations and starting failures. *Mathematical and Computer Modelling of Dynamical Systems*, *25*(5), 463-481. https://www.tandfonline.com/doi/pdf/10.1080/13873954.2019.1660378

[17] Zhang, X., Wang, J., & Ma, Q. (2017). Optimal design for a retrial queueing system with state-dependent service rate. *Journal of Systems Science and Complexity*, *30*, 883-900. https://www.researchgate.net/profile/Xuelu-Zhang/publication/316640288_Optimal_design_for_a_retrial_queueing_system_with_state-dependent_service_rate/links/5c21848892851c22a3443cb5/Optimal-design-for-a-retrial-queueing-system-with-state-dependent-service-rate.pdf