Chapter 1

Introduction

1.1 On-demand Computing

On-demand computing[1] is a model for enabling ubiquitous, convenient and on-demand network access to a shared pool of configurable computing resources. On-demand computing is popularily known as an enterprise-level model of technology and computing in which resources are provided on an as-needed and when-needed basis. It makes computing resources such as storage capacity, computational speed and software applications available to users as and when needed for specific temporary projects, known or unexpected workloads, routine work, or long-term technological and computing requirements. Concepts such as grid computing[2] [3], utility computing, autonomic computing, cloud computing[3] [4] and adaptive management seem very similar to the concept of on-demand computing. It requires sufficient number of resources to meet the peak requirements.

[1] https://www.techopedia.com/definition/1308/on-demand-computing-odc

[2] A grid is a very large-scale network computing system that scales to internet size environments with machines distributed across multiple organizations and administrative domains.

[3] A large-scale distributed computing paradigm that is driven by economies of scale, in which a pool of abstracted, virtualized, dynamically-scalable, managed computing power, storage, platforms, and services are delivered on demand to external customers over the Internet.

1.2 Transaction Processing in On-deamnd Computing

A transaction has a peculiar nature regarding its computation operation and its completion on various nodes in on-demand computing system [5]. Transaction processing by using on-demand computing system consists of various service calls executed by different peers of the system [6]. It is used when the number of users increases beyond the limit. It solves the problem of scalability[4] and reduces the completion time. Transaction processing in e-commerce[5], VisaNet[6], PayPal[7] and financial trading systems are the kind of transaction processing with the help of on-demand computing system including grid or cloud. Therefore, on-demand computing based transaction processing system deals with the challenge to an enterprise to meet fluctuating demands of sufficient resources efficiently.

On-demand computing based transaction is defined as a group of operations executed to perform some specific functions by accessing and/or updating a database [5]. It consists of various service calls executed by different peers of the computing system [6]. It is a reliable and coherent process unit that interacts with one or more systems, independently of the other. It is widely needed as an effective means of sharing a large number of resources among different organizations [7, 8, 9]. It is the platform that needs to share, access, transport, process, and manage large data collections distributed worldwide. It gives large e-commerce and financial transaction support to its customers to handle operations where speed and scalability are essential. It combines high-end computing technologies with high-performance networking and wide-area storage management techniques.

When transaction processing is performed using on-demand computing system, the transactions arrive at any node and may execute at several nodes before they commit [7, 8, 9]. Due to the dynamic nature of on-demand computing system, some nodes add while some nodes leave the system. As a result, the waiting time at heavily loaded nodes increases [10] and therefore, most of the transactions have to face rollback or abort

[4]Scalability is the capability of a system, network, or process to handle a growing amount of work, or its potential to be enlarged to accommodate that growth.

[5]E-commerce is a transaction of buying or selling online.

[6]VisaNet is a proprietary transaction processing network

[7]PayPal is one of the world's largest Internet payment companies. PayPal Holdings, Inc. is an American company operating a worldwide online payments system that supports online money transfers.

consequences. When the queues at the heavily loaded nodes are full, transactions' requests cannot be enqueued further. This situation causes unavailability problem which usually occurs during the peak time. At this peak time, several transactions' requests arrive in a short period. As a result, most of the transactions cannot be completed within their respective deadlines. To execute the transactions in such environment, distributed resources should be available to interact with resource management system and users. Processors inside a resource can fail to execute a transaction at any time. Load in the queue of the resources also causes the delay resulting deadline-miss of transactions and thus transaction aborts. In both the cases, resource availability is a prime factor. Therefore, throughput, resource utilization, availability[8], reliability[9] and performability[10] of on-demand computing to service the transaction can be highly influenced by the processor failure and the failure caused by deadline-miss. In order to meet and overcome these challenges, the system needs to be dependable. Therefore, dependability in on-demand computing system becomes important property.

1.3 Dependability

Dependability of a system is defined as the ability of the system to avoid service failures that are more frequent and more severe than are acceptable. In other words, dependability can be defined as the ability of the system to deliver a service that can justifiably be trusted. In **FIGURE 1.1** we see the relationship between Dependability and Attributes, Threats and Means [11]. It is first introduced as a global concept that subsumes the usual attributes of **reliability**, **availability**, **safety**, **integrity**, **maintainability**, **performability** etc. [11, 12]. As developed over the past four decades, dependability is an integrating concept that encompasses the following attributes [12]:

- **Availability**: The probability that the system will be up and running and able to deliver useful services to users.

[8]Availability of a system at a time is the probability that the system is functioning correctly at that instant of time.

[9]Reliability of a system is the probability that the system operates without a failure.

[10] Performability is defined as quantifying how well the object system performs in the presence of faults over a specified period of time.

- **Reliability**: The probability that the system will correctly deliver services as expected by users.
- **Safety**: A judgment of how likely it is that the system will not damage people or the system's environment. This property depends on the system being available and operating reliably.
- **Integrity**: It is the property with absence of improper system alterations.
- **Maintainability**: It is the ability of the system to undergo modifications and repairs.
- **Performability**: It is the property of quantifying how well the object system performs in the presence of faults over a specified period of time.

Dependability of a system reflects the extent of the user's confidence that it will operate as user's expect and it will not corrupt data or other systems and will not fail in normal use. Systems that are not dependable and are unreliable, unsafe or insecure may be rejected by their users. Dependability is affected by some basic threats.

1.4 The Threats to Dependability: Failures, Errors, Faults

A **failure**[11] is an event that occurs when the delivered service in the system deviates from the expected and correct service. A service failure occurs when the functional specification is not complied with it or the specification is unable to describe the system function. A service failure can be said to be a transition from correct service to incorrect service, i.e., to not implementing the system function [12].

Since a service consists of a sequence of the system's external states, a service failure is said to occur when at least one (or more) external state of the system deviates from the correct service state. This deviation is called an **error** [12].

[11] A failure which is referred to service failure.

The adjudged or hypothesized cause[12] of an error is called a **fault**. Faults can be internal or external of a system. The internal fault is the prior presence of vulnerability and enables an external fault to harm the system. This fault is necessary for an external fault to cause an error and possibly subsequent failure(s). There are many types of faults which the system may face. The general faults which occur in the system are hardware faults, software faults, and network faults. But in the case of transaction processing in on-demand computing, another type of fault comes and needs to be elaborated known as **deadline-miss fault**[13]. In this thesis, we incorporate **deadline-miss fault** while formulating the availability, reliability, and performability of the system. The means to attain the dependability are needed to overcome the effects of these faults.

1.5 The means to Attain Dependability

To attain the various attributes of dependability, many means have been developed over the course of the past 60 years [12]. The means are grouped into four major categories:

- **Fault prevention**: It prevents the occurrence or introduction of faults.
- **Fault tolerance**: It avoids service failures in the presence of faults.
- **Fault removal**: It reduces the number and severity of faults.
- **Fault forecasting**: It estimates the present number, the future incidence, and the likely consequences of faults.

The aim of fault prevention and fault tolerance is to provide the ability to deliver a service that can be trusted, whereas the aim of fault removal and fault forecasting is to reach confidence in that ability by justifying that the functional and the dependability and security specifications are adequate and that the system is likely to meet them [12].

[12]A hypothesis states a presumed relationship between two variables in a way that can be tested with empirical data. It may take the form of a cause-effect statement, or an “if x, ...then y” statement. The cause is called the independent variable; and the effect is called the dependent variable.

[13]Fault caused by deadline-miss.

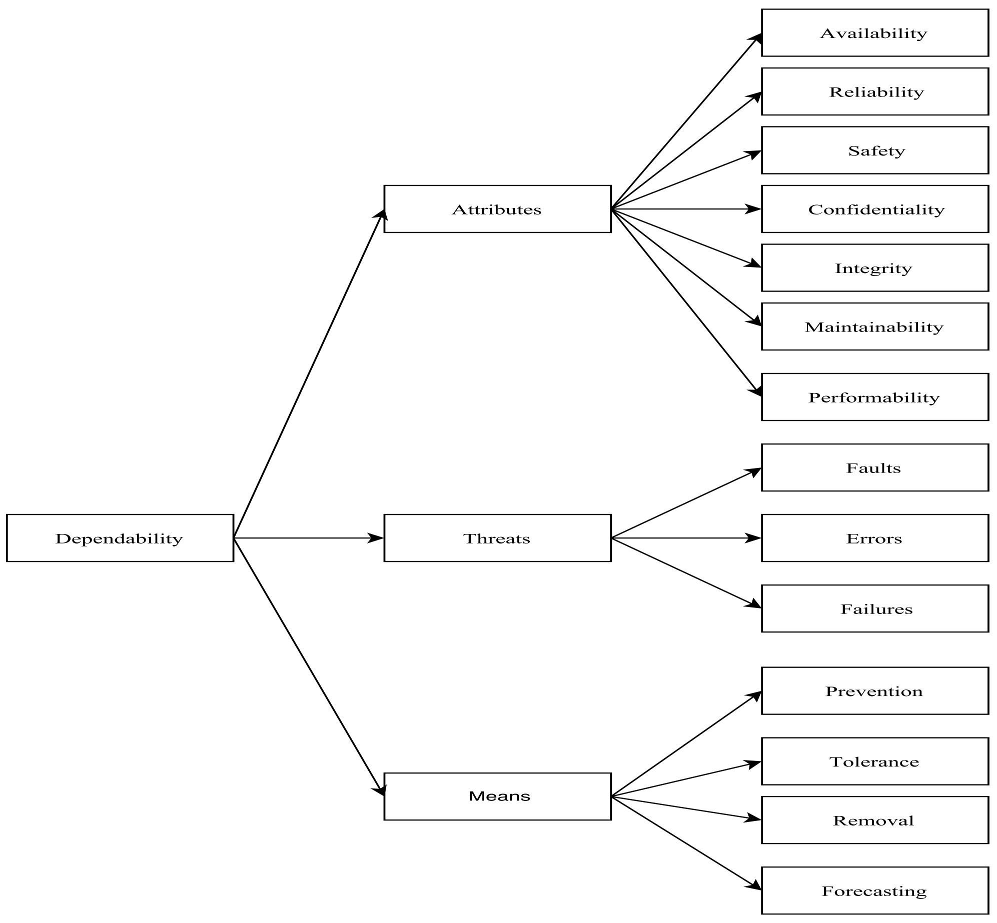

FIGURE 1.1: Taxonomy showing relationship between Dependability and Attributes, Threats and Means (after Laprie et al.)

In this thesis, we formulate the dependability model by using fault tolerance means. We model the dependability by considering the various attributes by using load balanced scheduling based on heuristic[14] or meta-heuristic[15] approach.

1.6 Load Balanced Scheduling

The task scheduling strategy in on-demand computing based transaction processing system schedules the deadline-constrained transactions based on failure possibility or

[14] A heuristic technique is any approach to problem solving, learning, or discovery that employs a practical method not guaranteed to be optimal or perfect, but sufficient for the immediate goals.

[15] A metaheuristic is a higher-level procedure or heuristic designed to find, generate, or select a heuristic (partial search algorithm) that may provide a sufficiently good solution to an optimization problem, especially with incomplete or imperfect information or limited computation capacity.

load of the distributed resources.

Load balancing is an important issue in distributed computing system, because the resources are geographically distributed computers or clusters which are aggregated logically to serve as a unified computing resource. Load balancing technique[16], when applied in a distributed system like grid, produces services with high availability, scalability, and predictability[17] characteristics. The conventional load balancing algorithms can not be harnessed successfully due to the obstacles caused by the real-time, heterogeneity, autonomy and dynamic nature of the transaction oriented grid service.

Load balanced scheduling consists of two primitive policies; transfer policy and location policy. Transfer policy determines the overload condition of the node. While location policy determines the underload condition of the node. Based on these two policies, load balanced scheduling is categorized as static and dynamic [13]. The static load balanced scheduling [14] takes decisions at compile time when resource requirements are already estimated while the dynamic load balanced scheduling takes instant decisions on the dynamic arrival of the transactions to reduce the waiting time and to improve the performance [13].

Therefore, load balanced transaction scheduling mechanism in on-demand computing system runs as follows: (i) receives new incoming transactions with their assigned deadlines, (ii) checks for available resources, (iii) selects the appropriate nodes according to availability and performance criteria and (iv) produces a planning of jobs to selected nodes within their deadlines [15].

1.7 Balanced Task Allocation

The balanced task allocation strategy in on-demand computing based transaction processing system which allocates the deadline-constrained transactions based on failure

[16]The work based load balancing approach was published as the paper entitled “Dynamic and adaptive load balancing in transaction oriented grid service” Green High Performance Computing (ICGHPC), 2016 2nd International Conference on, 2016, IEEE, https://doi.org/10.1109/ICGHPC.2016.7508067.

[17]Predictability is the degree to which a correct prediction or forecast of a system’s state can be made either qualitatively or quantitatively.

possibility of the distributed constituents becomes very important for the reliable execution of the transaction. The balanced task allocation is achieved by assigning the tasks to the balanced nodes. These nodes can be found by maximizing the resource availability in the system. The balanced task allocation in such distributed systems is not easy and is a non-deterministic polynomial-time hard (NP-hard)[18] problem. The problem becomes complicated by the fact that the resources of these systems may fail at any point of time.

1.8 Load Balanced Transaction Processing Model

The transaction processing model is depicted in **FIGURE 1.2**. The model consists of five modules. A set of the transaction requests T in the system is first generated. These transaction requests consist of a particular deadline $D(T_i)$ assigned to them. Every transaction T_i should execute within its assigned deadline $D(T_i)$. Otherwise it will rollback or abort. The load balanced transaction scheduling is applied to select the appropriate load balanced node from a set of n nodes N_j $(j = 1, 2, ..., n)$. Thus the probability of success of the transactions is increased within their prescribed deadline. The successful transactions are committed while unsuccessful ones are rolled back. After rollback of a transaction, if the deadline is not expired, the transaction is again sent to the new optimal processing node.

How each module of the model (**FIGURE 1.2**) works are given below:

- Transaction Generator: This module generates the transaction requests using exponential distribution. Transactions are sent to the resource manager for their needed resources. It receives the committed transactions and the acknowledgment of aborted transactions.

- Resource Manager: It manages the pool of resources that are available to the system, i.e., the scheduling of the nodes, network bandwidth, and disk storage. It manages

[18]NP-hardness (non-deterministic polynomial-time hardness), in computational complexity theory, is the defining property of a class of problems that are, informally, ”at least as hard as the hardest problems in NP”.

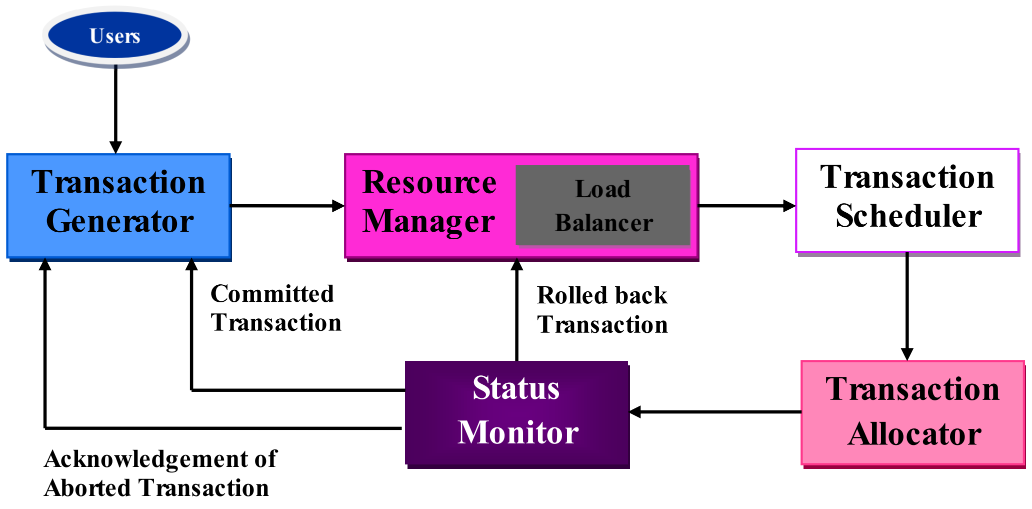

FIGURE 1.2: Transaction Processing Model with Load Balanced Scheduling

the resources for rolled back transactions also. Load Balancer is a sub-module of the resource manager which balances the load on each node of the system.

- Transaction Scheduler: It is responsible for mapping of the transactions to their suitable resources available in the system. As the system workload grows heavily, the number of transactions waiting in the queue increases. In this situation, the scheduler needs to perform load balancing also.
- Transaction Allocator: Based on the scheduling information, the load balanced nodes are selected for the execution of the transactions and prepare-messages are sent to them. After the lock is granted, the transaction allocator allocates the time schedule of the scheduled nodes to the transactions.
- Transaction Monitor: After the execution of the transactions at allocated nodes, their completion time is checked whether they are within their deadline or not. If they are within their deadline, the transactions are allowed to commit. Otherwise they rollback. Transactions will commit, rollback or abort is decided by transaction monitoring phase by using their respective deadlines.

The balanced task allocation algorithm in the on-demand computing system runs as follows: (i) receives new incoming transactions with their assigned deadlines, (ii) checks for available resources, (iii) selects the appropriate nodes according to availability and performance criteria and (iv) produces a planning of jobs to selected nodes within their deadlines [15], (v) Find out the availability of the system and lastly (vi) Find out the reliability.

1.9 Motivation and Challenges

A lot of work has been done on scheduling tasks of processes/jobs with load balancing and without load balancing in different distributed computing environments. Transactions have peculiar nature regarding their computation operation and their completion on various nodes of a grid processing system, and hence scheduling of transactions deserves special attention.

The issues such as throughput, reliability, resource availability, and performance in on-demand computing environment are very important user satisfaction factors [16]. The resource availability can be maximized by the balanced scheduling technique. The maximization of resource availability in on-demand computing based transaction processing system is crucial because it acts as the heart of the system [17, 18]. Since, the on-demand computing systems have the characteristics of heterogeneity[19], autonomy[20], scalability, and adaptability[21] [19], the maximization of resource availability in the task scheduling is a tough problem [3, 20, 21]. This problem remains an open research issue. Resource availability in such environment is very important for efficient scheduling of the tasks [16]. Maximization of resource availability is one of the objectives for efficient scheduling methods.

Performance of a system is the one of the key issues that service users' requests. Performance can be defined as considering the specification of the system and user expectations. The evaluation of performance in on-demand computing based transaction processing system mainly focuses on the service-time of the transactions, as transactions are deadline constrained tasks. In this respect, several related measures such as mean response time and mean waiting time of the tasks can be defined. But, there exists hardly any load balanced transaction scheduling method considering resource availability and performance in the literature. Therefore, we also formulate load balanced transaction

[19]The quality or state of being diverse in character or content.
[20]Autonomy is the quality or state of being self-governing.
[21]Adaptability is the ability of a system to adapt itself efficiently to changed circumstances.

scheduling model considering performability in on-demand computing environment. Load balanced transaction scheduling in on-demand computing system is a tough problem [3, 20]. This problem remains an open research issue.

The experiments in [22, 23, 24, 25] show that the consideration of reliability is also essential for dependability of the system. We formulate the reliability for transaction allocation problem in the system. The objective of the balanced task allocation is to minimize the load and to minimize the probability of failures of the transactions. There are mainly three terms which should be focussed while allocating the transactions; load, deadlines, and the probability of failures. It is a complex task to minimize all three objectives at the same time. Therefore, task allocation algorithm which can take into account all these three objectives must be devised. In order to achieve performance, reliability and consistency, data must be readily accessible in a data warehouse, backup procedures and the recovery process must be in place to deal with system failure, human failure, computer viruses, software applications or natural disasters. The reliability analysis and evaluation in grid transaction processing system becomes a challenging issue and attracts more and more attentions of researchers.

In such computing systems including grid, cloud, high-performance system, several works have been done on the topic of scheduling tasks of processes with load balancing as well as without load balancing. Since transactions have peculiar nature regarding their computation operation and their completion on various nodes of on-demand computing system, scheduling of transactions in such an environment deserves special attention.

The evaluation of performance in on-demand computing based transaction processing system mainly focuses on the service-time of the transactions, as transactions are deadline constrained tasks. In this respect, several related measures such as mean response time and mean waiting time of the tasks can be defined. But, there exists hardly any load balanced transaction scheduling method considering resource availability and performance in the literature. Therefore, one of our works focuses on load balanced transaction scheduling considering performability in on-demand computing environment. Load balanced transaction scheduling in on-demand computing system is a tough problem [3, 20, 21]. This problem remains an open research issue.

1.10 Dessertation Overview

The overarching goal of this dissertation is to propose models and algorithms which can improve the dependability attributes of on-demand computing based transaction processing system. First, we model the transaction processing and formulate throughput, resource utilization, load, makespan, and miss ratio of transactions. Considering this model, we propose a load balanced transaction scheduling algorithm which improves the throughput, resource utilization of the system. Secondly, we formulate the availability of the system and then propose another load balanced scheduling algorithm which improves availability of the system. After that we formulate performability of the system and propose a load balanced transaction scheduling algorithm which improves performability of the system. Lastly, we formulate reliability with the help of the availability of the system and propose a load balanced transaction allocation algorithm for improving the reliability of the system. All the algorithms in this dissertation are based on soft computing approach. We discuss the role and necessity of using these soft computing approaches. We also compare these scheduling techniques with other existing techniques. FIGURE 1.3 shows the overview of our dessertation's contributions.

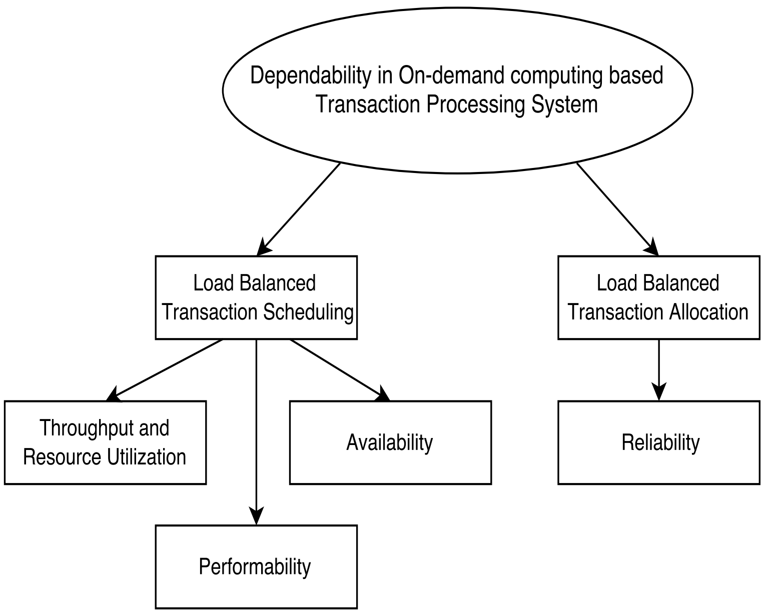
Dependability in On-demand computing based Transaction Processing System
Load Balanced Transaction Scheduling
Load Balanced Transaction Allocation
Throughput and Resource Utilization
Availability
Reliability
Performability

FIGURE 1.3: Overview of the dissertation's contributions

1.11 Contribution and Impact

This dissertation makes several fundamental contributions towards the improvement of dependability in on-demand computing based transaction processing system. Our contributions significantly advance the state-of-the-art by supporting transaction processing system with the improvement of throughput, availability, performability, and reliability of on-demand computing system. We also incorporated **deadline-miss fault** while formulating the availability, performability, and reliability of real-time tasks. **FIGURE 1.3** summarizes these contributions. We now highlight these contributions and their impact.

- We propose a transaction processing algorithm based on Ant Colony Optimization (LBTS_ACO)[22] [26] to solve the load balanced transaction scheduling problem by optimizing load and makespan [26]. We also consider other influential parameters such as node utilization, transaction throughput, transaction miss ratio, and speedup of load balancing. This algorithm balances the load before scheduling the transactions to the nodes. To compare the proposed algorithm, we implement some meta-heuristic algorithms such as Extremal Optimization (EO) [2, 27, 28] and Genetic Algorithm (GA) [29, 30, 31, 32, 33, 1], and also some heuristic algorithms such as HLBA (Hierarchical Load Balanced Algorithm) [34], DLB (Decentralized Load Balancing) [35], and Randomized [36] for transaction processing. In the field of the scheduling problem, we see several efforts with improved ACO algorithms. All these improved algorithms change only pheromone update mechanism. But the improvement in search speed and solution efficiency lead to premature convergence.
- The research in task scheduling for maximizing resource availability [37] in on-demand based computing based transaction processing system is considerably rare. We need an algorithm which not only maximizes the resource availability but also minimizes the makespan of the tasks. Therefore, we need the algorithm which can solve the bi-objective optimization problem. In order to meet and overcome these challenges, we need to formulate the resource availability for on-demand

[22]This work was published as the paper entitled "On Scheduling Transactions in Grid Processing System considering load through Ant Colony Optimization" in Applied Soft Computing, Elsevier, SCI, Impact Factor: 3.541, https://doi.org/10.1016/j.asoc.2017.08.047.

computing based transaction processing system and propose an algorithm. Thus we propose ant colony optimization based scheduling algorithm (MATS_ACO) [23] to maximize the resource availability and minimize the makespan. We compare MATS_ACO algorithm with some meta-heuristic algorithms such as Extremal Optimization (EO) [2, 27, 28] and Genetic Algorithm (GA) [29, 30, 31, 32, 33, 1].

- For maximizing the performability of the system, we first formulated the performability of the system using steady-state availability of transaction and then propose a load balanced transaction scheduling algorithm based on honey bee optimization (LBTS_HBO)[24]. In this work we incorporated a new fault, **deadline-miss fault**, while formulating the performability. In this method, first, the load of the system is balanced and then the transactions are scheduled using foraging behavior of honey bees to find the optimal solutions. We also modify four known scheduling algorithms such as Ant Colony Optimization (ACO), Hierarchical Load Balanced Algorithm (HLBA), Dynamic and Decentralized Load Balancing (DLB), and Randomized to obtain transaction scheduling algorithms for the purpose of comparison with our proposed algorithm.

- Maximization of the reliability is another challenge in on-demand computing. In order to meet and overcome this challenge, we again use meta-heuristic algorithm. This time we use load balanced task allocation algorithm to fulfill the requirement. For maximizing the reliability of on-demand computing based transaction processing system, we propose a load balanced transaction allocation algorithm

[23]This was published as the paper entitled "Maximizing availability for task scheduling in on-demand computing based transaction processing system using ant colony optimization" in Concurrency and Computation: Practice and Experience, Wiley Online Library, SCIE, Impact Factor: 1.133, DOI:10.1002/cpe.4405.

[24]The work reported in this chapter was published as the paper entitled "Load Balanced Transaction Scheduling using Honey Bee Optimization Considering Performability in On-demand Computing System" in Concurrency and Computation: Practice and Experience, Wiley Online Library, SCIE, Impact Factor: 1.133, DOI:10.1002/cpe.4253.

based on social spider optimization (LBTA_SSO) method[25]. The balanced task allocation in such environment is known to be an NP-hard. The reliability is a measure of trustworthiness of the system while executing the task. In addition, we formulate the reliability by incorporating a new fault **deadline-miss fault** and using steady-state availability of the system. We modify five existing algorithms to obtain the task allocation algorithms; Honey Bee Optimization (HBO), Ant Colony Optimization (ACO), Hierarchical Load Balanced Algorithm (HLBA), Dynamic and Decentralized Load Balancing (DLB), and Randomized Algorithm respectively. Then, we compare the proposed algorithm with these modified algorithms. We compare our algorithm on two different platform; grid and cloud.

1.12 Organization

The organization of this thesis is as follows. In Chapter 2, we provide a systematic survey and analysis of the state-of-the-art in load balanced scheduling and on-demand computing based transaction processing. Chapter 3 presents load balanced scheduling approach using soft computing. This chapter focuses on the parameters such as throughput, makespan, resource utilization, load, and miss ratio. Chapter 4 presents a meta-heuristic scheduling algorithm for maximizing the availability of on-demand computing based transaction processing system. Chapter 5 presents a meta-heuristic based load balanced scheduling algorithm for maximizing availability and performability of on-demand computing based transaction processing system. Chapter 6 presents a meta-heuristic based load balanced task allocation algorithm for maximizing the reliability of on-demand computing based transaction processing system. Chapter 7 concludes this dissertation and outlines some open challenges.

[25]This work reported in this chapter based on on-demand computing system was published as the paper entitled “Balanced task allocation in the on-demand computing based transaction processing system using social spider optimization” in Concurrency and Computation: Practice and Experience, Wiley Online Library, SCIE, Impact Factor: 1.133, DOI: 10.1002/cpe.4253.