Chapter - 3
CHAPTER 3

PROBLEM SPECIFICATION AND METHODOLOGY

3.1 Introduction

Research is a systematic inquiry to describe, explain, predict and control the observed phenomenon. Research involves inductive and deductive methods. Inductive methods analyze the observed phenomenon and identify the general principles, structures, or processes underlying the phenomenon observed; deductive methods verify the hypothesized principles through observations. The purposes are different: one is to develop explanation, and the other is to test the validity of the explanation. The heart of research is not on statistics, but the thinking behind it. Arguments are built about ideas and concepts, and evidences are provided to support to persuade people to accept those arguments.

3.1.1 Types of Research

Research can be classified in many different ways on the basis of the methodology of research, the knowledge it creates, the user group, the research problem it investigates, etc.

3.1.1.1 Basic Research

Basic, pure, fundamental research is the research which is done for knowledge enhancement, the research which does not have immediate commercial potential, the research which is done for human welfare, animal welfare and plant kingdom welfare. The main motivation is to expand man's knowledge, not to create or invent something. According to Travers, “Basic Research is designed to add to an organized body of scientific knowledge and does not necessarily produce results of immediate practical value.” Such a research is time and cost consuming.
3.1.1.2 Applied Research

Applied research is designed to solve practical problems of the modern world, rather than to acquire knowledge for its own sake. The goal of applied research is to improve the human condition. It focuses on analysis and solving social and real life problems. This research is generally conducted on large scale basis, it is expensive. As such, it often conducted with the support of some financing agency like Government, Public Corporation, World Bank, UNICEF, UGC, Etc. According to Hunt, “applied research is an investigation for ways of using scientific knowledge to solve practical problems” for example:- improve agriculture crop production, treat or cure a specific disease, improve the energy efficiency at homes and offices, how communication among workers in large companies can be improved. Applied research can be further classified as problem oriented and problem solving research.

Action Research: It is performed in classroom by teacher and the student on subject.

3.1.1.3 Problem oriented research

Research is done by industry apex body for sorting out problems faced by all the companies. Eg:- WTO does problem oriented research for developing countries. In India agriculture and processed food export development authority (APEDA) conducts regular research for the benefit of agri-industry.

3.1.1.4 Problem solving

This type of research is done by an individual company for the problem faced by it. Marketing research and market research are the applied research. For eg:- videocon international conducts research to study
customer satisfaction level. It is problem solving research. In short, the main aim of applied research is to discover some solution for some pressing practical problems.

3.2 Quantitative Research

This research is based on numeric figures or numbers. Quantitative research aims at measuring the quantity or amount and compares it with past records and tries to project for future periods. In social sciences, “quantitative research refers to the systematic empirical investigation of quantitative properties and phenomena and their relationships”. The objective of quantitative research is to develop and employ mathematical models, theories or hypothesis pertaining to phenomena.

The process of measurement is central to quantitative research because it provides fundamental connection between empirical observation and mathematical expression of quantitative relationships. Statistics is the most widely used branch of mathematics in quantitative research. Statistical methods are used extensively within fields such as economics and commerce.

Quantitative research involves the use of structured questions, where the response options have been Pre-determined and large number of respondents eg:-total sales of soap industry in terms of rupees cores and or quantity in terms of lakhs tones for a particular year, say 2008, could be researched, compared with the past 5 years and then projection for 2009 could be made.
3.3 Qualitative Research

Qualitative research presents non-quantitative type of analysis. Qualitative research is collecting, analyzing and interpreting data by observing what people do and say. Qualitative research refers to the meanings, definitions, characteristics, symbols, metaphors, and descriptions of things. Qualitative research is much more subjective and uses different methods of collecting information, mainly individual, in-depth interviews and focus groups.

The nature of this type of research is exploratory and open ended. Small number of people is interviewed in depth and or a relatively small number of focus groups are conducted. Qualitative research can be further classified in the following types.

- **Phenomenology:** A form of research in which the researcher attempts to understand how one or more individuals experience a phenomenon. Eg:- It can be interviewed 20 victims of Bhopal tragedy.

- **Ethnography:** This type of research focuses on describing the culture of a group of people. A culture is the shared attributes, values, norms, practices, language, and material things of a group of people. Eg:- The researcher might decide to go and live with the tribal in Andaman Island and study their culture and the educational practices.

- **Case study:** It is a form of qualitative research that is focused on providing a detailed account of one or more cases. Eg:- It may study a classroom that was given a new curriculum for technology use.

- **Grounded theory:** It is an inductive type of research, based or grounded on the observations of data from which it was developed; It
uses a variety of data sources, including quantitative data, review of records, interviews, observation and surveys

- **Historical research**: It allows one to discuss past and present events in the context of the present condition, and allows one to reflect and provide possible answers to current issues and problems. Eg:- the lending pattern of business in the 19th century.

In addition to the above, there is also the descriptive research. Fundamental research is based on establishing various theories. Also the research is classified into:

- Descriptive research
- Analytical research
- Fundamental research
- Conceptual research
- Empirical research
- One time research or longitudinal research
- Field-setting research or laboratory research or simulation research
- Clinical or diagnostic research
- Exploratory research
- Historical research
- Conclusion oriented research
- Case study research
- Short term research

### 3.4 Problem Statement

Grid computing is a form of distributed computing where a set of loosely coupled and heterogeneous computing nodes donate their unused processor cycles to create a pool of substantial processing capacity. In recent
years, grid computing has emerged as one of the most feasible alternatives to process compute intensive tasks, using co-operating processing nodes of ordinary capacity. The main advantage that grid computing offers, is that inexpensive computing nodes are coupled together to produce resource capacity comparable to high end supercomputers, although at a lower cost. The major bottleneck that grids face is that individual processors may not be well connected to one another and thus, the model is more suited to applications which can be broken into several independent and atomic sub tasks, without any requirement to communicate intermittent results between the grid nodes.

The principal challenge involved in any distributed computing environment, is that optimal scheduling of tasks dynamically entering the grid, becomes an NP-hard problem. Efficient grid scheduling is one of the key factors for achieving high performance in grid environments. Several heuristics have been proposed in literature to schedule the tasks efficiently to the most suitable node present in the grid. A majority of these heuristics show satisfactory results in static grid environments. However, it cannot be directly applied in dynamic environments where tasks are continuously arriving in the grid at regular intervals.

The principal motivation of the present work is to develop a scalable task scheduling algorithm which can operate efficiently without the services of a full-fledged prediction system providing prior information on workload of incoming tasks. The work proposes an enhancement of the existing round robin heuristic, where the user exploits information on the capacity of individual grid nodes, to prioritize tasks currently in execution, such that tasks currently allocated to slower machines are preferred for replication purpose over jobs executing on comparatively faster machines. The approach
facilitates replication of tasks, till now assigned to execute on slower machines, on machines with higher processing capacity. Hence, the research topic of the study selected by the researcher is “EFFICIENT TASK SCHEDULING IN GRID COMPUTING USING PRIORITIZED SSL WITH BACKEND FORWARDING MODEL”.

3.5 Objectives

Existing approaches of Grid scheduling do not give much emphasis on the performance of a Grid scheduler. Grid schedulers allocate resources to the jobs to be executed on First come First serve basis. If it provides an already optimize queue to the scheduler using various scheduling methods e.g. Shortest Job First, Priority based scheduling etc., the performance of a scheduler can be improved. The objective of this thesis work is to design, develop and implement a scheduling algorithm for Grid on top of existing scheduler in order to improve its performance by providing optimized queues to the scheduler. Queues can be optimized by using various scheduling algorithms depending upon the performance criteria to be improved e.g. response time, throughput. Main tasks undertaken and methodology followed in the thesis are:

A study of the current Grid schedulers, their architecture, Grid scheduling approaches and challenges involved in various types of existing Grid scheduling algorithms and how it evolved.

- A survey of Grid Simulation Tools and Techniques e.g. GridSim and Netbeans which are the open source tools.
- Design, Implementation and Simulation of the proposed Grid scheduling algorithm.
Round-robin scheduling was originally a computer science term, used to describe how an operating system handles priority when instructed to work through multiple tasks. Since its development, some organizations have begun using the term to describe a similar method of handling customers or tasks in the physical world. The advantages of round robin scheduling include even distribution of resources to different projects. However, the method does have some disadvantages, however.

Round-robin scheduling, like other first-come, first-served methods, doesn't give special priority to more important tasks. This means an urgent request doesn't get handled any faster than other requests in queue. The best round robin-based systems will include a tool that allows suspending of the usual priorities for a "rush job."

In the existing system, the thesis is using Round Robin [RR] model and SSL_with_Session model. Those models are not effective. Those models are not able to give the output in time and the throughput also lesser than their expected output. These models had made the Latency problem and minimal throughput. For this problem, SSL_with_bf model is to overcome the existing problems.

### 3.5.1 Coscheduled Web Servers

Request response time significantly affects user satisfaction. A significant volume of work has already been done regarding improvements in Web server throughput. However, latency time has garnered relatively little attention although it is also an important aspect in designing cluster-based Web servers. The steady growth of dynamic Web content has made latency reduction even more critical. In addition to user-level communication, coscheduling of communicating processes has been used as
an effective approach in minimizing the overall execution time of parallel application on clusters. Because the PRESS Web server and other distributed server models indicate a significant amount of intra-cluster communication to service client requests and to update caching information, a coscheduled distributed Web server model is proposed to facilitate faster communication. Coscheduling techniques are employed between the two threads which handle the intra-cluster communications in the sending and receiving nodes. Intra-cluster communication occurs when the requested file resides on a remote server cache in distributed Web servers. Since reading from the remote data cache takes much less time than accessing the disk, the request is forwarded to the server which has the requested file in the data cache. The impact of coscheduling techniques on Web server response time is to be examined.

3.5.2 Load Balancing of the Secure Servers

Application servers in a cluster-based Web server handle dynamic and sensitive Web content that requires protection from eavesdropping, tampering and forgery. Although the Secure Sockets Layer (SSL) is the most popular protocol to provide a secure channel between a client and an application server, adopting SSL results in degrading the server performance considerably due to its high overhead. This overhead becomes even more severe when it is deployed in application servers, since both the dynamic content of a Web server and the cryptographic algorithms of SSL require high CPU computation time.

Considering the fact that most dynamic content requires protection from unauthorized clients, performance improvement of SSL-enabled application servers is a pressing research topic. Thus, the impact of SSL offering is investigated in cluster-based application servers is investigated
and proposed a backend forwarding mechanism by exploiting the low overhead of user-level communication. For comparison, Three application server models are examined: RR (Round Robin), SSL-with-session and SSL-with-bf. The RR model, which is widely used in Web clusters, distributes requests from clients to servers using the Round Robin scheme. SSL-with-session uses a more sophisticated distribution algorithm, in which subsequent requests of the same client are forwarded to the same application server, thus avoiding expensive SSL setup costs. The proposed SSL-with-bf uses the same distribution policy as SSL-with-session, but also exploits an intelligent load balancing scheme that forwards client requests from a heavily loaded backend node to a lightly loaded node to improve the utilization across all nodes.

### 3.5.3 The NIC Caching Scheme

A user-level communication protocol that is employed in a cluster-based Web server requires an intelligent/programmable Network Interface Card (NIC) to facilitate direct communication between a user and a Network Interface (NI). NICs have, therefore, a profound impact on the overall communication performance. Traditionally, the on-chip/local memory of a NIC is usually small in size and is used primarily for establishing and maintaining connections in a cluster. However, recent trends show that many modern programmable NICs come with a large amount of on-chip memory compared to the Ethernet-based NICs. Thus, an NIC caching scheme is proposed that takes advantage of the large NIC memory for enhancing performance. In the proposed scheme, the NIC memory is used by Web servers to cache Web content locally or remotely. That is, requested data can be read from the local NIC memory or from the remote NIC memory. A read from the local NIC memory is done when a requested file is not in the local cache, but in the local NIC memory. This is beneficial since the NIC
memory can be used as extended memory. A read from a remote NIC memory occurs when a remote node serves a requested file in its NIC memory and forwards to it the requesting (initial) node. This is beneficial because it reduces the PCI traffic and DMA transfer time. Three Web server traces are used (i.e., Penn State CSE (CSE), UC Berkeley (UCB) and Penn State University (PSU) traces) to analyze the impact of the proposed NIC caching schemes.

3.6 Methodologies

3.6.1 A Generic Grid Scheduler Architecture

![Grid Scheduler Architecture Diagram]

Figure 3.1: General Grid Scheduler Architecture
The generic Grid scheduler architecture provides a high-level view of Grid schedulers. All existing Grid scheduling systems do not have corresponding components that appear in the architecture. But every component seems necessary for any comprehensive Grid scheduling system[13].

Real resources lay on the bottom of the architecture in the Figure 3.1 each being managed by a Local Resource Manager (LRM). An LRM is responsible for:

- Processing low level resource requests by executing a job that satisfies that request.
- Enabling remote monitoring and management of jobs created in response to resource.
- Updating the information service with information about the current availability and capabilities of the resources that it manages.

An LRM serves as the interface between a Grid scheduler and a local autonomous site being able to process low level resource requests. Grid Resource Allocation Management (GRAM) from Globus [6] is a good example of LRM.

### 3.6.1.1 Information Service (IS)

This component is responsible for maintaining the current state information of the resources such as CPU capacity, memory size, network bandwidth, software reliability and so on, and answering to queries for acquiring information about Grid Resources. To overcome the heterogeneous and dynamic nature of Grids, efficient information service plays a particularly important role in the Grid scheduling system. An
adequate scheduler must incorporate the current information of resources when making scheduling decisions. Globus Meta Director Service (MDS) and Network Weather Service (NWS) are two best examples that provide information service [13].

### 3.6.1.2 Analytical Benchmarking (AB)

This component provides a measure of how well a computational resource performs on a given job. The execution time of a given job varies with the capability of resources. Thus basically, the AB component provides a performance estimation of a given job on a specific computational resource.

### 3.6.1.3 Grid Scheduler (GS)

This is the core component in the architecture. The GS needs to do two jobs: one is resource selection and the other one is mapping. Resource selection is the process of selecting feasible and available resources for a given application to be scheduled. Mapping is the process of placing the jobs and communications of the application onto the resources and networks. Each mapping of jobs to feasible resources produces a candidate schedule. Each candidate schedule is then estimated for its performance potential based on the performance model.

### 3.6.1.4 Resource Allocation (RA)

This component implements a finally determined schedule through allocating the resources to the corresponding jobs. Resource allocation may involve data staging and binary code transferring before the job starts to execute on the computational resource.
3.7 Challenges in Grid Scheduling

Although Grids fall into the category of distributed parallel computing environments, it has a lot of unique characteristics, which make the Scheduling in Grids highly difficult. An adequate Grid scheduling system should overcome these challenges to leverage the promising potential of Grid systems, providing High-Performance services [13].

3.7.1 Resource Heterogeneity

A Grid has mainly two categories of resources: networks and computational resources. Heterogeneity exits in both of the two categories of resources. First, networks used to interconnect these resources may differ significantly in terms of their bandwidth and communicational protocols. A wide area Grid may have to utilize the best effort services provided by the internet. Second, computational resources are usually heterogeneous in that these resources may have different hardware, such as instruction set, computer architecture, number of processors, physical memory size, CPU speed and so on and also different software such as different operating systems, file systems, cluster management software and so on. The heterogeneity results in differing capability of processing jobs. Resources with different capacity cannot be considered uniformly. An adequate scheduling system should address the heterogeneity and further leverage different computing powers of diverse resources.

3.7.2 Site Autonomy

Typically a Grid may compromise multiple administrative domains. Each domain shares a common security and management policy. Each domain usually authorizes a group of users to use the resources in the domain. Thus applications from unauthorized users should not be eligible to
run on the resources in some specific domains. Furthermore, a site is an autonomous computational entity. A shared site will result in many problems. It usually has its own scheduling policy, which complicates the prediction of a job on the site. A single overall performance goal is not feasible for a Grid system since each site has its own performance goal and scheduling decision is made independently of other sites according to its own performance goal.

3.7.3 Local Priority

It is another important issue. Each site within the Grid has its own scheduling policy. Certain classes of jobs have higher priority only on certain specific resources. For example, it can be expected that local jobs will be assigned higher priorities such that local jobs will be better served on the local resources.

3.7.4 Resource Non-Dedication

Because of non-dedication of resources, resource usage contention is a major issue. Competition may exist for both computational resources and interconnection networks. Due to the non-dedication of resources, a resource may join multiple Grids simultaneously. The workloads from both local users and other Grids share the resource concurrently. The underlying interconnection network is shared as well. One consequence of contention is that behavior and performance may vary over the time; Contention free at the guaranteed level schedulers must be able to consider the effects of contention and predict the available resource capabilities.
3.7.5 Application Diversity

This problem arises because the Grid applications are from a wide range of users, each having its own special requirements. For example, some applications may require sequential execution, some applications may consist of a set of independent jobs, and others may consist of a set of dependent jobs. In this context, building a general-purpose scheduling system seems extremely difficult. An adequate scheduling system should be able to handle a variety of applications [6].

3.7.6 Dynamic Behavior

In Traditional parallel computing environments such as a cluster, the pool of resources is assumed to be fixed or stable. In a Grid Environment, dynamics exists in both the networks and computational resources. First, a network shared by many parties cannot provide guaranteed bandwidth. This is particularly true when wide areas networks such as the internet are involved. Second, both the availability and capability of computational resources will exhibit dynamic behavior. On one hand new resources may join the Grid and on other hand, some resources may become unavailable due to problems such as network failure. The capability of resources may vary overtime due to the contention among many parties who share the resources. An adequate scheduler should adapt to such dynamic behavior. After a new resource joins the Grid, the scheduler should be able to detect it automatically and leverage the new resources in the later Scheduling decision making. When a computational resource becomes unavailable resulting from an unexpected failure, mechanisms such as check pointing or rescheduling should be used to guarantee the reliability of Grid systems. These challenges pose significant obstacles on the problem of designing an efficient and effective scheduling system for Grid environments.
3.8 Grid Scheduling Process

A user goes through three stages to schedule a job when it involves multiple sites. Phase one is Resource Discovery, in which the user makes a list of potential resources to use. Phase two involves gathering information about those resources and choosing a best set to use. In phase three the user runs the job.

3.8.1 Phase 1: Resource Discovery

Resource discovery [3] involves the user selecting a set of resources to investigate in more detail; in phase two information gathering. At the beginning of this phase, the potential set of resources is empty set and at the end of this phase, the potential set of resources is some set that has passed a minimal feasibility requirement. Most users do this in three steps namely:

![Figure 3.2: Grid Scheduling Process](image)

Figure 3.2: Grid Scheduling Process
Authorization filtering: It is generally assumed that a user will know which resources he has access to in terms of basic services. At the end of this step the user will have a list of machines or resources to which he has access.

Application Requirement Definition: In order to proceed in resource discovery, the user must be able to specify some minimal set of job requirements in order to further filter the set of feasible resources. The set of possible job requirements can be very broad and vary significantly between jobs. It may include static details such as operating system or hardware for which a binary of the code is available. Or that the code is best suited to a specific architecture. Dynamic details are also possible e.g. a minimum RAM requirement, connectivity needed. This may include any information about the job that should be specified to make sure that the job could be matched to a set of resources.

Minimal Requirement Filtering: Given a set of resources to which a user has access and the minimal set of requirements the job has, the third step in the resource discovery step is to filter out the resources that do not meet the minimal job requirements. The user generally does this step by going through the list of resources and eliminating the ones that do not meet the job requirements as much as they are known. It could also be combined with the gathering more detailed information about each resource.

3.8.2 Phase 2: System Selection

Given a group of possible resources (or a group of possible resource sets), all of which meet the minimum requirements for the job, a single resource (or single resource set) must be selected on which to schedule the job. This is generally done in two steps [3]:

Gathering Information (QUERY): In order to make the best possible resource match, a user needs to gather dynamic information about the resources in question. Depending on the application and resource in question, different information may be needed. Take for instance the simple case of finding the best single resource for a job to run on. A user might want to know the load on the various machine(s) and queue lengths if the machine has queues. In addition, physical characteristics and software requirements play a role, is the disk big enough for the data etc. then there are location/connectivity issues is the machine close enough to the data store. All these issues are multiplied in the case of multiple resources. Making an advance reservation may or may not be a part of this step.

Select the system(s) to run on: Given the information gathered by the previous step, a decision of which resource (or set of resources) should the user submit a job is made in this step. This can be done in variety of ways. It is noted that this does not address the situation of speculative execution, where a job is submitted to multiple resources and when one begins to run the other submissions is cancelled.

3.8.3 Phase 3: Run the Job

The third phase of scheduling is running a job. This involves a number of steps [3]:

- Make an Advance Reservation (Optional): It may be the case that to make the best use of a given system, part or all of the resources will have to be reserved in advance. Depending on the resource, this can be easy or hard to do, may be done with mechanical means as opposed to human means, and the reservations may or may not expire with or without cost.
Submit Job to Resources: Once resources are chosen the application must be submitted to resources. This may be easy as running a single command or as complicated as running a series of scripts, and may or may not include setup or staging.

Preparation Tasks: The preparation stage may involve setup, claiming a reservation, or other actions needed to prepare the resource to run the application. One of the first attempts at writing a scheduler to run over multiple machines at America’s National Aeronautics and Space Agency (NASA) was considered unsuccessful because it did not address the need to stage files automatically.

Monitor Progress: Depending on the application and its running time, users may monitor the progress of their application.

Find out if Job is done: When the job is finished, the user needs to be notified.

Completion Tasks: After a job is run, the user may need to retrieve files from that resource in order to do analysis on the results, break down the environment and remove temporary settings etc.

3.9 Types of Grid Scheduling

There have been a number of efforts attempting to design scheduling systems for Grids, each having its unique features. A comprehensive set of different types of scheduling [14] is discussed in this section.

3.9.1 Knowledge of Application

Application Level Scheduling: The application level scheduling scheme makes use of knowledge of applications as much as possible. Such kind of scheduling results in custom schedulers for each application attempting
to maximize application performance, measured as runtime or speedup, with little regard to overall system performance. The complexity of application level scheduling is the order of the applications considered.

- **Resource Level Scheduling:** Resource level scheduling does not use much knowledge of Grid applications. In this scheme, applications neither specify resource requirements nor provide application characteristics. Generally, a scavenging Grid aiming at leveraging the idle computing power will use this scheduling scheme. Condor [15] is an example, which uses resource level scheduling.

### 3.9.2 Inter-Job Dependency

Given an application, the constituent jobs may either be dependent or independent. The mapping algorithms for a set of independent jobs differ significantly from those for a set of dependent jobs. An application with a set of jobs is usually represented by a DAG. The mapping algorithms for a set of dependent jobs are more complicated.

### 3.9.3 Information Service

The scheduler determines the state information of all the resources in a Grid system through the information service before making a scheduling decision. Different scheduling systems construct quite different structures to provide information services.

- **Centralized:** Under a centralized scheme, there exists a single centralized entity that maintains the state info. The centralized entity traverses every resource to get the most up to date state information periodically and keeps the information in its storage, waiting for queries issued by the schedulers. A centralized scheme is not scalable because it introduces the
risk of single point of failure. Furthermore, a centralized entity may become the performance bottleneck when the entity cannot afford a large number of resources and possible queries.

- Decentralized: Within this scheme, every resource is responsible for maintaining its current state information locally, and answering queries from different clients. A decentralized scheme may not be efficient due to the amplified quantity of queries. However, the decentralized scheme is more reliable because the risk of single point of failure is removed through distributing the responsibility evenly to every resource. The large overhead should be carefully considered. Example of decentralized scheme is NWS.

- Hybrid: The hybrid scheme resources are categorized into several groups. Within each group centralized scheme is applied. Thus each group has representative entity; which is in charge of the information of all resources in its group. Over the groups, the decentralized scheme applies. This scheme is the most practical method for real wide area Grids.

3.9.4 Scheduler Organization

The Grid Scheduler organization can be organized into three categories: centralized, decentralized & hierarchical.

![Scheduler Organization Diagram](image)

Figure 3.3 Scheduler Organization Types
Centralized: In the centralized scheme, all user applications are sent to the centralized scheduler. There is a queue in the centralized scheduler for holding all the pending applications. When user application is submitted to the scheduler, it may not be scheduled at once. Instead it will be put in the queue, waiting for scheduling and resource allocation. Typically, each site neither maintains its queue nor performs any scheduling decisions. A site receives jobs from the scheduler and executes them. The centralized scheme is not very scalable with increasing number of resources. The central scheduler may prove to be a bottleneck in some situation.

Decentralized: Decentralized scheme distributes the responsibility of scheduling to every. Each site in the Grid acts as both a scheduler and a computational resource. User applications are submitted to the local Grid scheduler where the applications originate. The local scheduler is responsible for scheduling its local applications, thus it possibly maintains a local queue to hold its own pending applications. Meanwhile, it should be able to respond to other schedulers’ requests by acknowledging or denying it.

Hierarchical: In the hierarchical scheme, different levels of schedulers share the scheduling process. The higher-level schedulers manage larger sets of resources and lower level schedulers manage smaller sets of resources. A higher-level scheduler has no direct control of a resource if there is one lower-level scheduler between the higher–level scheduler and the resource. A higher-level scheduler can only consider the capability of the set of resources managed by a lower-level scheduler as a whole entity, and utilizes the capability through invoking the lower-level scheduler. Compared with the centralized scheduling, hierarchical scheduling addresses the scalability and the problem of single point failure issue. Nevertheless, it also retains some of the advantages of the centralized scheme.
3.9.5 Rescheduling

After an application is scheduled, the performance of the application may not approach the desired performance due to the dynamic nature of the resources. It may be profitable to reschedule the application during execution to maintain good performance. At the minimum, an adequate Grid scheduler should acknowledge the resource failure and resend lost work to a live computational resource. In summary, rescheduling is done to guarantee the job’s completion and performance goal’s achievement.

3.10 Grid Scheduling Approaches

The scheduling policy determines how the scheduling should be performed. The performance goal defined in the scheduling policy plays a particularly important role in a Grid scheduling system. According to the different performance goals, the scheduling systems can be classified into three categories:

3.10.1 Application Centric

Scheduling systems that fall in the application centric category try to favor the performance of individual applications. Typical performance goals sought by application centric scheduling systems include minimizing execution time; maximizing the speed up etc. e.g. an application-centric scheduling system will exploit a greedy mapping algorithm, which allocates the application to the resources that are likely to produce the best performance without considering the rest of pending applications.

3.10.2 System-Centric

A system centric scheduling system concerns the overall performance of the whole set of applications and the whole Grid system. The performance
goals desired by a system centric policy typically include resource utilization, system throughput and average application response time. Ordering on the pending applications is usually performed in order to achieve a higher system centric performance.

3.10.3 Economy Based

An economy based scheduling system introduces the idea of market economy. Under this scheme, scheduling decisions are made based on the economy model. The economy model defines each application having the desired QoS, such as execution time and deadline and the cost that the application will pay for the desired QoS; each resource is specified by its cost and the capacity. For each application, it wants to get as higher QoS as possible within the budget constraint. For each resource, it wants to get profit as much as possible by keeping itself busy. Nimrod-G [17] is a scheduling system which exploits idea of economy mechanism.

3.11 Parallel and Distributed Vs Grid Scheduling Algorithms

In [18], Casavant et al propose a hierarchical taxonomy for scheduling algorithms in general-purpose parallel and distributed computing systems. Since Grid is a special kind of such systems, scheduling algorithms in Grid can be treated as a subset of scheduling algorithms for parallel and distributed computing. From the top to the bottom, this subset can be identified as follows [19].

Local vs. Global: At the highest level, a distinction is drawn between local and global scheduling. The local scheduling discipline determines how the processes resident on a single CPU are allocated and executed to the processor. While global scheduling policy uses information about the system
to allocate processes to multiple processors to optimize a system wide performance objective. Grid scheduling falls into the Global scheduling.

Static vs. Dynamic: The next level in the hierarchy (under the Global scheduling) is a choice between static and dynamic scheduling. This choice indicates the time at which the scheduling decisions are made. In the case of static scheduling, information regarding all resources in the Grid as well as all the tasks in an application is assumed to be available by the time the application is scheduled. By contrast, in the case of dynamic scheduling, the basic idea is to perform task allocation on the fly as the application executes. This is useful when it is impossible to determine the execution time, direction of branches and number of iterations in a loop as well as in the case where jobs arrive in a real-time mode. These variances introduce forms of non-determinism into the running program [20]. Both static and dynamic Scheduling are widely adopted in Grid computing.

Optimal vs. Suboptimal: In the case that all information regarding the state of resources and the jobs is known, an optimal assignment could be made based on some criterion function, such as minimum make span and maximum resource utilization. But due to the NP-Complete nature of scheduling algorithms and the difficulty in Grid scenarios to make reasonable assumptions which are usually required to prove the optimality of an algorithm, current research tries to find suboptimal solutions, which can be further divided into the following two general categories.

Approximate vs. Heuristic: The approximate algorithms use formal computational models, but instead of searching the entire solution space for an optimal solution, it is satisfied when a solution that is sufficiently “good” is found. In the case where a metric is available for evaluating a solution, this technique can be used to decrease the time taken to find an acceptable
schedule. The factors which determine whether this approach is worthy of pursuit include availability of a function to evaluate a solution.

- The time required to evaluate a solution.
- The ability to judge the value of an optimal solution according to some metric.
- Availability of a mechanism for intelligently pruning the solution space.

The other branch in the suboptimal category is called heuristic. This branch represents the class of algorithms which make the most realistic assumptions about a prior knowledge concerning process and system loading characteristics. It also represents the solutions to the scheduling problem which cannot give optimal answers but only require the most reasonable amount of cost and other system resources to perform their function. The evaluation of this kind of solution is usually based on experiments in the real world or on simulation. Not restricted by formal assumptions, heuristic algorithms are more adaptive to the Grid scenarios.

Distributed Vs Centralized: In dynamic scheduling, the responsibility for making global scheduling decisions may lie with one centralized scheduler, or can be shared by multiple distributed schedulers. The centralized strategy has the advantage of ease of implementation, but suffers from the lack of scalability, fault tolerance and the possibility of becoming a performance bottleneck.

Cooperative vs. Non-cooperative: If a distributed scheduling algorithm is adopted, the next issue that should be considered is whether the nodes involved in job Scheduling are working cooperatively or non-cooperatively. In the non-cooperative case, individual schedulers act alone as autonomous entities and arrive at decisions regarding their own optimum objects
independent of the effects of the decision on the rest of system e.g. application-level schedulers. In the cooperative case, each Grid scheduler has the responsibility to carry out its own portion of the Scheduling task, but all schedulers are working toward a common system-wide goal.

3.12 Summary of Current Approaches to Grid Scheduling

The Following table summarizes properties of widely used the Grid Resource Management Systems with emphasis on their scheduling attributes[21]:

Table 3.1 Summary of Current Approaches in Grid Scheduling

<table>
<thead>
<tr>
<th>System</th>
<th>Grid Type</th>
<th>Resources</th>
<th>Scheduling Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Condor [22]</td>
<td>Computational Flat</td>
<td>Extensible schema model, hybrid namespace, no QoS, network directory store, centralized queries discovery.</td>
<td>Centralized Scheduler</td>
</tr>
</tbody>
</table>
3.13 Challenges in Evaluation of Scheduling Algorithms

Besides the characteristics of Grid Environment which make the performance evaluation of scheduling algorithms tedious; the following factors also make the task difficult:

- Performance prediction is difficult because end to end internet performance itself is extremely hard to analyze and predict.

- End to end performance observed on internet exhibits great diversity and thus different algorithms work more effectively for different topologies and also for different time periods on same topology.

3.14 Grid Simulation

Simulation is the imitation of some real thing, state of affairs, or process. The act of simulating something generally entails representing certain key characteristics or behaviors of a selected physical or abstract system. Grid environment also can be simulated using several Grid simulators e.g. GridSim [22], Eclipse [23] etc. Grid simulators enable Grid users to work on Grid like environment without having to worry about the other external factors that may influence the Grid environment.
More so, be it Application Developer, Tool Developer or Grid Developer all of them need to test and verify their applications, tools & services respectively for fulfillment of intended design goals before the end product or logic is put in Real-Time Grid Computing Environment. The motivation for using simulation instead of directly using the Grid test-bed, especially in analyzing models and algorithms can be drawn from the following factors [22]:

- Setting up a Grid test-bed is expensive, resource intensive, and time consuming. Even if one is set up, it is mostly limited to some local area environments.

- Using a real test-bed with real jobs is time consuming as well. Hours of real job time can be simulated in seconds provided the simulation is having sufficient processor power.

- The real test-bed does not provide a repeatable and controllable environment for experimentation and evaluation of scheduling strategies.

- Simulation works well, without making the analysis mechanism unnecessary complex, by avoiding the overhead of co-ordination of real resources.

- Simulation is also effective in working with very large hypothetical problems that would otherwise require involvement of a large number of active users, which is very hard to coordinate and build at a large-scale research environment for investigation purposes.

- Simulation allows analyzing existing as well as new economic models and scheduling algorithms.
3.14.1 GridSim

GridSim [22] is a software platform that enables users to model and simulate the characteristics of Grid resources and networks with different configurations. Study Grids, or test new algorithms and strategies in a controlled environment. By using GridSim, it is able to perform repeatable experiments and studies that are not possible in a real dynamic Grid environment. Some of the GridSim features are outlined below:

- It allows modeling of different resource characteristics and their failure properties;
- It enables simulation of workload traces taken from real supercomputers;
- It allocates incoming jobs based on space-or time-shared mode;
- It supports reservation-based or auction mechanisms for resource allocation;
- It has the ability to schedule compute- and/or data-intensive jobs;
- It provides clear and well-defined interfaces for implementing different resource allocation algorithms;
- It allows modeling of several regional Grid Information Service (GIS) components.

In this thesis, the approach followed consists of a use of multiple priority based scheduling with different algorithm used in each queue. The scheduler allocates the resource to the rightmost job of the topmost queue. The multiple queue scheduling algorithms, allows a process to move between queues. Every queue has its own scheduling algorithm. There are three scheduling algorithm (First Come First Serve, Shortest Job First, Round Robin) used for different queues. The queues are arranged in apriority
order with topmost queue having the highest priority. The idea is to separate processes according to the priority associated with it. If a process does not have any priority, it will be moved to a lowest-priority queue. This scheme leaves I/O-bound and interactive processes in the higher-priority queues. In addition, a process that waits too long in a lower-priority queue may be moved to a higher-priority queue. This form of aging prevents starvation [16]. Here, the three algorithms are used in all possible combinations for each queue in order to find out a best combination for a particular type of job and according to certain performance criteria.

For example, consider a multiple queue scheduler with three queues, numbered from 0 to 2. The scheduler will only execute first queue, job will jump to the next higher priority queue after some time span. When job will come then it placed into the queue according to the priority [16]. It will explain in detail in the Chapter 4.

In general, a multiple queue scheduler is defined by following parameter:

- The number of queues.
- The scheduling algorithm for each queue.
- The method used to determine which job assign to which priority queue.
- The method used to determine when to upgrade a process to higher-priority queue.
- The method used to determine which queue a process will enter that process needs service.

In the proposed system, the SSL_with_Backend Forwarding model is implemented to overcome the problem of existing system. This model will reduce the latency and increase the throughput than the existing system
(Round Robin model and SSL_with_Session). The SSL_with_bf model is very helpful for load balancing of the server.

This will reduce the load of the server while the server is being busy. These are the advantages of the proposed system. The SSL_with_bf scheme can minimize the average latency by about 40 percent and improve throughput across a variety of workloads.

3.15 Summary

The problem statement of the research work is highlighted in this chapter. The research methodologies adopted to do the research work are also elaborated to provide a better understanding of this research study.