Chapter 1

Introduction

1.1 Motivation

The service industry interacts with our lives daily. Services can be defined as deeds, processes, and performances. The most critical issue for a customer-service system is to effectively and efficiently deliver the service to the customers. The difference between expected and perceived service is a reflection of service quality. There are eight dimensions for quality; they are features, performance, conformance, reliability, serviceability, durability, aesthetics, and value. The primary need of the customers in the modern service systems is to have a decent satisfaction and experience, along with added value to be served to them.

The increase in technology and the demand for quality service have caused severe congestion, delays. Congestion and delays are the escalating negative phenomenon. Congestion has become a veritable scourge that plagues industrialized sectors and customer care sectors alike. It affects both customers and servers of service systems, and as well as reducing economic efficiency. It also has other adverse effects on policy and society. The disturbing thing is that this expression of modern times has been intensifying, without any sign of having a limit, thus becoming a nightmare that threatens the quality of service.

The preceding few years have realized a rapid escalation in the number of customers in service sectors, as a result of various factors, such as the increase in the demand of quality service in socio-techno-economic sectors, the greater availability of service providers, the relative reduction in delays and blockings and the more excellent advent of new technology. Congestion has been cumulative in much of the real-time services, and everything designates that it will endure getting worse, representing an undoubted menace to the quality service. Its primary expression is a progressive drop in service rates, resulting in increases in response time, resource consumption, other operating costs, and strategic delay, as compared with uninterrupted services. Congestion is mainly due to the intensive use of service facility.

Performance modeling is one framework that can help the service organization to maintain service quality. Performance modeling plays a vital role in designing, developing, and characterizing the service system. The measurement and modeling are the two critical approaches of performance evaluation of the service system, in which performance modeling based on queueing theory has received significant attention from several researchers and system analysts. Performance modeling is required to predict the quality of various service systems such as a computer, communication, manufacturing, etc. Performance modeling through the queueing-theoretic approach for the service system gives the insight to ensure whether the appropriate grade of service (GoS) is rendered by the system facility in terms of response time. The system planner can analyze this by considering several alternatives and evaluating various performance indices by using analytical and computational queueing models.

The problem of congestion prevalent in every sector of day-to-day necessity from manufacturing to supply, from planning through operational to implementation, from social to technology and economic, transportation, scheduling, etc. The congestion in any service system is more influencing if there is limitation of resources due to constraints of facilities, supply, blocking, design, human resources, money, etc. Customer-oriented, server-oriented, and system-oriented investigation gives a better insight to system planners for upgrading the grade-of-service (GoS) service facility in the service system. For optimization perspective, cost analysis by setting appropriate cost function has been done in some models to optimize the expected total cost of the system incurred. The measurement of reliability/availability indices can provide a basic idea about system's operating efficiency. Sensitivity analysis has been facilitated to show the applied nature of the developed models. The possibility and importance of the investigation done have been featured by illustrative specific practical justifications. We hope that our work will be helpful in providing alternatives and optimal choices for improving the quality-of-service (QoS) in various congestion problems.

The content of the thesis is state-of-the-art for future study on service models by a research fellow, system managers, analysts, engineers, and practiciners. It includes various well-studied service model which imitate many real-time problems of the different service system. It also addresses performance evaluation, optimal design, comprehensive algorithm, mathematical tools, and cited references. The unique features of this thesis on optimal analysis of service system include

- Optimal analysis, sensitivity analysis, parametric analysis including in-depth literature survey and background knowledge on performance evaluation of service system.
- Markov chain imbeddable structures.

- Nature-based optimization techniques for the development of optimal invariant design.
- Application in understanding the behavior and advantages of the service system.

1.2 Objectives of the Thesis

The main objectives of the thesis are to predict and evaluate the performance characteristics of queueing modeling of the service and machining systems and to suggest optimal design. The whole study will be divided into the following categories

- Identification of new and challenging queueing problems which define realistic and random service system.
- To implement numerical solution techniques to compute steady or transient state probabilities.
- Classification and computation of various performance measures of queueing systems based on customer, server, or system outlook.
- To develop computational algorithms for the analysis of the governing system numerically. Some classical and meta-heuristic approaches are to be identified for sensitivity and optimal analysis.
- Recognition of critical parameters in designing of the queueing system or service system.

1.3 Queueing Systems

Queues are ubiquitous, which generally form everywhere in our day-to-day life whenever the existing demand exceeds the overall capacity of the service facility in any service system. Usually, a queue is a waiting line of customers (may also include data, jobs, raw materials, requests, etc.) who require the service from one or more servers (maybe machines, processors, etc.) at a point of time. The mathematical study of these queues or waiting lines using the fundamental law of Probability & Statistics is known as queueing theory. Queueing theory is applied in several congestions and delays management problems, and service systems for modeling purpose. The queueing systems are prevalent in each service sector, from business to market, from manufacturing to repair, from service to process, etc. for making decisions about the resources needed for providing service to the customers. The queueing theory has its origin in the early twentieth century when Agner Kraup Erlang, a Danish mathematician, and engineer, published the first research paper (Erlang, 1909) on the problem of congestion in telephonic exchange.

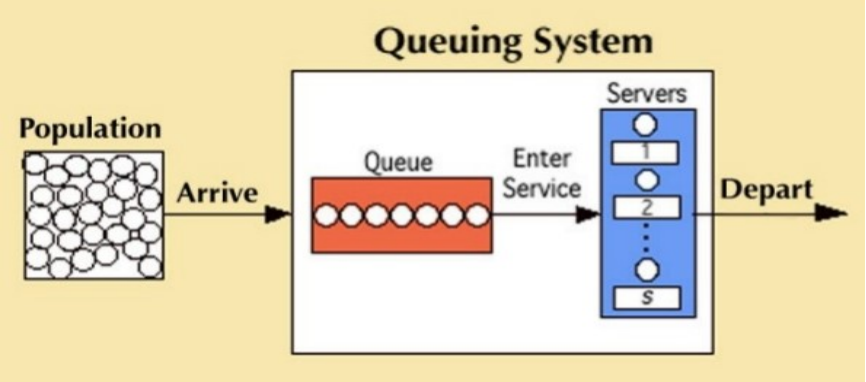

Figure 1.1: Block diagram of a basic queueing model.

The concept of queueing theory is extensively used in many service systems, namely, information technology (IT), computer & communication systems, manufacturing & repairing systems, supply-chain & scheduling, traffic flow, data processing, and many other management industries. Therefore, the future development of technology and management policies can further be upgraded by a comparative study of the past and present experiences, which will grow the requirements of more in-depth insights into the rapid advancement of queueing theory. Queues are formed because our resources are limited. In general, many real waiting situations that occur in our day-to-day life need to be exposed, as they waste a lot of our time, money, and many other useful resources. The concept of queueing theory, along with several queueing terminologies, is extensively used. There are several congestions, blockings, and delays situations where the term queueing or waiting is considerably used. Therefore, the performance quality of the service system and the behavior of users ensures to queue or not to queue.

Since waiting is a dominant part of many service-related operations, it is an essential area of investigation, research, and analysis. Each queueing system has its advantages and disadvantages, but with no doubt each customer-service system's goal is to reduce the waiting time and that customer returns. The main objective of queueing modeling in service systems is to derive a mathematical model required to serve customers and use that model to predict some significant quality performance measures viz, the expected length of the queue, mean-waiting time, et cetera. A basic queueing model is shown in Fig. 1.1.

The multiple-server waiting line strategy eliminates jockeying behavior. A single-line, multiple-server system has improved performance in terms of waiting times than a similar arrangement with a dedicated line for each server. The multiple-line configuration is suitable when skilled servers are used or when waiting space limitations make a single-line system troublesome. Contrary, the multiple-service line system

offers the customer the option of selecting his server, but check-out may be considerably slower. There are many conducts to minimize customer wait times. As expected, one strategy to reduce customer wait time is to diminution the expected service time. This can be proficient by identifying best practices among all the servers, customers, system designs, standardizing processes based on best practices, and enhancing training to guarantee that best practices are followed.

1.4 Applications of Queueing Based Service Systems

The service system consists of a heterogeneous group of services. The services can be classified into three categories:

- Stagnant personal services
- Substitutable personal services
- Progressive services

Stagnant personal services: These services frequently require direct contact between the customers and the service provider. Some examples are teaching, live artistic performance, shopping, etc.. Since the quality of such a service is highly correlated with labor time, these services offer low innovation potential and are difficult to standardize.

Substitutable personal services: Similar to stagnant personal services in characteristics; however, it is likely to substitute for these services with technological or other alternatives, for example, electronic surveillance systems, household appliances, television broadcasting, computation services, etc. A great leap in productivity and output in substitutable personal services is provided by technological innovation.

Progressive services: Progressive services can show phenomenal productivity growth and cost drops initially. This is due to the relatively significant impact of the first technology-intensive component, for example, communication systems, computer systems, automation, flexible manufacturing system, etc. In this class, productivity growth is self-extinguishing since, in due course of a productive period, the relative contribution of the advanced component exceeds that of the first component. The stagnant nature of the upgraded component dampens productivity growth.

A service system is an arrangement of technology and organizational networks intended to deliver services that satisfy the needs, wants, or aspirations of customers. Some common service systems which we observe around us for day-to-day activities are as follows.

1.4.1 Flexible Assembly Systems

Assembly is the capstone process for product realization where constituent parts and subassemblies are integrated to form the final products. As product diversity increases due to the shift from mass production to mass customization, assembly systems must be intended and operated to handle such high variety.

1.4.2 Fault-tolerant Machining Systems

Fault tolerance refers to the property of a system (computer, network, cloud cluster, etc.) to remain functioning without interruption when one or more of its units fail. The objective of developing a fault-tolerant system is to prevent disruptions arising from a single point of failure, safeguarding the high availability. Fault-tolerant systems use backup units that automatically take the place of failed units, ensuring no loss of service. It includes hardware systems, software system, power sources,

1.4.3 Production Systems

The methods, procedure, or arrangement governs with a set of instructions, which includes all functions required to collect the inputs, process, or reprocess the inputs and produce the marketable output. It also includes the artificial intelligence mechanism necessary to follow those instructions as the system responds to states of the demand.

1.4.4 Communication Systems

The communication system is a collection of specific communications networks, transmission systems, relay stations, tributary stations, and data terminal equipment (DTE), usually proficient in inter-connection and inter-operation to create a unified system in whole. The components of a communications system serve a common purpose, are technically compatible, use common procedures, respond to controls, and operate in union.

1.4.5 Computer System

A computer system is customary of unified devices that input, output, process, and store data and information. In general, computer systems are built around at least one digital processing device. There are five main hardware components in a computer system: input, processing, storage, output and communication devices.

1.4.6 Traffic System

Traffic systems include sensors to measure traffic flows and automatic inter-connected, guidance systems to manage traffic.

From the aforesaid different types of service systems, it is noticeable that the queueing system is applicable in all sectors and problems, and customers and servers may be human beings, data, machines, signals, etc. The congestion, blocking, and delay can be observed in any service system at any process and need to investigate and schedule properly for getting satisfactory services. It gives the notion of threshold-based service facility, additional service facility on rent, synchronized services, time-sharing, processor-sharing, distributive services, buffer, automation, robotics, breakdowns, replacement, repair, interruption, inspection, interference, bulk or batch, fork-join queue, optimal design, trade-off between services and waiting, etc. for investigation and analysis purposes.

1.5 Characteristics of Queueing Systems

Each queueing modeling based service system has its own unique set of characteristics. However, all such systems have the following essential components.

1.5.1 The Population of Prospective Customers

The population of prospective customers is an input source from which the customers, who require service from one or more servers depending on types of the service system, are generated. The most important parameter of the population is its size, which is the cardinality of the prospective customers, that might need service at some time instant or the other. Generally, the size is classified as finite or infinite. In practice, when the population size is too large to affect the arrival rate of the customers is known as infinite, otherwise finite.

1.5.2 Input or Arrival Pattern

The arrival pattern delineates how the customers arrive and join the service system, *i.e.* the statistical order by which the customers are generated over time. In general, it is not possible in advance to observe or identify the actual amount of arriving customers in the queue for service. Further, it is also noticed that different inter-arrival times (time between two successive arrivals) are not the same always. Therefore, the inter-arrival time can not be treated as a constant, but only as a random variable. In queueing literature, it is assumed that the inter-arrival times are independent random

variates that follow some specific probability distribution. This probability distribution may be approximated by collecting substantial data set of arriving customers. However, over a considerable time interval, the mean arrival rate remains constant and denoted by λ. The meantime between arriving customers is the reciprocal of λ, *i.e.* $\left(\frac{1}{\lambda}\right)$. Suppose that the probability density function (pdf) of the inter-arrival time T is $f(t)$. Then, mathematically, the meantime between arrivals is represented as

$$\frac{1}{\lambda} = \int_0^\infty t f(t) dt \tag{1.1}$$

Hence, the mean arrival rate (λ) can be calculated as

$$\lambda = \frac{1}{\int_0^\infty t f(t) dt} \tag{1.2}$$

1.5.3 System Capacity

The system capacity is defined as the maximum permissible number of customers that any waiting line can contain. It can be classified as a finite queue or infinite queue if the total number of customers in the waiting line is finite or infinite, respectively. When the system capacity is full, newly arrived customers do not enter the system at that moment and are treated as lost customers forever.

1.5.4 Service Discipline

The process of selecting a customer from the waiting queue for the service is termed as service discipline. In general, the customers are served following the First Come First Served (FCFS) service discipline. Depending on the service system, there are some other service disciplines also which are defined as Last Come First Served (LCFS), First Come Last Served (FCLS), Service in Random Order (SIRO), et cetera. Sometimes, the arriving customers are selected on a specific priority basis, *i.e.* Priority in Selection, such as the expected waiting time in the system, adequate service time, cost incurred due to waiting customer, and may more.

1.5.5 Service Mechanism

The service mechanism mainly defines the number of customers who get the service in one time-period, *i.e.* a service rate or the service time, which is defined as the time required to complete the service for a customer. The service time is also random variates, which follows some probability distribution with some specified parameter(s).

The distribution can be approximated, or the parameters can be estimated from the data set of provided services.

In a service system, there is one or more than one server that may be arranged in series or parallel. If the servers are working in parallel, an arriving customer may choose any of the idle servers, if any, on his arrival instant. If all the servers are busy providing the service to the customers then the newly arriving customer will have to wait up to a specific time in the queue and will be selected for service according to the service discipline when any of the servers are idle. If the servers provide the service in series, then the customers get the service from the sequence of service channels in tandem. In that case, the customer passes from one service channel to another following a definite rule, and the service is completed as soon as the customer moves through all the service channels allocated in series. In the tandem queue, customers may be blocked or starved.

1.5.6 State of the System

The state of the system denotes the total number of customers in the system at any time instant t, which includes those customers who are in the queue and even present in ongoing service. During the transitions between states of the system, states initially have a probability distribution $P_n(t)$, which is time-dependent. But in a sufficiently large interval of time, *i.e.* after infinite time has elapsed ($t \to \infty$) the probability distribution may be transformed to a distribution P_n which is independent of time. At that moment, all the state probabilities are independent of the initial condition. In other words, one can assert that the system is in equilibrium (statistically stable) or steady-state is achieved. The distribution of all such state probabilities is described as the steady-state probability distribution. The state of the system and its probability distribution provide the platform for transient or steady-state analysis. In practice, to demonstrate the dynamical behavior of any service system, we perform the transient analysis.

1.6 Some Important Random Processes

In this section, we delineate a brief description of some significant random processes used in queueing modeling of service systems. The random process is a probabilistic process that is used to characterize random events. The definition and properties of these processes help in understanding the events involved in modeling of the studied service systems and developing governing precedence relations between states.

These processes also involve in the computation and analysis of performance characteristics of service systems.

1.6.1 Stochastic Process

Consider a random experiment with a sample space S. If a time function $X(t,s); \forall t \in T$, index set, is assigned to each outcome (sample point) $s \in S$, then the family of all such functions, represented by $\{X(t,s), s \in S, t \in T\}$, is called a stochastic process. In other words, a stochastic process is known as a collection of random variates that are indexed by multiple mathematical sets, and each random variate is uniquely associated with an element in the set. Depending on index set and sample point, the stochastic process can be classified as:

- Discrete-time, discrete-state stochastic process
- Discrete-time, continuous-state stochastic process
- Continuous-time, discrete state stochastic process
- Continuous-time, continuous state stochastic process

A stochastic process is termed as a deterministic stochastic process if the value of future observations of any sample function can be predicted precisely from the past observations or past values.

If the value of the future observations of any sample function cannot be predicted precisely from previous observations or previous values, then the stochastic process is called a non-deterministic stochastic process.

1.6.2 Stationary Process

A stationary process is a random process whose probability distribution at a fixed time instant is the same for all time points. As a result, statistical parameters such as mean, variance, and the moments, if they exist, do not change over time for the stationary process.

1.6.3 Markov Process

A Markov process is a random process that holds the Markovian property (memoryless property), which is defined as the future behavior of the process depends only on the current observation or current state, and not on the states in the past. Mathematically, a random process $\{X(t), t \in T\}$ is called a Markov process if

$$\begin{aligned} P[X(t_{n+1}) \leq x_{n+1} | X(t_n) = x_n, X(t_{n-1}) = x_{n-1}, \cdots, X(t_0) = x_0] \\ = P[X(t_{n+1}) \leq x_{n+1} | X(t_n) = x_n] \end{aligned} \tag{1.3}$$

whenever $t_0 < t_1 < \cdots < t_n < t_{n+1}$.

1.6.4 Markov Chain

A discrete-state Markov process is called a Markov chain. Thus, a discrete-parameter Markov chain is defined as a set of random variables $\{X_n, n \geq 0\}$ with the memoryless property. *i.e.* $P(X_{n+1} = y|X_n = x, X_{n-1} = x_{n-1}, \cdots, X_0 = x_0) = P(X_{n+1} = y|X_n = x)$. The set of all possible values of the random variables X_i form a countable set Ω called the state space of the process.

1.6.4.1 Discrete-Time Markov Chain

A Markov chain is called a discrete-time Markov chain if the random variables X_i take values from the discrete time space. Therefore, any change in the system can be only possible when the change has been made between these discrete time values.

1.6.4.2 Continuous-Time Markov Chain

If the changes in the system can be made at any time instant within a continuous time interval, then the Markov chain is defined as a continuous-time Markov chain.

1.6.5 Bernoulli Process

If we can define a countably infinite sequence of random variables $\{X_n; n = 1, 2, 3, \ldots\}$ associated with a sequence of Bernoulli trials as

$$X_n = \begin{cases} 1; & \text{if the } n^{th} \text{ Bernoulli trial yields a success} \\ 0; & \text{if the } n^{th} \text{ Bernoulli trial yields a failure} \end{cases} \tag{1.4}$$

with probabilities

$$P(X_n = 1) = p \text{ and } P(X_n = 0) = q \tag{1.5}$$

where $0 < p < 1$ and $p + q = 1$. Then, the collection of such random variables $\{X_n; n \geq 1\}$ is defined as a Bernoulli process.

1.6.6 Binomial Process

Let $\{Z_n; n = 1,2,3,\ldots\}$ be a Bernoulli process and X_n denote the number of successes in the first n independent Bernoulli trials *i.e.*

$$X_n = Z_1 + Z_2 + \cdots + Z_n \tag{1.6}$$

Then $\{X_n; n = 1,2,3,\ldots\}$ is defined as a binomial process.

1.6.7 Counting Process

A random process $\{X(t); t \geq 0\}$ is defined as a counting process if $X(t)$ denotes the total number of events that have occurred in the interval $(0,t]$ and must satisfy the following conditions

(i) $X(t) \geq 0$ and $X(0) = 0$
(ii) $X(t)$ takes only integer values
(iii) $X(t)$ is monotonic non-decreasing, *i.e.* $X(s) \leq X(t)$ if $s < t$
(iv) $X(t) - X(s)$ is equal to the number of events that have occurred in the interval (s,t)

Moreover, a counting process $X(t)$ is said to have independent increments if the events that occur in disjoint time intervals are independent to each other.

1.6.8 Poisson Process

A counting process $X(t)$, which characterizes the number of occurrences of a specific event in the interval $(0,t]$, is called a Poisson process with mean rate of occurrences λ if it satisfies the following postulates

(i) $X(0) = 0$
(ii) $X(t)$ has independent and stationary increments
(iii) $P[\{X(t+\Delta t) - X(t)\} = 1] = \lambda \Delta t + o(\Delta t)$
(iv) $P[\{X(t+\Delta t) - X(t)\} \geq 2] = o(\Delta t)$

Specifically, if λ is constant, Poisson process is known as homogeneous Poisson process. While the Poisson process for which λ is a function of time, are characterized as non-homogeneous Poisson process.

1.6.9 Renewal Process

Suppose X_n be the interval between the $(n-1)^{th}$ count and n^{th} count of a counting process $\{X(t); t \geq 0\}$. Let $\{X_n; n = 1,2,\ldots\}$ be a sequence of non-negative and

independently & identically distributed (iid) random variable with mean μ. Then $\{X(t); t \geq 0\}$ is defined as a renewal process.

1.6.10 Birth and Death Process

Mathematically, a continuous-time Markov chain $X(t)$ with state space $\Theta = \{0, 1, 2, \ldots\}$ is called a birth and death process if the following axioms are satisfied

$$P[X(t+h) - X(t) = k | X(t) = n] = \begin{cases} \lambda_n h + o(h); & k = 1,\ n \geq 0 \\ \mu_n h + o(h); & k = -1,\ n \geq 1 \\ 1 - (\lambda_n + \mu_n)h + o(h) & k = 0,\ n \geq 1 \\ 0; & \text{otherwise} \end{cases} \tag{1.7}$$

where λ_0, λ_1, λ_2, ... are positive constants called birth rates, μ_1, μ_2, ... are positive constants called death rates and

$$\lim_{h \to 0} \frac{o(h)}{h} = 0 \tag{1.8}$$

1.6.11 Quasi-Birth and Death Process

A Markov chain with the state-space

$$\Theta = \big\{(j, n); 1 \leq j \leq n_j \ \& \ n \geq 0\big\} \tag{1.9}$$

is known as quasi-birth and death process, where the state space is divided into different levels and phases, *s.t.* the level n has n_j phases for each n. In a quasi-birth and death (QBD) process, the transitions are allowed between the adjacent states only. Therefore, a QBD process can be observed as a generator matrix in following way

$$Q = \begin{bmatrix} \mathbf{A}_0 & \mathbf{B}_0 & \mathbf{0} & \mathbf{0} & \cdots \\ \mathbf{C}_1 & \mathbf{A}_1 & \mathbf{B}_1 & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{C}_2 & \mathbf{A}_2 & \mathbf{B}_2 & \cdots \\ \mathbf{0} & \mathbf{0} & \mathbf{C}_3 & \mathbf{A}_3 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

where, each sub-matrix $\mathbf{B}_n$ can be obtained by the transitions from n^{th} level to $(n+1)^{th}$ level for $n \geq 0$. Similarly, the sub-matrices $\mathbf{C}_n$ can be generated by balancing the transitions from n^{th} level to $(n-1)^{th}$ level for $n \geq 1$ while the diagonal sub-matrices $\mathbf{C}_n$ are encoded within the n^{th} level for $n \geq 0$.

1.6.12 Chapman-Kolmogorov Theorem

Suppose $\{X_n; n \geq 0\}$ be a homogeneous Markov chain with transition matrix $\mathbf{P} = [p_{ij}]$ obtained using the state probabilities and n-step transition probability matrix $\mathbf{P}^{(n)} = [p_{ij}^{(n)}]$, where

$$p_{ij}^{(n)} = \Pr[X_n = j | X_0 = i] \quad \text{and} \quad p_{ij}^{(1)} = p_{ij} \tag{1.10}$$

Then

(i) $\mathbf{P}^{(n+m)} = \mathbf{P}^{(n)}\mathbf{P}^{(m)}$

(ii) $\mathbf{P}^{(n)} = \mathbf{P}^n$, *i.e.* the n^{th}-step transition matrix is equal to the n^{th} power of the one-step transition matrix $\mathbf{P}$.

1.6.13 Chapman-Kolmogorov Equations

Utilizing the memoryless property of the Poisson process, the developed equations

$$P_{i,j}(t+s) = \sum_{n=0}^{\infty} P_{i,n}(t) P_{n,j}(s) \tag{1.11}$$

are defined as the Chapman-Kolmogorov differential-difference equations. These equations states that in order to move towards from state i to j at time instant t, $X(t)$ moves to state k in time t and then from k to j in the remaining time s.

1.7 Finite and Infinite Queueing Systems

In literature, the queueing models are classified based on the capacity of the service system and the size of the population of prospective customers. If the size (capacity) of the service system is finite, then it is called a finite capacity service system otherwise infinite capacity service system. The population of potential customers in the service system is also defined as the calling population of prospective customers. The population of prospective arrivals may be finite or infinite, depending on the different environments of service systems. In a finite population service system, the arrival pattern of potential customers depends on the state of the system which generates more complicated and complex computations. In other words, the arrival rate of customers is influenced by the population in the system, which is generally be viewed as a closed system. Whereas, in the case of infinite population service systems, incoming customers are independent of the cardinality and states of other customers, as a result of which it is mathematically described as a tractable model. Usually, these are characterized as an open system. In such queueing models, each

state defines the different possibilities and connected to the other states in different ways. The queueing-based state-transition diagrams of basic finite capacity and finite population queueing models are pictured in next subsections.

1.7.1 Finite Capacity Queueing System

In classical queueing theory, the finite capacity queueing models are classified in the following two categories depending on number of service providers

1.7.1.1 Single Server Finite Capacity Queueing Model

In this queueing scenario, the service system can permit or accommodate only a finite number of customers (K) in the system. If a customer arrives and the queue capacity is full, then the customer has to leave the system without waiting in queue. Therefore, on the basis of this assumption, the mean arrival rate is

$$\lambda_n = \begin{cases} \lambda; & n = 0,1,2,\ldots,K-1 \\ 0; & n = K,K+1,K+2,\ldots \end{cases} \tag{1.12}$$

and the mean service rate is

$$\mu_n = \mu; \quad n = 1,2,3,\ldots \tag{1.13}$$

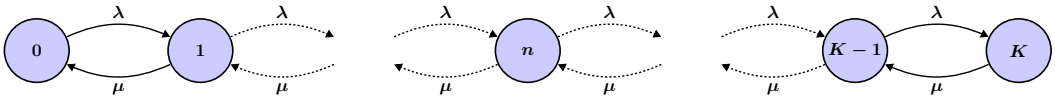

Figure 1.2: State transition diagram of a single server finite capacity queueing based service system.

Now, by balancing the input and output flow between the transitions in Fig. 1.2 and employing the postulates of the Poisson process, the differential-difference equations over the time t of the governing queueing modeling based service system are derived as

$$\frac{dP_0(t)}{dt} = -\lambda P_0(t) + \mu P_1(t) \tag{1.14}$$

$$\frac{dP_n(t)}{dt} = -(\lambda+\mu)P_n(t) + \lambda P_{n-1}(t) + \mu P_{n+1}(t);\ 1 \le n \le K-1 \tag{1.15}$$

$$\frac{dP_K(t)}{dt} = -\mu P_K(t) + \lambda P_{K-1}(t) \tag{1.16}$$

In a long run, *i.e.* in equilibrium (as $t \to \infty$), all the state probabilities attain steady-state as

$$\lim_{t\to\infty} P_n(t) = P_n;\ n = 0,1,2,\ldots,K \tag{1.17}$$

Using the recursive algorithm, the solution of above mentioned system of differential equations is obtained as

$$P_n = \rho^n P_0;\ n = 0,1,2,\ldots,K \tag{1.18}$$

where, $\rho = \left(\frac{\lambda}{\mu}\right)$ and $P_0 = \frac{(1-\rho)}{(1-\rho^{K+1})}; \rho \neq 1$. Thus,

$$P_n = \begin{cases} \frac{\rho^n(1-\rho)}{(1-\rho^{K+1})}; & n = 0,1,2,\ldots,K \ \&\ \lambda \neq \mu \\ 0; & n = K+1, K+2,\ldots \end{cases} \tag{1.19}$$

When traffic intensity is equal to one, *i.e.* $\lambda = \mu$, the probability distribution is defined as

$$P_n = \begin{cases} \frac{1}{(K+1)}; & n = 0,1,2,\ldots,K \\ 0; & n = K+1, K+2,\ldots \end{cases} \tag{1.20}$$

1.7.1.2 Multiple Server Finite Capacity Queueing Model

The present queueing model comprises with R servers, where $R > 1$, wherein the system can accommodate at most K customers in waiting. Thus the mean arrival rate is represented as

$$\lambda_n = \begin{cases} \lambda; & n = 0,1,2,\ldots,K-1 \\ 0; & n = K, K+1,\ldots \end{cases} \tag{1.21}$$

and the mean service rate is

$$\mu_n = \begin{cases} n\mu; & n = 0,1,2,\ldots,R-1 \\ R\mu; & n = R, R+1,\ldots \end{cases} \tag{1.22}$$

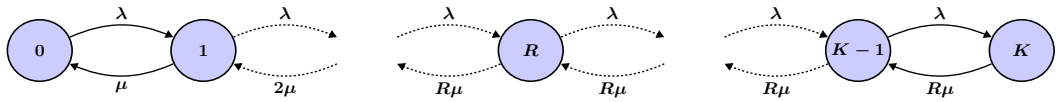

Figure 1.3: State transition diagram of a multiple server finite capacity queueing based service system.

Now, using the state transition diagram in Fig. 1.3, the governing Chapman-Kolmogorov differential-difference equations in transient state are depicted as

$$\frac{dP_0(t)}{dt} = -\lambda P_0(t) + \mu P_1(t) \tag{1.23}$$

$$\frac{dP_n(t)}{dt} = -(\lambda + n\mu)P_n(t) + \lambda P_{n-1}(t) + (n+1)\mu P_{n+1}(t);\ 1 \leq n \leq R-1 \tag{1.24}$$

$$\frac{dP_n(t)}{dt} = -(\lambda + R\mu)P_n(t) + \lambda P_{n-1}(t) + R\mu P_{n+1}(t);\ R \leq n \leq K-1 \tag{1.25}$$

$$\frac{dP_K(t)}{dt} = -R\mu P_K(t) + \lambda P_{K-1}(t) \tag{1.26}$$

In the equilibrium state, the probability distribution of the state of the system in steady-state is derived as

$$P_n = \begin{cases} \dfrac{\rho^n}{n!}; & n = 0,1,2,\ldots,R-1 \\ \dfrac{\rho^n}{R!R^{(n-R)}}; & n = R,R+1,\ldots,K \\ 0; & n = K+1,K+2,\ldots \end{cases} \tag{1.27}$$

Employing the normalization condition, $\sum_{n=0}^{K} P_n = 1$, we get

$$P_0 = \left[1 + + \sum_{n=1}^{R-1} \frac{\rho^n}{n!} + \rho^R \frac{\left\{ 1 - \left(\frac{\rho}{R}\right)^{K-R+1} \right\}}{R!\left(1 - \frac{\rho}{R}\right)} \right]^{-1} ; \quad \left(\frac{\rho}{R}\right) \neq 1 \tag{1.28}$$

1.7.2 Finite Population Queueing System

In this section, we analyze a particular type of queueing model based on the finite population of the service system. It is also known as machine repair model or machine interence problem in the queueing literature. For the modeling purpose, suppose a machining system has K machines and has R repairmen working in parallel. Let, the time-to-failure of a machine is identically and exponentially distributed random variable with parameter λ, and the time-to-repair by each of the repairman for the failed machine is exponentially distributed with parameter μ. Suppose there are

n failed machines in the repair facility, *i.e.* either under repair or waiting to be repaired at any instant, then $(K-n)$ are functioning well in the system. Therefore, the effective failure rate is $\lambda_n = (K-n)\lambda$; $n = 0,1,2,\ldots,K-1$, and the effective repair rate μ_n is defined as

$$\mu_n = \begin{cases} n\mu; & n = 1,2,\ldots,R-1 \\ R\mu; & n = R,R+1,\ldots,K \end{cases} \tag{1.29}$$

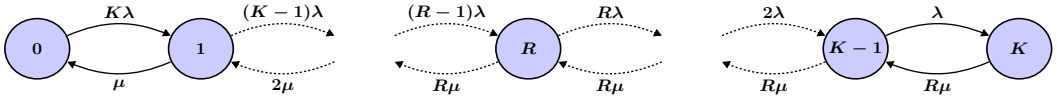

Figure 1.4: State transition diagram of a basic machine repair model.

The differential-difference equations in the transient-state can be formulated by balancing the input and output flow in the Fig. 1.4 as

$$\frac{dP_0(t)}{dt} = -K\lambda P_0(t) + \mu P_1(t) \tag{1.30}$$

$$\begin{aligned} \frac{dP_n(t)}{dt} = -((K-n)\lambda + n\mu)P_n(t) + (K-n+1)\lambda P_{n-1}(t) \\ +(n+1)\mu P_{n+1}(t); 1 \le n \le R-1 \end{aligned} \tag{1.31}$$

$$\begin{aligned} \frac{dP_n(t)}{dt} = -((K-n)\lambda + R\mu)P_n(t) + (K-n+1)\lambda P_{n-1}(t) + R\mu P_{n+1}(t); \\ R \le n \le K-1 \end{aligned} \tag{1.32}$$

$$\frac{dP_K(t)}{dt} = -R\mu P_K(t) + \lambda P_{K-1}(t) \tag{1.33}$$

Using the concept and results of birth and death process, the queue-size distribution in the steady-state (for a long run as $t \to \infty$) is computed as

$$P_n = \begin{cases} \binom{K}{n}\rho^n P_0; & n = 1,2,\ldots,R-1 \\ \dfrac{K!}{(K-n)!R!R^{n-R}}\rho^n P_0; & n = R,R+1,\ldots,K \\ 0; & \text{otherwise} \end{cases} \tag{1.34}$$

where

$$P_0 = \left[\sum_{n=0}^{R-1}\binom{K}{n}\rho^n + \sum_{n=R}^{K}\frac{K!\rho^n}{(K-n)!R!R^{n-R}}\right]^{-1} \tag{1.35}$$

1.8 Performance Measures

In practice, many generic performance measures are utilized to show the capability of working and the quality performance of a service system. Using the fundamental law of queueing modeling, system designers and scholars can compute these system performance measures for a given service system and utilize it in decision making. These performance measures are quite interrelated, and each assumes increased importance in a particular context. Next, we provide some primary system performance measures of service systems, which is indicative of its competitive status in the real world congestion problems. The performance measures of a queueing based service system is defined in terms of properties of one or more of the following stochastic (random) processes

$$\{N(t); t \geq 0\},\ \{N_q(t); t \geq 0\},\ \{W_j; j \in N\},\ \text{and}\ \{D_j; j \in N\} \tag{1.36}$$

We are concerned primarily with the long-run behavior ($t \to \infty$), where the performance measures are defined as limits that are averaged over either time or customers. Next, we define some generic performance measures for a queueing based service system, which are divided into two categories namely, customer-oriented, and system-oriented.

1.8.1 Customer-Oriented

From the customer's point of view, the performance of any service system is usually examined in terms of several constraints such as computation time, how much quicker the better results are achieved, et cetera. Some of them are defined as follows

- **Waiting Time:**

  It is defined as a period for which customers have to wait to get start their service in the system. The waiting time of the customer is further classified in two classes

  - **Average waiting time in the system:**

    It is defined as the overall waiting time of customers that they spend in the system.

$$W_S = \lim_{n \to \infty} \frac{1}{n} \sum_{j=1}^{n} W_{S_j} \tag{1.37}$$

– **Average waiting time in the queue:**

It is defined as the waiting time of customers that they spend in the waiting queue to get begin his service.

$$W_Q = \lim_{n\to\infty} \frac{1}{n} \sum_{j=1}^{n} W_{Q_j} \tag{1.38}$$

- **Elapsed Time**

A customer is interested in the elapsed time between the states of service initialization and service completion. In the breakdown state of the system, the elapsed time is also referred to as response time of the server.

- **Turnaround Time**

Turnaround time is the elapsed time for the batch arrival and batch service queueing based service systems.

1.8.2 System-Oriented

The customer specifically focuses on individual service progress through the service system, while the system itself makes it more useful than collective behavior and adopts a global view of situations. System analysts are more interested in the following indices

- **Expected length of the service system**

The expected length of the service system is defined as the average number of waiting customers in the system.

$$L_S = \lim_{t\to\infty} \frac{1}{t} \int_0^t N(x)dx \tag{1.39}$$

- **Expected length of the queue (mean queue length)**

Similarly, the expected length of the queue is defined as the average number of waiting customers in the queue.

$$L_Q = \lim_{t\to\infty} \frac{1}{t} \int_0^t N_q(x)dx \tag{1.40}$$

- **Average number of customers in service**

It is defined as the mean number of those customers for which the server provides the service or who present in the ongoing service.

$$L_O = \lim_{t\to\infty} \frac{1}{t} \int_0^t N_s(x)dx \tag{1.41}$$

- **Server utilization**

  The server utilization is defined as a fraction of time during which the server is busy.

$$U_S = \frac{L_O}{R} \tag{1.42}$$

- **Throughput of the system**

  The throughput of the system (τ_p) is defined as the mean number of customers whose job is completed in a unit time interval, *i.e.* the effective departure rate. In equilibrium condition, the throughput of the system is equal to the effective arrival rate

$$\tau_p = \lambda_{\text{eff}} \tag{1.43}$$

- **System Reliability**

  The reliability of the system is the probability that the system will work properly without interruption over the time interval $[0,t)$. Mathematically, if $f(t)$ is defined as the probability density function of the time-to-failure (T) of the system, then the reliability of the system is defined as

$$R_Y(t) = \int_t^\infty f(t)dt \tag{1.44}$$

- **System Availability**

  The system availability is defined as the probability that the system is working properly at time instant t.

- **Maintainability or Serviceability**

  The maintainability of the system is the probability of performing a successful repair action. In other words, the rate by which the system can be repaired/maintained.

In the above-defined closed expressions, the measures L_S, L_Q, L_O, and U_S are the averages over time, whereas the measures W_S, and W_Q are the averages over customers, and R be the number of identical servers in parallel in the service system.

1.9 Solution Techniques

When the service system is in the design phase, the sensitivity analysis of the design parameters is needed with the transient solution; else, if service system is in equilibrium, the optimal analysis is required with steady-state solution. The transient or steady-state solution of governing system of differential-difference equations can be computed using following numerical and analytical solution techniques.

1.9.1 Transient Solution

The transient solution of the system of differential equations is the solution, which is obtained by combining all the boundary conditions and the particular solution of the problem. In the transient state, the value of the system parameters is influenced with the passes of time. There are some standard solution techniques, which provide an efficient solution of highly nonlinear and complex queueing problems.

1.9.1.1 Laplace Transformation

The Laplace transformation technique is extensively used in queueing modeling of service systems for deriving and analyzing solutions of systems of differential equations. Mathematically, for the real-valued function $f(t)$, Laplace transformation in terms of variable s is

$$F(s) = \int_0^\infty f(t)e^{-st}dt \tag{1.45}$$

where the function $f(t)$ is integrable and defined for all $t \geq 0$. With Laplace transformation, the system of differential equations changes to the system of linear equations. Using key formulae and employing the inverse-Laplace transformation, the transient solution of queueing models can easily be determined.

1.9.1.2 Runge-Kutta Method

Let us consider a initial value problem

$$\frac{dy(t)}{dt} = f(t,y); \quad y(t_0) = y_0 \tag{1.46}$$

where $y(t)$ is a function of variable t, which we would like to approximate. Suppose h be the step-size, then

$$y_{n+1} = y_n + \frac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4) \tag{1.47}$$

$$s.t. \quad t_{n+1} = t_n + h \tag{1.48}$$

where

$$K_1 = h f(t_n, y_n) \tag{1.49}$$

$$K_2 = h f\left(t_n + \frac{h}{2}, y_n + \frac{K_1}{2}\right) \tag{1.50}$$

$$K_3 = h f\left(t_n + \frac{h}{2}, y_n + \frac{K_2}{2}\right) \tag{1.51}$$

$$K_4 = h f(t_n + h, y_n + K_3) \tag{1.52}$$

Using the iterative procedure of the Runge-Kutta method of fourth-order, we can obtain the transient solution of the governing system of differential equations of the queueing modeling based service system to examine the dynamical behavior at time instant t.

1.9.2 Steady-State Solution

If, in a sufficiently large interval of time, the states of the system are independent of the initial conditions *i.e.* the characteristics of the system remains unchanged concerning lower and higher values of time t. Then, the system is called in the steady-state. Usually, the steady-state of the system is obtained after some time-period of the initialization of the process. In queueing literature, there are several techniques to calculate the desired solution of queueing problems in a steady-state.

1.9.2.1 Successive Over-Relaxation Method

The successive over-relaxation (SOR) method is extensively used in queueing literature as a variant of the Gauss-Seidel method to compute the solution of a system of linear equations numerically. The convergence rate of SOR is comparatively higher than the other solution techniques. These methods are designed for higher-order computation by human-made calculators. The SOR method is used to solve a system of linear equations $\mathbf{AX} = \mathbf{B}$ that is derived by extrapolating the standard Gauss-Seidel method.

The coefficient matrix $\mathbf{A}$ in the matrix equation can be decomposed in terms of the diagonal pattern $\mathbf{D}$, and strictly upper & lower triangular forms U, L respectively

as

$$\mathbf{A} = (\mathbf{D} + \mathbf{L} + \mathbf{U}) \tag{1.53}$$

In terms of weighted average (relaxation factor), the matrix equation can be transformed as

$$(\mathbf{D} + \omega\mathbf{L})\mathbf{X} = \omega\mathbf{B} - [\omega\mathbf{U} + (\omega - 1)\mathbf{D}]\,\mathbf{X}; \quad \omega > 1 \tag{1.54}$$

and for the $(n+1)^{th}$ approximation, the expression is converted into

$$\mathbf{X}^{(n+1)} = (\mathbf{D} + \omega\mathbf{L})^{-1}\left(\omega\mathbf{B} - [\omega\mathbf{U} + (\omega - 1)\mathbf{D}]\,\mathbf{X}^{(n)}\right) = \mathbf{L}_{\omega}\mathbf{A}^{(n)} + \mathbf{e} \tag{1.55}$$

1.9.2.2 Gauss Elimination Method

In this method, the unknowns are eliminated by combining all the equations such that the n equations with n unknowns are reduced to an equivalent upper triangular system, which is further solved by employing the backward substitution method. Now, consider a system of linear equations with three equations and three unknowns

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1 \tag{1.56}$$
$$a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = b_2 \tag{1.57}$$
$$a_{31}x_1 + a_{23}x_2 + a_{33}x_3 = b_3 \tag{1.58}$$

Now, using the appropriate substitution and recursive algorithm, we get the following results

$$a_{11}^{(1)}x_1 + a_{12}^{(1)}x_2 + a_{13}^{(1)}x_3 = b_1^{(1)} \tag{1.59}$$
$$a_{22}^{(2)}x_2 + a_{23}^{(2)}x_3 = b_2^{(2)} \tag{1.60}$$
$$a_{33}^{(3)}x_3 = b_3^{(3)} \tag{1.61}$$

The obtained system is in the form of an upper triangular matrix, which can be solved efficiently using the backward substitution method.

1.9.2.3 Matrix Inverse Method

We can also find the solution of the system of linear equations

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1 \tag{1.62}$$

$$a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = b_2 \tag{1.63}$$

$$a_{31}x_1 + a_{23}x_2 + a_{33}x_3 = b_3 \tag{1.64}$$

by employing the matrix inverse method (only if the coefficient matrix is non-singular). Suppose $\mathbf{A} = [a_{ij}]$ be the coefficient matrix obtained by the above system of equations. The system of linear equations can easily be expressed in matrix form as

$$\mathbf{AX} = \mathbf{B} \tag{1.65}$$

where $\mathbf{X}$ is the vector of all unknown variables, and $\mathbf{B}$ is the constant matrix.

If $\mathbf{A}$ is a non-singular then on pre-multiplying the matrix equation by $\mathbf{A}^{-1}$ both sides, we get

$$\begin{aligned} \left(\mathbf{A}^{-1}\right)\mathbf{AX} &= \left(\mathbf{A}^{-1}\right)\mathbf{B} \\ \left[\mathbf{A}^{-1}\mathbf{A}\right]\mathbf{X} &= \left(\mathbf{A}^{-1}\right)\mathbf{B} \\ \mathbf{IX} &= \left(\mathbf{A}^{-1}\right)\mathbf{B} \end{aligned}$$

Therefore, we get the relation

$$\mathbf{X} = \left(\mathbf{A}^{-1}\right)\mathbf{B} \tag{1.66}$$

1.9.2.4 Newton's Method

In optimization theory, Newton's method is an iterative method to determine the roots of a twice-differentiable function. These roots are also referred as the stationary points. For the mathematical formulation, suppose g be a twice-differentiable function, and the second-order Taylor series expansion of the function g around x_n is

$$g(x_n + h) = g(x_n) + h\left\{\frac{d}{dx}g(x)\right\}_{x=x_n} + \frac{h^2}{2!}\left\{\frac{d^2}{dx^2}g(x)\right\}_{x=x_n} \tag{1.67}$$

Initially, we require a small value of h, such that $(x_n + h)$ is a stationary point of the function g. Now, using the Taylor expansion as an approximation, we get

$$\begin{aligned} 0 &= \frac{d}{dh}\left(g(x_n) + h\left\{\frac{d}{dx}g(x)\right\}_{x=x_n} + \frac{h^2}{2!}\left\{\frac{d^2}{dx^2}g(x)\right\}_{x=x_n}\right) \\ &= \left\{\frac{d}{dx}g(x)\right\}_{x=x_n} + h\left\{\frac{d^2}{dx^2}g(x)\right\}_{x=x_n} \end{aligned}$$

Therefore, we get

$$h = -\left\{\frac{\frac{d}{dx}g(x)}{\frac{d^2}{dx^2}g(x)}\right\}_{x=x_n} \tag{1.68}$$

provided the approximation by the Taylor series expansion is reasonably accurate, then an increasing value of h should yield very close to the stationary points of the function g. If the function g satisfies all the assumptions, then

$$x_1 = x_0 - \left\{\frac{g(x)}{\frac{d}{dx}g(x)}\right\}_{x=x_0} \tag{1.69}$$

and the whole process is repeated as

$$x_{n+1} = x_n - \left\{\frac{g(x)}{\frac{d}{dx}g(x)}\right\}_{x=x_n} \tag{1.70}$$

until a sufficiently more accurate value is achieved.

1.10 Optimization Techniques

In the modernization of queueing-based engineering and industrial problems, the evolutionary algorithms, and nature-inspired optimization techniques are broadly used because they can determine the solutions of highly complex problems in a significant manner. The primary deriving force to develop nature-inspired optimization techniques is that the traditional linear and nonlinear optimization techniques are unable to search the optimal solution for such complex issues efficiently. Most of the heuristic and metaheuristic algorithms have been inspired by the physical and biological behavior of animals and used Darwinṡ theory of survival of fittest. Over the last decade, these optimization algorithms have been effectively applied in many real-time decision-making problems such as queueing problems, telephony, computer and communication systems, inventory and production systems, scheduling

problems, Combinatorial & numerical optimization, et cetera. Some major heuristic and metaheuristic optimization techniques that are used throughout the research work are briefly explained in the next subsections.

1.10.1 Quasi-Newton Method

The main drawback of Newton's method is to estimate the second-order derivative matrix for which the user has to employ computationally difficult approach. Newton's approach can be applied when first derivatives exist. Therefore, to overcome this limitation, a quasi-Newton method is prospectively developed. Usually, it is an arduous task to calculate the analytic expressions of several system design parameters associated with the cost optimization problem of a queueing modeling based service system. Thus, we calculate the solution of optimization problems numerically by implementing non-classical optimization techniques. One of the most popular optimization techniques is the quasi-Newton algorithm, which is broadly used in queueing literature. The quasi-Newton method makes possible to find the optimal cost of the system under optimal operating conditions. We globally search the values of system design parameters until the optimal value of cost optimization problem is achieved. We initialize the values of decision parameters in a vector form Ω_0. In the next phase, we compute the gradients of the cost function numerically with respect to decision parameters and hence, the Hessian matrix is formulated. The quasi-Newton method finds the global minimum of expected cost function in the search domain. The pseudo-code of quasi-Newton (QN) method is given as follows

Quasi-Newton method: Pseudo code

Input: Input parameters, initial value $\Omega_{(0)} = (x_1, x_2, \ldots, x_n)^T$, tolerance ε.

Output: Approximate the solution $(\hat{x_1}, \hat{x_2}, \ldots, \hat{x_n})^T$ and calculate the value of objective function $f(\hat{x_1}, \hat{x_2}, \ldots, \hat{x_n})$.

Step 1: Interpolate the initial trial solution $\Omega_{(0)}$ and calculate $f(\Omega_{(0)})$.

Step 2: **while** $\left|\frac{\partial f}{\partial x_1}\right| > \varepsilon$ or $\left|\frac{\partial f}{\partial x_2}\right| > \varepsilon$, …, or $\left|\frac{\partial f}{\partial x_n}\right| > \varepsilon$ **do** steps 3–4.

Step 3: Compute the gradient of objective function $\vec{\nabla} f(\Omega) = \left[\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n}\right]^T$. Also, compute the Hessian matrix

$$\mathbf{H}(\Omega) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^n f}{\partial x_n \partial x_1} & \frac{\partial^n f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix} \text{ at point } \vec{\Omega}_i.$$

Step 4: Update the trial solution $\Omega_{(i+1)} = \Omega_{(i)} - \left[\mathbf{H}(\Omega_{(i)})\right]^{-1} \vec{\nabla} f(\Omega_{(i)})$. **end**
Step 5: Output

1.10.2 Direct-Search Method

Direct-search is a technique for solving optimization problems having discrete decision parameters. The direct-search method does not require the value of the derivative (gradient) of the function, which has to be optimized. The direct-search algorithm searches the solution locations around the current solution position, where the value of the objective function is less than the value at the current solution point in minimization problem or more for maximization problem. The pseudo-code of the direct-search method is as follows

Direct-Search Method: Pseudo-code

Input: Input system parameters.

Output: Approximate the solution $(x_1^*, x_2^*, \ldots, x_n^*)$ and compute the value of objective function $f(x_1^*, x_2^*, \ldots, x_n^*)$.

Step 1: Range the values of $(x_1^*, x_2^*, \ldots, x_n^*)$
Step 2: Set a intial trial solution.
Step 3: **if** the solution diverges, back to step 2 **end if**
Step 4: **if** $f(x_1^*, x_2^*, \ldots, x_n^*) < f^*$
Step 5: $f^* = f(x_1^*, x_2^*, \ldots, x_n^*)$
Step 6: **end if**
Step 7: Output $f^* = f(x_1^*, x_2^*, \ldots, x_n^*)$

1.10.3 Particle Swarm Optimization

Swarm intelligence is a modern intelligence optimization technique inspired by the essential information and biological behavior of animals, particularly birds and fishes. With swarm intelligence, birds & insects in the flock, bees in colonies and fishes in the school, other animals in a group communicate with each other, and members socialize in their territories while searching for food in a specific area. Swarm intelligence models are developed on decentralization, communication, and cooperation between the individuals of colonies. The mutual interaction is essential and emerges as a sophisticated global behavior, which is the base of swarm intelligence. Optimization techniques based on swarm intelligence have primarily been employed and found much more efficient over traditional optimization issues and challenges in every sphere of real-time applications.

In this section, the basic idea of the working process of the PSO algorithm is explained. The concept of the PSO algorithm is first proposed by American social

psychologist James Kennedy and engineer Russel Eberhart [67], having inspired by the social behavior of a group of birds/fish schooling. The PSO algorithm works on the notion of exploration and exploitation with a population of particles in the feasible search space. In PSO, each particle has its inherent velocity and position with which they move randomly within the search space. Also, the movement of every particle is influenced by its local/personal best (p-best) and global/common best (g-best) positions in the solution space. Suppose that $\mathbf{V}_i$ and $\mathbf{S}_i$ are the velocity and the position vector of the i^{th} particle respectively. The velocity component is updated using the following recursive formula

$$\mathbf{V}_i^{t+1} = \mathbf{V}_i^t + \kappa_1 \varphi_1 (\mathbf{G}^* - \mathbf{S}_i^t) + \kappa_2 \varphi_2 (\mathbf{S}_i^{*(t)} - \mathbf{S}_i^t) \tag{1.71}$$

wherein, κ_1 and κ_2 are the learning coefficients having standard value 2 for each, φ_1 and φ_2 are two random vectors having each entry between the range $[0,1)$. The position updating formula for the i^{th} particle is characterized as

$$\mathbf{S}_i^{t+1} = \mathbf{S}_i^t + \mathbf{V}_i^{t+1} \tag{1.72}$$

But to control the exploration and exploitation among particles, there is a requirement of an inertia function $\omega_2(t)$ which is introduced by [234] in the PSO algorithm. So, the improved velocity updating formula is given by

$$\mathbf{V}_i^{t+1} = \omega_2 \mathbf{V}_i^t + \kappa_1 \varphi_1 (\mathbf{G}^* - \mathbf{S}_i^t) + \kappa_2 \varphi_2 (\mathbf{S}_i^{*(t)} - \mathbf{S}_i^t) \tag{1.73}$$

The standard value of the inertia function $\omega_2(t)$ has been found in the literature to be between 0.5 to 0.9. The pseudo-code of the PSO algorithm is given below

Particle Swarm Optimization: Pseudo-code

Input: Input parameters, population size, learning parameters.

Output: Approximate solution $(x_1^*, x_2^*, \ldots, x_n^*)$ and the value of objective function $f(x_1^*, x_2^*, \ldots, x_n^*)$.

Step 1: Population Initialization: find the positions $\mathbf{S}_i$ of n particles.

Step 2: Find $\mathbf{G}^*$ (common best) from $\{f(\mathbf{S}_1), f(\mathbf{S}_2), \ldots, f(\mathbf{S}_n)\}$.

Step 3: **while** $(t <$ MaxGeneration$)$ or(stop criterion)

for loop over all the n particles and all d dimensions.

Step 4: Find new velocity vector for the i^{th} particle $\mathbf{V}_i^{t+1}$.

Step 5: Find new positions for the i^{th} particle $\mathbf{S}_i^{t+1} = \mathbf{S}_i^t + \mathbf{V}_i^{t+1}$.

Step 6: Evaluate cost function at new positions $\mathbf{S}_i^{t+1}$.

Step 7: Find the current best for each particle $\mathbf{S}_i^*$.

end for

Step 8: Update global best $\mathbf{G}^*$.

$t \to t+1$

end while

Step 9: Output final results $\mathbf{S}_i^*$ and $\mathbf{G}^*$.

Step 10: Output optimal value of the objective function: f^* at $\mathbf{G}^*$.

1.10.4 Cuckoo Search Algorithm

Cuckoo search algorithm takes inspiration of brood parasitic behavior wherein natural phenomenon is mapped to an optimization problem in the following way

- Each cuckoo which behaves as a search agent lays only one egg in the nest randomly. The randomly chosen nest refers to a solution to the problem.
- In each generation, some fixed number of eggs are taken to the next generation. The best solution is determined by evaluating fitness function at the current solution point for all agents.
- The host bird identifies cuckoo's egg in its nest with probability p_a. For optimization algorithm, this is called switch probability with which exploration and exploitation, both are executed.
- Positions of nests are adjusted in the local area such that eggs in these nests try to be best among all. It is defined as an exploitation step in the cuckoo search algorithm.
- The host bird is likely to leave the nest and make a new nest somewhere randomly. In this way, exploration for a new solution is done when some agents can not find a better solution in the neighborhood.

Local and global random walks and their behavior are significant concerns of this algorithm. This algorithm is based on a balanced combination of a local random walk and the global explorative random walk, controlled by a switching parameter p_a. The local random walk can be written as

$$x_i^{t+1} = x_i^t + \Theta_1 \, \kappa \otimes H(p_a - \varpi) \otimes (x_j^t - x_k^t) \tag{1.74}$$

where x_j^t and x_k^t are two different solutions selected randomly by random permutation, $H(u)$ is a Heaviside function, ϖ is a random number drawn from a uniform distribution, and κ is the step size. Here, $\otimes$ means the entry-wise product of two vectors.

As we have discussed earlier, global convergence is responsible for its efficiency and a wide range of applications. The global random walks of the cuckoo search algorithm are governed by Lévy flights. Isotropic random walks simulate such a population-based algorithm, but a significant step size in random walks are less likely because tails of the distribution function decrease exponentially. Lévy flights give

very intuitive random walks, which actually, animals, birds, and insects follow in any survival technique like searching food. Due to the power characteristics of Lévy distribution (1.76), it is heavy-tailed, which allows significant step sizes more than what normal distribution does.

$$x_i^{t+1} = x_i^t + \Theta_1 L_f(\kappa, \omega_1) \tag{1.75}$$

where

$$L_f(\kappa, \omega_1) = \frac{\omega_1 \Gamma(\omega_1) \sin(\frac{\pi\omega_1}{2})}{\kappa^{(1+\omega_1)}\pi}; \; \kappa \gg \kappa_0 > 0 \tag{1.76}$$

Here $\Theta_1 > 0$ is the step size scaling factor. It should be determined depending on the characteristics scale of the problem. We use $\Theta_1 = O(\frac{\Xi}{10})$, where Ξ is the characteristic scale of the problem of interest. These global optimization step would ensure that the solution would not get trapped in some local optimum.

In Mantegna's algorithm [310], the step length κ can be calculated by

$$\kappa = \frac{u}{|\aleph|^{\frac{1}{\omega}}} \tag{1.77}$$

where u and $\aleph$ are drawn from normal distributions *i.e.* $u \sim N(0, \sigma_u^2)$ and $\aleph \sim N(0, \sigma_v^2)$
where

$$\sigma_u = \left\{ \frac{\Gamma(1+\omega_1)\sin(\frac{\pi\omega_1}{2})}{\Gamma\left[\frac{(1+\omega_1)}{2}\right] \omega_1 2^{\{(\omega_1-1)/2\}}} \right\}^{\frac{1}{\omega_1}}, \qquad \sigma_v = 1 \tag{1.78}$$

The distribution for κ obeys the expected L*é*vy distribution for $|\kappa| \geq |\kappa_0|$, where κ_0 is the smallest step. In principle, $|\kappa_0| \gg 0$, but in reality κ_0 can be taken as a sensible value such as $\kappa_0 = 0.1$ to 1. The pseudo-code for the cuckoo search via Lévy flights for optimizing arbitrary function is as follows

Cuckoo Search Algorithm: Pseudo-code

Input: Fixed value of system parameters, population size, switching parameter.

Output: Approximate the solution $(x_1, x_2, \ldots, x_n)$ and compute corresponding value of the objective function f.

Step 1: Objective function $f(\mathbf{x})$; $\mathbf{x} = [x_1, x_2, \ldots, x_d]^T$ and initialize the population of n host nests $\mathbf{x_i}$

while ($t <$ MaxGeneration) or (Stop criterion)

Step 2: Get a cuckoo randomly

Step 3: Generate a solution by Lévy flights
Step 4: Evaluate its solution quality or objective value
Step 5: Choose a nest among n nests randomly
if $(f_i < f_j)$
Replace j by the new solution i
end
Step 6: A fraction (p_a) of worse nests are abandoned
Step 7: Generate new solutions and keep best solutions (or nests with quality solutions)
Step 8: Rank the solutions and find the current best
Step 9: Update $t \leftarrow t+1$
end while
Postprocess results and visualization

1.10.5 Bat Algorithm

Modern optimization techniques are often nature-inspired, typically based on swarm intelligence and biological behavior of animals. Each species, in general, has unique characteristics for survival as fittest ones. The ways for inspiration are miscellaneous, and therefore the algorithms can be of many different types. However, all these algorithms tend to utilize some specific characteristics for formulating the key updating formulae. Metaheuristics like particle swarm optimization and genetic algorithms can be very convenient, but still, they have some drawbacks in dealing with complex and multimodal optimization problems. One significant improvement is the bat algorithm (BA), which was first introduced by Yang [308] in 2010. Bat algorithm is an agent-based optimization technique inspired by the bio-sonar or echolocation characteristics of microbats.

Most of the microbats are insectivores in food habits. Microbats typically use a sonar technique called echolocation to detect prey or victims, and explore and locate their roaming crevices, even in the dark region. These bats produce a high pulse sound and listen to the resonance that produces back from the surrounding objects. Their pulses differ in physical values depending on the different species and can be interrelated with their hunting strategies. Most of the bats generally adopt a short frequency-modulated signal to roam through about an octave while the others more often use constant frequency signals for echolocation. Their signal bandwidth fluctuates with species and usually increases by using more harmonics. Studies demonstrate that microbats utilize the time delay from the emission and detection of the echo, the diversity of time interval between their two ears, and the loudness variations of the echoes to develop a three-dimensional dynamic image and scenario of the

surrounding. Also, they distinguish between distance and orientation of the object, the type of prey, and even the moving speed of the victim, such as small insects.

Based on the mentioned characteristics and depiction of echolocation, the bat algorithm works with the following three idealized rules

(i) All bats use the echolocation to detect the distance from a food source and also have the knowledge to distinguish between foods/victims and background barriers.

(ii) Bats fly randomly in the surroundings with velocity $\mathbf{V}_i$ at position $\mathbf{S}_i$ and produce fixed frequency f_i pulse. They can automatically regulate the frequency (or wavelength) of their emitted pulses and change the rate of pulse emission (r_i) correspondingly in the range between 0 and 1, depending on the proximity of their target.

(iii) Though the loudness can vary in a variety of ways, we consider that the loudness varies from a large (positive) L_0 to a minimum value $L_{\min}$.

In addition to these assumptions, for simplicity, the frequency f is taken in a range $[f_{\min}, f_{\max}]$ corresponding to a range of wavelengths $[\lambda_{\min}, \lambda_{\max}]$. Therefore, with the help of the above mentioned assumptions, the updated equations for frequency f_i, position $\mathbf{S}_i$ and velocity $\mathbf{V}_i$ are as follows

$$f_i = f_{\min} + (f_{\max} - f_{\min})\,\vartheta \tag{1.79}$$

$$\mathbf{V}_i^{t+1} = \mathbf{V}_i^t + \left(\mathbf{S}_i^t - \mathbf{S}^*\right) f_i \tag{1.80}$$

$$\mathbf{S}_i^{t+1} = \mathbf{S}_i^t + \mathbf{V}_i^{t+1} \tag{1.81}$$

where

- $\vartheta \in [0, 1]$ is a uniformly distributed random vector.
- f_i is the frequency that i^{th} bat emits and $f_{\min}$, $f_{\max}$ are the lower and upper bounds of frequencies, respectively.
- $\mathbf{V}_i^t$ is the velocity of i^{th} bat after t generations.
- $\mathbf{S}_i^t$ is the position of i^{th} bat after t generations.
- $\mathbf{S}^*$ is the current best position (solution) of the fitness function among all the n bats.

After selecting a solution among the current best solutions, for the local search, we use the random walk for each bat. Hence, the new position updating formula is generated locally and is expressed as

$$\mathbf{S}_{new} = \mathbf{S}_{old} + \varepsilon\, L^{(t)} \tag{1.82}$$

where $\varepsilon \in [-1, 1]$ is a random number and $L^{(t)} = \langle L_i^t \rangle$ is the average loudness of all

the bats at time instant t. Now to control the step size, we provide the scaling parameter ρ to avoid overshooting or undershooting the search space. The new position updating formula is rewritten as

$$\mathbf{S}_{new} = \mathbf{S}_{old} + \rho\, \varepsilon_t\, L^{(t)} \tag{1.83}$$

where the value of ε_t is taken from the Gaussian normal distribution $N(0,1)$, and ρ is a scaling factor having standard value 0.01.

The pseudo code of bat algorithm is as follows

Bat Algorithm: Pseudo code

Input: Input parameters, initialize the bat population $\mathbf{S}_i$ and $\mathbf{V}_i$; $(i = 1,2,\ldots,n)$.

Output: The global best solution ($\mathbf{S}^* = [\hat{x_1}, \hat{x_2}, \ldots, \hat{x_n}]^T$) and its analogously minimum functional value $f_{\min} = \min\{f(\mathbf{S}_1), f(\mathbf{S}_2), \ldots, f(\mathbf{S}_n)\}$.

Step 1: Initialize the frequencies f_i, pulse rates r_i and loudness L_i.

while $t <$ Max number of iterations

Step 2: Generate new solutions by adjusting frequency.

Step 3: Update velocities and locations/solutions [eq^n's(1.79)–(1.81)].

if (rand $> r_i$)

Select a solution among the best solutions.

Generate a local solution around the selected best solution.

end if

Step 4: Generate a new solution by flying randomly.

if (rand $< L_i$ & $f(\mathbf{S}_i) < f(\mathbf{S}^*)$)

Accept the new solutions.

Increase r_i and reduce L_i.

end if

Step 5: Rank the bats and find the current best $\mathbf{S}^*$.

end while

1.11 Some Basic Terminologies

The three main keywords around which the entire queueing modeling depend are customer, queue, and server. The meaning of these words is reasonably self-evident. Depending on behaviors and characteristics, several queueing models are developed in the literature. Some of the customers and server-based characteristics which we have taken into consideration in the present study are classified as follows

1.11.1 Customers Behavior

Usually, the customers behave impatiently in the waiting queue in the following manners

- **Balking:**

  Whenever the newly arriving customer finds the server busy or unavailable due to vacation, breakdown or any other reasons, he leaves the system without entering into it due to the high impatient. The random behavior of customer is defined as a balking which governs with some probability. In other words, the newly arriving customer may not like to wait in a queue due to lack of time or space or otherwise.

- **Reneging:**

  The customer, who joins the queue, but is abandoned from the waiting line after waiting for some time impatiently in the system, is known as a reneged customer in queueing modeling. It directly points the server efficiency, system satisfaction level, and customer loss.

- **Collusion:**

  When some customers collaborate, but only one of them joins the queue is defined as the collusion behavior of the customer. For example, in a cinema ticket counter one person enters the waiting queue and purchase the tickets for all his friends.

- **Jockeying:**

  Whenever the arriving customer joins a queue and then switches to another waiting queue in parallel to reduce his waiting time is defined as the jockeying behavior of the customer. This situation generally occurs in the supermarket.

1.11.2 Server's Behavior

1.11.2.1 Unreliable Server

Sometimes the server also breaks down due to continuous working and being busy for a long time. It is defined as an unreliable characteristic of the server, and state-ot-art of better service system is to repair or replace immediately to ensure the uninterrupted service.

1.11.2.2 Server's Vacation

In classical queueing modeling, generally, we assume that the servers are always available for providing the service whether there is no waiting customer. But in practice, continuous service seems to be uneconomical and impractical in global modernization. To overcome this limitation, a new perception is introduced in the queueing literature in which some times these servers may become unavailable for a while. That time-period is known as servers vacation. Some times these servers may opt a secondary service rather than altogether terminating his primary service. This phenomenon is defined as the working vacation of the server. In queueing theory, these vacations and working vacation policies are classified in following strategical class as

- **N-Policy Vacation:**

  In queueing modeling based service systems, if the server becomes unavailable at the end of a busy period and resumes to provide the service to the waiting customer(s) when the length of the queue reaches a threshold number N, then it is known as N-policy vacation of the server.

- **Single Vacation:**

  In the strategic service-management of service systems, the server may become unavailable deliberately for some time due to a variety of reasons like taking rest, reducing work stress, rejuvenating the efficiency, minimize the expected cost, etc. The random period during which the server is unavailable in the system is defined as a vacation of the server. In the single vacation policy, the server opts for the vacation only for one time. On returning from the vacation if the server finds no waiting customer for seeking service, he waits idly in the system otherwise provides the service to the waiting customer.

- **Multile Vacation:**

  According to this policy, a server leaves for a vacation of a random duration when there is no customer for service in the service facility. On return from the vacation, if he finds an empty queue, he takes another vacation and continues this process until he finds at least one waiting customer in the system after the end of vacation duration.

- **Bernoulli Vacation:**

  In this policy, after the end of a vacation, the server decides whether to leave

for another vacation of random duration with some probability or continue to provide the service to the waiting customer in the system with the complementary probability.

- **Variant Vacation Policy:**

  In the variant vacation policy, the server takes a vacation whenever the system becomes empty. However, the server can take vacations at most m times if the system remains empty after the end of the vacation. This vacation policy is defined as a particular case of both single and multiple vacation policies.

- **Working Vacation:**

  In the working vacation policy, the server works with a slower service rate rather than complete termination of his service during the vacation period.

- **Single Working Vacation:**

  In the single working vacation policy, at the end of a working vacation, if there is no waiting customer in the service facility to be served, the server stays in the system idle and is ready for providing the service to the new arrivals.

- **Multiple Working Vacation:**

  In this policy, at the end of a vacation period, if the server does not find any customer in the system, he repeatedly takes the working vacation of random duration otherwise provides the service to the waiting customer, if any, in the system with the normal service rate.

- **Synchronous and Asynchronous Vacation:**

  In a multi-server queueing modeling based service system, if all the servers take the vacations together at the same time, then the vacation policy is called the synchronous vacation policy. Whereas, in the asynchronous vacation policy, each server takes the vacations individually and independently to other servers.

1.12 Literature Review

1.12.1 Development of Queueing Theory

The history of queueing theory and its concepts is nearly 110 years old, when A K Erlang (*cf.* [73]) published a research paper in 1909 titled "The Theory of Probabilities and Telephone Conversions" in which a telephone system could be designed as a Poisson call input that represents the random arrival, exponential or continuous holding (service) time of a call, and a single or multiple channels. Later, Erlang enriched the literature with many essential concepts and techniques, such as the notion of classical equilibrium condition, method of writing the balance equations, and optimization of a queueing system. During the early 1930s, [209], [210] investigated queueing systems in finite-time intervals. At that time, many theorists showed interest in these types of problems, and many general models are developed that could be used in more complex situations. Hence, at the beginning of the 1940s, theoretical analysis of queue-based service systems increased substantially with the advent of operations research.

Queueing theory, as an identifiable body of literature, was primarily defined by the fundamental research ideas in the 1950s and 1960s. The first textbook on the subject, "Queue, Inventories, and Maintenance" was written by [197] in 1958. After that, in 1961, [218] wrote one of the most famous books in queueing literature, "Elements of Queueing Theory with Applications". Kovalenko [144] presented a survey of developments of mathematical research in queueing theory from the period 1964 to 1970. Since the 1970s, with the advent of new processes in manufacturing, the applications of queueing theory result, and the development of new techniques have occurred at a phenomenal rate. With the several emerging concepts of queueing and service systems, [143] provided a book "Queueing System" in 1976. In 1992, [30] reported open queueing network models of manufacturing systems. Later, [87], [3], and others discussed several queueing characteristics in their work. In addition, extensive research work has been done on queue-based service systems in various frameworks over the years. The literature on service systems has been vast. Here, we restrict the thesis works to the development of queue-based service systems, which are strictly related to our investigation on performance and optimization analysis using several critical queueing terminologies in service systems.

1.12.2 Customer Impatient

Queueing systems with impatient customers appear in many service models of our day-to-day life, such as emergency rooms in hospitals, inventory systems having storage of perishable goods, telephonic calls at telephone switchboard, et cetera. Due to the widespread use of queueing models, a large number of researchers have contributed in this direction and achieved many significant results. Palm [202] first introduced the reneging behavior of calls in telephonic transmission. Haight [95] was the first researcher to introduce the notion of the balking behavior of customers in queueing models. Keilson [135] investigated a general bulk queueing model with balking as a Hilbert problem. Rao [215] analyzed a single server queueing model with balking and reneging under the assumption of general service distribution. Barrer [27], [28] formulated a queueing problem with the impatient behavior of customers in processing systems under ordered processing. For a homogeneous queueing system, [198] provided the explicit expressions of queue-size distribution using general service distribution and discouragement. Dekok and Tijms [57] investigated a single-server queueing system with Poisson input and the general service times. Varshney et al. [269] analyzed a general input and general service-time distribution and analyzed several closed-form expressions for system performance measures using the diffusion approximation approach. Al seedy [5] envisaged a queueing model with phase-type service and impatient. Gupta [90] demonstrated the interrelationship between the concepts balking and reneging. Again, [91] considered the state-dependent queueing problem with the balking and reneging of customers. Using the terminologies of balking and reneging, [56] proposed a queueing model with the customer's loss in multi-server queues.

Shawky [226] provided the analytical solution to the finite population machine repair model with balking, reneging, and spare provisioning. Wen [287] evaluated the token-tray/weighted queueing time for near window on-demand system considering the concept of discouragement. Al Seedy [6] suggested a transient solution technique for a non-truncated two server queueing model involving balking behavior and additional server for long waiting lines. Ke [125] analyzed operating characteristics of a $M^{[X]}/G/1$ system with a variant of vacation policies and impatient using a supplementary variable technique. Wu and Ke [289] proposed an efficient computational solution technique for the multi-server system with impatient customers. Ammar [12] studied the single server Markovian queueing system with balking and reneging, and provided several closed-form expression through busy period analysis. Recently, [179] formulated the stability condition for an infinite capacity Markovian queue with a single unreliable service station and impatient customers utilizing the

matrix-analytic approach. Yassen and Tarabia [311] developed an infinite Markovian queueing system using the concept of balking and reneging, where the system may be used under repair for server failure. More recently, [151] proposed a model to study the infinite capacity queueing system with Poisson input, exponential service time distribution and retention of reneged customers.

From an economic point of view, customer impatience can be seen as a potential loss of customers. Thus, more and more organizations and industries are adopting efficient strategies by which impatient customers remain in the system. Therefore, inspired by this fact, new queueing terminology in literature, the retention of reneged customers, was proposed by [153]. Again, [150] extended their previous work, considering the balking behavior of customers. Sharma and Kumar [225] proposed steady-state solutions to a single server finite capacity queue with feedback and retention of reneged customers. Again, [154] established the transient and steady-state solutions to a multi-server finite capacity queueing model with impatient, and retention of reneged customers and performed a cost analysis for building an economic system. Recently, the optimal analysis of a finite capacity queueing model with working breakdown and retention of reneged customers is executed by [303]. Kumar et al. [152] studied two-heterogeneous servers' queueing problems with retention of reneged customers and provided some customer and server-oriented measures of effectiveness.

In the queueing literature, a significant number of research papers exist to deal with abandonment in queueing systems. However, there are only limited research papers in the literature to classify customers' abandonment behavior in different environments. For the detailed information and excellent study on the abandonment policy, readers can refer the following research papers and references therein (*cf.* [17], [68], [61], [60]). Sometimes, impatient customers abandon the system simultaneously rather than independently. The phenomenon of simultaneous abandon is defined as synchronized abandonment and is new terminology in literature, which was first introduced in 2009 by [2]. For the significant note on synchronized reneging, we refer the articles [70], [71], [123].

In designing a queue-based service system, the assumptions for arrival and service patterns can be made such that the system can work smoothly to achieve production goals despite unexpected failures of the system. It has been observed that after a fixed threshold level, newly arrived customers may not be allowed to join due to the capacity constraints of the service system. In the context of real-life applications, arrival control is one of the cost-effective as well as efficient managerial approaches that can be used under the constraints of the capacity of the service system. To control the arriving customers in the finite capacity service system, the admission control

policy based on the threshold level was first proposed by [92]. Tadj and Chaudhary [246] presented a comprehensive review of the literature to show the optimal design and control of queueing systems. A recursive approach to a controllable queueing model with a general arrival pattern was provided by [278] using the supplementary variable technique. Extensive works on controllable admission policy with many queueing variants are done in past (*cf.* [281], [301], [39], [111], [105], [108], [227]). Recently, [176] provided the reliability investigations of a k-out-of-n:F system under the regime of repair-rate differentiation policy. To develop a fault-tolerant machining system, [109] envisaged a Markovian queueing system with a working vacation and randomized arrival control policy.

1.12.3 Feedback

Sometimes, even after receiving the primary service, the customers are not satisfied with the service, the customer may reattempt with some probability to be completely satisfied or may leave the system with the complementary probability. It is defined as a feedback policy in the queueing literature. Many of the real-life queueing problems can be seen as feedback queues such as manufacturing processes, data transmission in networking, computers and communication systems, supply-chain management, et cetera. Finch [76] first presented a paper "Cyclic queues with feedback" for the terminal server, in which the author introduced the concept of feedback policy. Takacs [248] provided the steady-state probability distribution and the closed expressions for the expected waiting time of a customer of a single server queue with feedback. Delbrouck [58] considered a single channel feedback queueing system with batch arrivals, bulk service, and the queue-dependent service time. Arya [18] proposed a queueing system with two homogeneous servers, which are connected in series with a shared server. Montazer [196] studied a multi-servers' queueing system with feedback and determined customers' waiting time distribution and the static process. Garg [82] provided the probability of a fixed number of arrivals and departures for specific feedback queueing models. The Bernoulli feedback has also been introduced for polling models by [253]. Kalyanraman et al. [122] proposed the numerical scheme to determine the average waiting time of the customer in the system. Madan and Rawwash [190] considered a feedback queueing model for optional server vacations with bulk arrival. Using a matrix-geometric method, [19] calculated the state probabilities for the service systems in call centers. Transient-state queue size distributions and their Laplace transformation function were obtained by [239]. The approximation methods were developed by [194] to calculate several characteristics of the studied queueing system with feedback. Thombare et al. [258] used the round-robin algorithm for fair use of CPU in a queue with feedback policy in the first line

and SJF in the next waiting line to reduce the average waiting time. Recently, [35] presented the mathematical analysis for the Markovian feedback queueing model with impatient and Bernoulli scheduled vacation interruption. With the general service distribution, [141] delineated a feedback queueing system with a multi-class of customers and derived the functional equations for the stationary distribution of the queue size.

In the queueing literature, the concept of a breakdown server and its unreliable characteristics have been considered and studied extensively, but our main objective in this thesis is to provide a multidisciplinary view on the unreliable service characteristics of a service provider. Usually, it is assumed that the service provided by the server is successful and satisfactory at every instant. But, in real-time applications, the hypothesis may not be correct. Inspiring from this fact, a new terminology, unreliable service, has been introduced in the queueing literature by [205]. In this policy, the occurrence of unsuccessful service is neither because of the servers' fault, nor due to the customer, i.e., service failure instances occur due to some external and environmental shocks. Recently, [204] extended previous work [205] to an infinite capacity queueing model with a working vacation of the server. In the developed model, the authors employed the matrix-geometric technique to provide a closed-form expression of several system performance measures.

1.12.4 Vacation Policies

Since inception from Erlang's early work based on modeling of telephonic traffic systems, queueing theory and its modeling has been significantly developed manifold over nearly 110 years. Due to its extensive practical applications in many service fields, queueing theory has been one of the most active research areas in operations research, management science, and industrial engineering over the past several decades. Some preliminary research work related to queueing systems is relevant to queueing modeling with the vacation of the server. Cooper [52] firstly conferred a research work on queues placed in cyclic order, in which the service-times of serving other waiting lines could be considered as a service interruption of the queue under consideration. However, some significant and useful research results on vacation queueing systems were published in the second half of the twentieth century. Levy and Yechili [168] first introduced the concept of servers' vacation, which represents the duration of server work on some complementary projects. Later, several research results on vacation queueing models were published ([54], [77], [64], [65], [66]).

Takagi [249] gave a systematic treatment of the exhaustive service vacation model with the general service-time distribution. Kella [136] proposed a more general vacation policy, wherein, at service completion instant, if there are waiting customers

in the system, the server starts serving customers; otherwise, the server takes a vacation with probability p and enters the idle period with the complementary probability $(1-p)$. If $p=1$, it corresponds to the multiple vacation policy, and when $p=0$, it corresponds to the single vacation policy. Later, [260] presented another generalized form of multiple and single vacation policies. A batch arrival queueing model with a single vacation policy is studied by [164]. Tian et al. [261] proposed some conditional stochastic decomposition results for a muti-server queue with server vacation. A discrete-time $Geo/G/1$ queueing model is envisaged by [327] with multiple adaptive vacation. A mathematical analysis of queues with a single vacation was presented by [325]. Madan and Al-Rub [191] investigated a single server queue with optional phase-type server vacations based on a single vacation policy. Gupta and Sikdar [94] computed queue length distributions in MAP/G/1/N queue under single and multiple vacations. A short survey on vacation queueing models was presented by [132]. To analyze the vacation queueing models, the optimal operating policies in Markovian and Non-Markovian environment were used by many researchers and scientists (*cf.* [129], [315], [180], [243], [324]).

Yadin and Noar [297] was first to introduce the concept of N-policy of the server, with which the server shuts down when there is no customer in the system, and turns on the server if the number of waiting customers in the system reaches a threshold value N. Later, the concept of N-policy is used with several vacation queueing models ([100], [25], [223], [32], [263], [250], [163], [16], [161]). The recent researches on the N-policy for the server in service system have been observed in the following works (*cf.* [267], [316], [117], [283], [236], [116], [41], [20]), and references therein.

The upgraded generalization of vacation policies is working vacation policy, under which the vacationing server provides service at a lower service rate rather than stopping service completely. Due to reasonable assumptions, it has been extensively used in a wide range of applications like network service, mail services, web service, file transfer service, and many more. The concept of working vacation policy was first conceptualized by [222] for a single server queueing system. Later, [292] extended the result of [222] for general service distribution. Kim et al. [142] examined the queue-size distribution of a queueing system with Poisson arrival and general service times with working vacation policy. Liu et al. [185] demonstrated the stochastic decomposition results for a single server Markovian queue with a working vacation. Li et al. [169] analyzed a $M/G/1$ queue with an exponential working vacation regime. For a batch arrival queueing system with a single working vacation, [295] provided some closed-form expressions employing the matrix analytic approach. Later, [21] generalized the work of [295] to multiple working vacation

policy. Several researchers and scholars investigated queueing problems with multiple and single working vacation policy in the past (*cf.* [221], [89], [212], [171], [88], [137]).