Unit-1

1. Parallel Computing Architectures

(Vector Processing , Symmetric Multi Processing Amd Massively parallel processing Systems)

In computers, parallel processing is the processing of program instructions by dividing them among multiple processors with the objective of running a program in less time. In the earliest computers, only one program ran at a time. A computation-intensive program that took one hour to run and a tape copying program that took one hour to run would take a total of two hours to run. An early form of parallel processing allowed the interleaved execution of both programs together. The computer would start an I/O operation, and while it was waiting for the operation to complete, it would execute the processor-intensive program. The total execution time for the two jobs would be a little over one hour.

The next improvement was multiprogramming. In a multiprogramming system, multiple programs submitted by users were each allowed to use the processor for a short time. To users it appeared that all of the programs were executing at the same time. Problems of resource contention first arose in these systems. Explicit requests for resources led to the problem of the deadlock. Competition for resources on machines with no tie-breaking instructions lead to the critical section routine.

Vector processing was another attempt to increase performance by doing more than one thing at a time. In this case, capabilities were added to machines to allow a single instruction to add (or subtract, or multiply, or otherwise manipulate) two arrays of numbers. This was valuable in certain engineering applications where data naturally occurred in the form of vectors or matrices. In applications with less well-formed data, vector processing was not so valuable.

The next step in parallel processing was the introduction of multiprocessing. In these systems, two or more processors shared the work to be done. The earliest versions had a master/slave configuration. One processor (the master) was programmed to be responsible for all of the work

in the system; the other (the slave) performed only those tasks it was assigned by the master. This arrangement was necessary because it was not then understood how to program the machines so they could cooperate in managing the resources of the system.

Solving these problems led to the **symmetric multiprocessing system** (SMP). In an SMP system, each processor is equally capable and responsible for managing the flow of work through the system. Initially, the goal was to make SMP systems appear to programmers to be exactly the same as single processor, multiprogramming systems. (This standard of behavior is known as sequential consistency). However, engineers found that system performance could be increased by someplace in the range of 10-20% by executing some instructions out of order and requiring programmers to deal with the increased complexity. (The problem can become visible only when two or more programs simultaneously read and write the same operands; thus the burden of dealing with the increased complexity falls on only a very few programmers and then only in very specialized circumstances.) The question of how SMP machines should behave on shared data is not yet resolved.

As the number of processors in SMP systems increases, the time it takes for data to propagate from one part of the system to all other parts grows also. When the number of processors is somewhere in the range of several dozen, the performance benefit of adding more processors to the system is too small to justify the additional expense. To get around the problem of long propagation times, message passing systems were created. In these systems, programs that share data send messages to each other to announce that particular operands have been assigned a new value. Instead of a broadcast of an operand's new value to all parts of a system, the new value is communicated only to those programs that need to know the new value. Instead of a shared memory, there is a network to support the transfer of messages between programs. This simplification allows hundreds, even thousands, of processors to work together efficiently in one system. (In the vernacular of systems architecture, these systems "scale well.") Hence such systems have been given the name of **massively parallel processing (MPP) systems**.

The most successful MPP applications have been for problems that can be broken down into many separate, independent operations on vast quantities of data. In data mining, there is a need to perform multiple searches of a static database. In artificial intelligence, there is the need to

analyze multiple alternatives, as in a chess game. Often MPP systems are structured as clusters of processors. Within each cluster the processors interact as in a SMP system. It is only between the clusters that messages are passed. Because operands may be addressed either via messages or via memory addresses, some MPP systems are called NUMA machines, for Non-Uniform Memory Addressing.

SMP machines are relatively simple to program; MPP machines are not. SMP machines do well on all types of problems, providing the amount of data involved is not too large. For certain problems, such as data mining of vast data bases, only MPP systems will serve.

2.Cluster Computing

What is a Cluster?

A **computer cluster** is a group of linked computers, working together closely thus in many respects forming a single computer. The components of a cluster are connected to each other through fast local area networks

Cluster computing is a form of computing in which a group of computers are linked together so that they can act like a single entity .It is the technique of linking two or more computers into a network (usually through a local area network) in order to take advantage of the parallel processing power of those computers.

Need for Cluster Computing:

- Requirements for computing increasing fast.
    - More data to process.
    - More compute intensive algorithms available.
- Approaches to supply demand
    - Qualitative: Optimized algorithms, faster processors, more memory.

- Quantitative: Cluster computing, grid computing, etc

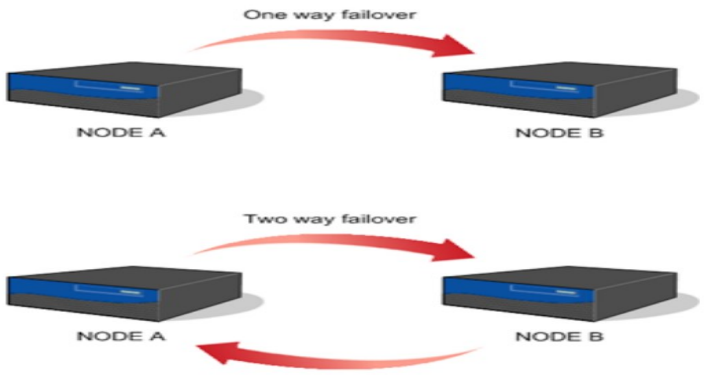
Cluster Categorizations:

- High Availability Cluster
- Load Balancing Cluster
- HPC Cluster

High Availability Cluster

High-availability clusters are groups of computers that support server applications that can be reliably utilized . The clusters are designed to maintain redundant nodes that can act as backup systems in the event of failure. The minimum number of nodes in a HA cluster is two-one active and one redundant-though most HA clusters will use considerably more nodes .

- Failover Clusters, mainly implemented to improve the availability of service that cluster provides
- They operates by having redundant nodes, upon failure the standby node take cares
- Types of High availability clusters: one way & two way

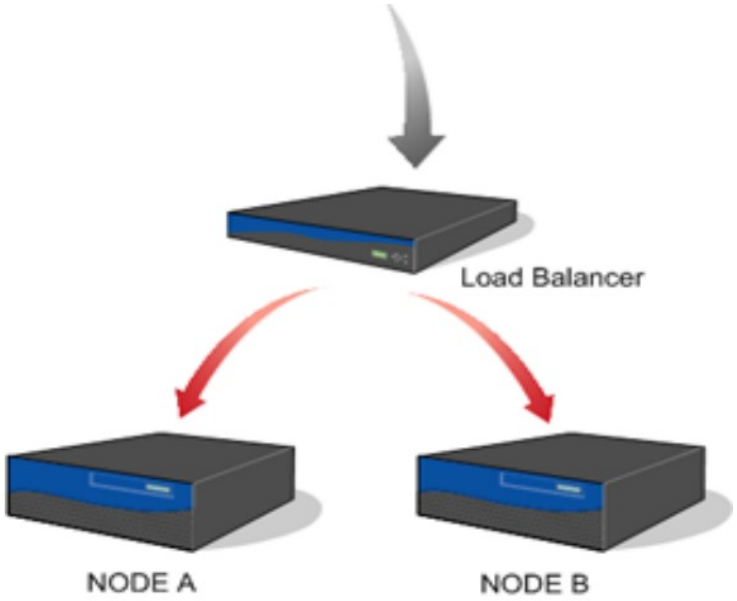


- Often used for critical databases, network file sharing and business applications

Load Balancing Cluster

Load balancing is a computer networking methodology to distribute workload across multiple computers or a computer clusters . Load-balancing clusters operate by routing all work through one or more load-balancing front-end nodes , which then distribute the workload efficiently between the remaining active nodes. Load-balancing clusters are extremely useful for those working with limited IT budgets . Devoting a few nodes to managing the workflow of a cluster ensures that limited processing power can be optimized.

- Multiple computers connected together to share computational workload
- Logically they are multiple computers but function as single virtual computer
- Request initiated from the user is distributed among all the nodes by one or more load balancer

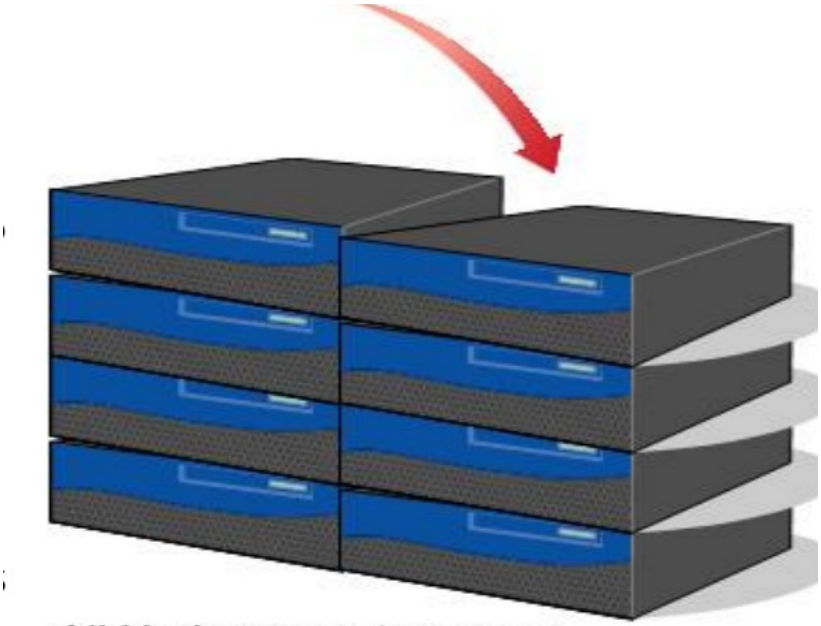


3. HPC Cluster (High Performance computing Cluster)

High-performance computing uses supercomputers and computer clusters to solve advanced computation problems. HPC clusters are designed to exploit the parallel processing power of

multiple nodes . They are most commonly used to perform functions that require nodes to communicate as they perform their tasks – for instance, when calculation results from one node will affect future results from another

- HPC clusters are mainly used to increases the performance by splitting the computational task into different nodes
- Mainly used in scientific computing
- Popular HPC cluster implementations are nodes running with linux os and free software's to implement the parallelism
- The job running on the cluster nodes requires little or no inter nodes communication is called "Grid Computing"✓
- The local Scheduling software manages the cluster nodes load balancing
- Middleware such as MPI (Message Passing Interface) or PVM (Parallel Virtual Machine) permits compute clustering programs to be portable to a wide variety of clusters



4. Grid Computing

Definition: **Grid computing** is a term referring to the combination of computer resources from multiple administrative domains to reach a common goal.

- Coordinates resources that are not subject to centralized control
- Uses standard, open, general-purpose protocols and interfaces
- Delivers nontrivial qualities of service

Why Grid?

- Large-scale science and engineering are done through the interaction of people, heterogeneous computing resources, information systems, and instruments, all of which are geographically and organizationally dispersed.
- The overall motivation for "Grids" is to facilitate the routine interactions of these resources in order to support large-scale science and Engineering.

Grid Architecture

Grid Architecture can be described as the layers of building blocks, where each layer has a specific function, to accomplish Grid Computing Infrastructure

The key components of grid computing include the following.

- Resource management: a grid must be aware of what resources are available for different tasks
- Security management: the grid needs to take care that only authorized users can access and use the available resources
- Data management: data must be transported, cleansed, parceled and processed
- Services management: users and applications must be able to query the grid

in an effective and efficient manner

Main characteristics of Grid Computing:

The main characteristics of a grid computing environment can be listed as follows:

- Large scale: A grid must be able to deal with a number of resources ranging from just a few to millions.
- Geographical distribution: Grid resources may be spread geographically.
- Heterogeneity: A grid hosts both software and hardware resources that can be ranging from data, files, software components or programs to sensors, scientific instruments, display devices, personal digital organizers, computers,super-computers and networks.
- Resource sharing and coordination: Resources in a grid belong to different organizations that allow other organizations (i.e. users) to access them. The resources must be coordinated in order to provide aggregated computing capabilities.
- Multiple administrations: Each organization may establish different security and administrative policies under which resources can be accessed and used.
- Accessability attributes: Transparency, dependability, consistency, and pervasiveness are attributes typical to grid resource access. A grid should be seen as a single virtual computing environment and must assure the delivery of services under established Quality of Service requirements