Name- Kumar Sachin
Reg. No. 20MCA0178

Digital Assignment-1

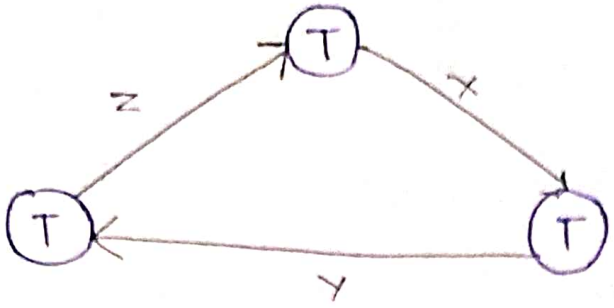
Q. Write a 2 page report on the following topics:-

1. Distributed Deadlock Detection
2. Distributed Shared Memory
3. Distributed Scheduling
4. Distributed fault Tolerence

1. Distributed Deadlock Detection :-

Deadlock is a state of a database system having two or more transactions, when each transaction is waiting for a data item that is being locked by some other transaction. A deadlock can be indicated by a cycle in the wait-for-graph. This is a directed graph in which the vertices denote transactions and the edges denote waits for data items.

For example, in the following wait-for-graph, transaction T1 is waiting for data item X which is locked by T3. T3 is waiting for Y which is locked by T2 and T2 is waiting for Z which is locked by T1. Hence, a waiting cycle is formed, and none of the transactions can proceed executing.



There are three classical approaches for deadlock handling:

→ Deadlock prevention
→ Deadlock avoidance
→ Deadlock detection and removal

Deadlock Prevention:- The deadlock prevention approach does not allow any transaction to aquire locks that will lead to deadlocks. The convention is that when more than one transactions request for locking the same data item, only one of them is granted the lock.

Deadlock Avoidance:- The deadlock avoidance approach handles deadlocks before they occur. It analyzes the transactions and the locks to determine whether or not waiting leads to a deadlock.

There are two algorithms for this purpose:-

- Wait-Die:- If T1 is older than T2, T1 is allowed to wait. otherwise, if T1 is younger than T2, T1 is aborted and later restarted.
- Wound-Wait:- If T1 is older than T2, T2 is aborted and later restarted. Otherwise, if T1 is younger than T2, T1 is allowed to wait.

Deadlock Detection and Removal:- The deadlock detection and removal approach runs a deadlock detection algorithm periodically and removes deadlock in case there is one. It does not check for deadlock when a transaction places a request for a lock. Some of the method used to victim selection are :-

- choose the youngest transaction
- choose the transaction with fewest data items.
- choose the transaction that has performed less no. of updates.
- choose the transaction having least restart overhead.
- choose the transaction which is commonly to two or more cycles.

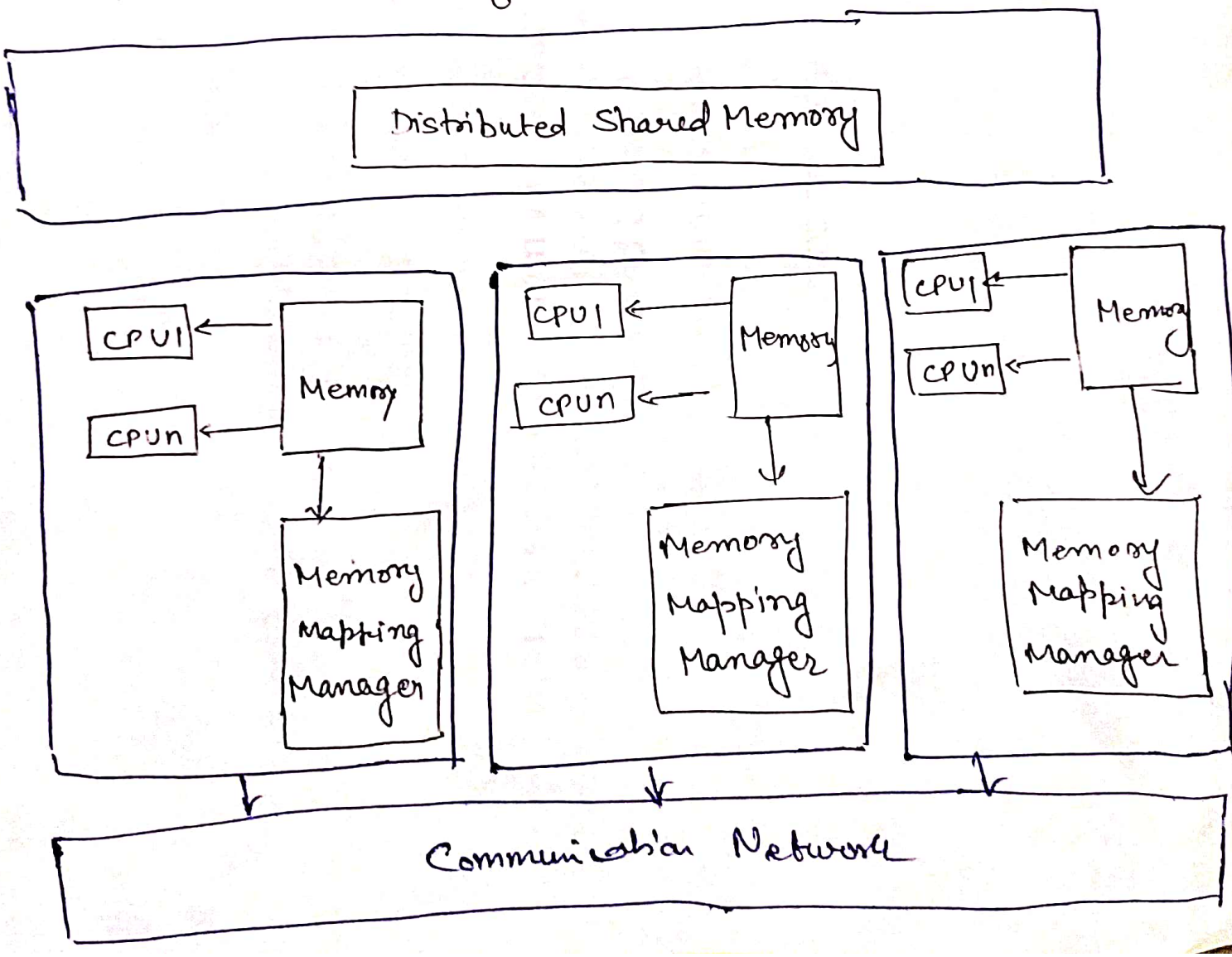
Distributed Deadlock Detection

Just like centralized deadlock detection approach, dea-dlocks are allowed to occurs and are removed if detected. The system does not perform any checks when a transaction places a lock request. For implementation, global wait-for-graphs are created. Existence of a cycle in the global wait-for-graph indicates deadlocks.

- Centralized Deadlock Detector:- One site is designated as the central deadlock detector.
- Hierarchical Deadlock Detector:- A number of detectors are arranged in hierarchy.
- Distributed Deadlock Detector:- All the sites participate in detecting deadlocks and removing them.

2. Distributed Shared Memory :-

Distributed Shared Memory implements the distributed Systems shared memory model in a distributed system, that hasn't any physically shared memory. Shared model provides a virtual address area shared between any or all nodes. To beat the high forged of communication in distributed system. Distributed shared Memory, model provides a virtual address area shared between all nodes. Systems move information to the placement of access. Information moves between main memory an secondary memory (with in a node) and between main recollections of various nodes.



Distributed Shared Memory permits programs running on separate reasons to share information while not the software engineer having to agitate causation message instead underlying technology can sent the messages to stay the DSM consistent between compute.

Architecture of Distributed Shared Memory:- Every node consi of 1 or additional CPU's and a memory unit. High-speed communication network is employed for connecting the nodes. A straight forward message passing system permits processes on completely different nodes to exchange one another.

Memory mapping manager unit:-

Memory mapping manager routine in every node maps the native memory onto the shared computed storage. For mapping operation, the shared memory house is divided into blocks.

Communication Network Unit:-

One method access information within the shared addre house mapping manager maps the shared memory address to the physical memory. The mapped layer of code enforced either within the operating kernel or as a runtime routine.

Physical memory on every node holds pages of shared virtual-address house. Native pages area unit gift in some node's memory, Remote pages in some other node's memory.

3. Distributed Scheduling :-

Distributed scheduling is an approach that enables local decision makers to create schedules that consider local objectives and constraints within the boundaries of the overall system objectives. Local decisions from different parts of the system are then integrated through coordination and communication mechanisms. Distributed scheduling attracts the interest of many researchers from a variety of disciplines, such as computer science, economics, manufacturing, and ~~device~~ service operations management.

Distributed System Modeling Mechanism :-

When a job is submitted to the system, job placement will be done. i.e, to decide which workstations to run the job cooperatively. Along with the job submission, a description of the attributes of the job is also submitted to the system in order to specify the resource requirement, such as memory size requirement, expect CPU time, deadline time, etc. In the meantime, the system always maintain an information table, either distributed or centralized, to record the current resource status of each workstation. e.g CPU load, free memory size, etc. Then, a matchmaking frame will do matching work to find the most suitable set of workstation to meet the requirements of the job.

Properties of a Good Scheduler :-

(i) General Purpose :- A scheduling approach make few assumptions about and have few restrictions to the types of applications that can be executed. Interative job, distributed and parallel applications, as well as non - interactive batch jobs, should all be supported with good performance.

(ii) Efficiency :- It has two meanings : one is that it should improve the performance of scheduled job as much as possible; the other is that the scheduling should incur reasonably low overhead so that it won't counteratteck the benefit.

(iii) fairness :- Sharing resources among users raises new challenges in guaranteeing that each user obtains his/her fair share when demand is heavy.

(iv) Dynamic :- The algorithm employed to decide where to process a task should respond to load changes, and exploit the full extent of the resources available.

(v) Transparency :- The behaviour and result of tasks execution should not be affected by the host (s) on which it executes. In particular, there should be no difference between local and remote execution.

4. Distributed fault Tolerance :-

A distributed System consists of several independent processing components that interact with each other via an interconnecting communication link network consisting of communication components. Distributed computing refers to the algorithmic controlling of the distributed systems processing components by means of a distributed program in order to reach a collective goal, that is, to provide a certain service. Unfortunately the components of literally every system are naturally imperfect and therefore prone to failures that may render the system unable to provide the service.

In any distributed system, these kind of problems can occur.

(i) fault (ii) Error (iii) failures.

All these are inter related. It is quite fair to say that fault is the root cause, where a problem starts, error is the result of fault and failure is the final out come.

Types of faults :-

Transient faults	Permanent faults
(i) Occur for a very short duration	(i) Permanent
(ii) Hard to locate	(ii) Easy to identified.
(iii) Do not affect the system to a great extent.	(iii) Can cause servere damage to the entire system.
(iv) Network fault, processor fault, Media fault are some of the examples	(iv) Node level faults - when an Entire node is unavialable,

Fault Tolerance :- The ability of a system to continue functioning in the event of a partial failure. Though the system continues to function but overall performance may get affected.

Distributed system are made up of a large number of components, developing a system which is hundred percent fault tolerance is practically very challenging.

Two main reasons for the occurrence of a fault :-

1) Node failure - Hardware or software failure.
11) Malicious Error - Caused by unauthorized Access.

Need of fault Tolerance :-

fault Tolerance is needed in order to provide 3 main feature to distributed systems :-

(i) Reliability :- focuses on a continuous services with out any interruptions.

(ii) Availability :- concerned with real readiness of the system.

(iii) Security :- Prevents any unauthorized access.

ex :- Patient Monitoring systems, flight control system, Banking system etc.