Unit 6
Parallel Systems

Structure of the Unit

1. Unit Outcomes
2. Parallel Systems- Definition
3. Advantages of Parallel Systems
4. Types of Parallel Systems
5. Parallel Versus Distributed Systems
6. Self-Assessment Questions
7. Self-Assessment Activities
8. Multiple Choice Questions
9. Keys to Multiple Choice Questions
10. Summary of the Unit
11. Keywords
12. Recommended Resources for Further Reading

6.1 Unit Outcomes

After the successful completion of this unit, the student will be able to:

1. Explain the concept of parallel systems and their fundamental principles.
2. Compare and contrast the performance benefits of parallel computing with sequential computing.
3. Differentiate between parallel and distributed systems in terms of their design and operation.

6.2 Parallel Systems

Parallel processing, also known as multiprocessing, is a computational technique that involves dividing extensive tasks into multiple segments and executing them concurrently using two or more central processing units (CPUs). This is a method aimed at speeding up computations by utilizing multiple processors simultaneously, rather than relying on a single processor This approach enhances performance and minimizes the time required to accomplish a task. Utilizing any operating system that supports multiple CPUs, including multi-core processors, allows for the implementation of multiprocessing techniques. This approach addresses the growing need for faster solutions and the ability to tackle more significant problems in various fields, such as fluid dynamics, weather forecasting, large-scale system modeling and simulation, data processing, image analysis, artificial intelligence, and automated manufacturing.

Three key factors have fueled the recent surge in parallel processing:

- Declining Hardware Costs: The cost of hardware has steadily decreased, making it feasible to construct systems with numerous processors at a reasonable expense.

- Advances in Integrated Circuit Technology: The technology of very large-scale integration (VLSI) has advanced to a point where it is feasible to design intricate systems that require millions of transistors on a single chip.
- Limitations of Single-Processor Speed: Traditional Von Neumann processors are reaching the physical limits of their cycle times, making further improvements increasingly difficult. As a result, parallel processing has gained prominence as a means to enhance computational performance.

In the context of parallel computing, a parallel computer is essentially a collection of processors, often of the same type, interconnected in a specific manner to facilitate the coordination of their tasks and data exchange. Parallel computing itself focuses on designing algorithms with minimal time complexity, taking into account the speedup factor, which measures the performance improvement achieved through parallelization.

The motivation for parallelism has typically been hindered by the perceived complexity of developing parallel software. This complexity stems from the need to specify and coordinate concurrent tasks, a lack of standardized algorithms, consistent environments, and software development tools. It raises the question of whether dedicating substantial effort to exploiting parallelism is worth it, especially when the underlying hardware and software platforms may become obsolete during the development period.

However, certain trends in hardware design suggest that uniprocessor architectures, which implicitly handle parallelism, may not be able to sustain the rate of performance improvement in the future. This is due to a lack of inherent parallelism and bottlenecks, such as in the data path and memory access. Simultaneously, standardized hardware interfaces have reduced the time required to transition from the development of a microprocessor to a parallel machine based on that microprocessor.

Moreover, significant progress has been made in standardizing programming environments, ensuring a longer life cycle for parallel applications. These developments present compelling arguments in favor of utilizing parallel computing platforms to address the increasing demand for faster and more efficient computational solutions.

6.3 Advantages of Parallel Systems

In recent years, multiprocessor systems, also known as parallel systems or multicore systems, have increasingly become the dominant paradigm in the field of computing. These systems incorporate two or more processors that closely collaborate, sharing the computer bus, and sometimes even the clock, memory, and peripheral devices. Initially, multiprocessor systems gained prominence in server environments, but they have since found their way into desktop and laptop systems. More recently, the proliferation of multiple processors has extended to mobile devices like smartphones and tablets.

Multiprocessor systems offer several key advantages:

Increased Throughput: One of the primary benefits of multiprocessor systems is their ability to enhance overall system throughput. By employing multiple processors, the expectation is to accomplish more work in less time. However, it's important to note that the speed-up ratio achieved with N processors is

not a straightforward N-fold increase. Multiple processors collaborating on a task introduce some overhead due to the need for coordination and resource sharing, which can reduce the expected gain from adding extra processors. Similarly, when multiple programmers work together, they do not necessarily produce N times the output of a single programmer.

Economy of Scale: Multiprocessor systems can often be more cost-effective than equivalent single-processor systems. This cost efficiency arises from the ability to share peripherals, mass storage, and power supplies among multiple processors. In cases where several programs work with the same dataset, it's more economical to store that data on a single disk and have all the processors access it, rather than maintaining many individual computers, each with its local storage and copies of the data.

Increased Reliability: Multiprocessor systems can enhance system reliability significantly. When functions can be effectively distributed among multiple processors, the failure of one processor does not lead to a system failure but merely a reduction in performance. For instance, if there are ten processors, and one fails, the remaining nine processors can collectively assume the workload of the failed processor. As a result, the entire system runs at a reduced speed, rather than crashing completely.

Enhanced reliability is crucial in numerous applications, and the ability to continue providing service in proportion to the surviving hardware is referred to as "graceful degradation." Some systems go a step further and are classified as "fault-tolerant." These systems can endure the failure of any single component and continue operating. Achieving fault tolerance necessitates mechanisms to detect, diagnose, and potentially correct failures. An example of a fault-tolerant system is the HP NonStop (formerly Tandem) system, which employs hardware and software duplication. It consists of multiple pairs of CPUs operating in lockstep. Each instruction is executed by both processors in the pair, and their results are compared. If discrepancies arise, it signifies a fault, and both CPUs are halted. The task is then shifted to another pair of CPUs, and the problematic instruction is retried. While this approach ensures fault tolerance, it can be expensive due to the need for specialized hardware and substantial duplication of components.

6.4 Types of Parallel System

6.4.1 Types of Mutiprocessor systems

Contemporary multi-processor systems fall into **two** categories: asymmetric multiprocessing and symmetric multiprocessing (SMP).

Symmetric Multiprocessing (SMP):

In modern computing, symmetric multiprocessing (SMP) is the prevalent approach. In an SMP system, all processors are treated as equals, with no boss-worker hierarchy among them. Each processor is capable of executing all tasks within the operating system. A typical SMP configuration includes multiple processors, each with its set of registers and private caches, but they share the physical memory. An example of an SMP system is AIX, a commercial version of UNIX by IBM. An SMP system is beneficial

because it allows many processes to run concurrently, with the number of processes matching the number of CPUs, without significant performance degradation. However, managing I/O and resource allocation is crucial to ensure data reaches the right processor, as idle and overloaded CPUs can lead to inefficiencies. These inefficiencies can be mitigated by enabling processors to share specific data structures. Building a system of this kind requires careful design. Virtually all modern operating systems, including Windows, Mac OS X, and Linux, provide support for SMP.

The distinction between symmetric and asymmetric multiprocessing can be due to either hardware or software choices. Special hardware can distinguish between processors, or the software can be designed to have a single boss and multiple workers. For instance, Sun Microsystems' SunOS Version 4 used asymmetric multiprocessing, whereas Version 5 (Solaris) employed symmetric multiprocessing on the same hardware.

Multiprocessing is employed to add CPUs and increase computational power. If a CPU includes an integrated memory controller, adding CPUs can also boost the addressable memory capacity of the system. Multiprocessing can shift a system's memory access model from uniform memory access (UMA) to non-uniform memory access (NUMA). UMA denotes a scenario where accessing any RAM from any CPU takes the same amount of time. In contrast, NUMA can lead to varying memory access times, creating a performance penalty. Operating systems can alleviate the NUMA penalty through resource management strategies.

A recent trend in CPU design is the incorporation of multiple computing cores on a single chip, leading to what we term multicore multiprocessor systems. These systems can be more efficient than using multiple chips with single cores because on-chip communication is faster than communication between separate chips. Furthermore, a single chip with multiple cores consumes significantly less power than multiple single-core chips.

Asymmetric Multiprocessing:

Asymmetric multiprocessing, in contrast to SMP, involves a specific task assignment for each processor within the system. It typically features a boss processor that oversees the system, while other processors either take direction from the boss or perform predefined tasks. This establishes a boss-worker relationship, where the boss processor handles task scheduling and work allocation to the worker processors.

For example, in an asymmetric multiprocessing system, one processor may have a dedicated role, such as managing system control, while other processors perform specialized tasks, like handling user requests or specific computations. The boss processor coordinates the activities of the worker processors, ensuring that each processor contributes to the overall functioning of the system.

The choice between symmetric and asymmetric multiprocessing can result from hardware or software decisions. Specialized hardware may be used to distinguish between processors, or the software can be written to enforce a single boss and multiple worker model.

In asymmetric multiprocessing, the key advantage is the clear allocation of responsibilities, which can simplify the design of systems with specific, dedicated tasks for each processor. However, this approach may lead to a more centralized control model, which can become a bottleneck as the system's complexity increases. Careful coordination and communication are essential to ensure that tasks are completed

efficiently in an asymmetric multiprocessing environment.

6.4.2 Clustered System

A clustered system is another type of multiprocessor system, distinct from the ones discussed in Section 6.4.1. In a clustered system, multiple CPUs are brought together by linking two or more individual systems, also known as nodes. These systems are considered to be loosely coupled, meaning that they operate semi-independently. Each node can be a single-processor system or a multicore system. It's important to note that there isn't a universally agreed-upon definition for clustered systems, and various commercial packages have different interpretations of what constitutes a cluster and why one form might be superior to another.

However, a generally accepted definition of clustered systems is that they share storage resources and are interconnected through a local-area network (LAN) or a high-speed interconnect like InfiniBand. Clustering is typically employed to ensure high availability of services, even if one or more systems within the cluster experience failures. This high availability is achieved by introducing redundancy into the system.

Cluster software runs on each node in the cluster, and these nodes can monitor the health of one another over the LAN. If one of the monitored machines fails, the monitoring machine can take control of its storage resources and restart the applications that were running on the failed machine. This approach results in only a brief interruption of service for the end users and clients of the applications.

Clusters can be structured asymmetrically or symmetrically. In asymmetric clustering, one machine remains in hot-standby mode while another actively runs the applications. The hot-standby host merely monitors the active server and becomes the active server if the primary one fails. In contrast, symmetric clustering involves multiple hosts running applications and monitoring each other, making more efficient use of available hardware. However, it does require that more than one application is available to run.

Because clusters consist of multiple computer systems connected via a network, they can also be used to create high-performance computing environments. These systems can deliver significantly greater computational power compared to single-processor or SMP systems because they can run applications concurrently on all the computers within the cluster. To fully exploit this capability, applications must be specifically designed to take advantage of the cluster through a technique called parallelization. Parallelization divides a program into separate components that run in parallel on individual computers within the cluster, and the results from all nodes are combined into a final solution.

Other cluster configurations include parallel clusters and clusters that operate over a wide-area network (WAN). Parallel clusters enable multiple hosts to access the same data on shared storage. However, this typically requires special versions of software and applications because most operating systems lack support for simultaneous data access by multiple hosts. For example, Oracle Real Application Cluster is a version of Oracle's database designed for parallel cluster operation. In this setup, each machine runs Oracle, and software manages access to the shared disk. To ensure no conflicting operations occur, a distributed lock manager (DLM) is commonly included in some cluster technology.

Cluster technology is evolving rapidly, and some cluster products now support dozens of systems within a cluster, even if these systems are separated by miles. Many of these advancements are made possible

by storage-area networks (SANs). SANs allow multiple systems to connect to a pool of shared storage. With applications and data stored on the SAN, the cluster software can assign an application to run on any host attached to the SAN. If one host fails, another can take over. In a database cluster, numerous hosts can share the same database, significantly enhancing performance and reliability. Figure 1.8 illustrates the general structure of a clustered system.

6.5 Parallel versus Distributed Operating System

6.5.1 Parallel Operating Systems:

Parallel computing involves the simultaneous execution of a task on multiple processors, tailored to divide and conquer the problem, with the aim of achieving faster results. This approach leverages the fact that most problem-solving processes can be broken down into smaller tasks, which can be executed simultaneously while coordinating their efforts. Systems employing parallel processing are often referred to as having a parallel processor or multiprocessing architecture, and some even reach the level of "massively parallel" with thousands of processors. In recent years, the advent of multicore processors (like Intel's dual-core processors and AMD's dual-core processors) has brought parallel computing to mainstream desktop systems, eliminating the need for multiple processor chips. However, existing software, typically written in sequential programming languages, must be modified or rewritten to effectively utilize modern parallel hardware for true multiprocessing within individual applications. While a system with n parallel processors may be less efficient than a single processor that is n times faster, parallel systems are often more cost-effective to build. They excel at tasks demanding extensive computational power, have strict time constraints, and are divisible into n execution threads. Nevertheless, achieving the anticipated linear speedup in applications with increased multiprocessor support can be challenging due to cache coherence, interprocess synchronization, and communication issues. Most high-performance computing systems, including supercomputers, rely on parallel architectures.

6.5.2 Distributed Operating Systems:

A Distributed System comprises independent computers that are physically and geographically separated and do not share physical memory or a clock. Each processor within the system has its own local memory and communicates using local and wide area networks. These processors may even have heterogeneous architectures. A Distributed Operating System aims to create a seamless and transparent user experience in this distributed architecture, facilitating the sharing of heterogeneous resources efficiently, flexibly, and robustly. It conceals the complexities of the architecture from the user and presents it as a time-shared centralized environment.

Distributed Operating Systems offer several advantages over centralized systems. They can distribute the workload across all processing nodes to enhance throughput and resource efficiency. Data and processes can be replicated at various sites to address node failures or eliminate performance bottlenecks. A well-designed system accommodates scalability by offering mechanisms for distributing control and data. Communication is a crucial aspect of distributed systems, as all interactions depend on it. Message exchange between system components incurs delays due to data propagation, execution of

communication protocols, and scheduling. These delays can lead to inconsistencies, making it challenging to gather global information for decision-making and distinguish between delays and failures. Fault tolerance is vital for distributed systems, as they are more prone to faults due to communication links and numerous processing elements. The system must recover from failures while preserving data integrity and ongoing computation performance.

A distributed architecture with heterogeneous hardware and complex issues can be user-unfriendly. The principle of transparency is applied at various levels in designing a distributed system to hide implementation details and complexities from the user. This includes uniform access to local and remote devices, automated data and process replication, load balancing, and recovery. The ultimate goal of a distributed operating system is to provide a high-performance and robust computing environment with minimal user involvement in managing and controlling distributed resources, while still offering flexibility for customization when needed.

6.6 Self-Assessment Questions

Q1. What is the primary goal of parallel processing? [2 marks, L2]

Q2. Explain advantages of parallel systems over single-processor systems. [2 marks, L2]

Q3. Describe the three key factors driving the recent surge in parallel processing. [4 marks, L2]

Q4. Differentiate between symmetric multiprocessing (SMP) and asymmetric multiprocessing in parallel systems. [4 marks, L3]

Q5. How can clustering enhance high availability in computer systems? [4 marks, L2]

Q6. Why is the decline in hardware costs significant for the adoption of parallel systems? [6 marks, L2]

Q7. What are the main challenges in achieving linear speedup in applications with increased multiprocessor support? [6 marks, L2]

Q8. Explain the concept of graceful degradation in multiprocessor systems and provide an example. [8 marks, L2]

Q9. Describe the advantages and disadvantages of symmetric clustering in clustered systems. [8 marks, L2]

Q10. Provide an in-depth analysis of the factors that have driven the recent advancements in parallel systems and their applications. [10 marks, L2]

Q11. In what scenarios is parallel processing more cost-effective than single-processor systems, and why? [4 marks, L2]

Q12. Explain the concept of graceful degradation in multiprocessor systems with an example [4 marks, L2]

6.7 Self-Assessment Activities

A1. Create a chart with two columns for advantages and disadvantages of parallel systems. Fill in the chart with points from the content. Rate your understanding of each point as high, medium, or low.

A2. Research a real-world case study where parallel processing has been applied (e.g., in scientific

research or industry). Analyze the case study, understanding the problem, solution, and the benefits of using parallel systems.

A3. Compare and contrast parallel processing and distributed computing. Identify their key differences and similarities in terms of architecture, goals, and applications.

A4. Pretend you are teaching someone who has no knowledge of parallel systems. Explain key concepts such as symmetric multiprocessing, graceful degradation, or transparent computing in a way that is easy for a beginner to understand.

A5. Identify a field or industry you're interested in (e.g., healthcare, finance, or gaming). Research how parallel processing is currently being applied in that industry, and discuss its benefits and challenges.

6.8 Multiple-Choice Questions

Q1. What is the primary goal of parallel processing? [1 mark, L1]

A. Reducing hardware costs
B. Enhancing computational performance
C. Isolating processors
D. Minimizing data exchange

Q2. Which of the following is NOT one of the key factors driving the recent surge in parallel processing? [1 mark, L1]

A. Declining Hardware Costs
B. Advances in Integrated Circuit Technology (VLSI)
C. Limitations of Single-Processor Speed
D. Increasing Software Complexity

Q3. What is the primary advantage of clustered systems? [1 mark, L1]

A. Reduced hardware costs
B. Improved single-processor performance
C. High availability
D. Enhanced data storage capacity

Q4. What is the primary goal of parallel operating systems? [1 mark, L1]

A. To enhance hardware costs
B. To divide tasks into smaller segments
C. To improve user-friendliness
D. To eliminate the need for parallel processing

Q5. Which type of operating system aims to create a seamless and transparent user experience in a distributed architecture? [1 mark, L1]

A. Parallel Operating System
B. Symmetric Operating System.
C. Asymmetric Operating System.
D. Distributed Operating System

Q6. What is the primary advantage of transparent computing in distributed operating systems? [1 mark, L1]

A. Improved hardware performance.
B. User-friendly access to resources

C. Reduced hardware complexity.
D. Enhanced fault tolerance.

Q7. In which scenarios is parallel processing more cost-effective than single-processor systems? [1 mark, L1]

A. When tasks can be divided into execution threads.
B. When hardware costs are increasing.
C. When tasks have no time constraints.
D. When software complexity is minimal

Q8. What is the primary challenge in achieving linear speedup in applications with increased multiprocessor support? [1 mark, L1]

A. Hardware costs.
B. Cache coherence.
C. Transparent computing
D. Single-processor performance

Q9. What is the primary goal of graceful degradation in multiprocessor systems?[1 mark, L1]

A. Improving hardware reliability.
B. Avoiding processor failures
C. Ensuring linear speedup.
D. Allowing the system to continue operating at reduced speed

Q10. What is the primary advantage of multiprocessor systems over single-processor systems in terms of fault tolerance? [1 mark, L1]

A. Multiprocessor systems are immune to failures..
B. Multiprocessor systems experience fewer failures.
C. Multiprocessor systems can continue operating with reduced performance in the event of failures.
D. Multiprocessor systems are less reliable than single-processor systems

Q11. In a clustered system, what is the role of hot-standby hosts? [1 mark, L1]

A. To actively run applications.
B. To monitor the health of the active server.
C. To coordinate communication between cluster nodes.
D. To replicate data across nodes.

Q12. In the context of distributed operating systems, what does scalability refer to? [1 mark, L1]

A. The ability to operate in a distributed environment
B. The ability to share resources across different nodes
C. The ability to increase or decrease the number of processing nodes as needed
D. The ability to hide system complexities from the user

Q13. What is the primary advantage of symmetric clustering in clustered systems?

A. Efficient use of hardware resources
B. Centralized control model
C. Higher fault tolerance
D. User-friendly transparency

Q14. In parallel systems, what does "massively parallel" typically refer to?

A. The use of symmetric multiprocessing (SMP)

B. The use of asymmetric multiprocessing
C. Systems with thousands of processors
D. Systems with high single-processor speed

Q15. What is the primary goal of parallelization in parallel computing?
A. To eliminate the need for multiple processors
B. To divide tasks into smaller components that can run in parallel
C. To synchronize all processor operations
D. To reduce cache coherence issues

6.9 Keys to Multiple-Choice Questions

Q1. Enhancing computational performance (B)
Q2. Increasing Software Complexity (D)
Q3. High availability (C)
Q4. To divide tasks into smaller segments (B)
Q5. Distributed Operating System. (D)
Q6. User-friendly access to resources. (B)
Q7. When tasks can be divided into execution threads. (A)
Q8. Cache coherence. (B)
Q9. Allowing the system to continue operating at reduced speed. (D)
Q10. Multiprocessor systems can continue operating with reduced performance in the event of failures. (C)
Q11. To monitor the health of the active server. (C)
Q12. The ability to increase or decrease the number of processing nodes as needed. (C)
Q13. Efficient use of hardware resources (A)
Q14. Systems with thousands of processors. (C)
Q15. To divide tasks into smaller components that can run in parallel(B)

6.10 Summary of the Unit

Parallel processing, also known as multiprocessing, involves breaking down complex tasks into multiple segments and executing them simultaneously using multiple CPUs. This technique aims to improve computational speed and efficiency by utilizing multiple processors rather than relying on a single CPU. Parallel systems have gained prominence in various fields, including fluid dynamics, weather forecasting, data processing, artificial intelligence, and more. The recent surge in parallel processing is driven by declining hardware costs, advances in integrated circuit technology (VLSI), and limitations in single-processor speed. These factors make parallel computing an attractive approach to enhance computational performance. Parallel systems can be categorized into symmetric multiprocessing (SMP), asymmetric multiprocessing, and clustered systems. SMP treats all processors as equals and is common in modern computing. Asymmetric multiprocessing assigns specific tasks to each processor, often with a boss-worker hierarchy. Clustered systems link multiple computers for high availability and enhanced computational power. Parallel operating systems focus on simultaneous execution of tasks on multiple processors within a single system, suitable for tasks that can be divided into execution threads.

Distributed operating systems, on the other hand, manage independent computers that are physically separated, aiming to create a seamless user experience in a distributed environment. These systems have advantages like workload distribution, fault tolerance, and scalability but also come with complexities that need to be hidden through transparency mechanisms. Parallel and distributed systems play a crucial role in addressing the growing demand for faster and more efficient computational solutions in various domains.

6.11 Keywords

Parallel processing, Multiprocessing, Multicore processors, Symmetric Multiprocessing (SMP), Asymmetric Multiprocessing, Clustered Systems, High Availability, Fault Tolerance, Scalability, Parallel Operating Systems, Graceful Degradation, Massively Parallel Computing

6.12 Recommended Learning Resources

[1] Silberschatz, A., Galvin, P., & Gagne, G. (2005). Operating System Concepts, 7th ed., Hoboken.

[2] William stalling. Operating Systems: Internal and design principles, 7th edition PHI

[3] D.M. Dhamdhere. Operating Systems: A concept-based Approach, 2nd Edition, TMH