7. Lambort's algorithm for distributed database.

Ans:- In a distributed database system, multiple sites (nodes) may need to access a shared resource such as a data item, file, or critical section (cs).

Since there is no shared clock or shared memory coordination is difficult.

Objectives .

1. Mutual Exclusion
2. No deadlock
3. No starvation
4. Fairness

- The algorithm is fair in the sense that a request for cs are executed in the order of their timestamps and time is determined by logical clocks.

→ when a site processes a request for the cs, it updates its local clock and assigns the request a timestamp.

→ The algorithm executes cs requests in the increasing order of timestamps.

→ Every site S_i keeps a queue, request-queue, which contains mutual exclusion request ordered by their timestamps.

→ The algorithm executes cs request in the increasing order of timestamps.

Algorithm

1. Requesting the critical section

- when a site S_i wants to enter the cs, it broadcast a REQUEST (ts_i, i) message to all other sites and places the request on request-queue, (cts_i, i) denotes the timestamp of the request).

• when a site S_j receives the REQUEST (ts_i, i) message from site S_i, it places site S_i's request in request_queue and returns a timestamped REPLY message to S_i.

2. Executing the ~~timp~~ Critical Section
   Site S_i enters the CS when the following two conditions hold:
   L1: S_i has received a message with timestamp larger than (ts_i, i) from all other sites.
   L2: S_i's cs request is at the top of request_queue.

3. Releasing the critical section.
   • Site S_i, upon exiting the CS, removes its request from the top of its request queue and broadcast a timestamp RELEASE message to all other sites.
   • when a site S_j receives a RELEASE message from site S_i, it removes S_i's request from its request queue.

Real time Applications of Lamport's Algorithm

1. Distributed database System
   * Controlling concurrent access to shared database records.
   * Ensuring transaction Consistency.

2. Distributed file system
   * synchronizing access to shared files.
   * Preventing write conflicts.

3. Cloud Computing Environments
   * managing access to shared cloud resources.
   * Coordinating distributed services.

4. Distributed Operating Systems:
   * Scheduling critical section execution across nodes

5. Replication Management

Requirements to Implement the algorithm

1. Reliable message passing - without loss and finite time.
2. Logical clock mechanism - Each process must maintain and update lamport timestamps.
3. Request Queue - A priority queue sorted by (timestamp, processid)
4. Stable Network.
5. knowledge of other process.
6. Unique process identifiers.