Cloud Computing – SWE 2022 [~KShawki]

***Parallel Computing***

* **Parallel Computer:** is a collection of processing elements that communicate and cooperate to solve large problems fast (HPC).
* Processors are closely connected, all processors share the same memory, and the processors communicate by accessing this shared memory
* **Examples:** Multicore processors, Graphics Processing Units (GPUs).

***Shared Memory Access***

There are three basic approaches: CRCW, CREW, EREW, ERCW:

1. **CRCW: Concurrent Read, Concurrent Write.** 
   * Simultaneous reads and writes are allowed to a memory cell.
   * The model must indicate how simultaneous writes are handled:

* **Common Write:** If processors write simultaneously, they must write **same value**.
* **Priority Write:** Processors have priority order, and the highest-priority processor's write wins in case of conflict.
* **Arbitrary Write:** In case of conflict, one of the requested writes will succeed. But the outcome is not predictable, and the program must work regardless of which processor wins.
* **Combining Write:** Simultaneous writes are combined with some function, such as adding values together.

1. **CREW: Concurrent Read, Exclusive Write**
   * Here different processors are allowed to read the same memory cell simultaneously, but writing a program must guarantee that only one processor can write to any memory cell at a time.
2. **EREW: Exclusive Read, Exclusive Write** 
   * The program must be written so that no memory cell is accessed simultaneously in any way.
3. **ERCW: Exclusive Read, Concurrent Write** 
   * There is no reason to consider this possibility.

­­­­­­­­­­­­­­­­­­



***Distributed System***

* **Distributed System:** A collection of independent computers that appears to its users as a single coherent system.
* Architectural design for software that executes on more than one processor

***Construction of Distributed Systems:***

Units/Node (Computers), Network, Software

***Enslow’s Definition***

Distributed System = Distributed hardware + Distributed control + Distributed data.

***Goals of Distributed System***

1. Resource Sharing: Hardware, Software and data.
2. Openness: Openness of main interface of the system, scalability of the current system.
3. Concurrency: Concurrent execution of the processes, high performance, nice rate of price and performance. (PC cluster = poor man’s supercomputer).
4. Fault Tolerance: Ability to tolerate the fault of system units, availability (using potential redundancy to overcome the system fault).
5. A distributed system can be looked as one computer (access transparency, position transparency, parallel transparency, …).

***Advantages and Disadvantages of Distributed Systems***

|  |  |
| --- | --- |
| **Advantages** | **Disadvantages** |
| Resource Sharing | Network reliance |
| Expandability | Complexities |
| Local autonomy | Security |
| Improved performance | Multiple point of failure |
| Improved reliability and availability | Manageability |
| Potential cost reductions | Unpredictability |

***Types of Distributed systems:***

* Client-Server
* Peer-To-Peer
* Processor Pool
* Network Operating Systems

***Challenges in Distributed systems:***

* Concurrency
* Heterogeneity
* Openness
* Security
* Scalability
* Failure handling
* Transparency

***Parallel Verses Distributed:***

|  |  |  |
| --- | --- | --- |
|  | **Parallel** | **Distributed** |
| **Hardware** | identical processors  regular interconnection | different types of processors Networks |
| **Memory** | Shared memory | Distributed memory |
| **Control** | synchronized (global clock) | A synchronized execution of tasks  (no global clock) |
| **Main focus** | Performance | Information / resource sharing Reliability / availability, … |
| **Homogeneity** | tasks perform similar functions | Inhomogeneity, tasks perform different functions |

***Middleware:***

* **Middleware:** Software that manages and supports the different components of a distributed system. In essence, it sits in the **middle** of the system.
* Middleware is usually off-the-shelf rather than specially written software.
* **Example:** Transaction Processing Monitors, Data convertors, Communication controllers.