NURTURE
Education Solutions
TOMORROW'S HERE

Subject Name: High Performance Computing

Module Number: 05

Module Name: Grid Computing

Syllabus

Grid Computing : Introduction, Evolution of the Grid, Definitions of Grid Computing, Infrastructure of hardware and software, Grid models, , High-Performance Schedulers, Grid Middleware: : Connectivity, Resource and Collective Layer, HPC and Grids, Scheduling HPC applications in Grids, Grid Computing Infrastructure and vulnerability- security Issues — Resource Management.

Aim

The aim of this module is to study grid computing with Hardware technologies for grid computing.

Objectives

The Objectives of this module are

- Learn how Grid computing helps in solving large scale scientific problems.
- Gain knowledge on the concept of virtualization that is fundamental to cloud computing.
- Learn how to program the grid and the cloud.
- Learn the security issues in the grid and the cloud environment.

Outcome

At the end of this module, you are expected to:

- Apply grid computing techniques to solve large scale scientific problems.
- Apply the concept of virtualization.
- Use the grid and cloud tool kits
- Apply the security models in the grid and the cloud environment.

Contents

Introduction, Evolution of the Grid, Definitions of Grid Computing

Infrastructure of hardware and software, Grid models,

High-Performance Schedulers, Grid Middleware: : Connectivity

Resource and Collective Layer, HPC and Grids,

Scheduling HPC applications in Grids,

Grid Monitoring Architecture (GMA) – An Overview of Grid Monitoring Systems,

Installation and configuration of Alchemi Grid.

Grid Computing Infrastructure and vulnerability Security Issues , Resource Management.

Grid Computing Introduction

Grid computing refers to a distributed computing infrastructure that utilizes widely distributed computer resources connected over a network to collectively achieve a common goal. It can be seen as a virtual supercomputer composed of multiple interconnected computers or nodes .

Grid computing offers advantages such as enhanced computing power, scalability, resource optimization, and the ability to tackle large-scale computational challenges. It finds applications in various fields, including scientific research, data analysis, weather forecasting, and simulations, where significant computational resources are required to process and analyze complex data sets.

Here are some key points about grid computing:

Distributed System: Grid computing involves connecting numerous computer nodes, often located in different geographical locations, to create a distributed architecture. These nodes can be part of private or public networks, including the internet .

Resource Pooling: Grid computing pools together the unused computing resources of multiple computers, such as processing power, storage, and memory. These resources are shared and made available for executing tasks or applications, enabling efficient utilization of resources .

Common Goal: The interconnected computers in a grid computing system work collectively towards accomplishing a shared objective or solving complex problems. This can involve tasks like data analysis, simulations, weather modeling, scientific research, or other computationally intensive operations.

Virtual Supercomputer: Grid computing creates a virtual supercomputer by aggregating the computational power and capabilities of individual nodes. This allows for high-performance computing and the ability to process large datasets or perform parallel computations.

Specialized Software: Grid computing requires specialized software installed on each participating computer to manage the entire system and coordinate tasks across the grid. This software acts as the manager and facilitates communication and coordination between the nodes .

CLUSTER COMPUTING VERSUS GRID COMPUTING

CLUSTER COMPUTING	GRID COMPUTING
A set of computers or devices that work together so that they can be viewed as a single system	Use of widely distributed computing resources to reach a common goal
Nodes have the same hardware and same operating system	Nodes have different hardware and various operating systems
Each node performs the same task controlled and scheduled by software	Each node performs different tasks
A homogenous network	A heterogeneous network

CLUSTER COMPUTING VERSUS GRID COMPUTING

CLUSTER COMPUTING	GRID COMPUTING
Located in a single location	Devices are located in different locations
Devices are connected through a fast local area network	Devices are connected through a low-speed network or internet
Resources are managed by centralized resource manager	Each node has its own resource manager that behaves similarly to an independent entity
Used to solve issues in databases or WebLogic Application Servers	Used to solve predictive modelling, simulations, Engineering Design, Automation etc.

Grid Computing

Evolution of the Grid

The evolution of Grid computing can be traced back to the early 1970s when the concept of utilizing unused CPU cycles was first introduced. During this time, computers were being connected together using network engineering techniques .

- In recent years, there has been a substantial change in the way computing resources and services are perceived and utilized. Previously, computing needs were primarily serviced by localized platforms and infrastructures. However, there has been a shift towards a new perspective that emphasizes the utilization of distributed computing resources and the integration of services .
- The evolution of Grid computing can be divided into different generations. **The first generation** focused on linking supercomputing sites and addressing challenges such as communication, resource management, and remote data manipulation. **The second generation** emphasized the development of middleware to support large-scale data and computation. **The third generation** shifted the focus towards distributed computing and scalability .
- Within the domain of Grid computing, evolution can be classified into two distinct types: external and internal. **External evolution involves** extending the boundaries of the Grid, while **internal evolution** deals with enhancing the functionality of individual Grids .

Grid Computing

Evolution of the Grid

- Furthermore, there is ongoing research on the standardization of Grid computing architecture principles. Achieving some form of standardization is seen as significant in order to integrate Grid computing principles into mainstream distributed computing and web services .

In summary, the evolution of Grid computing has witnessed the emergence of different generations, advancements in middleware, a focus on distributed computing, the integration of services, and ongoing efforts to standardize Grid computing architecture principles.

Grid Middleware

The software that allows the different parts of the grid to communicate and coordinate with one another by sitting between the application layer and the underlying hardware infrastructure is referred to as middleware. To facilitate the effective and efficient distribution of computing jobs over a network of computers, middleware can contain a wide range of technologies, including data management systems, job scheduling software, and resource management tools.

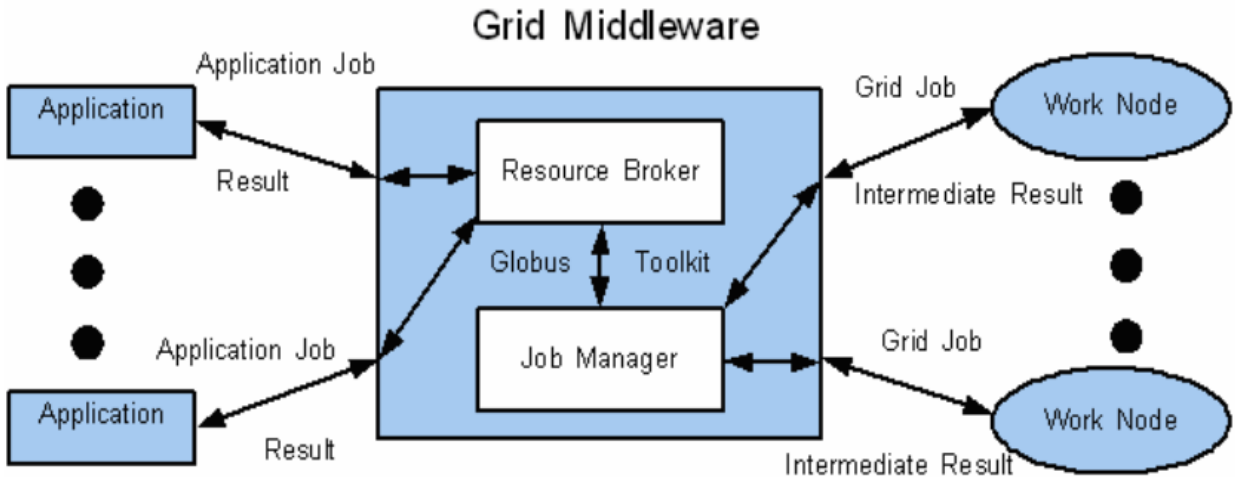
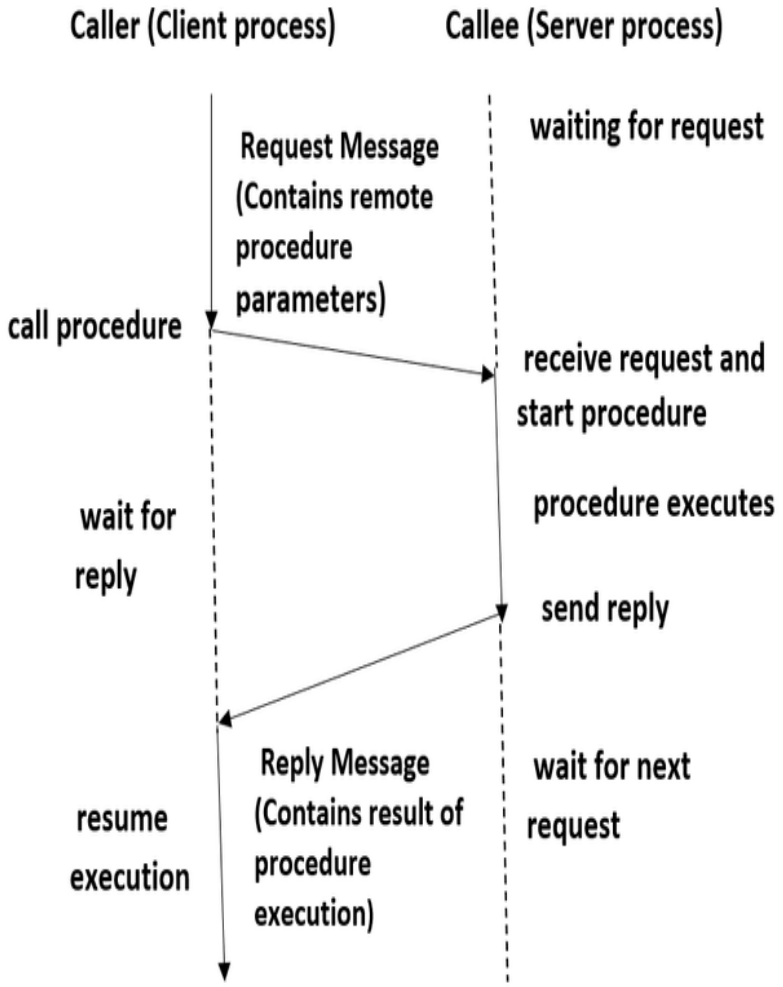


Fig. 1. Basic principle of Globus Toolkit Grid Middleware

Grid Middleware

Some common types of middleware used in grid computing include:

- **Message-oriented middleware (MOM):** This type of middleware enables the communication between different components of the grid by providing a messaging infrastructure.
- **Remote procedure call (RPC) middleware:** RPC middleware allows different parts of the grid to communicate with each other by calling functions remotely.
- **Grid services middleware:** This type of middleware provides a set of standard services that can be used to build grid applications, such as resource discovery and allocation, data management, and security.



Grid Middleware

Uses of middleware

- **Resource Management:** Middleware can be used to manage the resources available in the grid, including computing nodes, storage systems, and network resources. This can involve allocating resources to different tasks, monitoring resource usage, and balancing the workload across the grid.
- **Data Management:** Middleware can be used to manage the data that is used and generated by grid applications, including transferring data between nodes, replicating data for reliability, and storing data in a centralized repository.
- **Job Scheduling:** Middleware can be used to schedule the execution of tasks on the grid, including deciding which tasks should run on which nodes, and allocating resources to those tasks.
- **Workflow Management:** Middleware can be used to manage the flow of tasks in a grid application, including defining the dependencies between tasks and coordinating their execution.

Infrastructure of hardware and software

Grid computing infrastructure consists of both hardware and software components that work together to enable the efficient utilization of distributed resources for large-scale computing tasks. Here is a comprehensive overview of the hardware and software aspects of grid computing infrastructure based on the provided search results.

Hardware Infrastructure:

- Computer Nodes: Grid computing involves connecting multiple computers distributed geographically. These computers, often referred to as nodes, form the basis of the hardware infrastructure in grid computing .
- High-End Computers: Grid computing infrastructure typically includes high-performance or high-end computers that provide substantial computational power .
- Networking: The hardware infrastructure encompasses the networking components that connect the distributed computers, enabling data transfer and communication among them .
- Storage Systems: Multi-petabyte storage systems are utilized within grid computing centers to store and manage the vast amount of data generated and processed by grid applications .

Infrastructure of hardware and software

Software Infrastructure:

- The software infrastructure in grid computing plays a crucial role in coordinating and managing the tasks and resources within the grid .
- Specialized software is installed on each computer within the grid network to facilitate the interaction and coordination of tasks. This software divides the main task into subtasks and assigns them to individual computers .
- Grid middleware, a layer of software, provides a set of tools, services, and protocols to enable efficient resource management, job scheduling, data management, security, and other grid-related operations .
- Grid middleware packages such as Globus Toolkit, gLite, and ARC middleware are commonly used in grid computing environments .
- The software infrastructure also includes software components for monitoring, security, fault tolerance, and data management to ensure the reliable and secure operation of the grid .

Grid models

A unique type of distributed computing model is the grid computing model. Different computers connected to the same network share one or more resources when computing is dispersed. Every resource is shared in the ideal grid computing system, transforming a computer network into a potent supercomputer. A grid computing system's resources could be accessed via a user interface that is identical to that of a local machine. There would be abundant processing and storage available to every authorised computer.

Grid computing, at its most basic, is a network of computers wherein each computer's resources are distributed across all other computers in the system. Authorised users can access and utilise the community's processing power, memory, and data storage for specified purposes. A grid computing system can be as straightforward as a group of comparable computers using the same operating system, or it can be as sophisticated as interconnected networks made up of every type of computer platform imaginable.

Grid models

For example, say we have the following equation:

- Z = (2 x 3) + (1 x 7) + (4 x 6)

On a desktop computer, the steps needed to calculate a value for Z might look like:

- Step 1: Z = 6 + (1 x 7) + (4 x 6)
- Step 2: Z = 6 + 7 + (4 x 6)
- Step 3: Z = 6 + 7 + 24
- Step 4: Z = 37

In a Grid Computing scenario, the steps might look like (with three processors or computers):

- Step 1: Z = 6 + 7 + 24
- Step 2: Z = 37

Grid models

As you can see, due to the abundance of resources, steps are combined in the grid computing scenario. Fewer stages are required as a result of the entire process, which reduces the amount of time required. This is accomplished by segmenting each computer task into smaller units and distributing them for execution across the available computing resources.

You might now mistake a parallel computing model for a grid computing model. They are actually rather similar, but you need to be aware of the scale differences between the two. A parallel computing model can be imagined as a single box with numerous CPUs or processors working in parallel to solve the same issue.To solve a very big computational task, the Grid Computing Model may instead be a very large collection of systems. You can add or remove an infinite number of systems almost fast. There is no demand for central management of the grid's boxes. The systems on the grid might not be exclusively for it and might also be located on another grid.

High-Performance Schedulers

Since the inception of Grid computing, research on scheduling has been ongoing. However, because to the steep learning curve of grid computing, newcomers find it exceedingly challenging to master associated ideas. Therefore, a clear understanding of scheduling is required in the field of grid computing. The goal of this work is to provide a clear grasp of scheduling and the related concept of the Grid computing system.

Scheduling Fundamentals and Terminology

What is planning? In order to accomplish the intended objective(s), scheduling is the decision-making process of allocating the relatively large number of jobs to the relatively small number of workers in either space or time or both dimensions. Depending on the target environment of employees, the types of jobs, and desirable requirements for the completion of jobs, this decision may be influenced by the duration of tasks/activities, preceding tasks/activities, resource availability, and target completion time. For instance, the operating system's scheduling function assigns a worker (a CPU) to the active processes and jobs that are now running. In multi-processor systems, automated manufacturing systems, aviation scheduling, train scheduling, etc.

High-Performance Schedulers

Scheduling Terminology

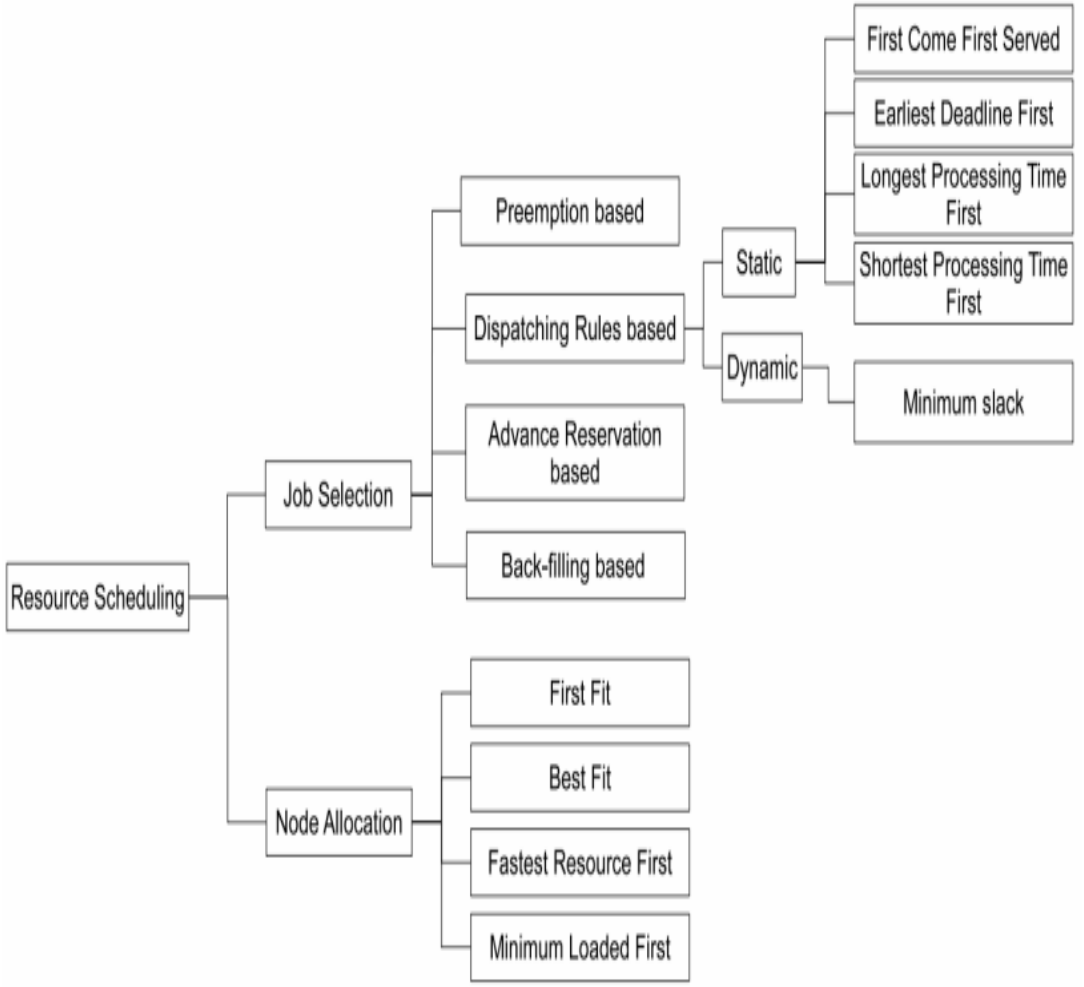
- Release-time (date): The earliest time at which a job can start its processing.
- Processing-time: The time-duration needed by a job to finish its processing.
- Start-time: The actual time at which a job starts its processing.
- Finish-time: The actual time at which a job finishes its processing.
- Expected execution time: It is a time expected to be taken up by a job to finish its execution. It is derived empirically.

Two main types of scheduling are involved in Grid, one at local resource level and another at application level. We concisely explore them in following discussion.

a. Resource Scheduling b. Application Scheduling

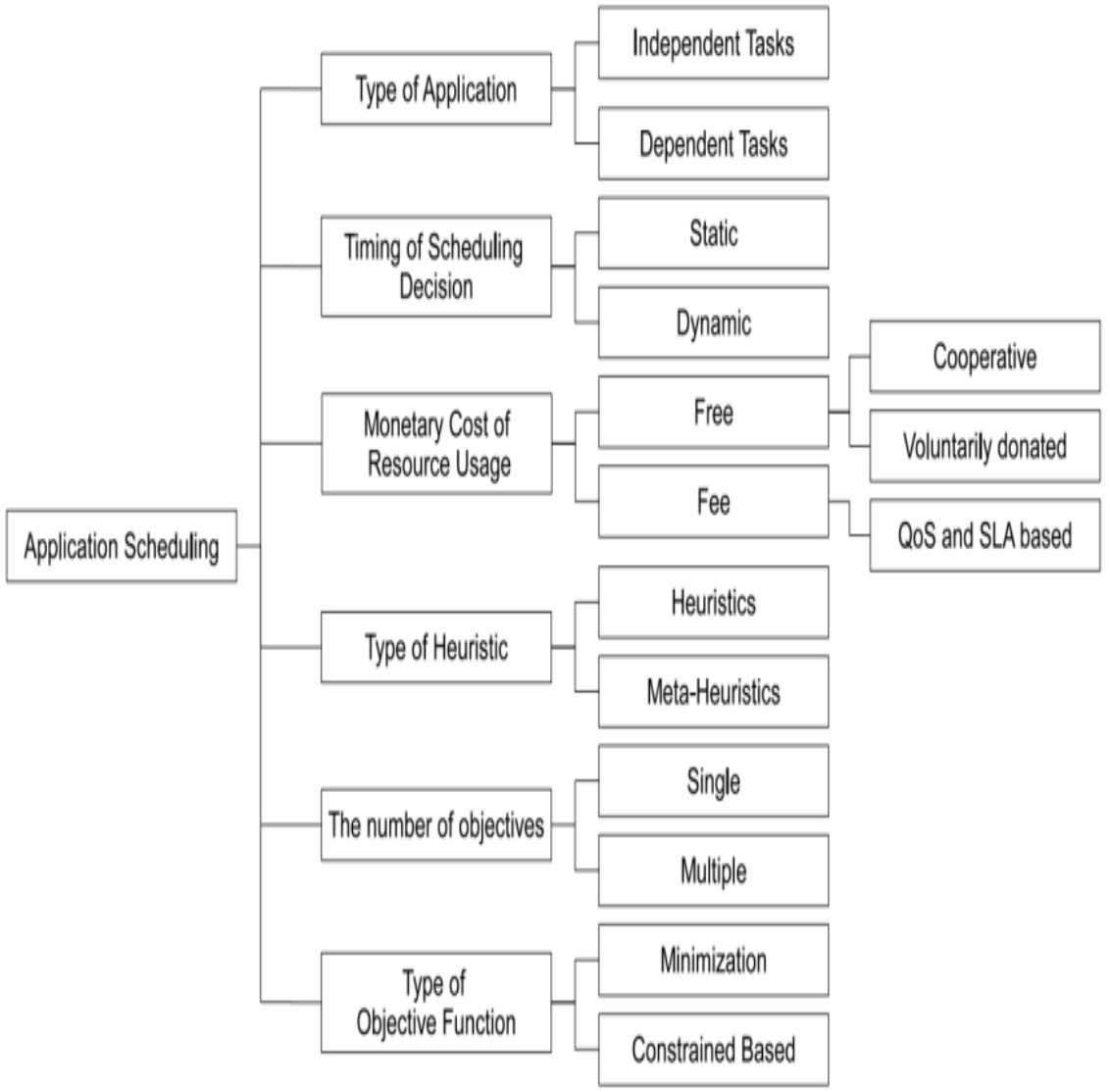
High-Performance Schedulers

Resource Scheduling At the local resource level, there are two forms of scheduling involved: time sharing and space sharing. Each machine (computer) in the cluster uses time sharing scheduling, such as the CPU scheduler on each machine that is a part of the batch system, whereas Local Resource Manager uses space sharing scheduling to schedule the job that is in the batch queue on the idle machine in the cluster. Resource scheduling is used to balance the load on the resources or boost resource utilisation. A batch queue controlled cluster is typically present at a Grid site, as seen in Figure.



High-Performance Schedulers

Application Scheduling Users submit their applications to the grid scheduler, which then handles application scheduling by making decisions about the tasks for the entire application. The term "Grid scheduler" is used in a variety of contexts. These terms include application scheduler, application broker, global scheduler, superscheduler, and meta-scheduler. Application level schedulers assign a Grid site to a job, but resource schedulers assign a machine from a cluster, which is a key distinction between resource scheduling and application scheduling. Based on several criteria, Figure presents a wide classification of application scheduling techniques.



Grid Middleware

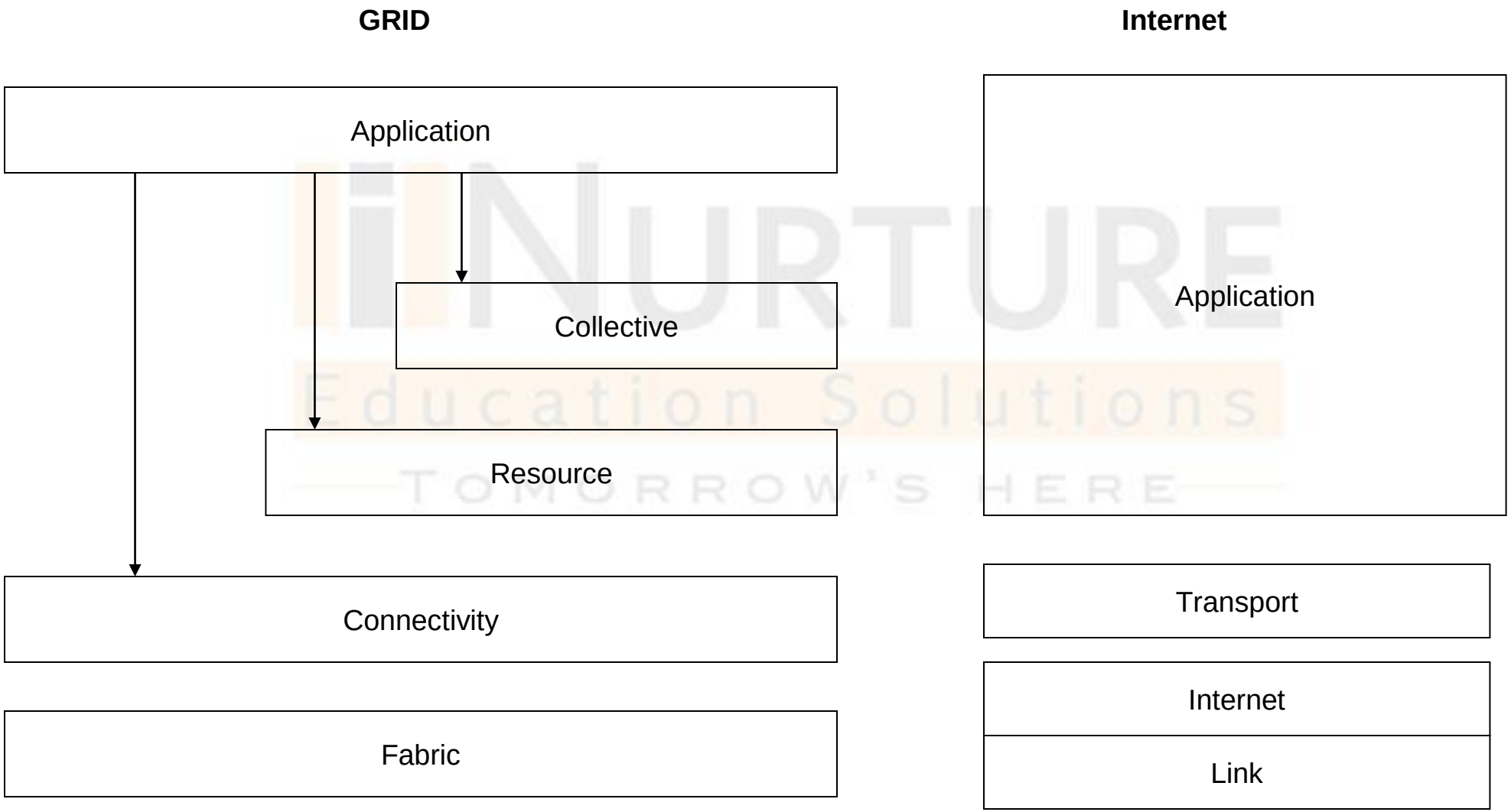
Uses of middleware

- **Security:** Middleware can be used to provide security services for grid applications, including authentication, authorization, and encryption.
- **Monitoring and Management:** Middleware can be used to monitor the status of the grid and the tasks running on it, and to provide tools for managing and administering the grid.

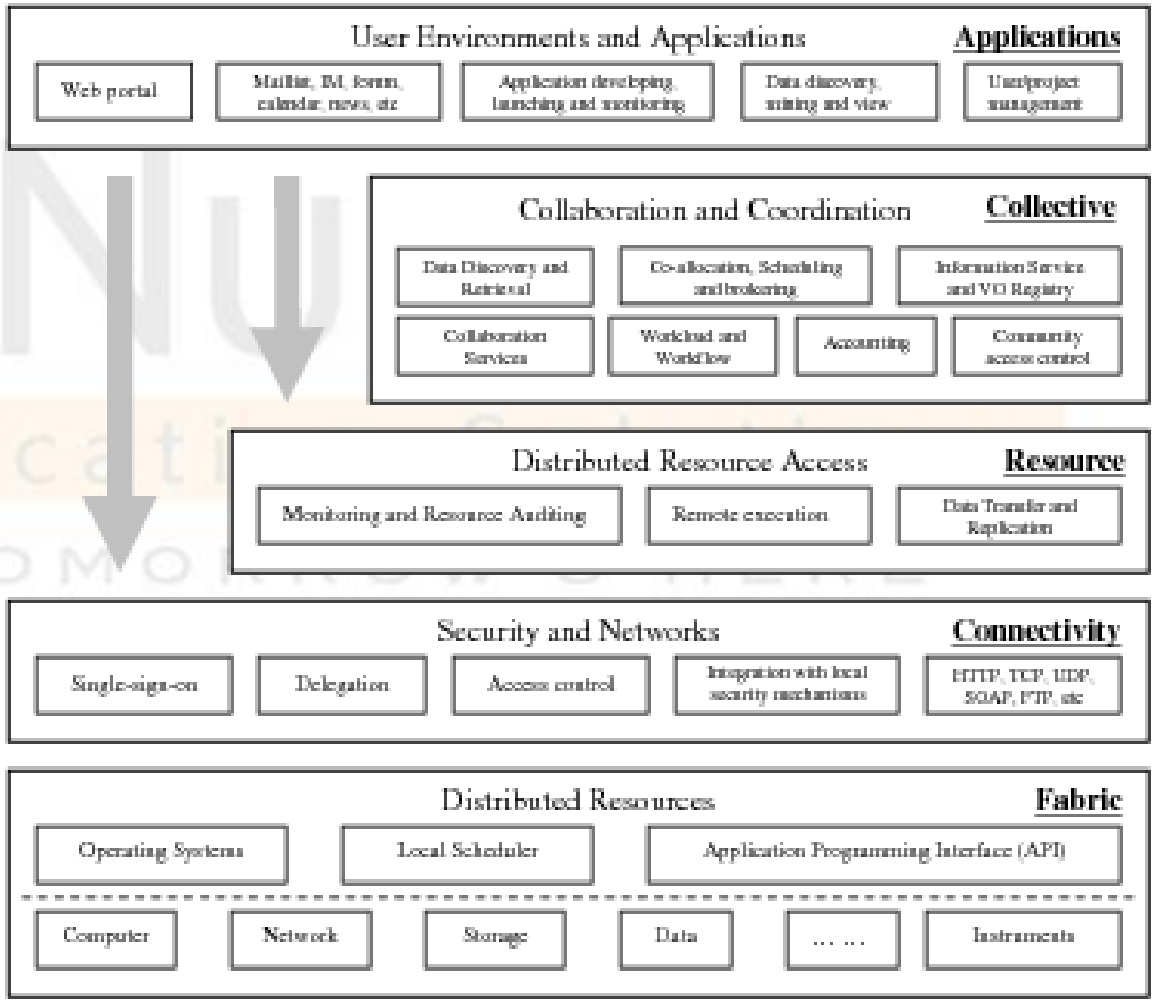
Grid Architecture

- Architecture identifies the fundamental system components, specifies purpose and function of these components, and indicates how these components interact with each other.
- Grid architecture is a protocol architecture, with protocols defining the basic mechanisms by which VO users and resources negotiate , establish, manage and exploit sharing relationships.
- Grid architecture is also a services standards-based open architecture that facilitates extensibility, interoperability, portability and code sharing.
- API and Toolkits are also being developed.

Grid Architecture



Grid Architecture



Grid Architecture

Fabric Layer

- Fabric layer: Provides the resources to which shared access is mediated by Grid protocols.
- Example: computational resources, storage systems, catalogs, network resources, and sensors.
- Fabric components implement local, resource specific operations.
- Richer fabric functionality enables more sophisticated sharing operations.
- Sample resources: computational resources, storage resources, network resources, code repositories, catalogs.

Grid Architecture

Connectivity Layer

- Communicating easily and securely.
- The essential authentication and communication protocols necessary for grid-specific network services are defined by the connectivity layer.
- This enables the swap of data between fabric layer resources.
- maintain for this layer is drawn from TCP/IP's IP, TCL and DNS layers.
- verification solutions: single sign on, etc.

Grid Architecture

Resources Layer

- Resource layer defines protocols, APIs, and SDKs for secure negotiations, initiation, monitoring control, accounting and payment of sharing operations on individual resources.
- Two protocols information protocol and management protocol define this layer.
- Information protocols are used to gather details on the configuration, current load, and usage policy of resources as well as their structure and state.
- Access to the shared resource is negotiated via management protocols, which specify things like qos, advanced reservations, etc..

Grid Architecture

Collective Layer

- Coordinating multiple resources.
- Contains protocols and services that incarcerate interactions among a anthology of resources.
- It supports a variety of sharing behaviors lacking placing new necessities on the resources being shared.
- Sample services: directory services, coallocation, brokering and scheduling services, data replication service, workload management services, collaboratory services.

Grid Architecture

Applications Layer

- These are client applications that control within VO environment.
- Applications are constructed by vocation upon services defined at any layer.
- all of the layers are well defined using protocols, provide access to useful services.
- Well defined APIs also survive to work with these services.
- A toolkit Globus implements all these layers and supports grid application development.

HPC and Grids

Similar to its namesake, the electric power grid, a computational grid offers nearly universal access to capabilities that are difficult to duplicate at network endpoints. These capabilities, in the context of a high-performance grid, encompass both high-performance hardware (networks, computers, storage devices, visualisation devices, etc.) and special services that rely on it, such as smart instruments, collaborative design spaces, and meta computations.

A network of independent computers situated across several places forms the grid computing system. By connecting various, inexpensive computers into a single, vast infrastructure, grid computing is able to utilise their idle processing and other compute resources. Grid computing, as opposed to high performance computing (HPC) and cluster computing, allows for the assignment of various tasks to various nodes. Resources on each machine, including CPUs, GPUs, software, memory, and data storage

HPC and Grids

etc. – can be shared as community resources. Load-sharing software or middleware evenly distributes tasks to several personal computers and servers on the grid. The nodes run independent tasks and are loosely linked by the Internet or low-speed networks, connecting to the grid directly or via scheduling systems. Grid environments are fault tolerant and have no single point of failure. If one node in the grid fails, many others are available to take over its load.

Grid computing is best suited for applications in which many parallel computations can happen independently, without the need to communicate between processors. Most grid computing projects have no time dependency, and large projects typically deploy across many countries. Some grid projects use the idle power of computers, known as cycle-scavenging, and may run in the background for many weeks.

HPC and Grids

High Performance Computing (HPC) is the IT practice of aggregating computing power to deliver more performance than a typical computer can provide. Originally used to solve complex scientific and engineering problems, HPC is now used by businesses of all sizes for data-intensive tasks. Companies that provide automotive engineering, pharmaceutical design, oil and gas exploration, renewable energy research, entertainment and media, financial analytics, and consumer product manufacturing rely on HPC for scalable business computing.

HPC infrastructures are complex in both design and operation, involving a large set of interdependent hardware and software elements that must be precisely configured and seamlessly integrated across a growing number of compute nodes. SUSE Linux Enterprise for HPC is designed for scalability and includes tools that simplify configuration and management. SUSE Linux Enterprise Real Time is ideal for time-sensitive HPC applications.

Scheduling HPC applications in Grids

BIG DATA applications play an increasing role in High Performance Computing (HPC). They are perfect candidates for co-scheduling, as they obey flexible speedup models, alternating I/O operations and intensive computation phases.

A simple scheduling strategy on HPC platforms is to execute each application in dedicated mode, assigning all resources to each application throughout its execution. However, it was shown recently that rather than using the whole platform to run one single application, both the platform and the users may benefit from coscheduling several applications, thereby minimizing the loss due to the fact that applications are not perfectly parallel. Sharing the platform between two applications already leads to significant performance and energy savings , which become even more important when the number of co-scheduled applications increases

Scheduling HPC applications in Grids

With the advent of multi core platforms, HPC applications can be efficiently parallelized on a flexible number of processors. Usually, a speedup profile determines the performance of the application for a given number of processors In the Computational Grid, scheduling can be static or dynamic.

The scheduling in the static mode is simple and straightforward; all the jobs arrive at a certain known time to the system. The resources are always available for the whole scheduling process. Whereas in the dynamic mode, the jobs arrive to the system in different lengths of time, the scheduler has no idea about the arrival time until job reaches to the system. Moreover, the availability of the resources is not guaranteed. Most frequently, the candidate resources may sign out/in to the system for various reasons (such as resource failure).

Scheduling HPC applications in Grids

The scheduler may anticipate the static mode, and it is simple to implement. For a tiny grid computing system, a traditional scheduling technique (such as FCFS) can work effectively. On the other hand, a more sophisticated policy that can handle these conditions is extremely necessary if thousands of jobs (that reach in different times) are waiting to be allocated for numerous available resources.

In the scheduling field, priority rule algorithms based on **queue-based systems** are frequently utilised. Maintaining the balance of performance for various measures is the main challenge faced by priority-based algorithms. These algorithms can perform admirably for certain measures but poorly when compared to other factors. Schedule-based approach is more flexible than queuebased approach due to applying the backfilling technique alongside runtime estimates. The backfilling performs the scheduling without a fixed order.

Short jobs are used to fill in the gaps created in the machine's schedule as a result of the jobs' varying timings of arrival. As a result, the effectiveness of the resources will be increased.

Scheduling HPC applications in Grids

arrival times of the jobs. Thus, the efficiency of the resources will be further improved.

Backfilling was referred to provide a benefit without a cost, something for free. The schedule-based strategy backfills the incoming jobs in advance using runtime estimates provided by the user, whereas the queue-based approach does so on the fly as the queues execute the scheduling.

Extensible Argonne Scheduling System (EASY) and Conservative backfilling (CONS) are the two main backfilling scheduling techniques that have been presented. While CONS uses a schedule-based method, EASY uses a queue-based approach. EASY is straightforward, strong, and combative. This method fills in the gaps with brief jobs in order to completely utilise the available resources.

Scheduling HPC applications in Grids

CONS implements schedule-based approach in order to backfill the jobs. To avoid the flaws in EASY, run time estimation is provided by the user. Run time estimates provide information to the scheduler about the resources capabilities, resources availability, job arrival time and execution time for the jobs.

Grid Computing Infrastructure and vulnerability- security Issues — Resource Management.

Every application you download from the Internet has security risks. When connecting two or more computers, you must be ready to answer a few questions. How do you protect the privacy of personal information? How can malevolent hackers be stopped from accessing the system? How do you manage who can use the system's resources and gain access to it? How do you ensure that the user doesn't consume the entire system's resources? Thus The grid design must meet security criteria. Resource discovery, authentication, authorization, and access mechanism are the major issues. The integrity and confidentiality of the data handled within the grid would be at danger without this functionality. Let's talk about how the grid system handles authentication and authorization.

Grid Computing Infrastructure and vulnerability- security Issues — Resource Management.

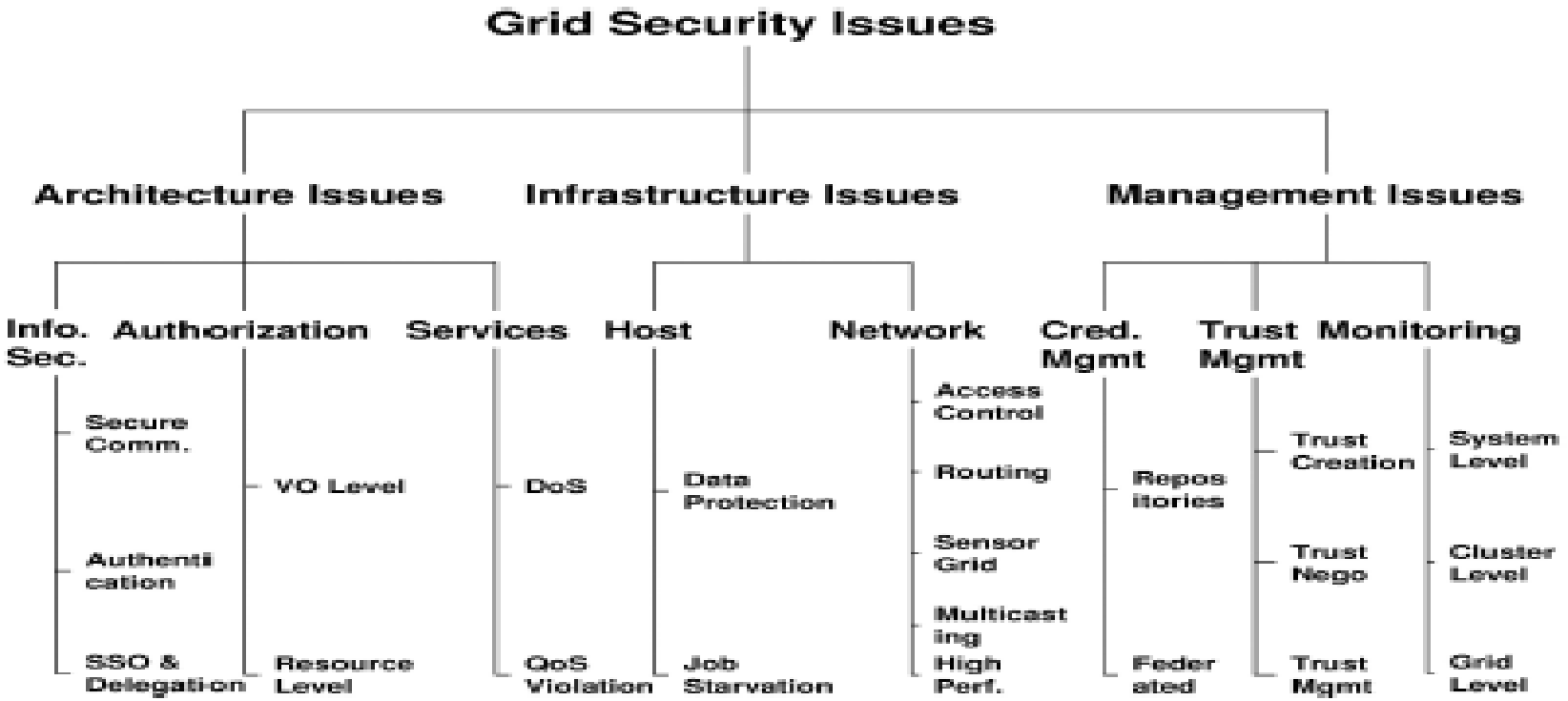
- Grid is a extremely huge network of computers
- Grid computing is application of a number of computers to a single problem at the same time
- Examples of Grids are SETI@Home, World Community Grid.
- Grid Security has Architecture, Infrastructure and Management issues

Grid Computing Infrastructure and vulnerability- security Issues — Resource Management.

- Offers less exclusive alternative to purchasing novel, larger server platforms.
- Sometimes workload supplies exceed existing server platform capabilities.
- Useful for small responsibilities like movie rendering to solving vast computational tribulations of future.
- Future infrastructure.

Grid Computing Infrastructure and vulnerability- security Issues — Resource Management.

SECURITY ISSUES IN GRID COMPUTING



Grid Computing Infrastructure and vulnerability- security Issues — Resource Management.

ARCHITECTURE ISSUES

- Grid should defend flow of information to trust parties
- User’s Data should be confined
- User’s individual data and system data should be illustrious
- Resource level agreement
- Delegation of Identity and Single Sign On

Grid Computing Infrastructure and vulnerability- security Issues — Resource Management.

INFRASTRUCTURE ISSUES

- Issues related to network and host components
- Host Issues-prevention of misuse of user's data and resources
- Prioritizing local jobs over system jobs
- Job Starvation
- Availability
- Multicasting, traffic censoring are issues to be handled

Grid Computing Infrastructure and vulnerability- security Issues — Resource Management.

MANAGEMENT ISSUES

- Scheduling, Rescheduling.
- Monitoring, Auditing and Logging
- Host and network component compatibility
- Management is difficult due to heterogeneous nature of the Grid.
- Routing

Grid Computing Infrastructure and vulnerability- security Issues — Resource Management.

OTHER ISSUES

- Assurance Mechanisms are seldom used in Grids
- Accounting-required to assist Auditing mechanism
- Auditing- essential to monitor system performance

Parallel Algorithms

Summary

- An **algorithm** is a sequence of instructions followed to solve a problem
- The problem is divided into sub-problems and are executed in parallel to get individual outputs. Later on, these individual outputs are combined together to get the final desired output.

Self Assessment Questions

Q.1 The word Grid in Grid Computing comes from an analogy to the ___ Power Grid.

A. Electrical

B. Mechanical

C. Both A and B

D. None of these

Answer» A. Electrical

Self Assessment Questions

Q.2 ___ Grids consist of one or more systems working together to provide a single point of access to users.

A. Group

B. Cluster

C. Batch

D. None of these

Answer» B. Cluster

Self Assessment Questions

Q.3 **Type of grid computing is**

A. Collaborative grid

B. System grid

C. Process grid

D. Channel grid

Answer» A. Collaborative grid

Self Assessment Questions

Q.4 ___ are types of applications responsible for the management of jobs, such as allocating resources needed for any specific job.

A. Schedulers

B. Sequencer

C. Resource allocator

D. Resource sharing

Answer» A. Schedulers

Self Assessment Questions

Q.5 A ___ is a web-based gateway that provides seamless access to heterogeneous back-end resources.

A. Grid portal

B. Grid model

C. Grid block

D. Grid Plan

Answer» A. Grid portal

Self Assessment Questions

Q.6 ___ offers core services such as remote process management, co-allocation of resources, storage access.

A. Grid

B. Grid Middleware

C. Grid Bottom

D. Grid Top

Answer» B. Grid Middleware

Self Assessment Questions

Q.7 A ___ infrastructure is key to the success or failure of a grid environment.

A. Security

B. Management

C. Guardianship

D. Fault

Answer» A. Security

Assignment

1. Explain Grid Computing with diagram.
2. Describe Advantages and Disadvantages of Grid Computing.
3. List out Grid Computing security Issues.
4. Explain Connectivity in Grid Middleware.
5. Differentiae Resource and Collective Layer.
6. Define High-Performance Schedulers.

Document Links

Topic	URL	Notes
Complete Introduction to Scheduling in Grid Computing Environment	https://arxiv.org/ftp/arxiv/papers/1407/1407.3879.pdf	**this link explain Scheduling in Grid Computing Environment**
vector Add with CUDA	http://selkie.macalester.edu/csinparallel/modules/ConceptDataDecomposition/build/html/Decomposition/CUDA_VecAdd.html	This link explains about cuda vector code

Video Links

Topic	URL	Notes
Grid Computing	https://www.youtube.com/watch?v=wFlejBXX9Gk&list=PL3xCBlatwrsXCGW4SfEoLzKiMSUCE7S_X	Explains about GPU PROGRAMMING
CUDA	https://www.youtube.com/watch?v=2NgpYFdsduY&list=PLxNPSjHT5qvtYRVdNN1yDcdSl39uHV_sU	This video talks about CUDA architure

E-Book Links

Topic	URL	Notes
Handbook of Grid Computing Making The Global Infrastructure a Reality	https://docs.google.com/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxncmlkY291cnNlfGd4OjI1N2M3ZmNjODNlZjUzNTI	Ebook gives an comprehensive knowledge on grid computing
A Networking Approach to Grid Computing	https://docs.google.com/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxncmlkY291cnNlfGd4OjFlZmU5ZGVlNTYyODY4NDI	Ebook gives an comprehensive knowledge on grid computing with network approach
Introduction To Grid Computing	https://docs.google.com/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxncmlkY291cnNlfGd4OjNhMDA1OTExYjY0ZjI0YTQ	Introduction To Grid Computing
Introduction to Gird Computing and Globus	https://docs.google.com/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxncmlkY291cnNlfGd4OjRmZTBiMDM0ZWZiMTU4ZQ	