Chapter 1

Introduction

Distributed computing is often used to describe a type of computing, where computers are not only networked, but also smartly distribute their workload across each computer so that they can provide dependable, consistent and pervasive access to high-end computation.

The new inventions and developments in the distributed system design, collaborative environments, high performance computing and high throughput computing made Grid the logical step thereafter. This led to the emergence of Grids in the 1990s, which collaborate resources from multiple organizations to fulfill the computing needs of applications with varying resource requirements. The participating organizations together form Virtual Organization (VO) and pool their resources into a common shared resource pool. The resources in this shared Grid infrastructure are used for applications which would have been otherwise impossible without massive computing power.

This chapter provides a high level view of the thesis. It discusses the fundamental concept behind the Grid technology, its evolution, Grid architecture, its key areas along with the major issues of this area. It further provides the motivation to propose resource provisioning and scheduling framework for Grid systems. It culminates with discussion of the organization of the rest of the thesis along with its contributions.

1.1 Grid Computing : An Overview

The performance and reliability of IT networks has led to the idea of extending the concept of remote computing to a significantly large set of users, similar to the provisioning of electric power to every user even in the most remote places with the help of power nets. Borrowing the name from the electrical power infrastructure, Grid computing was born in the mid 1990s [1]. Grid computing originated as a distributed paradigm for scientific high performance computing (Distributed Supercomputing, High-Throughput Applications, Data-Intensive Applications, etc.) and as an alternative to expensive supercomputers by virtually joining a large number of interconnected computers.

The term Grid was coined to describe technologies that would allow consumers to obtain computing power on demand. Grid computing can be described as a distributed computing paradigm in which virtualized applications, softwares, platforms, computation and storage can be provisioned, scaled and released instantly through the use of self-manageable services [2][3]. The technology flourished as it allowed consumers to obtain computing power on-demand by offering computing as a utility to the consumers.

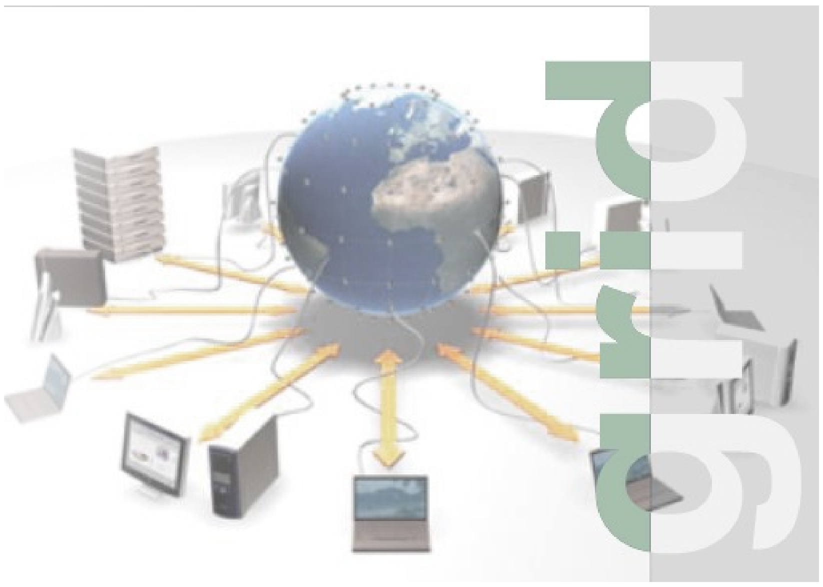

Figure 1.1: Simple View of Grid[2]

Grid has emerged as a computing paradigm for solving grand challenge applications in science, engineering and economics through the sharing and collaboration of numerous heterogeneous resources [4]. Grid computing connects computers that are scattered over a wide geographic area, allowing their computing power to be shared, which implies coordinated resource sharing and problem solving by multi-institutional VOs as shown in Figure 1.1 [5][6]. Grid computing is a promising technology for providing a uniform and transparent access to geographically dispersed computing resources, such as computers, databases, experimental equipments and observational equipments [7]. Although it has been used within the academic and scientific community; its standards, enabling technologies, toolkits and products are readily available that allow businesses to use and reap the advantages of Grid computing.

The concept of Grid computing started as a project to link supercomputing sites, but now it has grown far beyond its original intent [8][9]. Grid computing not only provides the resources that allow our scientists to manage the infinite collection of data but it also allows this data to be distributed all over the world, which means that the scientific teams can work on international projects together and simultaneously.

On the basis of the utilities, Grid can be categorized as follows [10]:

i. Computational Grid: A Computational Grid is a hardware and software infrastructure that allows components of information technology infrastructure, computational capabilities, databases, sensors, and people to be shared flexibly as a true collaborative tool[11]. Computational Grid coordinates resources that are not subject to centralized control using standard, open general purpose protocols and interface to deliver nontrivial Quality of Service (QoS).

ii. Data Grid: A Data Grid is utilized for large-scale data storage and management of data-intensive services. It also provides the main infrastructure to store and handle large amount of data that is required by many scientific and engineering applications. The main aim of data Grid is the management and controlled sharing of large amounts of distributed data; however, different data can have their own formats. The European Data Grid is one of the Grids which provides the facility to large projects like Large Hadron Collider Computing Grid (LCG) [12]. The Grid PhyN [13], Tera Grid [14] [15] and PPDG [16] data Grid are other well known examples of data Grids.

iii. e-Science Grid: e-Science Grids are known for providing solutions to problems arising in fields like medicine, finance, weather forecast and engineering, etc. Such Grids give support to the computational infrastructure (access to computational and data resources) needed to solve many complex problems arising in the areas of science and engineering. Representative examples are

EGEE Grid Computing [17], UKe-ScienceGrid [18], German D-Grid [19], BIGGRID (the Dutch e-Science Grid) [20] and French Grid'5000 [21].

iv. Enterprise Grid: An Enterprise Grid can be loosely defined as a distributed system that aims to dynamically aggregate and co-ordinate various resources across the enterprise and improve their utilization such that there is an overall increase in productivity. Enterprise Grids enable running several projects within one large enterprise to share resources in a transparent way. Examples of enterprise Grids are Sun Grid Engine (SGE) [22], IBM Grid [23], Oracle Grid [24] and HP Grid [25].

v. Knowledge Grid: These are Grid-based environments that enable interoperation among users, applications, and resources to effectively manage knowledge resources used in virtual organizations, e-learning, online collaboration, etc.

vi. Desktop Grids: Desktop Grids are a new form of enterprise Grids emerging in institutions, which use the idle cycles of desktops. Small enterprises and institutions are usually equipped with hundreds or thousands of desktops, mainly used for office tasks. These PCs are thus a good source for setting up a Grid system for the institution. In this case, the particularity of the Grid system is its unique administrative domain, which makes it easier to manage due to the low heterogeneity and volatility of resources. Of course, the desktop Grid can cross many administrative domains and in this case the heterogeneity and volatility of the resources is not an issue, as in a general Grid system setting.

vii. Application Grid: Application Grid provides application server level infrastructure for the processing of applications thus allowing them to meet their goals and increase performance by resource sharing.

After an overview of Grid computing, the next section would be discussing its evolution till date.

1.1.1 Evolution of Grid Computing

The popularity of the Internet and the availability of powerful computers and high-speed networks as low-cost commodity components are changing the way we use computers today [26][9]. Distributed computing is transforming the IT landscape towards an increasingly global and knowledge-based economy. Distributed computing is critical to the development of new, innovative and scalable applications and infrastructures that helps in the advancement of commerce and science. Grid computing emerged in the second half of the 1990s as a new computing paradigm for advanced science and engineering [6][27]. The term Grid, however, may mean different things to different people. To some users, a Grid is any network of machines, including personal or desktop computers within an organization. To others, Grids are networks that include computer clusters, cluster of clusters, or special data sources. As defined by Foster & Kesselman in 1999, "The word "Grid" is chosen by analogy with the electric power Grid, which provides pervasive access to power and thus having a dramatic impact on human capabilities and society". Four distinct phases of this evolution have been illustrated in Figure 1.2.

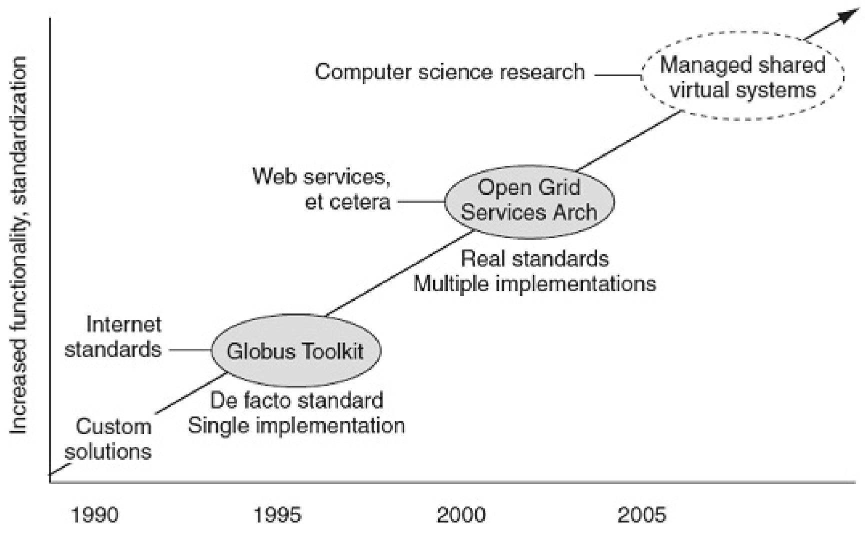

Figure 1.2: The Evolution of Grid Technologies [27]

Starting in the early 1990s, work in "metacomputing" and related fields involved the development of custom solutions to Grid computing problems. From 1997 onwards, the open source Globus Toolkit version 2 (GT2) emerged as the defacto standard for Grid computing. Focusing on usability and interoperability, GT2 defined implemented protocols, APIs and services used in thousands of Grid deployments worldwide. The year 2002 saw the emergence of the Open Grid Services Architecture (OGSA), a true community standard with multiple implementations, including the OGSA-based GT 3.0, in particular, which was released in 2003. The definition of the initial OGSA is an important step forward, but much more remains to be done before the full Grid vision is realized [27]. For more than ten years, Grid computing has been promoted as the global computing infrastructure of the future. Today advances in distributed computing have enabled the creation of national and international Grids such as the TeraGrid [14]

[15], Open Science Grid [28], Enabling Grids for E-sciencE (EGEE) [17] , APAC-Grid in Australia [29], K*Grid in Korea [30], NAREGI in Japan [31], Garuda in India [32], E-Science Grid in the UK [18], Our Grid in Brazil [33], Grid'5000 [21] and Auver Grid in France [34] [35] and DAS in the Netherlands [36] [37].

With this background, the next section discusses the Grid architecture.

1.1.2 Grid Architecture

Grid architecture [2] is often described in terms of layers, where each layer has a specific function. The higher layers are generally user-centric, whereas the lower layers are more hardware-centric. Figure 1.3 shows the layered Grid architecture and its relationship to the Internet Protocol architecture [2]. Grid architecture comprises of five layers: Fabric, Connectivity, Resource, Collective and Applications as shown in Figure 1.3.

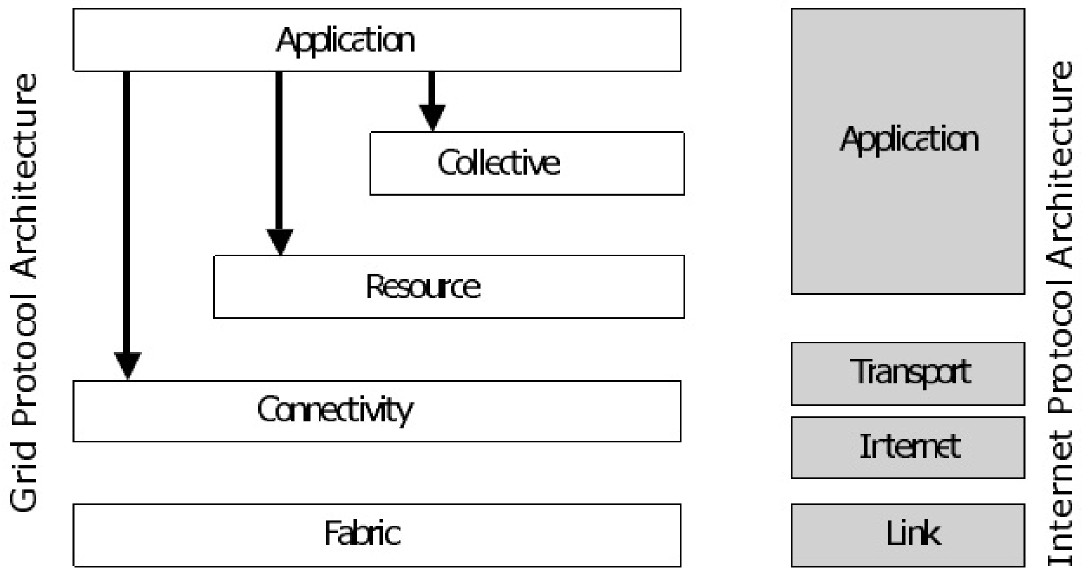

Figure 1.3: The layered Grid architecture and its relationship to the Internet protocol architecture [2]

Fabric layer: The Grid Fabric layer provides the resources to which shared access is mediated by Grid protocols, for example, computational resources, storage systems, catalogs, network resources, sensors etc.

Connectivity Layer: The Connectivity layer defines core communication and authentication protocols required for Grid-specific network transactions.

Resource layer: The role of the Resource layer is to implement core communications and authentication protocols call on fabric layer functions to access and control local resources. Resource layer protocols are concerned entirely with indi-

vidual resources.
Collective layer: Collective layer contains protocols and services that are not associated with any one specific resource but instead capture interactions across a collection of resources.
Application layer: The final layer in Grid architecture comprises of the user applications that operate within a VO environment.

The Grid architecture is based on Grid characteristics as discussed in the next section.

1.1.3 Grid Characteristics

Grids share certain common characteristics of distributed computing as enlisted below [38] [26]:

- Heterogeneity: The resources in Grid are heterogeneous in nature. The extent of heterogeneity spans across multiple institutions that are a part of Grid computing resources like processors, data storage devices, and bandwidth vary with different resource providers.

- Adaptability & Scalability: Grid adapts itself to the changing user needs. However, the distinguishing characteristic of Grid is scalability as per the user demand. Resources may scale up equally well as they scale down with the changing user's needs.

- Resource Sharing: Grid is based on virtualization to enable resource sharing. Heterogeneous resources located across multiple administrative domains are pooled to fulfill the resource needs of user applications.

- Dynamic: Grids are dynamic in nature. Resources can join or leave the computing environment any time as per the requirement.

- Security: Grid computing allows users to access a shared infrastructure. Users must be authenticated and authorized to maintain the confidentiality and integrity of data and shared resources.

After an introduction to Grid characteristics, next section will discuss the Grid standards.

1.1.4 Grid Standards

There are many standards involved in building a Grid architecture, which form the basic building block that allow applications to execute service requests. In this section, prevalent open Grid standards used for implementing Grid are discussed which are as follows:

- Web services: Grid services, defined by OGSA, is an extension of web services. So, Grid service can leverage the available web service specifications[39]. Four basic types of web services used in Grid systems are described as follows:

  a. Extensible Markup Language (XML): XML is a markup language that forms the basis of web services. It is used to store and transport the data.

  b. Simple Object Access Protocol (SOAP): It is an XML based, platform independent protocol providing simple and relatively light weight mechanism for exchanging structured information in the implementation of web services in computer networks like Grid.

  c. Web Service Description Language (WSDL): WSDL[40] is an XML based language used for describing the model of web services.

  d. Universal Description, Discovery, and Integration (UDDI): UDDI [41] protocol is approved by the Organization for the Advancement of Structured Information Standard (OASIS) as a standard for service registries in the context of SOA. It is a key member of the web services stack and defines the ways in which web services are published and discovered across the network.

- Open Grid Service Architecture (OGSA): OGSA represents an evolution towards a Grid system architecture based on web services concepts and technologies. The Global Grid Forum (GGF) has embraced the OGSA as the blueprint for standards-based Grid Computing [3]. OGSA is a Service-Oriented Architecture (SOA) that addresses the need for standardization by defining a set of core capabilities and behaviors to address the key concerns in Grid systems [42]. OGSA provides services such as resource discovery, resource provisioning, resource management, data management, information services and security.

- Open Grid Services Infrastructure (OGSI): OGSI [43] defines mechanisms for creating, managing, and exchanging information among entities called Grid services. OGSI provides the features that are needed for the implementation of the Grid services. It also provides the facility of Web Service Description Language (WSDL) that defines Grid services.

- Web Service Resource Framework (WSRF): WSRF is the de-facto standard for modeling and accessing resources using web services [44].
- Open Grid Service Architecture-Data Access and Integration (OGSA-DAI): OGSA-DAI project allows data resources to be federated and accessed via web services on the web or within Grids. With the help of these web services, data can be queried, updated, transformed and combined by different methods [45].

1.1.5 Grid Applications

A Grid application is a collection of work items to solve a certain problem or to achieve desired results using a Grid infrastructure. For example, Grids generally support many different kinds of applications, ranging from High Performance Computing (HPC) to High Throughput Computing (HTC) [46]. Grids evolved to comply with the computing needs of high performance scientific applications. As a result much of the driving force and assistance for Grids is provided by the academia. A Grid application can be the simulation of business scenarios, like stock market development, that requires a large amount of data as well as a high demand for computing resources in order to calculate and handle the large number of variables and their effects [47]. In other words, a Grid application may consist of a number of jobs that together fulfill the whole task.

Grid is utilized in a number of areas including:

- Molecular modelling for drug design
- Neuro science brain activity analysis
- Cellular microphysiology
- High Energy Physics and the Grid Network (HEP Grid)
- Access Grid
- Globus Applications
- The International Grid (iGrid) and many more

Other applications like LHC Computing Grid (LCG) [12], The Large Hadron Collider will produce roughly 15 petabytes (15 million gigabytes) of data annually enough to fill more than 1.7 million dual-layer DVDs a year. Thousands of scientists around the world at CERN, the European Organization for Nuclear Research want to access and analyze this data, so CERN organized LCG project by collaborating with institutions in 34 different countries to operate a distributed computing and data storage infrastructure [12]. Similarly other applications like Enabling Grid for E-science (EGEE) [17], SETI@home [48], etc require huge amount of power, usually over a short period of time. Grid provides the required computation power through the sharing of computational resources. Apart from these, Life sciences, Biology, Aerospace, Earth sciences and E-commerce are other grand challenge applications of Grid computing .

After discussing the Grid essentials such as Grid architecture, its characteristics and standards, the next section would focus on the key challenges of this area.

1.2 Grid Computing Key Issues

Grid computing also presents certain issues that needs to be addressed similar to the issues in other emerging technologies. Current obstacles related to the growth and adoption of Grid computing are:

- Data Management
- Resource Management and Scheduling
- Security

1.2.1 Data Management Issues

Data management is one of the key features of Grid computing, specifically for data Grid in which large amount of data are distributed over dynamic and remote sites, potentially all over the world [49]. Data submission, data location, data dissemination, data replication, data consistency control, multi-protocol reliable file transfer, autonomic data management, data security and privacy are the main issues of data management [50][51]. Data can be static or dynamic in nature. Static data cannot be modified and updated but can only be read and analyzed. On the flip side, dynamic data can be modified and updated. Data replication on distributed infrastructures is a static data problem while E-business type applications belong to dynamic data type category [39]. A data Grid must allow dynamically scalable storage services to store data in such a way that data intensive applications can be handled efficiently and effectively.

1.2.2 Resource Management and Scheduling Issues

Resource management plays a vital role in Grid computing. Resource management issues often need to be addressed as the complexity of Grid increases. Grid resource management requires discovery, monitoring and scheduling of resources in an efficient manner and at an advance level. The resource management system also provides functionalities such as QoS that enables guaranteed provisioning of resource capacity [52]. An efficient functioning of a complicated and dynamic Grid environment requires resource manager to monitor and identify the idling resources and to schedule user's submitted jobs (or programs) accordingly [53]. In general, Grid scheduling is the process of mapping Grid jobs to resources over multiple administrative domains. Traditionally the scheduler is responsible for resource discovery, resource trading, resource selection and job assignment, but emerging deadline driven Grid applications require access to several resources and predictable QoS [54]. The resource allocation and job scheduling mechanism used at the global and local level play a crucial role for the performance and availability of Grid applications [6]. QoS is a constraint imposed on the scheduling process instead of the final objective function [26]. In Grid resource management system, QoS management aims to provide assurance for accessing resources, while maintaining the security level between domains. Scheduling should be done by considering the QoS parameters like cost of the resources at the time of job execution. QoS requirements must be fulfilled according to the user and provider's satisfaction. In such a complex, dynamic and distributed environment, resource provisioning and resource scheduling are the key issues which further need to be addressed, to improve the resource usage efficiency on the Grid [53].

1.2.3 Security Issues

Grid is increasingly being taken up and used by all sectors of business, industry, academia and the government as the middleware infrastructure of choice. So, Grid security is an essential aspect of its overall architecture [55]. Security within the Grid environment is driven by the need to support scalable, dynamic and distributed VOs.

Considering the Grid environment's diverse and geographically separated resources and wide variety of users, each with unique needs and goals for the Grid

system, the issue of managing the security of users and resources becomes an issue [56]. There are major security issues in Grid systems which can be categorized in the following disciplines: authentication, delegation, single sign-on, credential lifespan and renewal, confidentiality, message integrity, non repudiation, secure logging, privacy, trust, policy exchange, authorization, assurance and manageability [57]. Security is also of main concern at the time of job submission. OGSA Security Roadmap [58] itemizes the various security services: auditing, anonymity, credential processing services, credential conversion services, authorization, identity mapping services, etc. to provide the security in Grid computing. Trust being the fundamental and common security aspect at the time of resource sharing in Grid environments led to the development of several trust models.
This section discussed the major issues of Grid computing. The next section discusses the Resource Provisioning and Scheduling in the Grid resource management systems, which has been chosen as the area of research for this thesis.

1.3 Grid Resource Provisioning and Scheduling : The Research Motivation

A number of issues need to be dealt with in the area of Grid resource provisioning and scheduling. Resource provisioning allocates the resources dynamically and it also works at both the user's and provider's level. Resource provisioning and scheduling have been picked up due to the reasons discussed in the subsequent paragraphs. The Grid computing model, where resources tend to be both heterogeneous and distributed across multiple management domains, faces all the traditional IT management issues. It also brings new challenges - not only in the management of its component resources, but also of the Grid itself. Due to this reason, many issues come to the surface, which have been summarized as follows:

- The resource owners and the users have different goals, objectives, strategies, and demand patterns. Compared to traditional distributed systems, the resource provisioning is confronted with the complication that, even if resources are granted to a job, these resources may be withdrawn or disappear before the job actually uses them. Moreover, it is necessary to cater to the users' needs optimally which may fluctuate from time to time [59].

- In Grid resource management system, QoS management aims to provide assurance for accessing resources, while maintaining the security level between domains. Scheduling can be done by considering the optimization of QoS parameters like cost, time etc of the resources at the time of job execution.

- The complex, heterogeneous and dynamic nature of Grid systems presents new challenges in resource management such as the provision of QoS parameters like cost, time, reliability and security to resource consumers.
- In Grid resource scheduling, it's not an easy task to allocate resources. There is no facility of accessing the resources by calling an intermediate agent and the resources are available in two forms viz stable and un-stable. Moreover, the scheduling algorithms need to have less communication overhead and should meet user's QoS requirements in comparison to any other conventional algorithms available [26].
- Scheduling task on a set of heterogeneous and dynamically changing resources is itself a complex problem that requires sophisticated algorithms that take into account multiple-optimization criteria [5].
- In providing QoS, Grid resource management system must be able to handle resource provisioning and scheduling in an efficient manner. Investigation into related areas; such as real-time resource allocation strategies and capacity planning remains important research areas to support collaborative applications- such as those that require computational steering support [60].
- Due to the dynamic nature and an uncertainty of Grid computing system behavior, QoS depends crucially on the selection of appropriate subset of the available resources [4].
- A computational Grid must allow resource owners and resource consumers to make autonomous scheduling decisions and both parties must have sufficient incentives to stay and play in the Grid [61].
- One of the primary goals of the Grid is to provide non-trivial QoS which is very important for the stability distributed resources [60].

Due to all these factors, the resource provisioning and scheduling issues specific to the Grid environment have been the motivation behind this work. The next section discusses the organization of this thesis

1.4 Thesis Organization

After giving an Introduction to the thesis in chapter 1, the rest of the thesis is structured as follows:

Chapter 2 presents the literature survey on Grid resource management. Resource provisioning strategies and resource provisioning with and without QoS has been discussed. A detailed description of the existing Grid scheduling work has been presented. Existing scheduling techniques in different middleware have also been discussed. A taxonomy of Grid schedulers, types of Grid scheduling algorithms and heuristic approaches for Grid scheduling have been presented and a comparison of different heuristic techniques has also been discussed. The chapter finally concludes with the Problem Formulation. Chapter 2 is partially derived from:

- **Rajni**, Chana, I., "Resource Provisioning and Scheduling in Grids: Issues, Challenges and Future Directions", in IEEE International Conference on Computer and Communication Technology (ICCCT10), pp:306-310, MNNIT, Allahabad, September 17-19, 2010.
- **Aron R.**, Chana I., "Enhancement of Resource Provisioning in Grid Resource Management Systems", 11th annual Grace Hopper Celebration of Women in Computing, organized by Anita Borg Institute for Women and Technology and ACM, Oregon Convention Center, Portland, Oregon, November 9-12, 2011.

Chapter 3 describes the proposed Resource Provisioning and Scheduling Framework and the goals which it tends to achieve. The requirements analyzed on the basis of QoS parameter(s) have been illustrated in detail. Mode of Operation of QoS based Resource Provisioning and QoS parameters based Resource Provisioning Policies corresponding to resource provisioning and scheduling framework have been presented. Chapter 3 derives from:

- **Aron R.**, Chana I., "QoS based Resource Provisioning and Scheduling in Grids", Journal of Supercomputing, **Springer**, DOI 10.1007/s11227-013-0903-1, Impact Factor : 0.578, 2013.
- **Aron R.**, Chana I., "Formal QoS Policy based Grid Resource Provisioning Framework", Journal of Grid Computing, **Springer**, Impact Factor : 1.310, Vol 10, No 2, Pages: 249-264, June 2012.
- **Aron R.**, Chana I., "Cost based Resource Provisioning Policy for Grids", in International Conference on Parallel and Distributed Computing, Proceedings of the World Congress on Engineering , (Awarded Certificate of Merit) Vol II, WCE 2011, July 6 - 8, 2011, London, U.K.

- **Aron R.**, Chana I., “Resource Provisioning for Grids: A Policy Perspective”, in International Conference on Contemporary Computing, LNCS, Springer, pp-546-547, JIIT University Noida, August 8-10, 2011.

Chapter 4 presents design of resource scheduling algorithm based on resource provisioning policies. This chapter provides the objective function formulation that takes care of the interests of both resource providers and resource consumers. A resource scheduling algorithm based on Bacterial Foraging Optimization (BFO) has been proposed in this chapter. Details about the proposed algorithm are discussed in this chapter. Chapter 4 derives from:

- **Aron R.**, Chana I., “QoS based Resource Provisioning and Scheduling in Grids”, Journal of Supercomputing, **Springer**, DOI 10.1007/s11227-013-0903-1, 2013, Impact Factor : 0.578.

- **Rajni**, Chana I., “Bacterial Foraging based Hyper-heuristic for Resource Scheduling in Grid Computing”, Future Generation of Computer Systems (FGCS), **Elsevier**, Vol 29, No 3, pp. 751-762, March 2013, Impact Factor: 1.978.

- **Rajni**, Chana I., “Resource Provisioning Policy based Scheduling for Grid Environment”, 12th annual Grace Hopper Celebration of Women in Computing, organized by Anita Borg Institute for Women and Technology and ACM, Baltimore Convention Center, Baltimore, Maryland, October 3-6, 2012.

Chapter 5 illustrates the verification of the framework. Further, the framework has been validated and compared with the existing systems. Verification of the framework and QoS parameters(s) based resource provisioning policies have been done through Z formal specification language. Experimental results have been collected from the implementation of the policies using the GridSim toolkit. This chapter provides the design and implementation details of the proposed algorithm. The implementation of the proposed algorithm has been done in a Grid Environment using Ali’s simulation model. Chapter 5 derives from:

- **Aron R.**, Chana I., “QoS based Resource Provisioning and Scheduling in Grids”, Journal of Supercomputing, **Springer**, DOI 10.1007/s11227-013-0903-1, 2013, Impact Factor : 0.578,

- **Rajni**, Chana I., “Bacterial Foraging based Hyper-heuristic for Resource Scheduling in Grid Computing”, Future Generation of Computer Systems (FGCS), **Elsevier**, Vol 29, No 3, pp. 751-762, March 2013, Impact Factor: 1.978.

- **Aron R.**, Chana I., "Formal QoS Policy based Grid Resource Provisioning Framework", Journal of Grid Computing, **Springer**, Vol 10, No 2, pp: 249-264, June 2012, Impact Factor: 1.310.

Chapter 6 finally concludes the thesis and discusses the future scope of the work.

1.5 Thesis Contribution

This Thesis contributes in the following ways:

- It critically analyzes the detailed literature of Grid resource provisioning and scheduling along with a detailed study of Grid schedulers and Grid middleware. It provides a qualitative comparison of the existing heuristic approaches with respect to their parameters like convergence, premature convergence, service, etc.
- QoS parameters have been identified to propose a resource provisioning and scheduling framework.
- To address the challenges of resource provisioning and scheduling in Grid resource management, a Resource Provisioning and Scheduling Framework has been proposed that offers resource provisioning policies which cater to provisioned resource allocation and resource scheduling.
- The proposed QoS parameter(s) based Resource Provisioning Policies have been designed using XML and the implementation of the policies has been shown in resource provisioning and scheduling framework.
- The usability of the proposed policies, their validation has been shown using Z formal specification language. Formal specification and verification of the framework helps in predicting possible errors before scheduling process itself and thus results in an efficient resource provisioning and scheduling of Grid resources.
- Grid resource scheduling problem has been formulated in the form of combinatorial optimization problem. The scheduling problem has been considered from both the user's and resource provider's point of view.
- Bacterial Foraging Optimization (BFO) based hyper-heuristic resource scheduling algorithm for scheduling of application with QoS constraints has been

proposed. Its main aim is to minimize the cost and execution time simultaneously by considering both the resource consumer's and resource provider's benefits. The algorithm has been evaluated by comparing it with other well-known scheduling algorithms.

- To demonstrate the validation of the proposed algorithm, it has been implemented in the GridSim toolkit using Ali's simulation model, the existing benchmark for heterogenous environments.

- The experimental results show that the proposed algorithm outperforms in comparison to the existing resource scheduling algorithms in all respects such as effect of resource heterogeneity, number of applications, number of resources, etc.

- Statistical analysis of simulation output has been performed in order to assess the accuracy of the estimated performance indices by using coefficient of variation. The coefficients of variation have been calculated for cost and makespan results achieved by the BFO based hyper-heuristic resource scheduling algorithm and existing scheduling algorithms.