CONTENTS

KCS 077 : Distributed Systems

UNIT 1

Characterization of Distributed System

CONTENTS

PART-1

Introduction, Example of Distributed System.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.1. **What is a distributed system ? Describe the main characteristics of distributed system. Give two example of distributed system.** **AKTU 2014-15, Marks 05**

Answer

Distributed system :

1. A distributed system is a system in which software or hardware components connected via communication network communicates and coordinates their actions only by passing messages.
2. Computers that are connected by a network may be spatially separated by distance.
3. Resources may be managed by servers and accessed by clients.

Characteristics of distributed system :

1. **Heterogeneity :** Distributed system enables the users to access services and run application over a heterogeneous collection of computers and networks.
2. **Openness :** The openness of a computer system is the characteristics that determine whether the system can be extended and re-implemented in various ways.
3. **Concurrency :** Concurrency in distributed system is use to help different users to access the shared resource at the same time.
4. **Scalability :** A system is described as scalable if it remains effective when there is significant increase in the number of resources and the numbers of users.
5. **Security :** Security provides confidentially, integrity and availability of the information resources.

Example of distributed system :

1. **Internet :** The Internet is a very large distributed system. It enables users to make use of services such as the World Wide Web, e-mail and file transfer.

2. **Intranet :**
   a. An intranet is a private network that is contained within an enterprise.
   b. An intranet is connected to the internet via router, which allows the users inside the intranet to make use of services such as web or e-mail.

Que 1.2. **What are distributed systems ? What are significant advantages and applications of distributed system ?**

AKTU 2018-19, Marks 10

Answer

Distributed system : Refer Q. 1.1, Page 1–2B, Unit-1.

Advantages of distributed system :

1. **Data sharing :** It allows many users to access to a common database.
2. **Resource sharing :** Expensive peripherals like color printers can be shared among different nodes (or systems).
3. **Communication :** Enhance human-to-human communication, e.g., email, chat.
4. **Flexibility :** Spread the workload over the available machines.

Applications of distributed systems :

1. Telecommunication networks such as telephone networks and cellular networks.
2. Network applications, world wide web and peer-to-peer networks.
3. Real-time process controls aircraft control systems.
4. Parallel computation.

Que 1.3. **How the distributed computing system is better than parallel processing system ? Explain.** **AKTU 2017-18, Marks 10**

Answer

Distributed computing system is better than parallel processing system because of following advantages :

1. **Economics :** Microprocessors offer better performance than parallel processing system.
2. **Speed :** A distributed system may have more total computing power than parallel processing system.
3. **Reliability :** If some of the machines are downed, the distributed system as a whole can still survive with small degradation of performance.

4. **Incremental growth :** Computing power can be added in small increments.
5. **Data sharing :** Allow many users access to a common database.
6. **Device sharing :** Allow many users to share expensive peripherals.
7. **Flexibility :** In distributed computing workload can be spread over the available machines in the most cost effective way.

Que 1.4. **What is distributed transparency ? Explain the different types of distributed transparencies.** **AKTU 2017-18, Marks 10**

Answer

Distributed transparency is the property of distributed databases by the virtue of which the internal details of the distribution are hidden from the users.

Types of transparencies :

1. **Access transparency :** It enables local and remote resources to be accessed using identical operations.
2. **Location transparency :** It enables resources to be accessed without knowledge of their physical or network location.
3. **Concurrency transparency :** It enables several processes to operate concurrently using shared resources without interference between them.
4. **Replication transparency :** It enables multiple instances of resources to be used to increase reliability and performance without knowledge of the replicas by users or application programmers.
5. **Failure transparency :** It enables the concealment of faults, allowing users and application program to complete their tasks despite the failure of hardware or software components.
6. **Performance transparency :** It allows the system to be reconfigured to improved performances as load varies.

PART-2

Resource Sharing and Web Challenges, Architectural Models, Fundamental Models.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.5. How the resource sharing is done in distributed system ? Explain with an example.

Answer

1. Resource sharing is one of the major advantages which is obtained from distributed system.
2. In a distributed system, the resources are enclosed within computers and can only be accessed from other computers by communication.
3. Each resource must be managed by a program that offers a communication interface enabling the resource to be accessed.
4. The program also helps the resources to be updated reliably and consistently.

For example :

1. The client-server model provides an effective general purpose approach to the sharing of information and resources in distributed systems.
2. In this model, a client sends a request to a server for getting some services such as reading a block of a file.
3. The server executes the request and sends back a reply to the client that contains the result of request processing.

Que 1.6. Discuss the major issue in designing a distributed system.

AKTU 2017-18, Marks 10

Answer

Major issues in designing a distributed system :

1. **Heterogeneity :**
   a. Distributed system must be constructed from variety of different networks, operating systems, computer hardware's and programming languages.
   b. Internet communication protocol mask the difference (heterogeneity) in networks and middleware can deal with the other differences.
2. **Openness :** Distributed system should be extensible *i.e.* to develop interface for the distributed system component so that they can be integrated to new extension of distributed system.
3. **Security :**
   a. Encryption can be used to provide adequate amount of shared resources and to keep sensitive information secret when it is transmitted in messages over a network.
   b. Denial of Services (DoS) attack is one of the big problems for security.

4. **Scalability :**
   a. Scalability refers to the capability of a system to adapt to increased service load.
   b. It is inevitable that a distributed system will grow with time since it is very common to add new machines to take care of increased work load. Therefore, a distributed system should be designed to easily cope with the growth of nodes and users in the system
5. **Fault avoidance :**
   a. Fault avoidance deals with designing the components of the system in such a way that the occurrence of faults in minimized.
   b. Conservative design practice such as using high reliability components are often employed for improving the system's reliability based on the idea of fault avoidance.
6. **Transparency :**
   a. Transparency aims to hide the details of distribution from the users.
   b. For an example, user or programmer need not be concerned with its location or the details of how its operations are accessed by other components, or whether it will be replicated or migrated.

Que 1.7. Why is scalability an important feature in the design of distributed system ? Discuss some of the guiding principles for designing a scalable distributed system.

AKTU 2014-15, Marks 10

Answer

Scalability is important features in design of distributed system because :

a. It helps the system to work efficiently with an increase in number of users.
b. It increases the system performance by incorporating additional resources.

Guiding principle for designing scalable distributed system :

1. **Avoid centralized entities :**
   a. Use of centralized entities should be avoided in the design of scalable distributed system because :
      i. In centralized system, if centralized entity fails then the entire system will also fail.
      ii. Capacity of the network that connects the centralized entity gets saturated.
      iii. In case of wide-area network system, traffic in the network increases.

b. Replication of resources and distributed control algorithms are frequently used techniques to achieve scalable system.
c. For better scalability, functionally symmetric configuration should be used in which all nodes of the system should play equal role in the operation of the system.

2. **Avoid centralized algorithms :**
   a. A centralized algorithm is one that operates by collecting information from all nodes, processing this information on a single node and then distributing the result to other nodes.
   b. Time complexity of centralized algorithm may be very high which creates heavy network traffic and consumes network bandwidth.
   c. Therefore, in the design of a distributed operating system, only decentralized algorithm should be used.
3. **Perform most operations on client workstations :**
   a. If possible, an operation should be performed on the client's workstation.
   b. This principle enhances the scalability of the system, since it allows graceful degradation of system performance as the system grows in size.
   c. Caching is a frequently used technique for the realization of this principle.
4. **Controlling the cost of physical resources :** As the demand for a resource grows, it should be possible to extend the system, at reasonable cost, to meet it.

Que 1.8. **Discuss architectural models of distributed system.**

Answer

1. An architecture model of a distributed system simplifies and abstracts the functions of the individual components of a distributed system
2. It also considers the placement of the components across a network of computers and the interrelationship between the components.
3. The main objective of these models is to make the system reliable, manageable, adaptable and cost-effective.
4. The two main types of architectural model are :
   a. **Client-server model (Search engine) :**
      i. Fig. 1.8.1 illustrates the simple structure in which client processes interact with individual server processes in separate host computers in order to access the shared resources that they manage.

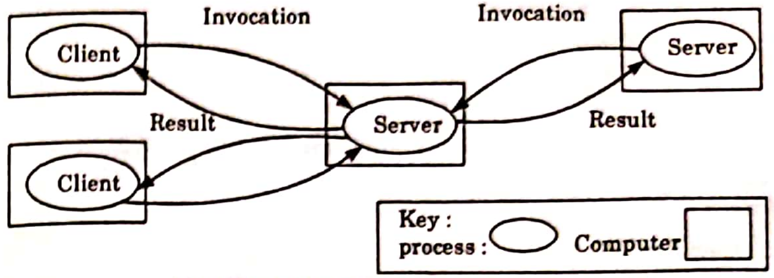


Fig. 1.8.1. Clients invoke individual servers.

      ii. This is the architecture that is most widely employed.
      iii. Client-server model offers a direct and simple approach to the sharing of data and other resources.
      iv. Servers may acts as a client of other servers.
      v. For example, a web server is often a client of a local file server that manages the files in which the web pages are stored.
   b. **Peer-to-Peer model :**
      i. In this architecture, all of the processes which are involved in a task play similar roles, interacting cooperatively as peers without any distinction between client and server processes.
      ii. The Fig. 1.8.2 illustrates the form of a peer-to-peer application.

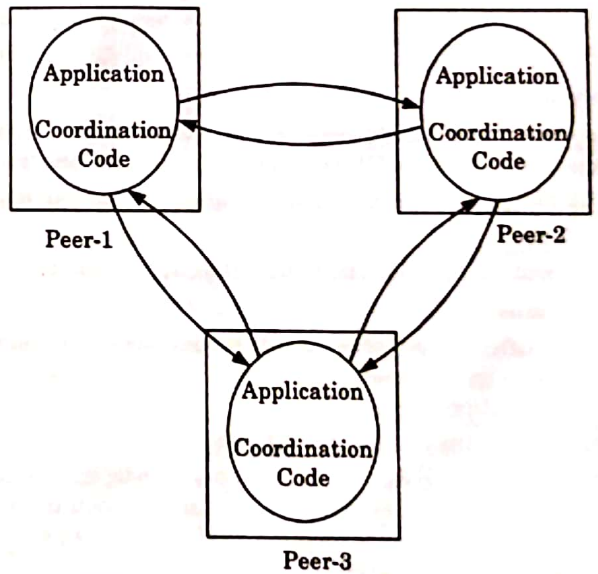


Fig. 1.8.2. Distributed application based on peer-to-peer processes.

iii. Applications are composed of large numbers of peer processes running on separate computers and the pattern of communication between them depends on application requirements.

Que 1.9. **Explain the fundamental models of distributed system.**

Answer

1. Fundamental models are based on the fundamental properties that allow us to be more specific about their characteristics, failures and security risks that they might exhibit.
2. The purpose of a model is :
   a. To make explicit all the relevant assumptions about the systems.
   b. To make generalizations concerning what is possible or impossible, given those assumptions.

Following are the fundamental model :

1. **Interaction models :**
   a. It is concerned with performance of process communication channels and absence of global clock.
   b. Interaction model is further classified as synchronous and asynchronous system.
   c. Interacting process perform all of the activity in a distributed system.
   d. Each process has its own state, consisting of the set of data that it can access and update, including the variable in its program.
   e. The state belonging to each process is completely private.
2. **Failure model :**
   a. In a distributed system both processes and communication channels may fail, so this model is capable of handling all the failure.
   b. The failure model defines and classifies the faults that occur in the system.
   c. It provides a model to understand the effects of faults in the system.
3. **Security model :**
   a. It identifies the possible threats to processes and communication channels in an open distributed system such as integrity, authentication, privacy etc.
   b. The architectural model provides the basis for our security model :
      i. The security of a distributed system can be achieved by securing the processes and the channels used for their interactions and by protecting the objects that they encapsulate against unauthorized access.
      ii. Protection is described in terms of objects, although the concepts apply equally well to resources of all types.

PART-3

Theoretical Foundation for Distributed System : Limitations of Distributed System, Absence of Global Clock, Shared Memory.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.10. **Explain the limitations of distributed system with example.**

AKTU 2018-19, Marks 10

Answer

Limitations of distributed systems are as follows :

1. **Absence of global clock :**
   a. In a distributed system, global clock (or common clock) is not present.
   b. Suppose a global clock is available for all the processes in the system.
   c. In this case, two different processes can observe a global clock value at different instants due to unpredictable message transmission delays.
   d. Therefore, two different processes, may falsely perceive two different instants in physical time to be a single instant in physical time.
2. **Absence of shared memory :**
   a. The computer in a distributed system do not share common memory, an up-to-date state of the entire system is not available to any individual process.
   b. It is necessary for reasoning about the system's behaviour, debugging, recovering from failures, etc.
   c. A process in a distributed system can obtain a coherent but partial view of the system or a complete but incoherent view of the system.
   d. A view is said to be coherent if all the observations of different processes (computers) are made at the same physical time.
   e. Because of the absence of a global clock in a distributed system, obtaining a coherent global state of the system is difficult.

Example :

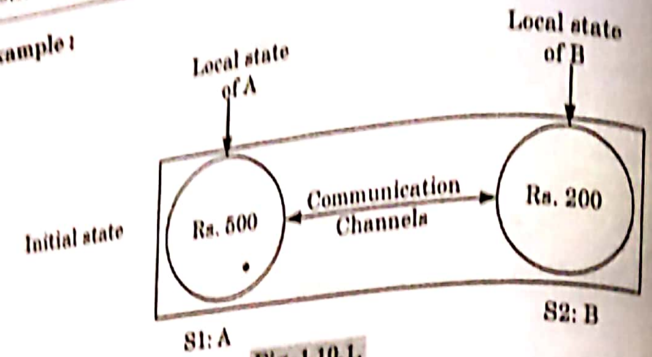


Fig. 1.10.1.

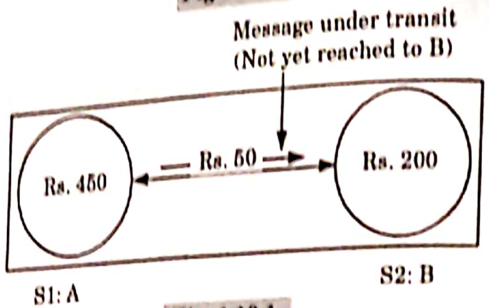


Fig. 1.10.1.

a. S1 records its local state (Rs. 450) just after debit (– 50) and S2 records its location (200) before receiving.

b. If transit message is not taken care off

Global state = Local state S1 + Local state S2

= 450 + 200

= 650 = Rs. 50 missing *i.e.*, in coherent system

Que 1.11. **What are distributed systems ? What are significant advantages, applications and limitations of distributed systems ? Explain with examples, what could be the impact of absence of global clock & shared memory.**

AKTU 2015-16, Marks 10

Answer

Distributed systems : Refer Q. 1.1, Page 1–2B, Unit-1.

Significant advantages and applications of distributed systems : Refer Q. 1.2, Page 1–3B, Unit-1.

Limitations of distributed systems :

1. Absence of shared memory.
2. Absence of global clock.
3. The initial deployment cost of a distributed system is very high.

Impact of the absence of global clock :

1. It is difficult in a distributed system to reason about the temporal order at events.
2. It is difficult to design and debug algorithms for a distributed system as compared to centralized systems.
3. Collecting up-to-date information on the state of the entire system in harder.

Impact of the absence of shared memory :

1. An up-to-date state of the entire system is not available to any of the individual processes.
2. Recovery from failure cannot be possible.

For example : Refer Q. 1.10, Page 1–10B, Unit-1.

PART-4

Logical Clock, Lamport's and Vectors Logical Clocks.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.12. **What are Lamport logical clocks ? List the important conditions to be satisfied by Lamport logical clocks. If A and B represent two distinct events in a process and if $A > B$ then $C(A) < C(B)$ but vice-versa not true. Justify the statement.**

AKTU 2015-16, Marks 10

Answer

Lamport logical clocks :

A Lamport logical clock is a monotonically increasing software counter, whose value need bear no particular relationship to any physical clock.

Following conditions are to be satisfied by Lamport logical clocks :

1. If a and b are two events within the same process P_i and a occurs before b, then $C_i(a) < C_i(b)$.
2. It a is the sending of a message by process P_i and b is the receipt of that message by process P_j, then $C_i(a) < C_j(b)$.
3. A clock C_i associated with a process P_i must always go forward, never backward. That is, corrections to time of a logical clock must always be made by adding a positive value to the clock, never by subtracting value.

Justification : Event 'A' casually affects event 'B' if $A \rightarrow B$. Now, if $A \rightarrow B$ then $C(A) < C(B)$, but it vice-versa (reverse) is not here, because nothing can be said about events by comparing timestamps.

Que 1.13. **Discuss the limitations of Lamport's logical clock with suitable example.**

AKTU 2016-17, Marks 05

OR

What are Lamport logical clocks ? List the important conditions to be satisfied by Lamport logical clocks. Discuss the limitations of Lamport logical clocks.

AKTU 2018-19, Marks 10

Answer

Lamport logical clock and important conditions to be satisfied by Lamport logical clocks : Refer Q. 1.12, Page 1–12B, Unit-1.

Limitation of Lamport's clocks :

1. According to Lamport's system of logical clocks, if $a \rightarrow b$ then $C(a) < C(b)$.
2. However, the reverse is not necessarily true if the events have occurred in different processes.
3. That is, if a and b are events in different processes and $C(a) < C(b)$, then $a \rightarrow b$ is not necessarily true; event a and b may be causally related or may not be causally related.
4. Thus, Lamport's system of clocks is not powerful enough to capture such situations.

For example :

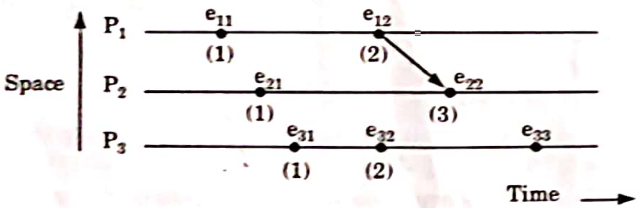


Fig. 1.13.1.

a. Fig. 1.13.1 shows a computation over three processes clearly, $C(e_{11}) < C(e_{12})$ and $C(e_{11}) < C(e_{32})$.

b. However, we can see from the Fig. 1 that event e_{11} is causally related to event e_{22} but not to e_{32}.

c. Note that the initial clock values are assumed to be zero and d is assumed to equal 1.

d. In other words, in Lamport's system of clocks, we can guarantee that if $C(a) < C(b)$ then $b \not\rightarrow a$, however we cannot say whether events a and b are causally related or not by just looking at the timestamps of the events.

e. The reason for the above limitation is that each clock can independently advance due to the occurrence of local events in a process.

f. The Lamport's clock system cannot distinguish between the advancements of clocks due to local events from those due to the exchange of messages between processes.

g. Therefore, using the timestamps assigned by Lamport's clocks we cannot reason about the causal relationship between two events occurring in different processes by just looking at the timestamps of the events.

Que 1.14. **What are vector clocks ? Explain with the help of implementation rule of vector clocks, how they are implemented ? Give the advantages of vector clock over Lamport clock.**

AKTU 2014-15, Marks 05

Answer

Vector clocks :

1. Vector clocks are used in a distributed system to determine whether pairs of events are causally related.
2. Using vector clocks, timestamps are generated for each event in the system, and their causal relationship is determined by comparing those timestamps.

Implementation of vector clocks :

1. Let 'n' be the number of processes in a distributed system. Each process P_i is equipped with a clock C_i, which is an integer vector of length n.
2. Let a, b be a pair of events. Let $C[a][i]$ be the i^{th} element of the vector clock for the event a.
3. $C(a)$ is dominated by $C(b)$ *i.e.*, $C(a) < C(b)$, if and only if the following two conditions hold :
   a. $\forall\, i, 0 \le i \le n-1 : C[a][i] \le C[b][i]$
   b. $\exists\, i, 0 \le i \le n-1 : C[a][i] \le C[b][i]$
4. To implement a system of vector clocks, initialize the vector clock of each process to 0, 0, ..., 0, (n component).

Justification : Event 'A' casually affects event 'B' if $A \rightarrow B$. Now, if $A \rightarrow B$ then $C(A) < C(B)$, but it vice-versa (reverse) is not here, because nothing can be said about events by comparing timestamps.

Que 1.13. **Discuss the limitations of Lamport's logical clock with suitable example.**

AKTU 2016-17, Marks 05

OR

What are Lamport logical clocks ? List the important conditions to be satisfied by Lamport logical clocks. Discuss the limitations of Lamport logical clocks.

AKTU 2018-19, Marks 10

Answer

Lamport logical clock and important conditions to be satisfied by Lamport logical clocks : Refer Q. 1.12, Page 1–12B, Unit-1.

Limitation of Lamport's clocks :

1. According to Lamport's system of logical clocks, if $a \rightarrow b$ then $C(a) < C(b)$.
2. However, the reverse is not necessarily true if the events have occurred in different processes.
3. That is, if a and b are events in different processes and $C(a) < C(b)$, then $a \rightarrow b$ is not necessarily true; event a and b may be causally related or may not be causally related.
4. Thus, Lamport's system of clocks is not powerful enough to capture such situations.

For example :

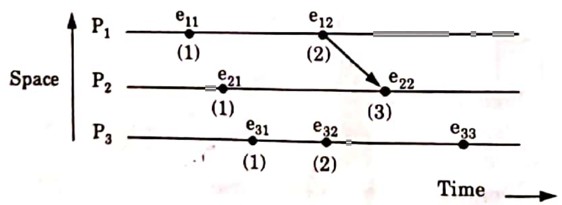


Fig. 1.13.1.

a. Fig. 1.13.1 shows a computation over three processes clearly, $C(e_{11}) < C(e_{12})$ and $C(e_{11}) < C(e_{32})$.

b. However, we can see from the Fig. 1 that event e_{11} is causally related to event e_{22} but not to e_{33}.

c. Note that the initial clock values are assumed to be zero and d is assumed to equal 1.

d. In other words, in Lamport's system of clocks, we can guarantee that if $C(a) < C(b)$ then $b \not\rightarrow a$, however we cannot say whether events a and b are causally related or not by just looking at the timestamps of the events.

e. The reason for the above limitation is that each clock can independently advance due to the occurrence of local events in a process.

f. The Lamport's clock system cannot distinguish between the advancements of clocks due to local events from those due to the exchange of messages between processes.

g. Therefore, using the timestamps assigned by Lamport's clocks we cannot reason about the causal relationship between two events occurring in different processes by just looking at the timestamps of the events.

Que 1.14. **What are vector clocks ? Explain with the help of implementation rule of vector clocks, how they are implemented ? Give the advantages of vector clock over Lamport clock.**

AKTU 2014-15, Marks 05

Answer

Vector clocks :

1. Vector clocks are used in a distributed system to determine whether pairs of events are causally related.
2. Using vector clocks, timestamps are generated for each event in the system, and their causal relationship is determined by comparing those timestamps.

Implementation of vector clocks :

1. Let 'n' be the number of processes in a distributed system. Each process P_i is equipped with a clock C_i, which is an integer vector of length n.
2. Let a, b be a pair of events. Let $C[a][i]$ be the i^{th} element of the vector clock for the event a.
3. $C(a)$ is dominated by $C(b)$ *i.e.*, $C(a) < C(b)$, if and only if the following two conditions hold :
   a. $\forall\, i,\ 0 \le i \le n-1 : C[a][i] \le C[b][i]$
   b. $\exists\, i,\ 0 \le i \le n-1 : C[a][i] \le C[b][i]$
4. To implement a system of vector clocks, initialize the vector clock of each process to 0, 0, ..., 0, (n component).

5. **The implementation rules for vector clocks :**

Rule 1 : Each local event at process P_i increments the i^{th} component of its vector clock (*i.e.*, $C[i]$ by 1).

Rule 2 : The sender appends the vector timestamp to every message that it sends.

Rule 3 : When process P_j receives a message with a vector timestamp from another process, it first increments the j^{th} component $C[j]$ of its own vector clock *i.e.*, $C[j] = C[j] + 1$ and then updates its vector clock as

$$\forall\, i, 0 \le i \le n-1,\ C[i] = \max(T[i], C[i])$$

6. When the vector clock values of two events are incomparable, the events are concurrent.

7. An example of vector timestamp is shown in Fig. 1.14.1. The event with vector timestamp (2, 1, 0) is causally ordered before the event with the vector timestamp (2, 1, 4), but is concurrent with the event having timestamp (0, 0, 2).

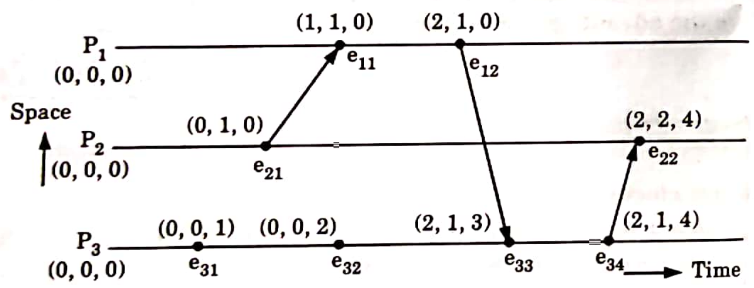


Fig. 1.14.1.

Advantage of vector clock over Lamport's clock :

1. Events '*a*' and '*b*' are causally related if $t^a < t^b$ or $t^b < t^a$. Otherwise, the events are concurrent.
2. In the system of vector clocks,

$$a \to b \text{ iff } t^a < t^b.$$

Thus, the system of vector clocks allows us to order events and decide whether two events are causally related or not by simply looking at the timestamps of the event.

3. In Lamport clock, it is not possible because $t^a < t^b$, does not always implies that $a \to b$.

PART-5

Concept in Message System : Causal Order, Total Order, Total Causal Order, Techniques for Message Ordering, Causal Ordering of Messages, Global State and Termination Detection.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 1.15. **What do you mean by casual ordering of messages ? If process *P* sends two message m_1 and m_2 to another process *Q*, what problems may arise if the two messages are not received by recipient *Q*, in the order they were sent by process *P*. Develop an algorithm which guarantees the casual ordering of message in distributed system.**

AKTU 2015-16, Marks 10

OR

Discuss causal ordering of messages. Give one algorithm which can order the messages according to causal dependencies.

AKTU 2016-17, Marks 10

Answer

Casual ordering of message : The casual ordering of message deals with the concept of maintaining same casual relationship that holds among "message send" event with corresponding "message receive" event.

Problem :

If the two messages m_1 and m_2 are not received by recipient Q in the order they were sent by process P, this means message delivery will not be causal.

Algorithm :

Schiper-Eggli-Sandoz algorithm :

Instead of maintaining a vector clock based on the number of messages sent to each processes, the vector clock for this algorithm can be incremented at any rate and has no additional meaning related to the number of messages spent to the processes.

Sending a message :

1. All messages are timestamped and sent out with a list of all timestamps of messages sent to other processes.

2. Locally store the timestamp of the sent message.

Receiving a message :

1. A message cannot be delivered if there is a predate message in the list of timestamps.
2. Otherwise, a message can be delivered, performing the following steps :
   a. Add the timestamp of delivered message in the list :
      i. Add knowledge of messages destined for other processes to our list of processes.
      ii. If the new list has a timestamp greater than one we already had stored, update our timestamp to match.
   b. Update the local logical clock.
   c. Check all the local buffered messages to see if they can now be delivered.

Que 1.16. **Explain the algorithm for casual ordering of message in distributed system.**

Answer

Algorithm for casual ordering of message in distributed system:

a. **Birman-Schiper-Stephenson algorithm :**

There are three basic principles to this algorithm :

1. All messages are time stamped by the sending process.
2. A message cannot be delivered until:
   i. All the messages before this one have been delivered locally.
   ii. All the other messages that have been sent out from the original process have been accounted as delivered at the receiving process.
3. When a message is delivered, the clock is updated.

This algorithm requires that the processes communicate through broadcast messages which ensure that only one message could be received at any one time.

b. **Schiper-Eggli-Sandoz algorithm :** Refer Q. 1.15, Page 1–16B, Unit -1.

Que 1.17. **Write short note on global state.**

Answer

1. The global state of a distributed computation is the set of local states of all individual processes involved in the computation and the state of the communication channels.
2. The global state of the system is a collection of the local states (LS) of a processing system.

$$GS = \{LS_1, LS_2, LS_3 \ldots\ldots LS_N\}$$

where N is number of sites in the system.

Consistent global state :

1. A global state GS is a consistent global state iff it satisfies the following two conditions :
   a. Every message m_{ij} that is recorded as sent in the local state of a process P_i must be captured in the state of the channel C_{ij} or in the collected local state of the receiver process P_j.
   b. If a message m_{ij} is not recorded as sent in the local state of process P_i, then it must neither be present in the state of the channel C_{ij} nor in the collected local state of the receiver process P_j.
2. Thus, in a consistent global state, for every received message a corresponding send event is recorded in the global state.
3. In an inconsistent global state, there is at least one message whose receive event is recorded but its send event is not recorded in the global state.
4. In Fig. 1.17.1, the global state $\{LS_{12}, LS_{23}, LS_{33}\}$ and $\{LS_{11}, LS_{22}, LS_{32}\}$ correspond to consistent and inconsistent global states, respectively.

Transitless global state : A global state is transitless if and only if

$\forall i, \forall j : 1 \le i, j \le n ::$ transit $(LS_i, LS_j) = \phi$. Thus, all communication channels are empty in a transitless global state.

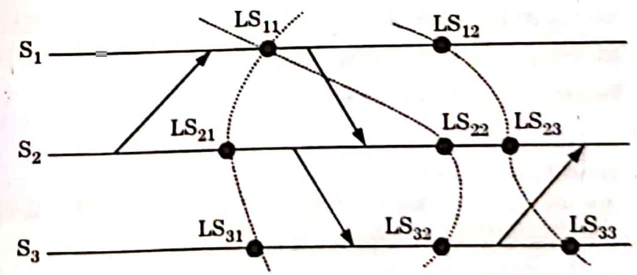


Fig. 1.17.1. Global states in a distributed computation.

Strongly consistent global state :

1. A global state is strongly consistent if it is consistent and transitless.
2. In a strongly consistent state, not only the send events of all the recorded received events are recorded, but the receive events of all the recorded send events are also recorded.
3. Thus, a strongly consistent state corresponds to a consistent global state in which all channels are empty.
4. In Fig. 1.17.1, the global state $\{LS_{11}, LS_{21}, LS_{31}\}$ is a strongly consistent global state.

Que 1.18. **Give the Chandy-Lamport's global state recording algorithm.**

AKTU 2016-17, Marks 05

Answer

Chandy-Lamport global state recording algorithm :

1. The Chandy-Lamport algorithm uses a control message, called a marker whose role in a FIFO system is to separate messages in the channels.
2. After a site has recorded its local state, it sends a marker, along all of its outgoing channels before sending out any more messages.
3. A marker separates the messages in the channel into those which are included in the local state and which are not to be recorded in the local state.
4. A process must record its local state before it receives a marker on any of its incoming channels.

Chandy-Lamport algorithm :

1. **Marker receiving rule for process *j* :**

   On receiving a marker along channel C:

   If (*j* has not recorded its state) then

   Record its process state

   Record the state of C as the empty set

   Follow the "marker sending rule"

   else

   Record the state of C as the set of messages received along C after *j*'s state was recorded and before *j* received the marker along C.

2. **Marker sending rule for process *i* :**

   a. Process *i* records its state.

   b. For each outgoing channel C on which a marker has not been sent, *i* sends a marker along C before *i* sends further messages along C.

Que 1.19. **What is termination detection in distributed system ? Explain any algorithm for termination detection.**

AKTU 2017-18, Marks 10

Answer

1. The termination detection problem involves detecting whether an ongoing distributed computation has finished all its activities.
2. The termination detection problem arises when a distributed computation terminates implicitly, that is, once the computation finishes all its activities, no single process knows about the termination. Therefore a separate algorithm has to be run to detect termination of the computation.

Notations used in algorithm :

1. **B(DW) :** Computation message sent as a part of the computation and DW is the weight assigned to it.
2. **C(DW) :** Control message sent from the processes to the controlling agent and DW is the weight assigned to it.

Huang's termination detection algorithm :

Rule 1 : The controlling agent or an active process having weight W may send a computation message to a process P by doing :

Derive W_1 and W_2 such that

$$W_1 + W_2 = W,\ W_1 > 0,\ W_2 > 0\,;$$

$$W := W_1\,;$$

send $B\,(W_2)$ to P ;

Rule 2 : On receipt of B(DW), a process P having weight W does :

$W := W + (DW)$

If P is idle, P becomes active;

Rule 3 : An active process having weight W may become idle at any time by doing :

send $C(W)$ to (controlling agent)

$W := 0$;

(Process becomes idle);

Rule 4 : On receiving C(DW), the controlling agent having weight W takes the following actions :

$W := W + DW$;

If $W = 1$, conclude that the computation has terminated.

Strongly consistent global state :

1. A global state is strongly consistent if it is consistent and transitless.
2. In a strongly consistent state, not only the send events of all the recorded received events are recorded, but the receive events of all the recorded send events are also recorded.
3. Thus, a strongly consistent state corresponds to a consistent global state in which all channels are empty.
4. In Fig. 1.17.1, the global state $\{LS_{11}, LS_{21}, LS_{31}\}$ is a strongly consistent global state.

Que 1.18. **Give the Chandy-Lamport's global state recording algorithm.**

AKTU 2016-17, Marks 05

Answer

Chandy-Lamport global state recording algorithm :

1. The Chandy-Lamport algorithm uses a control message, called a marker whose role in a FIFO system is to separate messages in the channels.
2. After a site has recorded its local state, it sends a marker, along all of its outgoing channels before sending out any more messages.
3. A marker separates the messages in the channel into those which are included in the local state and which are not to be recorded in the local state.
4. A process must record its local state before it receives a marker on any of its incoming channels.

Chandy-Lamport algorithm :

1. **Marker receiving rule for process *j* :**

 On receiving a marker along channel C:

 If (*j* has not recorded its state) then

 Record its process state

 Record the state of C as the empty set

 Follow the "marker sending rule"

 else

 Record the state of C as the set of messages received along C after *j*'s state was recorded and before *j* received the marker along C.

2. **Marker sending rule for process *i* :**

 a. Process *i* records its state.

 b. For each outgoing channel C on which a marker has not been sent, *i* sends a marker along C before *i* sends further messages along C.

Que 1.19. **What is termination detection in distributed system ? Explain any algorithm for termination detection.**

AKTU 2017-18, Marks 10

Answer

1. The termination detection problem involves detecting whether an ongoing distributed computation has finished all its activities.
2. The termination detection problem arises when a distributed computation terminates implicitly, that is, once the computation finishes all its activities, no single process knows about the termination. Therefore a separate algorithm has to be run to detect termination of the computation.

Notations used in algorithm :

1. **B(DW) :** Computation message sent as a part of the computation and DW is the weight assigned to it.
2. **C(DW) :** Control message sent from the processes to the controlling agent and DW is the weight assigned to it.

Huang's termination detection algorithm :

Rule 1 : The controlling agent or an active process having weight W may send a computation message to a process P by doing :

Derive W_1 and W_2 such that

$$W_1 + W_2 = W, W_1 > 0, W_2 > 0;$$

$$W := W_1;$$

send $B(W_2)$ to P;

Rule 2 : On receipt of B(DW), a process P having weight W does :

$W := W + DW$

If P is idle, P becomes active;

Rule 3 : An active process having weight W may become idle at any time by doing :

send $C(W)$ to controlling agent

$W := 0$;

(Process becomes idle);

Rule 4 : On receiving C(DW), the controlling agent having weight W takes the following actions :

$W := W + DW$;

If $W = 1$, conclude that the computation has terminated.

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. What is a distributed system ? Describe the main characteristics of distributed system. Give two example of distributed system.

Ans. Refer Q. 1.1.

Q. 2. How the distributed computing system is better than parallel processing system ? Explain.

Ans. Refer Q. 1.3

Q. 3. What is distributed transparency ? Explain the different types of distributed transparencies.

Ans. Refer Q. 1.4.

Q. 4. Discuss the major issue in designing a distributed system.

Ans. Refer Q. 1.6.

Q. 5. Why is scalability an important feature in the design of distributed system ? Discuss some of the guiding principles for designing a scalable distributed system.

Ans. Refer Q. 1.7.

Q. 6. What are distributed systems ? What are significant advantages, applications and limitations of distributed systems ? Explain with examples, what could be the impact of absence of global clock & shared memory.

Ans. Refer Q. 1.11.

Q. 7. Discuss the limitations of Lamport's logical clock with suitable example.

Ans. Refer Q. 1.13.

Q. 8. What are vector clocks ? Explain with the help of implementation rule of vector clocks, how they are implemented ? Give the advantages of vector clock over Lamport clock.

Ans. Refer Q. 1.14.

Q. 9. What do you mean by casual ordering of messages ? If process *P* sends two message m_1 and m_2 to another process *Q*, what problems may arise if the two messages are not received by recipient *Q*, in the order they were sent by process *P*. Develop an algorithm which guarantees the casual ordering of message in distributed system.

Ans. Refer Q. 1.15.

Q. 10. Give the Chandy-Lamport's global state recording algorithm.

Ans. Refer Q. 1.18.

Q. 11. What is termination detection in distributed system ? Explain any algorithm for termination detection.

Ans. Refer Q. 1.19.

☺☺☺

2 UNIT

Distributed Mutual Exclusion

CONTENTS

PART-1

Distributed Mutual Exclusion : Classification of Distributed Mutual Exclusion, Requirement of Mutual Exclusion Theorem.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 2.1. **What do you mean by mutual exclusion in distributed system ? What are requirements of a good mutual exclusion algorithm ?**

AKTU 2014-15, Marks 05

OR

State the classification of distributed mutual exclusion. What is requirement of mutual exclusion theorem ?

AKTU 2018-19, Marks 10

Answer

Mutual exclusion :

1. Mutual exclusion is a problem that arises if the process relies on a common resource that can be used only by one process at a time.
2. Concurrent access to shared resources is prevented.
3. Mutual exclusion algorithm guarantees that only one request accesses the critical section (CS) at a time.
4. There are two classes of distributed mutual exclusion algorithm :
   a. Non-token based algorithm
   b. Token based algorithm

Requirements of good mutual exclusion algorithm :

1. **Freedom from deadlocks :** Two or more sites should not endlessly wait for messages that will never arrive.
2. **Freedom from starvation :** A site should not be forced to wait indefinitely to execute CS *i.e.*, every requesting site should get an opportunity to execute CS in a finite time.
3. **Fairness :** Fairness dictates that requests must be executed in the order in which they arrive in the system.
4. **Fault tolerance :** A mutual exclusion algorithm is fault-tolerant if in the wake of a failure, it can recognize itself so that it continues to function without any (prolonged) disruptions.

Que 2.2. **How distributed mutual exclusion is different from mutual exclusion in single-computer system ?**

Answer

Difference :

S. No.	Mutual exclusion in distributed system	Mutual exclusion in single-computer
1.	Shared memory does not exist.	Shared memory exists.
2.	Both shared resources and the users may be distributed.	Both shared resources and the users are present in shared memory.
3.	Mutual exclusion problem is solved by using message passing approach.	Mutual exclusion problem is solved by using shared variables approach *i.e.*, semaphores.

Que 2.3. **What is mutual exclusion ? Describe the requirements of mutual exclusion in distributed system. Is mutual exclusion problem more complex in distributed system than single computer system ? Justify your answer.**

AKTU 2017-18, Marks 10

Answer

Mutual exclusion and its requirements : Refer Q. 2.1, Page 2–2B, Unit-2.

Yes, the problem of mutual exclusion becomes more complex in distributed system as compared to single computer systems because of absence of both shared memory and a common physical clock and because of unpredictable message delays. So, considering these factors, it is virtually impossible for a site to have correct and complete knowledge of the state at the system.

PART-2

Token Based and Non-Token Based Algorithm, Performance Metric for Distributed Mutual Exclusion Algorithm.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 2.4. **What is token based algorithm and non-token based algorithm in distributed system ? Explain with example.**

AKTU 2018-19, Marks 10

Answer

Token based algorithm :

1. In the token based algorithm, a unique token is shared among all sites. If sites possess the token then it is allowed to enter its CS (Critical Section).
2. Token based algorithms use sequence numbers instead of timestamps.
3. Every request for the token contains a sequence number and the sequence number of sites advances independently. A site increment its sequence number counter every time when it makes a request for the token.

Example :

Suzuki-Kasami algorithm :

In the Suzuki-Kasami's algorithm, if a site attempting to enter the CS but does not have the token, it broadcasts a request message for the token to all other sites.

Non-token based algorithm :

1. In non-token based mutual exclusion algorithms, a site communicates with a set of other sites to arbitrate who should execute the CS next.
2. Non-token based mutual exclusion algorithms use timestamps to order request for the CS and to resolve conflicts between simultaneous requests for the CS.
3. Each request for the CS gets a timestamp and small timestamp requests have priority over larger timestamp requests. Each process freely and equally competes for the right to use the shared resource; requests are arbitrated by a central control site or by distributed agreement.

Example :

Lamport's algorithm :

In Lamport's algorithm, $\forall\, i : 1 \le i \le N :: R_i = \{S_1, S_2, ...S_N\}$. Every site S_i keeps a queue, request-queue$_i$, which contains mutual exclusion requests ordered by their timestamps. This algorithm requires message to be delivered in the FIFO order between every pair of sites.

Que 2.5. **Differentiate between token and non-token based algorithms.**

AKTU 2014-15, 2016-17; Marks 05

Answer

S. No.	Token based algorithm	Non-token based algorithm
1.	In token based algorithm, token is shared among all the sites (or nodes).	In non token based algorithm, central site will communicates with all other sites.
2.	Token contains sequence number of the sites in order to request for critical section.	It uses timestamp value in order to request for critical section.
3.	A site having token can only enter the critical section.	A site with smaller timestamp value can only enter the critical section.
4.	Token based algorithms are : a. Lamport's algorithm b. Rickart Agarwala algorithm c. Maekowa's algorithm	Non-token based algorithm are : a. Suzuki-Kasami's broadcast algorithm b. Raymond tree based algorithm c. Singhal's Heuristic algorithm

Que 2.6. **Explain any one token based mutual exclusion algorithm with its performance.**

OR

Explain Lamport's algorithm for mutual exclusion.

Answer

Lamport's algorithm :

1. In Lamport's algorithm, $\forall\, i : 1 \le i \le N :: R_i = \{S_1, S_2, ...S_N\}$. Every site S_i keeps a queue, request-queue$_i$, which contains mutual exclusion requests ordered by their timestamps.
2. This algorithm requires message to be delivered in the FIFO order between every pair of sites.

Algorithm :

1. **Requesting the critical section :**
   a. When a site S_i wants to enter the critical section (CS), it sends a REQUEST (ts_i, i) message to all the sites in its request set R_i and places the request on request-queue$_i$. Where (ts_i, i) is the timestamp of the request.
   b. When a site S_j receives the REQUEST (ts_i, i) message from site S_i, it returns a timestamped REPLY message to S_i and places site S_i's request on the request-queue$_j$.

2. **Executing the critical section :** Site S_i enters the CS when the following two conditions hold :
   a. S_i has received a message with timestamp larger than (ts_i, i) from all other sites.
   b. S_i's request is at the top of request-queue$_i$.
3. **Releasing the critical section :**
   a. Site S_i, upon exiting the CS, removes its request from the top of its request queue and sends a timestamped RELEASE message to all the sites in its request set.
   b. When a site S_j receives a RELEASE message from site S_i, it removes S_i's request from its request queue.
   c. When a site removes a request from its request queue, its own request may come at the top of the queue, enabling it to enter the CS. The algorithm executes CS requests in the increasing order of timestamps.

Performance : Lamport's algorithm requires $3(N-1)$ messages per CS invocation :

1. $(N-1)$ REQUEST, $(N-1)$ REPLY, and $(N-1)$ RELEASE messages. Synchronization delay in the algorithm is T.
2. Lamport's algorithm can be optimized to require between $3(N-1)$ and $2(N-1)$ messages per CS execution by suppressing REPLY messages in certain situations.

Que 2.7. **Explain Suzuki-Kasami algorithm.**

Answer

Suzuki-Kasami algorithm :

1. In the Suzuki-Kasami's algorithm, if a site attempting to enter the CS but does not have the token, it broadcasts a request message for the token to all other sites.
2. The main design issues in this algorithm are :
   a. It distinguishws between outdated request messages and current request messages.
   b. It determines which site has an outdated request for the critical section.

Algorithm :

1. **Requesting the critical section :**
   a. If the requesting site S_i does not have the token, then it increments its sequence number, $RN_i[i]$, and sends a REQUEST (i, sn) message to all other sites. (sn is the updated value of $RN_i[i]$).

b. When a site S_j receives this message, it sets $RN_j[i]$ to max $(RN_j[i], sn)$. If S_j has the idle token, then it sends the token to S_i if $RN_j[i] = LN[i] + 1$. ($LN[i]$ is the sequence number of the request that site S_i executed most recently).

2. **Executing the critical section :**
   a. Site S_i executes the CS when it has received the token.

3. **Releasing the critical section :** After finishing the execution of the CS, site S_i takes the following actions :
   a. It sets $LN[i]$ element of the token array equal to $RN_i[i]$.
   b. For every site S_j whose ID is not in the token queue, it appends its ID to the token queue if $RN_i[j] = LN[j] + 1$.
   c. If token queue is non-empty after the above update, then it deletes the top site ID from the queue and sends the token to the site indicated by ID.

Thus, after having executed its CS, a site gives priority to other sites with outstanding requests for the CS (over its pending requests for the CS).

Que 2.8. **Explain the Ricart-Agrawala algorithm for mutual exclusion. Mention the performance of this algorithm.**

AKTU 2014-15, Marks 05

Answer

The Ricart-Agrawala algorithm is an optimization of Lamport's algorithm that dispenses with RELEASE messages by merging them with REPLY messages. In this algorithm,

$$\forall i : 1 \leq i \leq N :: R_i = \{S_1, S_2, ...S_N\}.$$

Algorithm :

1. **Requesting the critical section :**
   a. When a site S_i wants to enter the CS, it sends a timestamped REQUEST message to all the sites in its request set.
   b. When site S_j receives a REQUEST message from site S_i, it sends a REPLY message to site S_i, if site S_j is neither requesting nor executing the CS or if site S_j is requesting and S_i's request's timestamp is smaller than site S_j's own request's timestamp. The request is deferred otherwise.

2. **Executing the critical section :**
   a. Site S_i enters the CS after it has received REPLY messages from all the sites in its request set.

3. **Releasing the critical section :**
   a. When site S_i exits the CS, it sends REPLY message to all the deferred requests.

A site's REPLY messages are blocked only by sites that are requesting the CS with higher priority (*i.e.* a smaller timestamp). Thus, when a site sends out REPLY messages to all the deferred requests, the site with the next highest priority request receives the last needed REPLY message and enters the CS. The execution of CS requests in this algorithm is always in the order of their timestamps.

Performance : The Ricart-Agrawala algorithm requires $2(N - 1)$ messages per CS execution : $(N - 1)$ REQUEST and $(N - 1)$ REPLY messages. Synchronization delay in the algorithm is T.

Que 2.9. **How the performance of mutual exclusion algorithms is measured ?**

OR

Discuss the performance metric for distributed mutual exclusion algorithms.

AKTU 2016-17, Marks 7.5

OR

How distributed mutual exclusion is different of mutual exclusion in single computer system ? How the performance of mutual exclusion algorithm is measured ?

AKTU 2018-19, Marks 10

Answer

Difference : Refer Q. 2.2, Page 2–3B, Unit-2.

Performance of mutual exclusion algorithm :

Performance of mutual exclusion algorithm is measured by the following four metrics :

1. **Response Time (RT) :** It is time between R and CS *i.e.*, time between request message is sent out and completion of critical section.

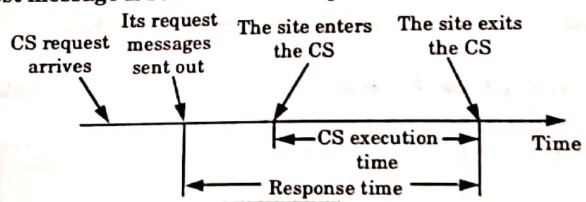


Fig. 2.9.1.

2. **Synchronization delay (sd) :** It is time between two consecutive CS, that is, time between end of one CS and beginning of another CS. In this period, messages are exchanged to arrive at mutual exclusion decision.

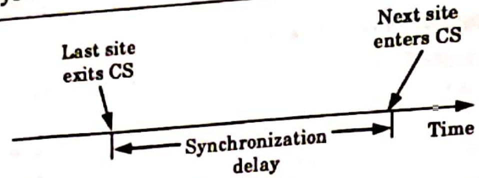


Fig. 2.9.2.

3. **Number of message per CS :** As number of message exchange reduc the performance will improve.
4. **System throughput :** It is the rate at which the system executes the CS. If *sd* is the synchronization delay and *E* is the average critic section time then the throughput is given by the following equation:

$$\text{System throughput} = \frac{1}{(sd + E)}$$

Que 2.10. **How distributed mutual exclusion is different fro mutual exclusion in single-computer system ? Classify mutu exclusion algorithms. How the performance of mutual exclusio algorithms is measured ? Compare the performance of token an non-token based algorithms. How the Ricart- Agrawala algorith optimize the Lamport's algorithm.**

AKTU 2015-16, Marks 1

Answer

Difference : Refer Q. 2.2, Page 2–3B, Unit-2.

Classification of mutual exclusion algorithm :

1. **Token based algorithm :** In the token based algorithm, a uniq token is shared among all sites. If sites possess the token then i allowed to enter its CS (critical section).
2. **Non-token based algorithm :** In non-token based algorithms, a s communicates with a set of other sites to arbitrate who should exec the CS next.

Performance of mutual exclusion algorithm : Refer Q. 2.9, Page 2- Unit-2.

Comparison of performance of token and non-token bas algorithms : (*ll* = light load, *hl* = heavy load)

Non-token	Response time (*ll*)	Synchronization delay	Message (*ll*)	Messa (*hl*)
Lamport	$2T + E$	T	$3(N - 1)$	$3(N - 1$
Ricart-Agrawal	$2T + E$	T	$2(N - 1)$	$2(N - 1$
Maekawa	$2T + E$	$2T$	$3\sqrt{N}$	$5\sqrt{N}$

Token	Response time (*ll*)	Synchronization delay	Message (*ll*)	Message (*hl*)
Suzuki-Kasami	$2T + E$	T	N	N
Singhal's Heuristic	$2T + E$	T	$N/2$	N
Raymond	$T (\log N) + E$	$T \log (N)/2$	$\log (N)$	4

Ricart-Agrawala algorithm : Refer Q. 2.8, Page 2–7B, Unit-2.

PART-3

Distributed Deadlock Detection : System Model Resource vs Communication Deadlocks, Deadlock Prevention Avoidance, Detection & Resolution.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 2.11. **What is deadlock ? What are necessary conditions for the occurrence of deadlock in distributed system ?**

Answer

Deadlock is defined as the permanent blocking of the process *i.e.*, a set of process is waiting for an event that is held by other process.

Necessary conditions for deadlock :

1. **Mutual exclusion :** A resource can be held by at most one process.
2. **Hold and wait :** Processes that already hold resources can wait for other resources.
3. **No preemption :** A resource, once granted, cannot be taken away from a process till its complete execution.
4. **Circular wait :** Two or more processes are waiting for resources used by one of the other processes. We can represent resource allocation as a graph where : $P \leftarrow R$ means a resource R is currently held by a process P. Deadlock exists when a resource allocation graph has a cycle.

Que 2.12. **What is distributed deadlock ? Explain various deadlock handling strategies.**

Answer

1. Deadlock is a situation in which a set of processes are blocked because each process is holding a resources and waiting for another resources acquired by some other process.
2. The detection of deadlocks in a distributed DBMS is more complicated, because it involves several different sites.
3. Thus, in a distributed DBMS it is necessary to draw a global wait-for graph (GWFG) for the entire system to detect a deadlock situation.

Deadlock handling strategies in distributed system :

1. **Deadlock prevention :**
   a. Deadlock prevention is achieved by having a process collect all the needed resources at once before it beings executing or by preempting a process that holds the needed resource.
   b. Now, mutual exclusion, hold-and-wait, no pre-emption and circular-wait are the four necessary conditions for a deadlock to occur. If one of these conditions is never satisfied then deadlock can be prevented.
   c. Deadlock prevention methods are :
      i. **Collective requests :** These methods deny the hold and wait condition by ensuring that whenever a process requests a resource it does not hold any other resource.
      ii. **Ordered requests :** In this method circular-wait is denied such that each resource type is assigned a unique global number to impose total ordering of all resource types.
      iii. **Preemption :** A preemptable resource is one whose state can be easily saved and restored later. Deadlocks can be prevented using resource allocation policies to deny no-preemption condition.
   d. Deadlock prevention is highly incompetent and unrealistic in distributed system.
2. **Deadlock avoidance :**
   a. For deadlock avoidance in distributed system, a resource is assigned to a process if the state of global system is safe.
   b. State of global system includes all processes and resources in distributed system.
   c. Deadlock avoidance algorithm can be done in the following steps:
      i. When a process requests for a resource, if the resource is available for allocation it is not immediately allocated to the process. Rather, the system assumes that the request is granted.

      ii. Using advance knowledge of resource usage of processes and the assumption made in step (i), the system performs some analysis to decide whether granting the process request is safe or unsafe.
      iii. The resource is allocated to the process only if it is safe to do so, otherwise the request is deferred.
3. **Deadlock detection :**
   a. In this approach for deadlock detection, the system does not make any attempt to prevent deadlock but allows processes to request resources and wait for each other in uncontrolled manner.
   b. Deadlock detection requires status of the process and resources interaction for availability of cyclic wait.
   c. Deadlock detection algorithms are easily implemented by maintaining Wait-for-graph (WFG) and searching for cycles.

Que 2.13. **Distinguish between resource deadlock and communication deadlock.**

AKTU 2014-15, Marks 05

Answer

S. No.	Resource deadlock	Communication deadlock
1.	The dependence of one transaction on actions of other transactions is not directly known.	A process can know the identity of those processes on the action of which it depends.
2.	A process cannot proceed with its execution until it receives all the resources for which it is waiting.	A process cannot proceed with its execution until it can communicate with atleast one of the processes for which it is waiting.

PART-4

Centralized Deadlock Detection, Distributed Deadlock Detection, Path-Pushing Algorithm, Edge Chasing Algorithm.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 2.14. **What are the deadlock handling strategies in distributed file system ? What is control organization for distributed deadlock detection ? Discuss an algorithm which can remove phantom deadlock.**

AKTU 2016-17, Marks 05

OR

Explain edge chasing algorithm.

Answer

Deadlock handling strategies : Refer Q. 2.12, Page 2–10B, Unit-2.

Control organizations : Algorithm for detecting distributed deadlock can be handled in following ways :

1. **Centralized control :**
   a. In centralized deadlock detection algorithms, a designated site (control site) has the responsibility of constructing the global WFG and searching it for cycles.
   b. Centralized deadlock detection algorithms are conceptually simple and are easy to implement.
2. **Distributed control :**
   a. In these algorithms, the responsibility for detecting a global deadlock is shared equally among all sites.
   b. The global state graph is spread over many sites and several sites participate in the detection of a global cycle.
3. **Hierarchical control :** In hierarchical deadlock detection algorithms, sites are arranged in a hierarchical fashion, and a site detects deadlocks involving only its descendant sites.

Ede chasing algorithm :

1. Edge chasing algorithm is used for phantom deadlock removal in distributed systems.
2. In this, the global wait-for-graph is not constructed, but each of the servers involved has knowledge about some of its edges.
3. The servers attempt to find cycles by forwarding messages called probes, which follows the edges of the graph throughout the distributed system.
4. A probe message consists of transactions wait-for-relationship representing a path in the global wait-for-graph.
5. Edge chasing has three steps :
   a. **Initiation :** The server initiates to detect deadlock.
   b. **Detection :** Detection consists of receiving probes and deciding whether deadlock has occurred and forward probes.
   c. **Resolution :** When a cycle detected, a transaction in the cycle is aborted to break deadlock.

Que 2.15. **Give the deadlock handling strategies in distributed system ? What are the differences in centralized, distributed and hierarchical control organizations for distributed deadlock detection ?**

AKTU 2014-15, Marks 10

Answer

Deadlock handling strategies : Refer Q. 2.12, Page 2–10B, Unit-2.

Difference :

S. No.	Centralized control	Distributed control	Hierarchical control
1.	A control site has the responsibility to detect global wait for graph.	All sites have the responsibility to detect a global wait for graph.	Descendant site can detect a global wait for graph.
2.	Have single point of failure.	No single point of failure.	No single point of failure.
3.	Easy to implement.	Difficult to implement.	Simple to implement.
4.	Completely centralized and Ho-Ramamoorthy algorithm are used for deadlock detection.	Path pushing and edge chasing algorithm are used for deadlock detection.	Menasce-Muntz and Ho-Ramamoorthy algorithm are used for deadlock detection.

Que 2.16. **Classify the deadlock detection algorithms. Describe the path-pushing deadlock detection algorithm.**

AKTU 2017-18, Marks 10

OR

Discuss Obermarck's path-pushing algorithm.

AKTU 2016-17, Marks 7.5

Answer

Distributed deadlock detection algorithms can be divided into four classes :

a. **Path-pushing algorithm :** In path-pushing algorithms, wait for dependency information of the global WFG (wait for graph) is circulated in the form of paths.
b. **Edge chasing algorithm :** In edge chasing algorithms, special messages called probes are circulated along the edge of the WFG to detect a cycle. When a blocked process receives a probe, it propagates the probe along its outgoing edges in the WFG.
c. **Diffusion computation based algorithm :** Diffusion computation type algorithms make use of echo algorithms to detect deadlocks.

Deadlock detection messages are successively propagated (*i.e.*, "diffused") through the edges of the WFG.

d. **Global state detection based algorithm :** These algorithms detect deadlocks by taking a snapshot of the system and by examining it for the condition of a deadlock. Several sites in distributed system participate in detection of global cycle. Thus global state graph is spread over many sites. The responsibility of detecting deadlock is shared equally among all sites.

Obermarck's path-pushing algorithm :

1. Obermarck's push-pushing algorithm was designed for distributed database system.
2. In path-pushing deadlock detection algorithms, information about the wait-for dependencies is propagated in the form of a path.

Algorithm : Deadlock detection at a site follows the following iterative process :

1. The sites wait for deadlock-related information from other sites in the system.
2. The site combines the received information with its local TWF graph to build an updated TWF graph. It then detects all cycles and breaks local cycles which do not contain the node Ex (External node).
3. For all cycles, 'Ex $\rightarrow T_1 \rightarrow T_2 \rightarrow$ Ex' which contains the node 'Ex' the site transmits them in string form 'Ex T_1, T_2, Ex' to all other sites where a subtransaction of T_2 is waiting to receive a message from the subtransaction of T_2 at this site. The algorithm reduces message traffic by lexically ordering transaction and sending the string 'Ex, T_1, T_2, T_3, Ex' to other sites only if T_1 is higher than T_3 in the lexical ordering. Also for a deadlock, the highest priority transaction detects the deadlock.

Que 2.17. Discuss various centralized deadlock detection algorithms.

Answer

Various centralized deadlock detection algorithms are :

1. **Completely centralized algorithm :**
   a. It is the simplest type of deadlock detection algorithm, wherein a designated site called the control site, maintains the WFG (Wait for graph) of the entire system and checks it for the existence of deadlock cycles.
   b. All sites request and release resources (even local resources) by sending request resource and release resource messages to the control site, respectively.
   c. When the control site receives a request resource or a release resource message, it correspondingly updates its WFG.

   d. The control site checks the WFG for deadlocks whenever a request edge is added to the WFG.
2. **The Ho-Ramamoorthy algorithms :** Ho and Ramamoorthy gave two centralized deadlock detection algorithms called two-phase and one-phase algorithms.
   a. **The two-phase algorithm :**
      i. In the two-phase algorithm, every site maintains a status table that contains the status of all the processes initiated at that side.
      ii. Periodically, a designated site requests the status table from all sites, constructs a WFG from the information received, and searches it for cycles.
      iii. If there is no cycle, then the system is free from deadlocks, otherwise, the designated site again requests status tables from all the sites and again constructs a WFG using only those transactions which are common to both reports.
      iv. If the same cycle is detected again, the system is declared deadlocked.
   b. **The one-phase algorithm :**
      i. The one-phase algorithm requires only one status report from each site; however each site maintains two status tables; a resource status table and a process status table.
      ii. The resource status table at a site keeps track of the transactions that have locked or are waiting for resources stored at that site.
      iii. The process status table at a site keeps track of the resources locked by or waited for by all the transactions at that site.
      iv. Periodically, a designated site requests both the tables from every site, construct a WFG using only those transactions for which the entry in the resource table matches the corresponding entry in process table, and searches the WFG for cycles.
      v. If no cycle is found, then the system is not deadlocked, otherwise a deadlock is detected.

Que 2.18. Explain various hierarchical deadlock detection algorithms.

Answer

In hierarchical algorithms, sites are logically arranged in hierarchical fashion, and a site is responsible for detecting deadlocks involving only its children sites.

Deadlock detection messages are successively propagated (*i.e.*, "diffused") through the edges of the WFG.

d. **Global state detection based algorithm :** These algorithms detect deadlocks by taking a snapshot of the system and by examining it for the condition of a deadlock. Several sites in distributed system participate in detection of global cycle. Thus global state graph is spread over many sites. The responsibility of detecting deadlock is shared equally among all sites.

Obermarck's path-pushing algorithm :

1. Obermarck's push-pushing algorithm was designed for distributed database system.
2. In path-pushing deadlock detection algorithms, information about the wait-for dependencies is propagated in the form of a path.

Algorithm : Deadlock detection at a site follows the following iterative process :

1. The sites wait for deadlock-related information from other sites in the system.
2. The site combines the received information with its local TWF graph to build an updated TWF graph. It then detects all cycles and breaks local cycles which do not contain the node Ex (External node).
3. For all cycles, 'Ex $\rightarrow T_1 \rightarrow T_2 \rightarrow$ Ex' which contains the node 'Ex' the site transmits them in string form 'Ex T_1, T_2, Ex' to all other sites where a subtransaction of T_2 is waiting to receive a message from the subtransaction of T_2 at this site. The algorithm reduces message traffic by lexically ordering transaction and sending the string 'Ex, T_1, T_2, T_3, Ex' to other sites only if T_1 is higher than T_3 in the lexical ordering. Also, for a deadlock, the highest priority transaction detects the deadlock.

Que 2.17. **Discuss various centralized deadlock detection algorithms.**

Answer

Various centralized deadlock detection algorithms are :

1. **Completely centralized algorithm :**
   a. It is the simplest type of deadlock detection algorithm, wherein a designated site called the control site, maintains the WFG (Wait for graph) of the entire system and checks it for the existence of deadlock cycles.
   b. All sites request and release resources (even local resources) by sending request resource and release resource messages to the control site, respectively.
   c. When the control site receives a request resource or a release resource message, it correspondingly updates its WFG.

   d. The control site checks the WFG for deadlocks whenever a request edge is added to the WFG.
2. **The Ho-Ramamoorthy algorithms :** Ho and Ramamoorthy gave two centralized deadlock detection algorithms called two-phase and one-phase algorithms.
   a. **The two-phase algorithm :**
      i. In the two-phase algorithm, every site maintains a status table that contains the status of all the processes initiated at that side.
      ii. Periodically, a designated site requests the status table from all sites, constructs a WFG from the information received, and searches it for cycles.
      iii. If there is no cycle, then the system is free from deadlocks, otherwise, the designated site again requests status tables from all the sites and again constructs a WFG using only those transactions which are common to both reports.
      iv. If the same cycle is detected again, the system is declared deadlocked.
   b. **The one-phase algorithm :**
      i. The one-phase algorithm requires only one status report from each site; however each site maintains two status tables; a resource status table and a process status table.
      ii. The resource status table at a site keeps track of the transactions that have locked or are waiting for resources stored at that site.
      iii. The process status table at a site keeps track of the resources locked by or waited for by all the transactions at that site.
      iv. Periodically, a designated site requests both the tables from every site, construct a WFG using only those transactions for which the entry in the resource table matches the corresponding entry in process table, and searches the WFG for cycles.
      v. If no cycle is found, then the system is not deadlocked, otherwise a deadlock is detected.

Que 2.18. **Explain various hierarchical deadlock detection algorithms.**

Answer

In hierarchical algorithms, sites are logically arranged in hierarchical fashion, and a site is responsible for detecting deadlocks involving only its children sites.

Various hierarchical deadlock detection algorithms :

1. **The Menasce-Muntz Algorithm :**
   a. In this algorithm all the controllers are arranged in tree fashion.
   b. The controllers at the bottom-most level (leaf controllers) manage resources and others (non-leaf controllers) are responsible for deadlock detection.
   c. Whenever a change occurs in a controller's TWF (Transa) graph due to a resources allocation, wait or release, it is propagated to its parent controller.
   d. The parent controller makes changes in its TWF graph, searches for cycles, and propagates the changes upward, if necessary.
   e. A non-leaf controller can receive up-to-date information concerning the TWF graph of its children continuously or periodically.

2. **The Ho-Ramamoorthy algorithm :**
   a. In this algorithm, sites are grouped into several disjoint clusters.
   b. Periodically, a site is chosen as a central control site, which dynamically chooses a control site for each cluster.
   c. The central control site requests from every control site their intercluster transaction status information and wait-for relations.

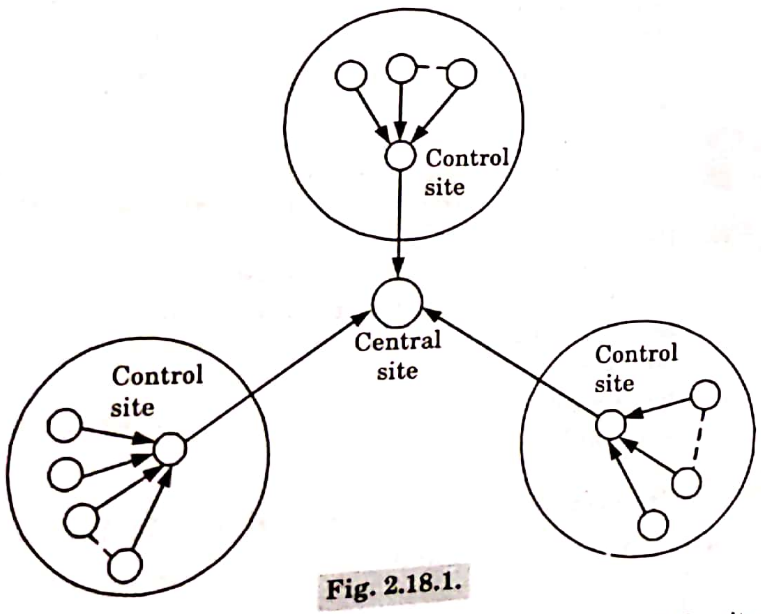


Fig. 2.18.1.

   d. As a result, a control site collects status table from all the sites in its cluster and applies the one-phase deadlock detection algorithm to detect all deadlocks involving only intracluster transactions.

   e. Then, it sends intercluster transaction status information and wait-for relations to the central control site.
   f. The central site split the intercluster information it receives, constructs a system WFG, and searches it for cycles.
   g. Thus, a control site detects all deadlocks located in its cluster, and the central control site detects all intercluster deadlocks.

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. What do you mean by mutual exclusion in distributed system ? What are requirements of a good mutual exclusion algorithm ?

Ans. Refer Q. 2.1.

Q. 2. What is token based algorithm and non-token based algorithm in distributed system ? Explain with example.

Ans. Refer Q. 2.4.

Q. 3. Differentiate between token and non-token based algorithms.

Ans. Refer Q. 2.5.

Q. 4. Explain the Ricart-Agrawala algorithm for mutual exclusion. Mention the performance of this algorithm.

Ans. Refer Q. 2.8.

Q. 5. Discuss the performance metric for distributed mutual exclusion algorithms.

Ans. Refer Q. 2.9.

Q. 6. Classify the deadlock detection algorithms. Describe the path-pushing deadlock detection algorithm.

Ans. Refer Q. 2.16.

Q. 7. What are the deadlock handling strategies in distributed file system ? What is control organization for distributed deadlock detection ? Discuss an algorithm which can remove phantom deadlock.

Ans. Refer Q. 2.14.

☺☺☺

3

UNIT

Agreement Protocols

CONTENTS

PART-1

Agreement Protocol : Introduction, System Models.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.1. **Explain agreement protocol.**

Answer

Agreement protocol :

1. Process of sending and reaching the agreement to all sites is called agreement protocol.
2. In distributed system, the agreement protocols are very much useful for error free communication among various sites.
3. In distributed system, the chances of the faulty processors are more. The faulty processor may lead to wrong message communication, no response for a message etc.
4. Also the presence of faulty processor is not known to the non-faulty processors. So, the non-faulty processors do not restrict the message transfer to the faulty processors.
5. The agreement protocols allow the non-faulty processors to reach a common agreement in the distributed system, whether there are other processors which are faulty or not.
6. The common agreement among the processors is taken through the agreement protocol.

Que 3.2. **What is agreement protocol ? Discuss the general system model where agreement protocols are used.**

Answer

Agreement protocol : Refer Q. 3.1, Page 3–2B, Unit-3.

System model : Following are the system models where agreement protocols are used :

1. If there are n processors in the distributed system, then only m processors out of them may be found as faulty processors.
2. Every processor in the system is free to communicate with other processors in the system due to their logical connections with each other.

3. A receiver processor always knows the identity of the sender processor of message.
4. The communication medium is reliable (*i.e.*, it delivers all messages without introducing any errors) and only processors are prone to failures.

Que 3.3. **Discuss various aspects for recognizing the agreement protocol.**

Answer

Following are various aspects to consider for recognizing the agreements in distributed system :

1. **Synchronous and asynchronous computations :**
   a. In a synchronous computation, processes in the system run in lock step manner, where in each step, a process receives messages, performs a computation and sends messages to other processes.
   b. In synchronous computation, a process knows all the messages which it expects to receive in a round.
   c. A message delay or a slow process can slow down the entire system or computation.
   d. In an asynchronous computation, processes in the system does not proceed in lock step.
   e. A process can send and receive messages and perform computation at any time.
2. **Process failure model :**
   a. The processor failure is the most common system model considered in finding the agreement protocol.
   b. A processor can fail in three models :
      i. **Crash fault :** In a crash fault, a processor stops functioning and never resumes operation.
      ii. **Omission fault :** In an omission fault, a processor "omits" to send messages to some processors.
      iii. **Malicious fault :** In a malicious fault, a processor behaves randomly and arbitrarily.

Authenticated and non-authenticated messages :
   a. In an authenticated message system, a (faulty) processor cannot forge a message or change the contents of a received message.
   b. A processor can verify the authenticity of a received message.
   c. An authenticated message is also called a signed message.
   d. In a non-authenticated message system, a (faulty) processor can forge a message and claim to have received it from another processor or change the contents of a received message before it relays the message to other processors.
   e. A processor is not able to verify the authenticity of a received message.
   f. A non-authenticated message is also called an oral message.
   g. It is easier to reach agreement in an authenticated message system because faulty processors are capable of doing less damage.

4. **Performance aspects :**
   a. The performance of agreement protocols is generally determined by the following three metrics :
      i. **Time :** Time refers to the time taken to reach an agreement under a protocol.
      ii. **Message traffic :** Message traffic is measured by the number of messages exchanged to reach an agreement.
      iii. **Storage overhead :** Storage overhead measures the amount of information that needs to be stored at processors during the execution of a protocol.

PART-2

Classification of Agreement Problem Byzantine Agreement Problem, Consensus Problem, Interactive Consistency Problem, Solution to Byzantine Agreement Problem.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.4. **What are agreement protocols ? Explain Byzantine agreement problem, the consensus problem and interactive consistency problem.** Most important questions. **AKTU 2016-17, Marks 10**

Answer

Agreement protocols : Refer Q. 3.1, Page 3–2B, Unit-3.

Classification of agreement protocol :

1. **The Byzantine agreement problem :**
   a. In the Byzantine agreement problem, an arbitrarily chosen processor, called the source processor, broadcasts its initial value to all other processors.

b. A solution to the Byzantine agreement problem should meet the following objectives :

i. **Agreement :** All non-faulty processors agree on the same value.

ii. **Validity :** If the source processor is non-faulty, then the common agreed upon value by all non-faulty processors should be the initial value of the source.

iii. **Termination :** Each non-faulty processor must eventually decide on a value.

2. **Consensus problem :**

a. In the consensus problem, every processor broadcasts its initial value to all other processors.

b. Initial values of the processors may be different.

c. A protocol for reaching consensus should meet the following conditions :

i. **Agreement :** All non-faulty processors agree on the same single value.

ii. **Validity :** If the initial value of every non-faulty processor is v, then the agreed upon common value by all non-faulty processors must be v.

iii. **Termination :** Each non-faulty processor must eventually decide on a value.

3. **The interactive consistency problem :**

a. In the interactive consistency problem, every processor broadcasts its initial value to all other processors.

b. The initial values of the processors may be different.

c. A protocol for the interactive consistency problem should meet the following conditions :

i. **Agreement :** All non-faulty processors agree on the same value vector $(v_1, v_2, \ldots\ldots v_n)$.

ii. **Validity :** If the i^{th} processor is non-faulty and its initial value is v_i, then the i^{th} value to be agreed on by all non-faulty processors must be v_i.

iii. **Termination :** Each non-faulty processor must eventually decide on different value of vectors.

Que 3.5. **What are agreement protocols ? Explain Byzantine agreement problem, the consensus problem and interactive consistency problem. Describe Lamport-Shostak-Pease algorithm.**

AKTU 2014-15, Marks 10

Answer

Agreement protocol : Refer Q. 3.1, Page 3–2B, Unit-3.

Byzantine agreement problem, the consensus problem and interactive consistency problem : Refer Q. 3.4, Page 3–4B, Unit-3.

Lamport-Shostak-Pease algorithm : Isso Yaad krna hai?

Lamport algorithm, also referred to as Oral Message algorithm $OM(m)$, $m > 0$, solves the Byzantine agreement problem for $3m + 1$ or more processors in the presence of at most m faulty processors.

Let n denote the total number of processors (where, $n \geq 3m + 1$). The algorithm is recursively defined as follows :

Algorithm OM(0) :

1. The source processor sends its value to every processor.
2. Each processor uses the value it receives from the source. If it receives no value, then it uses a default value of 0.

Algorithm OM (m), $m > 0$:

1. The source processor sends its value to every processor.
2. For each i, let v_i be the value processor i receives from the source (if it receives no value, then it uses a default value of 0). Processor i acts as the new source and initiates algorithm $OM(m-1)$ wherein, it sends the value v_i to each of the other $n-2$ processors.
3. For each i and j (where i and j are not equal), let v_j be the value processor i received from processor j in step 2 using Algorithm $OM\ (m-1)$. If it receives no value, then it uses a default value of 0. Processor i uses the value majority $(v_1, v_2 \ldots\ldots v_{n-1})$.
4. The majority function is used to select the majority value out of values received in round of message exchange.
5. The function majority $(v_1, v_2, \ldots\ldots v_{n-1})$ computes majority of values $v_1, v_2, \ldots\ldots v_{n-1}$ if it exists (otherwise returns 0).

Que 3.6. **Describe Lamport-Shostak-Pease algorithm. How does vector clock overcome the disadvantages of Lamport clock ? Explain with an example.**

AKTU 2016-17, Marks 15

Answer

Lamport-Shostak-Pease algorithm : Refer Q. 3.5, Page 3–5B, Unit-3.

Vector clock overcome advantage of Lamport clock : With Lamport clocks, we cannot determine whether two events are casually related by looking at the timestamps, because if $C(A) < C(B)$ does not always mean $A \rightarrow B$ while vector clock allow to compare the timestamps of the events to determine whether they are casually related or not.

For example : Vector timestamp is shown in Fig. 3.6.1. The event e_{12} with vector timestamp (2, 1, 0) is causally ordered before the event e_{34} with the vector timestamp (2, 1, 4), but is concurrent with the event e_{32} having timestamp (0, 0, 2).

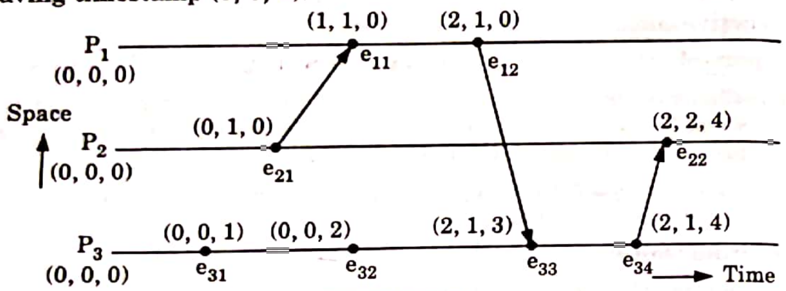


Fig. 3.6.1.

Que 3.7. What do you understand by Byzantine agreement problem ?

AKTU 2018-19, Marks 10

OR

What is Byzantine agreement problem ? Provide the solution to Byzantine agreement problem.

AKTU 2018-19, Marks 10

Answer

1. In Byzantine agreement problem a single value, which is to be agreed on is initializes by an arbitrary processes and all non-faulty processes have to agree on that value.
2. There are n processes, $\pi = \{p_1, p_2, \ldots p_n\}$ with unique names over $N = \{1, \ldots, n\}$ and at most Byzantine participants $t < n$ of the processes can be Byzantine.
3. Each process starts with an input value v from a set of values.
4. The goal of this protocol is to ensure that all non-faulty processes eventually output the same value.
5. The output of a non-faulty process is called the decision value.
6. An algorithm solves the Byzantine agreement if the following conditions hold :
   a. **Agreement :** All non-faulty processes agreed on the same value (*i.e.*, there are no two non-faulty processes that decide different values).
   b. **Validity :** If all non-faulty processes start with the same value v, the decision value of all non-faulty participants is v.
   c. **Termination :** All non-faulty processes decide a value.

7. Reaching agreement in presence of Byzantine processes is expensive as the number of messages grows quadratically with the number of participants n and the number of round (time) grows linearly with the number of Byzantine participants t (with $n > 3t$).

Solution to Byzantine agreement : Solution to Byzantine agreement problem is given by Lamport Shostak-Pease algorithm.

Lamport Shostak-Pease algorithm : Refer Q. 3.5, Page 3–5B, Unit-3.

Que 3.8. Show that a Byzantine agreement cannot be reached among three processors, where one processor is faulty.

OR

Explain treatment of impossible result for the solution of Byzantine agreement problem.

Answer

1. Sometimes, the agreement problem may lead to such a condition which is quite impossible to solve.
2. The situation where the agreement is impossible, called as impossible result.
3. This type of problem cannot be reached to agreement.
4. In a system, the impossible result situation is found with more than two processors.
5. Let us check the situation of impossible result in a system with three processors.
6. Consider a system with three processors, P_0, P_1 and P_2.
7. We assume that there are only two values, 0 and 1, on which processors agree and processor P_0 initiates the initial value.
8. There are two possibilities :

Case I : P_0 is not faulty :

1. Assume P_2 is faulty.
2. Suppose that P_0 broadcasts an initial value of 1 to both P_1 and P_2.
3. Processor P_2 acts maliciously and communicates a value of 0 to processor P_1.
4. Thus, P_1 receives conflicting values from P_0 and P_2.
5. However, since P_0 is non-faulty, processor P_1 must accept 1 as the agreed upon value.

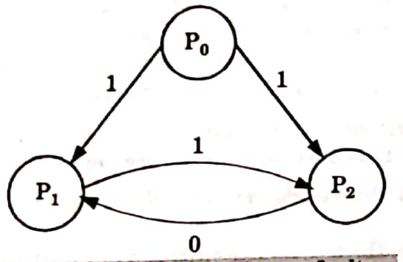


Fig. 3.8.1 Processor P_0 is non-faulty.

Case II : P_0 is faulty :

1. Suppose that processor P_0 sends an initial value of 1 to P_1 and 0 to P_2.
2. Processor P_2 will communicate the value 0 to P_1.
3. As far as P_1 is concerned, this case will look identical to Case I.
4. So any agreement protocol which works for three processors cannot distinguish between the two cases and must force P_1 to accept 1 as the agreed upon value whenever P_1 is faced with such situations.
5. However, in case II, this will work only if P_2 is also made to accept 1 as the agreed upon value.
6. Using a similar argument, we can show that if P_2 receives an initial value of 0 from P_0, then it must take 0 as the agreed upon value, even if P_1 communicates a value of 1.
7. However, if this is followed in case II, P_1 will agree on a value of 1 and P_2 will agree on a value of 0.

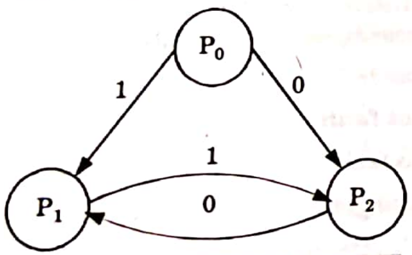


Fig. 3.8.2 Processor P_0 is faulty.

Therefore, no solution exists for the Byzantine agreement problem for three processors, which can work under single processor failure.

Que 3.9. **Describe Byzantine agreement problem, and explain its solution. Show that Byzantine agreement cannot always be reached among four processors if two processors are faulty.**

AKTU 2017-18, Marks 10

Answer

Byzantine agreement problem and its solution : Refer Q. 3.7, Page 3–7B, Unit-3.

Proof :

1. Consider a system with four processors as P_1, P_2, P_3, P_4. Assume that processors are exchanging three values x, y and z to each other, P_1 initiate the initial value and processors P_2 and P_4 are faulty.
2. To initiate the agreement, processor P_1 execute algorithm $OM(1)$ and sends its value 'x' to all other processor as shown in Fig. 3.9.1.

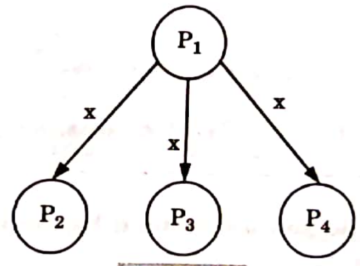


Fig. 3.9.1.

3. After receiving the value x from source processor P_1, processors P_2, P_3 and P_4 execute the algorithm $OM(0)$.
4. Processor P_3 is non-faulty and send value x to processor P_2 and P_4. Faulty processors P_2 and P_4 sends value y to (P_3, P_4) and z to (P_2, P_3) respectively as shown in Fig. 3.9.2.

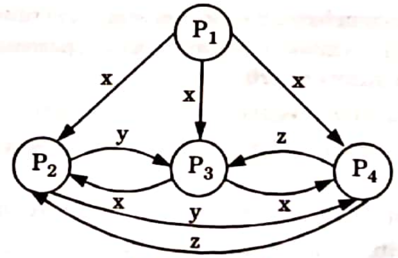


Fig. 3.9.2.

5. After receiving all the messages, processor P_1, P_2, P_3 and P_4 decide on the majority value.

Majority values for Byzantine solution :

Processor	Received majority values	Common majority values
P_2	(x, x, z)	x
P_3	(x, y, z)	0
P_4	(x, x, y)	x

6. According to majority value table, processors does not agree on single common majority value, which violates the condition of Byzantine agreement problem.
7. This proves that Byzantine agreement cannot always reach among four processors if two processors are faulty.

PART-3

Application of Agreement Protocol, Atomic Commit in Distributed Database System.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.10. What do you understand by the fault-tolerant clock synchronization ?

Answer

1. In distributed systems, it is often necessary that sites (or processes) maintain physical clocks that are synchronized with one another.
2. Since physical clocks have a drift problem, they must be periodically resynthesized.
3. Such periodic synchronization becomes extremely difficult if the Byzantine failure is allowed because a faulty process can report different clock values to different processes.
4. Now the assumptions regarding the system are :
   a. All clocks are initially synchronized to approximately the same values.
   b. A non-faulty process's clock runs at approximately correct rate.
   c. A non-faulty process can read the clock value of another non-faulty process with at most a small error ϵ.
5. A clock synchronization algorithm should satisfy the following two conditions :
   a. At any time, the values of the clocks of all non-faulty processes must be approximately equal.
   b. There is a small bound on the amount by which the clock of a non-faulty process is changed during each resynchronization. This condition implies that resynchronization does not cause a clock value to jump arbitrarily far, thereby preventing the clock rate from being too far from the real time.

Que 3.11. Explain the interactive convergence algorithm for clock synchronization.

Answer

The interactive convergence algorithm :

1. It is called interactive convergence algorithm because it causes the convergence of non-faulty clocks.
2. This algorithm assumes that the clocks are initially synchronized and they are resynthesized often enough so that two non-faulty clocks never differ by more than δ.
3. According to the algorithm, each process reads the value of all other processes clocks and sets its clock value to the average of these values.
4. If a clock value differs from its own clock value by more than δ, it replaces that value by its own clock value when taking the average.
5. The processing of the algorithm is very simple.
6. Now, assume the situation where, there are two processes p and q respectively, use C_{pr} and C_{qr} as the clock values of a third process r when computing their averages.
7. If r is non-faulty, then

$$C_{pr} = C_{qr}$$

8. If r is faulty, then

$$| C_{pr} - C_{qr} | \leq 3\delta$$

when p and q compute their averages for the n clock values, they both use identical values for the clocks of $n - m$ non-faulty processes and the difference in the clock values of m faulty processes they use is bounded by 3δ.

9. In this way, the average value computed by p and q differ by at most $(3m/n)\,\delta$.

Since, $n > 3m$

and $(3m/n)\,\delta < \delta$

10. Thus, each resynchronization brings the clocks closer by a factor of $(3m/n)$.
11. This implies that we can keep the clocks synchronized within any desired degree by resynchronizing them often enough using the algorithm.

Que 3.12. Explain interactive consistency algorithm.

Answer

1. In this algorithm we consider two important conditions :

C_1 : Any two processes obtain approximately the same value for a process *P*'s clock (even if *P* is faulty).

This condition is important because any two processes always compute approximate same median if they get approximate same set of clock values for other processes.

C_2 : If *q* is a non-faulty process, then every non-faulty process obtains approximately the correct value for process *q*'s clock.

Thus, if a majority of the processes are non-faulty, the median of all the clock values is either approximately equal to a good clock's value or it lies between the values of two good clocks.

2. The algorithm is as follows :
   a. All the processes in a system execute an algorithm to collect values for clock that satisfy the conditions C_1, C_2.
   b Every process uses the median of collected values to compute its new clock value.

Que 3.13. **Write a short note on atomic commit in distributed database system.**

Answer

1. In the problem of atomic commit, sites of a distributed system must agree whether to commit or abort a transaction.
2. In the first phase of the atomic commit, sites execute their part of a distributed transaction and broadcast their decisions (commit or abort) to all other sites.
3. In the second phase, each site, based on what it received from other sites in the first phase, decides whether to commit or abort its part of the distributed transaction.
4. If every site receives an identical response from all other sites, they will reach the same decision.
5. If some sites behave maliciously, they can send a conflicting response to other sites, causing them to make conflicting decisions.
6. In these situations, we can use algorithms for the Byzantine agreement to insure that all non-faulty processors reach a common decision about a distributed transaction.
7. It works as follows :
   a. In the first phase, after a site has made a decision, it starts the Byzantine agreement.
   b. In the second phase, processors determine a common decision based on the agreed vector of values.

Que 3.14. **What are agreement protocols ? Discuss the general system model where agreement protocols are used. Give the applications of agreement protocols.** **AKTU 2015-16, Marks 10**

Answer

Agreement protocols : Refer Q. 3.1, Page 3–2B, Unit-3.

System model : Refer Q. 3.2, Page 3–2B, Unit-3.

Applications of agreement protocols :

1. **Fault-tolerant clock synchronization :**
   a. Distributed systems require physical clocks to synchronize.
   b. Physical clocks have drift problem.
   c. Agreement protocols may help to reach a common clock value.
2. **Atomic commit in distributed database system (DDBS) :**
   a. DDBS sites must agree whether to commit or abort the transaction.
   b. Agreement protocols may help to reach a consensus.

PART-4

Distributed Resource Management : Issues in Distributed File System, Mechanism for Building Distributed File System.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.15. **Explain typical architecture of Distributed File System (DFS).**

Answer

1. In distributed file system, files can be stored at one machine and the computation can be performed at other machine.
2. When a machine needs to access a file stored on a remote machine, the remote machine performs the necessary file access operations and returns data if a read operation is performed.
3. File server are higher performance machines which are used to store file and performs storage and retrieval operations.
4. Client machines are used for computational purpose and to access the files stored on servers.

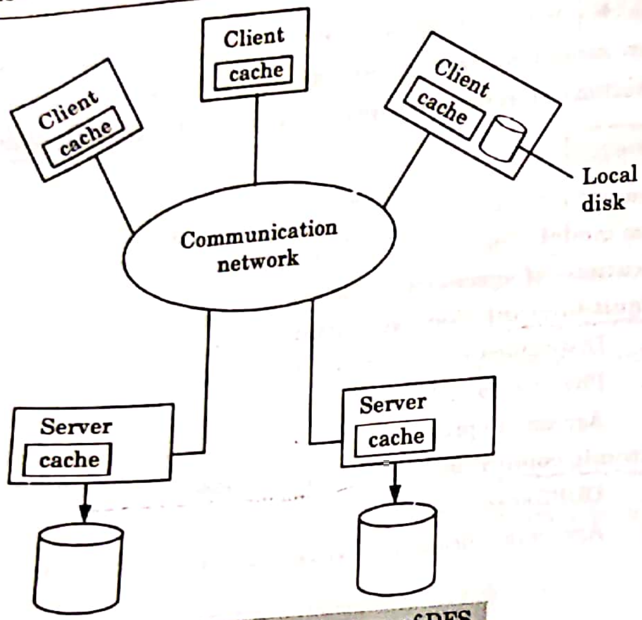


Fig. 3.15.1. Architecture of DFS.

5. The two most important services present in a distributed file system are :

a. **Name server :**

i. A name server is a process that maps names specified by clients to stored objects such as files and directories.

ii. The mapping occurs when a process references a file or directory for the first time.

b. **Cache manager :**

i. A cache manager is a process that implements file caching.

ii. In file caching, a copy of data stored at a remote file server is brought to the client's machine when referenced by the client.

iii. Cache managers can be present at both clients and file servers.

iv. Cache managers at the servers, cache files in the main memory to reduce delays due to disk latency.

v. If multiple clients are allowed to cache a file and modify it, the copies can become inconsistent.

vi. To avoid this inconsistency problem, cache managers at both servers and clients coordinate to perform data storage and retrieval operations.

Que 3.16. Explain the mechanism for distributed file system.

Answer

Mechanism for building distributed file system :

1. **Mounting :**

a. A mount mechanism allows the binding of different filename spaces to form a single hierarchically structured name space.

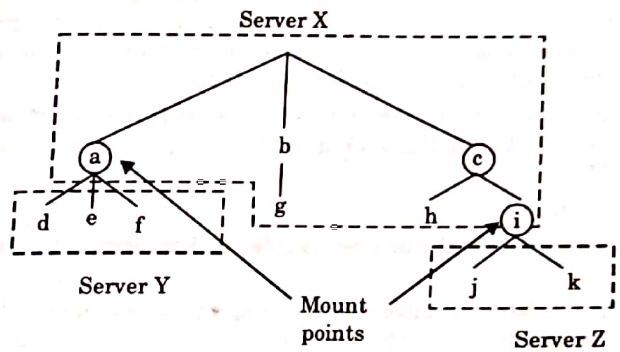


Fig. 3.16.1. Name space hierarchy.

b. A name space (or a collection of files) can be bounded to or mounted at an internal node or a leaf node of a name space tree.

c. A node onto which a name space is mounted is known as mount point. The kernel maintains a structure called the mount table which maps mount points to appropriate storage devices.

2. **Caching :**

a. In caching, a copy of data stored at a remote file server is brought to the client when referenced by the client.

b. On further request, the data is searched locally at the client side only and it reduces the access delay through the network.

c. Data can either be cached in the main memory or on the local disk of the clients or at the servers to reduce disk access latency.

3. **Hints :**

a. Caching is an important mechanism to improve the performance of a file system.

b. Guaranteeing the consistency of the cached items requires elaborate and expensive client/server protocol.

c. An alternative approach to maintaining consistency is to treat cached data as hints.

d. With this scheme, cached data is not expected to be completely consistent, but when it is, it can dramatically improve performance.

e. To prevent the occurrence of negative consequences if a hint is erroneous, the classes of applications that use hints must be restricted to those that can recover after discovering that the cached data is invalid, that is, the data should be self-validating upon use.

4. **Bulk data transfer :**
   a. In this mechanism, multiple consecutive data blocks are transferred from server to client.
   b. This reduces file access overhead by obtaining multiple number of blocks with a single seek, by formatting and transmitting multiple number of large packets in single context switch and by reducing the number of acknowledgement that need to be sent.
   c. Bulk transfer amortizes the cost of the fixed communication protocol overheads and disk seek time over many consecutive blocks of a file.

5. **Encryption :**
   a. Encryption is the process used for data security in the distributed system.
   b. A number of possible threats exist, such as unauthorized release of information, unauthorized modification of information.
   c. Encryption prevents unauthorized release and modification of information.
   d. For performance, encryption/decryption may be performed by special hardware at the client and server.

Que 3.17.

i. **Explain typical architecture of distributed file system. Give the mechanisms for building distributed file system.**

ii. **What is caching ? How is useful in DFS ?**

AKTU 2014-15, Marks 10

Answer

i. **Architecture of distributed file system :** Refer Q. 3.15, Page 3–14B, Unit-3.

Mechanisms for building DFS : Refer Q. 3.16, Page 3–15B, Unit-3.

ii. **Caching :** Refer Q. 3.16, Page 3–15B, Unit-3.

Uses of caching in DFS :
   a. The file system performance can be improved by caching; since accessing remote disks is much slower than accessing local memory or local disks.
   b. Caching reduces the frequency of access to the file servers and the communication network, thereby, improving the scalability of a file system.

   c. Caching in the distributed file system is used to reduce delays in accessing of data.

Que 3.18. **What is cache ? Discuss read operation with cache and write operation with cache.**

AKTU 2017-18, Marks 05

Answer

Cache :
1. A cache is a data storing technique that provides the ability to access data or files at a higher speed.
2. Caches are implemented both in hardware and software.
3. Caching serves as an intermediary component between the primary storage appliance and the recipient hardware or software device to reduce the latency in data access.

Read operation :
1. The sequence of steps in a cache read operation is shown in Fig. 3.18.1.
2. Read operation starts with a lookup operation and has a partial overlap between the lookup and data read operations.
3. If there is a cache hit, then cache returns the value to the processor, or the higher level cache.
4. If there is a cache miss, then we need to cancel the data read operation, and send a request to the lower level cache.
5. The lower level cache will perform the same sequence of accesses, and return the entire cache block.
6. The cache controller can then extract the requested data from the block, and send it to the processor.
7. Simultaneously, the cache controller invokes the insert operation to insert the block into the cache.

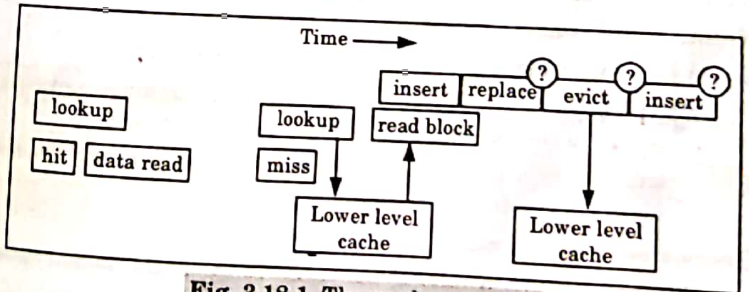


Fig. 3.18.1. The read operation.

Cache write operation (write back cache) :
1. Fig. 3.18.2 shows the sequence of operations for a cache write operation for a write back cache.

2. The sequence of operations is similar to that of a cache read.
3. If there is a cache hit, then we invoke a data write operation, and set the modified bit to 1.
4. Otherwise, we issue a read request for the block to the lower level cache.
5. After the block arrives, most cache controllers store it in a small temporary buffer.
6. In some processors, the cache controller might wait till all the sub-operations complete.
7. After writing into the temporary buffer, cache controller invokes the insert operation for writing the contents of the block.

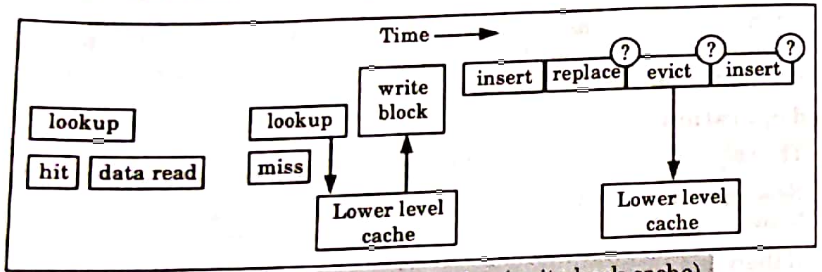


Fig. 3.18.2. The write operation (write back cache).

Que 3.19. **Write and explain various issues that must be addressed in design and implementation of distributed file system.**

AKTU 2017-18, Marks 05

Answer

Following are various issues that must be addressed in the design and implementation of distributed file system :

1. **Naming and name resolution :**
   a. A name in file systems is associated with an object.
   b. Name resolution refers to the process of mapping a name to an object or, in the case of replication, to multiple objects.
   c. A name space is a collection of names which may or may not share an identical resolution mechanism.
2. **Caches on disk or main memory :**
   a. The advantages of having the cache in the main memory are as follows :
      i. Diskless workstations can take advantages of caching as they are cheaper.

      ii. Accessing a cache in main memory is much faster than accessing a cache on local disk.
      iii. The server-cache is in the main memory at the server, hence, a single design for a caching mechanism is both clients and servers.
   b. Caches deal with the memory contention between cache and virtual memory system.
3. **Writing policy :**
   a. The writing policy decides when a modified cache block at a client should be transferred to the server.
   b. The simplest policy is write-through.
   c. In write-through, all writes requested by the applications at clients are also carried out at the servers immediately.
   d. The main advantage of write-through is reliability.
4. **Availability :**
   a. Availability is one of the important issues in the design of distributed file systems.
   b. The failure of servers or the communication network can severely affect the availability of files.
   c. Replication is the primary mechanism used for enhancing the availability of files in distributed file systems.
5. **Scalability :** The issue of scalability deals with the suitability of the design of a system to handle the demands of the growing system.
6. **Semantics :**
   a. The semantics of a file system characterizes the effects of accesses on files.
   b. The basic semantics are easily understood by programmers and are easy to handle.
   c. A read operation will return the data (stored) due to the latest write operation.

Que 3.20. **Explain naming in distributed system. What is flat naming and structured naming ?**

AKTU 2017-18, Marks 05

Answer

Naming :

1. The naming facility enables users and programs to assign character-string names to objects and use the names to refer those objects.
2. The locating facility, which is an integral part of the naming facility, maps an object's name to the object's location in a distributed system.

3. The naming and locating facilities jointly form a naming system that hides the details of how and where an object is actually located in the network.
4. The naming system plays a very important role in achieving the goal of location transparency, facilitating transparent migration and replication of objects, object sharing.

Flat naming :

1. Flat name is a simplest name space where names are character strings.
2. Flat names are fixed size bit strings that can be efficiently handed by machines.
3. Names defined in a flat name space are called primitive or flat names.
4. Flat names do not have any structure.
5. Flat names are suitable for use either for small name spaces having names for only a few objects or for system-oriented name spaces.

Structured naming :

1. Structured names are organized into name spaces.
2. A structured name space is represented as a labeled directed graph with two types of nodes.
3. A leaf node represents a named entity and stores information about the entity.
4. A directory node stores a directory table of (edge label, node ID) pairs.

Que 3.21. **Explain shared memory architecture and distributed memory architecture.**

AKTU 2017-18, Marks 10

Answer

Shared memory architecture : It contains following features :

1. All processors can directly access all memory locations in the system, thus providing a convenient mechanism for processors to communicate.
2. It is convenient in the sense of :
   a. Location transparency.
   b. Abstraction support.
3. Memory usually is centrally placed.
4. Symmetric multiprocessor (SMP) systems use this centralized memory approach, where each processor is connected to a shared bus. This shared bus handles all accesses to main memory and I/O.
5. A schematic view of the centralized shared memory model in an SMP system is given in Fig. 3.21.1. Each processor (denoted by *P*) accesses memory through a shared bus and has a local cache (denoted by $).

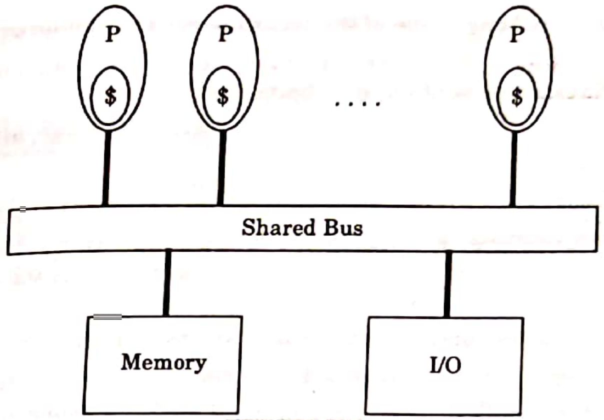


Fig. 3.21.1.

6. It belongs to the MIMD (Multiple Instruction, Multiple Data) computational model.

Distributed memory architecture : Its main features are :

1. All processors in the system are directly connected to own memory and caches. Any processor cannot directly access another processor's memory.
2. Each node has a network interface (NI).
3. All communication and synchronization between processors happens via messages passed through the NI.
4. Since this approach uses messages for communication and synchronization, it is often called message passing architecture.
5. This architecture belongs to the MIMD (Multiple Instruction Stream, Multiple Data Stream) programming model.
6. A schematic view of the distributed memory approach is shown in the Fig. 3.21.2, where each processor has local memory and processors (each denoted by *P*) communicate through an interconnection network. Also, each processor has its own cache (denoted by $).

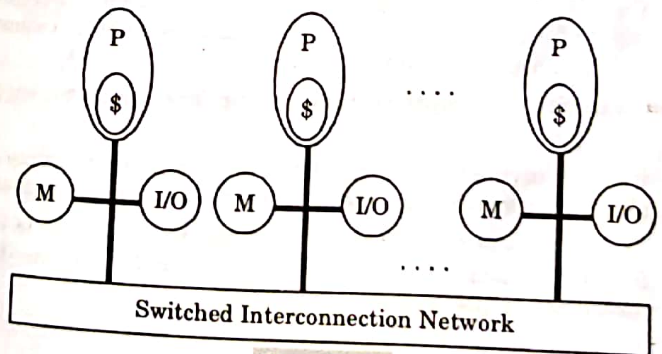


Fig. 3.21.2.

Que 3.22. **Caching is one of the techniques used to improve access to naming data. What are the benefits of caching and what assumptions must hold for it to be useful ?**

AKTU 2015-16, Marks 10

Answer

Benefits of caching :

a. Due to caching, data is cached in the main memory at the servers to reduce disk access latency.

b. File system performance can be improved by caching.

c. It improves the scalability of a file system.

Assumptions : Following assumptions must hold for caching to be useful :

1. **Client-specific assumptions :** The assumptions that are client-specific are :
   a. The client might issue a list of requests instead of individual requests. This is to be known as a request-list. The client suffers insignificant performance penalty in the construction of this list.
   b. The client will receive a bundle that is a collection of responses to some or all of its requests (in a request-list). The client will exert a fixed amount of resources to dissemble the bundle.
   c. The client is prepared to receive responses not in the same order as that of its requests. In addition, the order of requests and responses is not important to the client.
2. **Server-specific assumptions :** The assumptions that are server-specific are :
   a. The server will determine whether or not to use the PC-Bundle mechanism whenever a request-list is received. The server incurs insignificant overhead for this action.
   b. The server will determine which of the responses to a request-list should be bundled. It will exert a fixed amount of resources to determine and assemble a bundle.
3. **Proxy-specific assumptions :** The assumptions that are server-specific are :
   a. Upon receiving the request-list in a PC-bundle, the proxy has the ability to parse and understand all individual object requests.
   b. The proxy will remove from the request-list all objects that are found in its cache. It will form a new PC-bundle's request-list to be forwarded to the server or next proxy.

PART-5

Design Issues in Distributed Shared Memory.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.23. **What is Distributed Shared Memory (DSM) ? Explain with diagram the architecture of distributed shared memory.**

AKTU 2014-15, Marks 05

Answer

1. Distributed Shared Memory (DSM) is a form of memory architecture where the (physically separate) memories can be addressed as one (logically shared) address space.
2. The shared memory model provides a virtual address space shared between all nodes.
3. DSM is primarily a tool for any distributed application in which individual shared data items can be accessed directly.

Architecture :

1. DSM consists of number of nodes or computers each of which is connected to other through high speed communication channel.

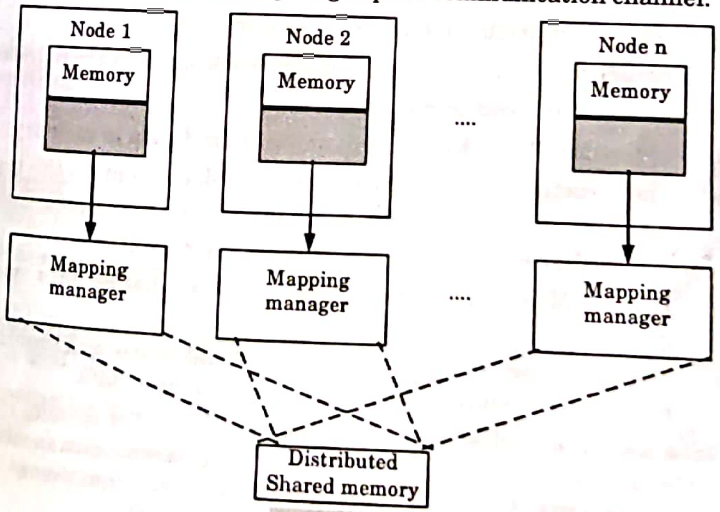


Fig. 3.23.1.

2. Each node consists of one or more Central Processing Units and a single memory unit.
3. Message is passed from one node to another by means of simple message passing technique.
4. Data moves between main memory and secondary memory (within a node) and between main memories of different nodes.
5. The main memory of individual nodes is used to cache pieces of shared memory space using mapping manager.
6. When a process accesses data in the shared address space, the mapping manager maps shared memory address to physical memory.
7. The mapping manager is layer of software implemented either in the operating kernel or as runtime library routine.
8. For mapping operation, the shared memory space is partitioned into blocks.
9. A simple message passing system allows on different node to exchange message with each other.

Que 3.24. **Explain design issues in distributed shared memory.**

OR

Explain the mechanism for building distributed file system also explain the design issues in distributed shared memory.

AKTU 2018-19, Marks 10

Answer

Mechanisms for building distributed file system : Refer Q. 3.16, Page 3–15B, Unit-3.

Design issues in distributed shared memory :

1. **Granularity :** Granularity refers to the block size of a DSM system.
2. **Structure of shared memory space :**
   a. Structure refers to the layout of the shared data in memory.
   b. The structure of the shared-memory is dependent on the type of applications that the DSM system is intended to support.
3. **Memory coherence and access synchronization :**
   a. In a DSM system, concurrent access to shared data may be generated.
   b. Therefore, memory coherence protocol and access synchronization is needed to maintain the consistency of shared data.
4. **Data location and access :** To share data in a DSM system, it must implement some form of data block locating mechanism in order to service network data block fault and to meet the requirement of the memory coherence semantics which are being used.

5. **Replacement strategy :**
   a. If the local memory of a node is full, a cache miss at that node implies not only a fetch of the accessed data block from a remote node but also a replacement.
   b. Therefore, a cache replacement strategy is also necessary in the design of a DSM system.

PART-6

Algorithm for Implementation of Distributed Shared Memory.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 3.25. **Explain typical architecture of distributed file system.**

State the algorithm for implementation of distributed shared memory.

AKTU 2018-19, Marks 10

OR

Give the design issues in distributed shared memory. State the algorithm for implementation of distributed shared memory.

AKTU 2018-19, Marks 10

OR

Explain design issues in distributed shared memory and also write algorithm for implementation of shared memory.

AKTU 2016-17, Marks 10

Answer

Architecture of distributed file system : Refer Q. 3.15, Page 3–14B, Unit-3.

Design issues in distributed shared memory : Refer Q. 3.24, Page 3–25B, Unit-3.

Following four algorithms are used to implement DSM systems :

1. The central-server algorithm

2. The migration algorithm
3. The read-replication algorithm
4. The full-replication algorithm

Migration algorithm :

1. In migration algorithm, data is migrated to the location of the data access request, allowing subsequent accesses to the data to be performed locally.
2. The migration algorithm allows only one node to access the shared data.
3. Data is migrated between servers in a fixed size unit called block to facilitate the management of the data.

Algorithm :

1. Client will check if the block is local or not.
2. If block is not local then determine the location of block.
3. It sends the request to the server to determine the location of block.
4. Server receives the request from the client.
5. Server will send the location of block to the client.
6. Client receives the location of block from the server and access the data block.

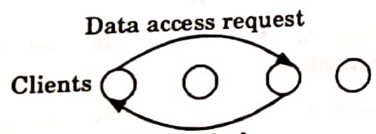


Fig. 3.25.1.

Que 3.26. **Describe the following algorithm for implementing DSM :**

i. The Migration Algorithm

ii. The Full-Replication Algorithm

AKTU 2014-15, Marks 05

Answer

i. **The Migration Algorithm :** Refer Q. 3.25, Page 3–26B, Unit-3.

ii. **The Full-Replication Algorithm :**

1. The full replication algorithm allows multiple nodes to have both read and write access to shared data blocks.

2. Because many nodes can write shared data concurrently, the access to shared data must be controlled to maintain its consistency.
3. One simple way to maintain consistency is to use a gap-free sequencer.
4. In this scheme, all nodes wishing to modify shared data will send the modifications to a sequencer.
5. The sequencer will assign a sequence number and multicast the modification with the sequence number to all the nodes that have a copy of the shared data item.
6. Each node processes the modification requests in the sequence number order.
7. A gap between the sequence number of a modification request and the expected sequence number at a node indicates that one or more modifications have been missed.

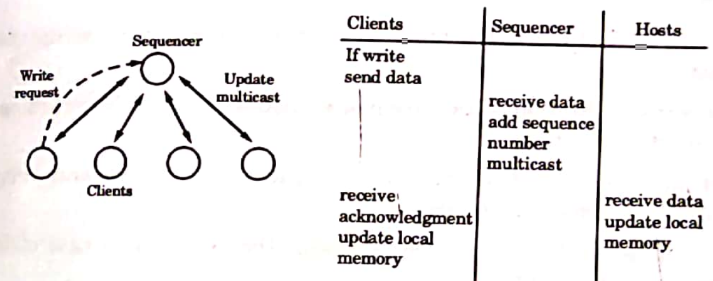


Fig. 3.26.1. Full replication algorithm.

8. Under such circumstances, the node will ask for the retransmission of the modifications it has missed.

Que 3.27. **What are the advantages of DSM ?**

Answer

Following are the advantages of DSM :

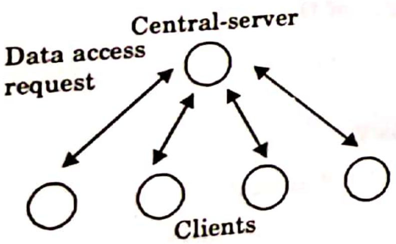
1. DSM provides a simple abstraction for sharing data.
2. DSM systems allow complex structures to be passed by reference, thus simplifying the development of algorithms for distributed applications.
3. DSM takes advantages of the locality of reference exhibited by programs and thereby cuts down on the overhead of communicating over the network.
4. DSM systems are cheaper to build.

5. The physical memory available at all the nodes of a DSM system combined together is enormous. This large memory can be used to run programs efficiently that require large memory without incurring disk latency.
6. In tightly coupled multiprocessor systems with a single shared memory, main memory memory is accessed via a common bus that limits the size of the multiprocessor system to a few tens of processors. DSM systems do not suffer from this drawback and can easily be scaled upwards.
7. Programs written for shared memory multiprocessors can be run on DSM systems without any changes.

Que 3.28. **Explain central-server algorithm for implementing distributed shared memory.**

Answer

1. In the central-server algorithm, a central-server maintains all the shared data.
2. It services the read request from other nodes or clients by returning the data items to them.
3. It updates the data on write request by clients and returns acknowledgment messages.
4. A timeout can be employed to resent the requests in case of failed acknowledgements.
5. Duplicate write requests can be detected by associating sequence numbers with write requests.
6. A failure condition is returned to the application typing to access shared data after several retransmission without a response.



Clients	Central server
Send data request	
	Receive request perform data access send response
Receive response	

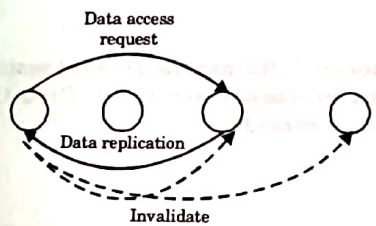
Fig. 3.28.1. Central- server algorithm.

Que 3.29. **Describe the read replication algorithm for implementing distributed shared memory.**

Answer

Read replication algorithm :

1. The read replication algorithm extends the migration algorithm by replicating data blocks and allows multiple nodes to have read access or one node to have read-write access (the multiple readers-one writer protocol).
2. Read replication can improve system performance by allowing multiple nodes to access data concurrently.
3. However, the write operation is expensive as all the copies of a shared block at various nodes will either have to be invalidated or update with the current value to maintain the consistency of the shared data block.
4. In the read replication algorithm, DSM must keep track of the location of all the copies of data blocks.



Clients	Remote host
If block not local determine location send request	
	Receive request send block
Receive block multicast invalidate	
	Receive invalidate invalidate block
Access data	

Fig. 3.29.1. Write operation in the read replication algorithm.

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. What are agreement protocols ? Explain Byzantine agreement problem, the consensus problem and interactive consistency problem.

Ans. Refer Q. 3.4.

Q. 2. What are agreement protocols ? Explain Byzantine agreement problem, the consensus problem and interactive consistency problem. Describe Lamport-Shostak-Pease algorithm.

Ans. Refer Q. 3.5.

Q. 3. What is Byzantine agreement problem ? Provide the solution to Byzantine agreement problem.

Ans. Refer Q. 3.7.

Q. 4. Describe Lamport-Shostak-Pease algorithm. How does vector clock overcome the disadvantages of Lamport clock ? Explain with an example.

Ans. Refer Q. 3.6.

Q. 5. Describe Byzantine agreement problem, and explain its solution. Show that Byzantine agreement cannot always be reached among four processors if two processors are faulty.

Ans. Refer Q. 3.9.

Q. 6. What are agreement protocols ? Discuss the general system model where agreement protocols are used. Give the applications of agreement protocols.

Ans. Refer Q. 3.14.

Q. 7.

i. Explain typical architecture of distributed file system. Give the mechanisms for building distributed file system.

ii. What is caching ? How is useful in DFS ?

Ans. Refer Q. 3.17.

Q. 8. What is cache ? Discuss read operation with cache and write operation with cache.

Ans. Refer Q. 3.18.

Q. 9. Write and explain various issues that must be addressed in design and implementation of distributed file system.

Ans. Refer Q. 3.19.

Q. 10. Explain naming in distributed system. What is flat naming and structured naming ?

Ans. Refer Q. 3.20.

Q. 11. Explain shared memory architecture and distributed memory architecture.

Ans. Refer Q. 3.21.

Q. 12. What is Distributed Shared Memory (DSM) ? Explain with diagram the architecture of distributed shared memory.

Ans. Refer Q. 3.23.

Q. 13. Explain the mechanism for building distributed file system also explain the design issues in distributed shared memory.

Ans. Refer Q. 3.24.

☺☺☺

UNIT 4

Failure Recovery in Distributed System

CONTENTS

PART-1

Failure Recovery in Distributed System : Concepts in Backward and Forward Recovery, Recovery in Concurrent System.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.1. Define forward recovery and backward recovery. List advantages and disadvantages of forward recovery. Explain two approaches of backward-error recovery.

AKTU 2014-15, 2016-17; Marks 10

Answer

Failure recovery is a process that involves restoring an erroneous state to an error-free state. There are two approaches for restoring an erroneous state to an error-free state :

1. **Forward-error recovery :**
   a. If the nature of errors and damages caused by faults can be completely and accurately assessed, then it is possible to remove those errors in the process state and enable the process to move forward. This technique is known as forward-error recovery.
   b. The forward-error recovery technique incurs less overhead because only those parts of the state that deviate from the intended value need to be corrected.
   c. This technique can be used only where the damages due to faults can be correctly assessed.

   Advantages of forward recovery : Relatively low overhead.

   Disadvantages of forward recovery :
   1. Dependent on damage assessment and error recovery and error prediction.
   2. Cannot provide a general mechanism for recovery design specifically for a particular system.

2. **Backward-error recovery :**
   a. If it is not possible to foresee the nature of fault and to remove all the errors in the process state, then the process state can be restored to a previous error-free state of the process. This technique is known as backward-error recovery.

b. In backward-error recovery, a process is restored to a prior state in the hope that the prior state is free of errors.

c. Backward-error recovery is simpler than forward-error recovery as it is independent of the fault and the errors caused by the fault.

d. Thus, a system can recover from an arbitrary fault by restoring to a previous state. This enables backward-error recovery to act as a general recovery mechanism to any type of system.

e. The points in the execution of a process to which the process can later be restored are known as recovery points.

f. A recovery point is said to be restored when the current state of a process is replaced by the state of the process at the recovery point.

Approaches for implementing backward-error recovery :

1. **Operation-based approach :**
   a. In the operation-based approach, all the modifications that are made to the state of a process are recorded in sufficient detail so that a previous state of the process can be restored by reversing all the changes made to the state.
   b. The record of the system activity is known as an audit trail or a log.
2. **State-based approach :**
   a. In the state-based approach, the complete state of a process is saved when a recovery point is established and recovering a process involves reinstating its saved state and resuming the execution of the process from the state.
   b. The process of saving state is also referred to as checkpointing. The recovery point at which checkpointing occurs is often referred to as a checkpoint.
   c. The process of restoring a process to a prior-state is referred to as rolling back the process.

Que 4.2. **Define forward and backward recovery. Also list the advantages and disadvantages of both.** AKTU 2018-19, Marks 10

Answer

Forward and backward recovery, advantages and disadvantages of forward recovery : Refer Q. 4.1, Page 4–2B, Unit-4.

Advantages of backward recovery :

1. Backward recovery can handle unpredictable errors caused by residual design faults.
2. Backward recovery can be used regardless of the damage sustained by the state.
3. Backward recovery can handle transparent or permanent arbitrary faults.

Disadvantages of backward recovery :

1. Backward recovery requires significant resources (*i.e.*, time, computation, and stable storage) to perform checkpointing and recovery.
2. The implementation of backward recovery often requires that the system be halted temporarily.
3. Restoring the previous state of a system or process (performance wise) relatively costly.

Que 4.3. **What do you mean by recovery in concurrent system ? Explain.** AKTU 2014-15, Marks 05

OR

What do you mean by backward and forward error recovery ? Discuss recovery in concurrent systems in detail. AKTU 2015-16, Marks 10

Answer

Backward and forward error recovery : Refer Q. 4.1, Page 4–2B, Unit-4.

Recovery in concurrent system :

1. In concurrent systems, several processes cooperate by exchanging information to accomplish a task.
2. The information exchange can be through a shared memory in the case of shared memory machines or through messages in the case of distributed system.
3. Recovery in concurrent system means to rollback all the processes at the time of failure.
4. During failure the system assigns a recovery points at the point where failure occur in the process.
5. If the failed process is associated with active process then the active process must also rollback at an earlier state.
6. Recovery point helps to undo the effect caused by failed process.

PART-2

Obtaining Consistent Checkpoints, Recovery in Distributed Database System.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.4. **What is checkpointing ? Explain strongly consistent set of checkpoint.**

Answer

1. Checkpointing is a mechanisms that enables transactions to recover from inconsistent state using backward error recovery.
2. Checkpointing in distributed system involves taking a checkpoint by all the processes (sites) or at least by a set of processes (sites) that interact with one another in performing a distributed computation.
3. In distributed systems, all the sites save their local states, which are known as local checkpoints, and the process of saving local states is called local checkpointing.
4. All the local checkpoints, one from each site, collectively form a global checkpoint.

Strongly consistent set of checkpoints :

1. The domino effect is caused by orphan messages, which themselves are due to rollbacks.
2. To overcome the domino effect, a set of local checkpoints is needed such that no information flow takes place between any pair of processes in the set, as well as in between any process in the set and any process outside the set during the interval spanned by the checkpoints.
3. Such a set of checkpoints is known as a recovery line or a strongly consistent set of checkpoints.

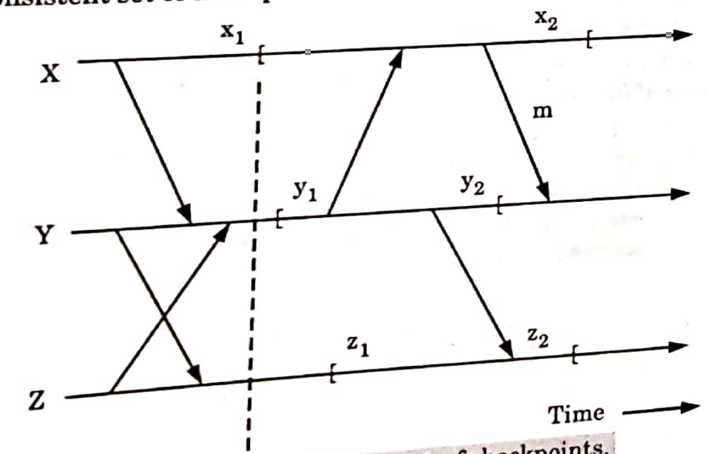


Fig. 4.4.1. Consistent set of checkpoints.

In the Fig. 4.4.1, the set $\{x_1, y_1, z_1\}$ is a strongly consistent set of checkpoints and the thinly dotted lines denote the interval spanned by the checkpoints.

5. A strongly consistent set of checkpoints corresponds to a strongly consistent global state, wherein all messages have been delivered and processed, and no message is in transit.
6. Processes X, Y, and Z can be rolled back to their respective checkpoints x_1, y_1, and z_1 and resume execution in the event of a failure.
7. No further rollbacks due to the domino effect would be necessary as no information exchange took place in the interval spanned by the set of checkpoints.
8. That is, no local checkpoint includes an effect whose cause would be undone due to the rollback of another process.

Que 4.5. **Write short note on consistent checkpoints.**

Answer

Consistent checkpoints :

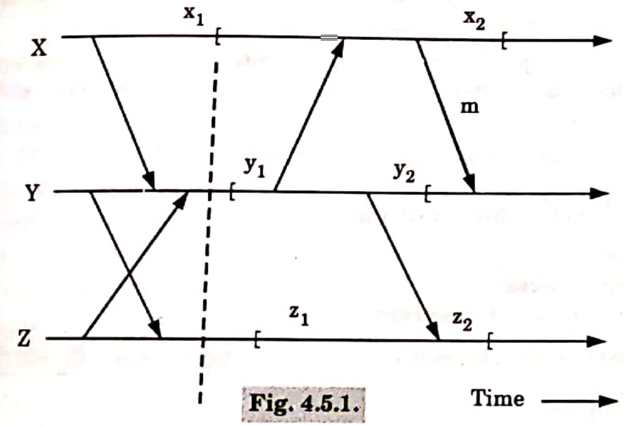


Fig. 4.5.1.

1. Suppose that Y fails after receiving message m as shown in Fig. 4.5.1.
2. If Y restarts from checkpoint y_2, message m is lost due to rollback. The set $\{x_2, y_2, z_2\}$ is referred to as a consistent set of checkpoints.
3. A consistent set of checkpoints is similar to a consistent global state in that it requires that each message recorded as received in a checkpoint (state) should also be recorded as sent in another checkpoint (state).
4. Therefore, systems that do not establish a strongly consistent set of checkpoints have to deal with lost messages during rollback recovery.
5. While the systems which establish strongly consistent set of checkpoints do not have to deal with lost messages during rollback recovery, they experience delays during the checkpointing process as processes cannot exchange messages while checkpointing is in progress.

Que 4.4. **What is checkpointing ? Explain strongly consistent set of checkpoint.**

Answer

1. Checkpointing is a mechanisms that enables transactions to recover from inconsistent state using backward error recovery.
2. Checkpointing in distributed system involves taking a checkpoint by all the processes (sites) or at least by a set of processes (sites) that interact with one another in performing a distributed computation.
3. In distributed systems, all the sites save their local states, which are known as local checkpoints, and the process of saving local states is called local checkpointing.
4. All the local checkpoints, one from each site, collectively form a global checkpoint.

Strongly consistent set of checkpoints :

1. The domino effect is caused by orphan messages, which themselves are due to rollbacks.
2. To overcome the domino effect, a set of local checkpoints is needed such that no information flow takes place between any pair of processes in the set, as well as in between any process in the set and any process outside the set during the interval spanned by the checkpoints.
3. Such a set of checkpoints is known as a recovery line or a strongly consistent set of checkpoints.

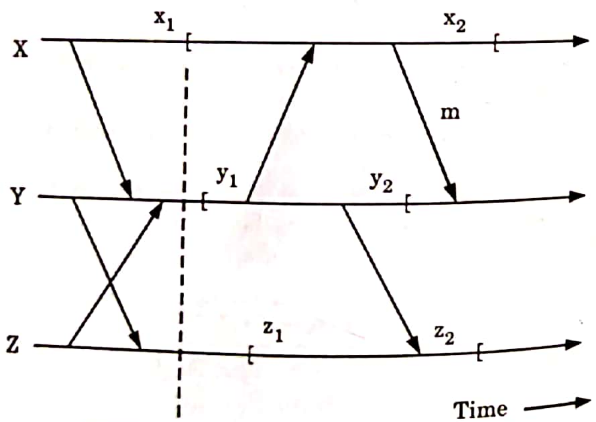


Fig. 4.4.1. Consistent set of checkpoints.

4. In the Fig. 4.4.1, the set $\{x_1, y_1, z_1\}$ is a strongly consistent set of checkpoints and the thinly dotted lines denote the interval spanned by the checkpoints.

5. A strongly consistent set of checkpoints corresponds to a strongly consistent global state, wherein all messages have been delivered and processed, and no message is in transit.
6. Processes X, Y, and Z can be rolled back to their respective checkpoints x_1, y_1, and z_1 and resume execution in the event of a failure.
7. No further rollbacks due to the domino effect would be necessary as no information exchange took place in the interval spanned by the set of checkpoints.
8. That is, no local checkpoint includes an effect whose cause would be undone due to the rollback of another process.

Que 4.5. **Write short note on consistent checkpoints.**

Answer

Consistent checkpoints :

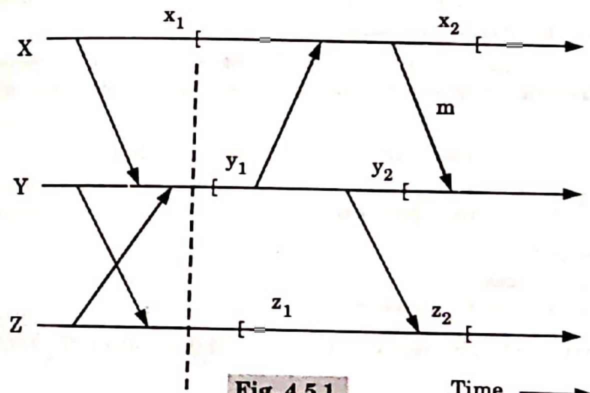


Fig. 4.5.1.

1. Suppose that Y fails after receiving message m as shown in Fig. 4.5.1.
2. If Y restarts from checkpoint y_2, message m is lost due to rollback. The set $\{x_2, y_2, z_2\}$ is referred to as a consistent set of checkpoints.
3. A consistent set of checkpoints is similar to a consistent global state in that it requires that each message recorded as received in a checkpoint (state) should also be recorded as sent in another checkpoint (state).
4. Therefore, systems that do not establish a strongly consistent set of checkpoints have to deal with lost messages during rollback recovery.
5. While the systems which establish strongly consistent set of checkpoints do not have to deal with lost messages during rollback recovery, they experience delays during the checkpointing process as processes cannot exchange messages while checkpointing is in progress.

Que 4.6. **Write a short note on method to obtain consistent set of checkpoints.**

AKTU 2016-17, Marks 10

Answer

Method to obtain consistent set of checkpoints :

1. Assume that the action of taking a checkpoint and the action of sending or receiving a message are indivisible; that is, they are not interrupted by any other events.
2. If every process takes a checkpoint after sending every message, the set of the most recent checkpoints is always consistent.
3. The set of latest checkpoints is consistent because the latest checkpoint at every process corresponds to a state where all the messages recorded as received in it have already been recorded elsewhere as sent.
4. Therefore, rolling back a process to its latest checkpoint would not result in any orphan messages.
5. Taking a checkpoint after sending each message is sent is expensive, so we reduce the overhead by taking a checkpoint after every $K(K > 1)$ messages sent.
6. However, this method suffers from the domino effect.

Que 4.7. **Write short note on :**

i. Livelocks
ii. Domino effects
iii. Failure resilient processes
iv. Consistent checkpoints

AKTU 2014-15, Marks 10

Answer

i. **Livelocks :**

1. In rollback recovery, livelock is a situation in which a single failure can cause an infinite number of rollbacks, preventing the system from making progress.
2. A livelock situation in a distributed system is shown in Fig. 4.7.1. Here Fig. 4.7.1(*a*) illustrates the activity of two processes X and Y until the failure of Y.
3. Process Y fails before receiving message n_1, sent by X.
4. When Y rolls back to y_1, there is no record of sending message m_1, hence X must rollback to x_1.
5. When process Y recovers, it sends out m_2 and receives n_1 [Fig. 4.7.1(*b*)].

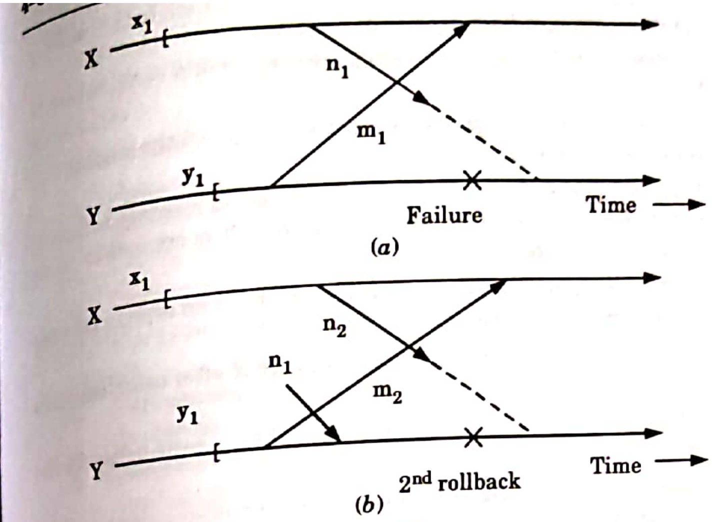


Fig. 4.7.1.

6. Process X, after resuming from x_1, sends n_2 and receives m_2.
7. However, because X is rolled back, there is no record of sending n_1 and hence Y has to rollback for the second time.
8. This forces X to rollback too, as it has received m_2, and there is no record of sending m_2 at Y.
9. This situation can repeat indefinitely, preventing the system from making any progress.

ii. **Domino effect :**

1. Consider the system activity illustrated in the Fig. 4.7.2.
2. In the Fig. 4.7.2, X, Y, and Z are three processes that cooperate by exchanging information (shown by the arrows).
3. Each symbol '[' marks a recovery point to which a process can be rolled back in the event of a failure.

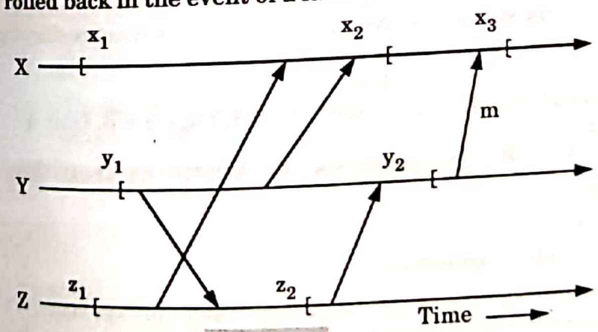


Fig. 4.7.2.

4. If process X is to be rolled back, it can be rolled back to the recovery point x_3 without affecting any other process.
5. Suppose that Y fails after sending message m and is rolled back to y_2.
6. In this case, the receipt of m is recorded in x_3, but the sending of m is not recorded in y_2.
7. Now we have a situation where X has received message m from Y, but Y has no record of sending it, which corresponds to an inconsistent state.
8. Under such circumstances, m is referred to as an orphan message and process X must rollback.
9. X must rollback because Y interacted with X after establishing its recovery point y_2.
10. When Y is rolled back to y_2, the event that is responsible for the interaction is undone.
11. Therefore, all the effects at X caused by the interaction must also be undone.
12. This can be achieved by rolling back X to recovery point x_2.
13. In the same way, if Z is rolled back, all three processes must rollback to their very first recovery points, namely, x_1, y_1, and z_1.
14. This effect, where rolling back one process causes one or more other processes to rollback, is known as domino effect.

iii. Failure resilient processes :

1. The fundamental unit of execution is a process.
2. Hence, in order for any system to be fault-tolerant, the processes of that system must be resilient to system failure.
3. A process is said to be resilient, if it marks failures and guarantees progress despite a certain number of system failures.
4. In other words, a minimum disruption is caused to service provided by the process in event of a system failure.
5. Two approaches have been proposed to implement resilient process :
   a. Backup processes
   b. Replicated execution

iv. Consistent checkpoints : Refer Q. 4.5, Page 4–6B, Unit-4.

Que 4.8. Explain synchronous checkpointing algorithm.

Answer

Synchronous checkpointing :

1. In this approach, processes synchronize their checkpointing activity so that a globally consistent set of checkpoints is always maintained in the system.

2. In this method, consistent set of checkpoints are used which avoids livelock problems during recovery.

Algorithm :

1. It assumes the following characteristics :
   a. Processes communicate by exchanging messages through communication channels.
   b. Channels are FIFO in nature.
   c. Communication failures do not partition the network.
2. The checkpoint algorithm takes two kinds of checkpoints on stable storage :
   a. **Permanent checkpoint :** A permanent checkpoint is a local checkpoint at a process and is a part of a consistent global checkpoint.
   b. **Tentative checkpoints :** A tentative checkpoint is a temporary checkpoint that is made a permanent checkpoint on the successful termination of the checkpoint algorithm.
3. Processes rollback only to their permanent checkpoints.
4. The algorithm has two phases :
   a. **First phase :**
      i. An initiating process P_i takes tentative checkpoint and requests all the processes to take tentative checkpoints.
      ii. Each process informs process P_i whether it accepts or rejects the request of taking tentative checkpoint.
      iii. When all the process has successfully accepted the tentative checkpoints then P_i decides to make this checkpoint a permanent checkpoint. Otherwise tentative checkpoint is discarded.
   b. **Second phase :**
      i. P_i informs all the processes of the decision it reached at the end of the first phase.
      ii. A process, on receiving the message from P_i, will act accordingly.
      iii. Therefore, either all or none of the processes accept permanent checkpoints.

Que 4.9. Write short note on lost message.

Answer

Lost messages :

a. Suppose that checkpoints x_1 and y_1 (Fig. 4.9.1) are chosen as the recovery points for processes X and Y, respectively.

b. In this case, the event that sent message m is recorded in x_1, while the event of its receipt at Y is not recorded in y_1.

c. If Y fails after receiving message m, the system is restored to state $\{x_1, y_1\}$, in which message m is lost as process X is past the point where it sends message m.

d. This condition can also arise if m is lost in the communication channel and processes X and Y are in state x_1 and y_1, respectively.

e. Both the above conditions are indistinguishable.

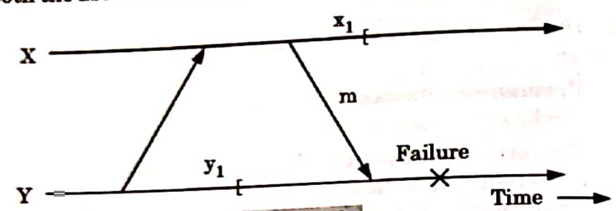


Fig. 4.9.1.

Que 4.10. **Explain the approaches to implement resilient process.**

Answer

Two approaches have been proposed to implement resilient process :

1. **Backup processes :**
   a. In the backup processes approach, each resilient process is implemented by a primary process and one or more backup processes. The primary process executes while the backup processes are inactive.
   b. If the primary process terminates because of a failure, one of the backup processes becomes active and takes over the functions of the primary process.
   c. To minimize the computation that has to be redone by the backup process, the state of the primary process is stored (checkpointed) at appropriate intervals.
   d. The checkpointed state is stored in a suitable place such that the failure of the primary process's machine does not affect the checkpoint's availability.
2. **Replicated execution :**
   a. In the replicated execution approach, several processes execute the same program concurrently.
   b. As long as one of the processes survives failures, the computation or the service continues.

   c. A significant advantage of replicated execution is that it can be used to increase reliability as well as availability.
   d. The reliability of a computation can be increased by taking a majority consensus among the results generated by all the process.
   e. This final result can then be used in subsequent computations.

Que 4.11. **Explain rollback recovery algorithm in distributed database system.**

Answer

1. This algorithm assumes that a single process invokes the algorithm as opposed to several processes concurrently invoking it to rollback and recover.
2. It also assumes that the checkpoint and the rollback recovery algorithms are not concurrently invoked.
3. This algorithm has two phases :
   a. **First phase :**
      i. An initiating process P_i checks whether all the processes are willing to restart from their previous checkpoints.
      ii. A process may reject the request of P_i if it is already participating in a checkpointing or a recovering process initiated by some other process.
      iii. If all the processes accept the request of P_i to restart from their previous checkpoints, P_i decides to restart all the processes.
      iv. Otherwise all the processes continue with their normal activities.
   b. **Second phase :**
      i. P_i propagates its decision to all the processes. On receiving P_i's decision, a process will act accordingly.
      ii. The recovery algorithm requires that every process do not send messages related to underlying computation while it is waiting for P_i's decision.

Que 4.12. **Write the difference between deadlock and livelock.**

Answer

Difference :

S. No.	Deadlock	Livelock
1.	A deadlock is a situation where a process or a set of processes is blocked and waiting for an event that will never occur.	Livelock is a situation in which single failure of a process can cause an infinite number of rollbacks, preventing the systems from making progress.
2.	The problem of deadlocks is common in computer system where resource sharing is frequent.	Livelocks are common in distributed system where synchronization is maintained by message passing.
3.	Deadlock can be handled by various deadlock handling strategies like prevention, detection and avoidance.	Livelocks can be prevented by coordinating the processes either at the time of establishing checkpointing or at the beginning of recovery.

PART-3

Fault Tolerance : Issues in Fault Tolerance.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.13. Define fault and failure. What are different approaches to fault-tolerance ? Explain. AKTU 2014-15, Marks 05

Answer

Faults : A fault is an anomalous physical condition. The causes of a fault include design errors, manufacturing problems, damage fatigue or other deterioration, and external disturbances.

Failure : Failure of a system occurs when the system does not perform its services in the manner specified. An erroneous state of the system is a state which could lead to a system failure by a sequence of valid state transaction.

Different fault tolerance approaches :

1. **Replication :**
   a. Replication is the process of creating and maintaining multiple copies of data objects or processes on several nodes.
   b. Therefore, if failure on one node occurs then data will be accessible to the user from other node.
   c. Replication provides high data availability and performance.
2. **Checkpointing :**
   a. Fault tolerance can be achieved through checkpointing.
   b. Checkpointing means to periodically save the consistent state of the system in a reliable storage medium. Each such instance when a system is in the consistent state is called a checkpoint.
   c. Checkpointing is primarily used to avoid losing all the useful processing done before a fault has occurred.
   d. In case of a fault, checkpoint enables the execution of a program to be resumed from a previous consistent state rather than resuming the execution from the beginning.

Que 4.14. Discuss at least three main issues that are relevant to the understanding of distributed fault tolerance system. Explain how that makes it important. AKTU 2015-16, Marks 10

Answer

Issues in the fault tolerance are as follows :

1. **Process deaths :**
   a. When a process dies, it is important that the resources allocated to that process are recouped, otherwise they may be permanently lost.
   b. Many distributed systems are structured along the client-server model in which a client requests a service by sending a message to a server.
   c. If the server process fails, it is necessary that the client machine be informed so that the client process, waiting for a reply can be unblocked to take suitable action.
   d. Similarly, if a client process dies after sending a request to a server, it is imperative that the server be informed that the client process no longer exists.
   e. This will facilitate the server in reclaiming any resources it has allocated to the client process.

2. **Machine failure :**
   a. In the case of machine failure, all the processes running at the machine will die.
   b. As far as the behaviour of a client process or a server process is concerned, there is not much difference in their behaviour in the event of a machine failure or a process death.
   c. In case of machine failure, an absence of any kind of message indicates either process death or a failure.
3. **Network failure :**
   a. A communication link failure can partition a network into subnets, making it impossible for a machine to communicate with another machine in a different subnet.
   b. A process cannot give the difference between a machine and a communication link failure, unless the underlying communication network (such as a slotted ring network) can recognize a machine failure.
   c. If the communication network cannot recognize machine failures and thus cannot return a suitable error code (such as ethernet), a fault-tolerant design will have to assume that a machine may be operating and processes on that machine are active.

PART-4

Commit Protocol, Voting Protocol, Dynamic Voting Protocol.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.15. **Explain two phase commit protocol.**

Answer

Two phase commit protocol : Two phase commit protocol is designed to allow any participant to abort its part of transaction. Due to the requirement for atomicity, if one part of a transaction is aborted then the whole transaction must also be aborted.

Phase 1 (Voting phase) :

1. The coordinator sends a canCommit request to each of the participants in the transaction.
2. When a participant receives a canCommit request it replies with its vote (yes or no) to the coordinator. Before voting yes, it prepares to commit by saving objects in permanent storage. If the vote is no, the participant aborts immediately.

Phase 2 (Completion according to outcome of vote) :

1. The coordinator collects the votes (including its own) :
   a. If there are no failures and all the votes are yes, the coordinator decides to commit the transaction and sends a doCommit request to each of the participants.
   b. Otherwise, the coordinator decides to abort the transaction and sends doAbort requests to all participants that voted yes.
2. Participants that voted yes are waiting for a doCommit or doAbort request from the coordinator. When a participant receives one of these messages, it acts accordingly and in the case of commit, makes a haveCommitted calls as confirmation to the coordinator.

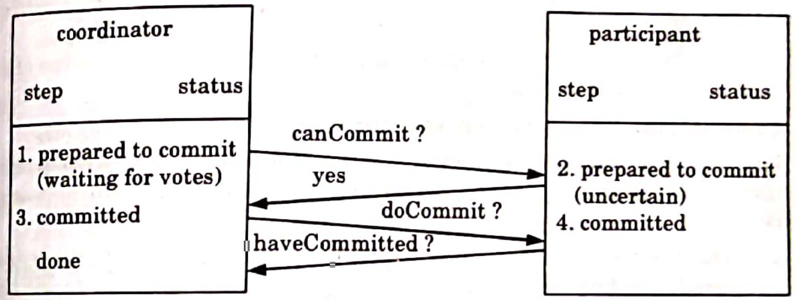


Fig. 4.15.1. Communication in two phase commit protocol.

Que 4.16. **What are commit protocols ? Explain how two phase protocols respond to failure of participating site and failure of coordinator.**

AKTU 2014-15, Marks 05

Answer

Commit protocols :

1. In distributed system commit protocols ensure the atomicity across the sites, *i.e.*, when a transaction executes at multiple sites it must either be committed at all the sites or aborted at all the sites.
2. The goal of commit protocols is to have all the concern participants agree either to commit or to abort a transaction.

Handling a failure of a participating site :

Let us assume that the failed site is S_i and the Transaction Coordinator is TC. There are two things we need to look into to handle failure of a participating site :

1. **The response of the Transaction Coordinator of transaction *T* :**
   a. If the failed site have not sent any message (<ready *T*>), the TC cannot decide to commit the transaction. Hence, the transaction *T* should be aborted and other participating sites is to be informed.
   b. If the failed site have sent a message (<ready *T*>), the TC can assume that the failed site also was ready to commit, hence the transaction can be committed by TC and the other sites will be informed to commit. In this case, the site which recovers from failure has to execute the two phase (2PC) protocol to set its local database up-to-date.
2. **The response of the failed site when it recovers :**
   a. When recover from failure, the recovering site S_i must identify the fate of the transactions which was going on during the failure of S_i. This can be done by examining the log file entries of site S_i.
   b. This is how the two phase (2PC) protocol handles the failure of a participating Site.

Handling the failure of a coordinator site :

Let us suppose that the coordinator site failed during execution of two phase (2PC) protocol for a transaction *T*. This situation can be handled in following two way :

1. The other sites which are participating in the transaction *T* may try to decide the fate of the transaction. That is, they may try to decide on commit or abort of *T* using the control messages available in every site.
2. The second way is to wait until the coordinator site recovers.

Que 4.17. **Describe three phase commit protocol. How three phase commit protocol is different than two phase commit protocol ?**

AKTU 2017-18, Marks 10

Answer

Phases in three phase commit protocol :

1. The three-phase commit (3PC) protocol is a distributed algorithm which lets all nodes in a distributed system agrees to commit a transaction.
2. 3PC is non-blocking protocol.
3. 3PC places an upper bound on the amount of time required before a transaction either commits or aborts.
4. This property ensures that if a given transaction holds some resource locks, it will release the locks after the time-out.
5. The three-phase commit (3PC) protocol is more complicated and more expensive phase in 3PC protocol.

Phase 1 : Voting/Prepare phase :

1. Transaction Coordinator (TC) of the transaction writes BEGIN_COMMIT message in its log file and sends PREPARE message to all the participating sites and waits.
2. On receiving PREPARE message, if a site is ready to commit, then the site's Transaction Manager (TM) writes READY in its log and send VOTE_COMMIT to TC.
3. If any site is not ready to commit, it writes ABORT in its log and responds with VOTE_ABORT to the TC.

Phase 2 : Buffering/Pre-commit phase :

1. If TC received VOTE_COMMIT from all the participating sites, then it writes PREPARE_TO_COMMIT in its log and sends PREPARE_TO_COMMIT message to all the participating sites.
2. If TC receives any one VOTE_ABORT message, it writes ABORT in its log and sends GLOBAL_ABORT to all the participating sites and also writes END_OF_TRANSACTION message in its log.
3. On receiving the message PREPARE_TO_COMMIT, the TM of participating sites write PREPARE_TO_COMMIT in their log and respond with READY_TO_COMMIT message to the TC.
4. If they receive GLOBAL_ABORT message, then TM of the sites write ABORT in their logs and acknowledge the abort.

Phase 3 : Decision/Commit or abort phase :

1. If all responses are READY_TO_COMMIT, then TC writes COMMIT in its log and send GLOBAL_COMMIT message to all the participating sites' TMs.
2. The TM of all sites then writes COMMIT in their log and sends an acknowledgement to the TC. Then, TC writes END_OF_TRANSACTION in its log.

Three phase vs. two phase commit protocol :

1. In two-phase commit protocol, when coordinator fails during execution then participating sites are unable to determine whether the coordinator has made a decision to abort or commit the transaction, which cause participating sites to be in blocked state.
2. To remove this blocking problem in 2PC, three phase commit protocol was proposed. Three-Phase Commit protocol is able to prevent this blocking problem by taking the decision based on the decision of all sites.

Que 4.18. **What is voting protocol ? Explain static voting and dynamic voting protocols.**

AKTU 2014-15, Marks 05

OR

Describe in detail :

a. Dynamic voting protocols
b. Method to obtain consistent set of checkpoint

AKTU 2016-17, Marks 10

Answer

Voting protocol :

1. Voting protocol is a common approach to provide fault tolerance in distributed system by replicating data at many sites (or nodes).
2. If a site is not available, the data can still be obtained from copies at other sites.
3. With the voting mechanism, each replica is assigned some number of votes, and a majority of votes must be collected from a process before it can access a replica.
4. The voting mechanism is more fault tolerant than a commit protocol in that it allows access to data under network partitions, site failures, and message losses without compromising the integrity of the data.

Static voting protocol :

1. In static voting scheme, the replicas of files are stored at different sites.
2. Every file access operation requires that an appropriate lock is obtained.
3. The lock granting rules allow either 'one writer and no readers' or 'multiple readers and no writers' to access a file simultaneously.
4. It is assumed that at every site there is a lock manager that performs the lock related operations, and every file is associated with a version number, which gives the number of times the file has been updated.
5. The version numbers are stored on stable storage, and every successful write operation on a replica updates its version number.

Dynamic voting protocol :

1. Suppose that in the system shown in Fig. 4.18.1, site 4 becomes unreachable from the rest of the sites due to its failure or due to a network partition.

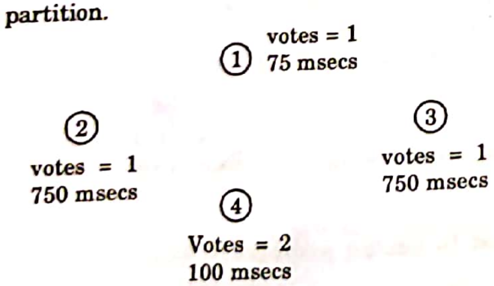


Fig. 4.18.1.

2. Site 1, 2, and 3 can still collect a quorum (also referred to as majority) while site 4 cannot collect a quorum.
3. If another partition or a failure of a site occurs, making any site unavailable, the system cannot serve any read or write requests as a quorum cannot be collected in any partition.
4. In other words, the system is completely unavailable which is a serious problem.
5. Dynamic voting protocols solve this problem by adapting the number of votes or the set of sites that can form a quorum, to the changing state of the system due to site and communication failures.
6. In the dynamic protocols, following two approaches are used to enhance availability :
   a. **Majority based approach :** The set of sites that can form a majority to allow access to replicated data changes with the changing state of the system.
   b. **Dynamic vote reassignment :** The number of votes assigned to a site changes dynamically.

Method to obtain consistent set of checkpoint : Refer Q. 4.6, Page 4–7B, Unit-4.

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. Define forward recovery and backward recovery. List advantages and disadvantages of forward recovery. Explain two approaches of backward-error recovery.

Ans. Refer Q. 4.1.

Q. 2. What do you mean by recovery in concurrent system ? Explain.

Ans. Refer Q. 4.3.

Q. 3. Write a short note on method to obtain consistent set of checkpoints.

Ans. Refer Q. 4.6.

Q. 4. Write short note on :
i. **Livelocks**
ii. **Domino effects**

iii. **Failure resilient processes**
iv. **Consistent checkpoints**

Ans. Refer Q. 4.7.

Q. 5. Define fault and failure. What are different approaches to fault-tolerance ? Explain.

Ans. Refer Q. 4.13.

Q. 6. What are commit protocols ? Explain how two phase protocols respond to failure of participating site and failure of coordinator.

Ans. Refer Q. 4.16.

Q. 7. Explain two phase commit protocol.

Ans. Refer Q. 4.15.

Q. 8. Describe three phase commit protocol. How three phase commit protocol is different than two phase commit protocol ?

Ans. Refer Q. 4.17.

Q. 9. What is voting protocol ? Explain static voting and dynamic voting protocols.

Ans. Refer Q. 4.18.

☺☺☺

5 UNIT

Transaction and Concurrency Control

CONTENTS

PART-1

Transaction and Concurrency Control : Transaction, Nested Transactions.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 5.1. **Explain transaction and its properties.**

Answer

Transaction is a sequence of read, compute and write statements that refer to the data objects of a database.

Properties of transactions : Transactions have four essential properties, known as ACID properties :

1. **Atomicity :** Atomicity means that the transaction completes its processing as an indivisible and atomic unit either successfully or unsuccessfully. After the transaction, the database changes consistency or not changed at all.
2. **Consistency :** Consistency means that the database is in consistent state before transaction and after the termination of transaction, the database will also be in the consistent state.
3. **Isolation :** Isolation property indicates that every action processed by a transaction is kept isolated until the completion of transaction. During the transaction processing, the processing will be hidden from outside the transaction.
4. **Durability :** Durability means any failure made after the commit operation will not affect the database. The commit action will reflect its results to the database after its termination.

Que 5.2. **Write short note on nested transaction.**

AKTU 2016-17, Marks 10

Answer

Nested transaction :

1. In a nested transaction, the top-level transaction can open further subtransactions, and each subtransaction can open further subtransactions down to any depth of nesting.

2. The Fig. 5.2.1 shows a client's transaction T that opens two subtransactions T_1 and T_2, which access objects at servers X and Y.
3. The subtransactions T_1 and T_2 open further subtransactions T_{11}, T_{12}, T_{21} and T_{22} which access objects at servers M, N, O and P.
4. In the nested case, subtransactions at the same level can run concurrently, so T_1 and T_2 are concurrent, and as they invoke objects in different servers, they can run in parallel.
5. The four subtransactions T_{11}, T_{12}, T_{21} and T_{22} also run concurrently.

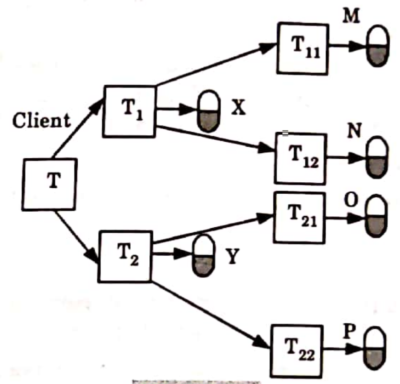


Fig. 5.2.1.

Que 5.3. **Explain how the two phase commit protocol for nested transaction ensures that if the top level transactions commit, all the right descendents are committed or aborted ?**

AKTU 2015-16, Marks 10

Answer

1. Consider the top-level transaction T and its subtransactions shown in Fig. 5.3.1.

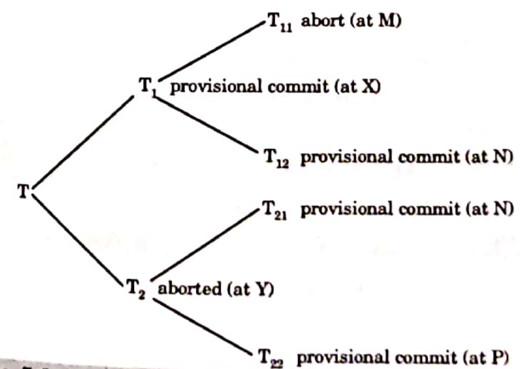


Fig. 5.3.1. Transaction T decides whether to commit.

2. Each subtransaction has either provisionally committed or aborted. For example,
   a. T_{12} has provisionally committed and T_{11} has aborted, but the fate of T_{12} depends on its parent T_1 and eventually on the top-level transaction, T.
   b. Although T_{21} and T_{22} have both provisionally committed, T_2 has aborted and this means that T_{21} and T_{22} must also abort.
   c. Suppose that T decides to commit in spite of the fact that T_2 has aborted, also that T_1 decides to commit in spite of the fact that T_{11} has aborted.
4. When a top-level transaction completes, its coordinator carries out a two phase commit protocol. The only reason for a participant subtransaction being unable to complete is if it has crashed since it completed its provisional commit.
5. When each subtransaction was created, it joined its parent transaction.
6. Therefore, the coordinator of each parent transaction has a list of its child subtransactions.
7. When a nested transaction provisionally commits, it reports its status and the status of its descendants to its parent.
8. When a nested transaction aborts, it just reports abort to its parent without giving any information about its descendants.
9. Eventually, the top-level transaction receives a list of all the subtransactions in the tree, together with the status of each.
10. Descendants of aborted subtransactions are actually omitted from this list.

PART-2

Locks, Optimistic Concurrency Control, Timesstamp Ordering, Comparison of Methods for Concurrency Control.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 5.4. What is lock ? What are the different modes in which transaction can lock a data object.

Answer

1. A lock is a variable associated with shared resources such as data item that determines whether read/write operation can be performed on that data item.
2. In lock based techniques, each data object has a lock associated with it.
3. A transaction can hold, request or release the lock on a data object, as required by the transaction.
4. The transaction is said to have the locked data object, if it holds a lock.
5. There are two modes of locking in which transaction can lock data object :
   a. **Exclusive :**
      i. If a transaction has locked the data object in exclusive mode, no other transaction can lock it in any mode.
      ii. In this locking scheme, the server attempts to lock any object.
      iii. If a client requests access to an object, the request is suspended and the client must wait until the object is unlocked.
   b. **Shared :**
      i. If the transaction has locked the data object in shared mode, other transaction can concurrently lock it but only in shared mode.
      ii. If a client requests access to an object, the request is always successful.
      iii. All the transactions reading the same object can share their read lock.

Que 5.5. Discuss 2PL and strict 2PL in context of distributed system.

AKTU 2015-16, 2016-17; Marks 7.5

Answer

Two phase locking :

1. This locking scheme is also called as 2 PL.
2. Two phase locking is a dynamic locking scheme in which a transaction requests a lock on a data object when it needs the data object.
3. Two phase locking is called two phase as it has two phases :
   a. **Growing phase :** It is a phase during which new locks are acquired.
   b. **Shrinking phase :** It is a phase during which locks are released.
4. Two phase locking imposes a constraint on lock acquisition and the lock release actions of a transaction to guarantee consistency.
5. The state of a transaction in which it releases locks and hold locks on all the needed data objects is referred to as lock point. An execution is shown in Fig. 5.5.1.

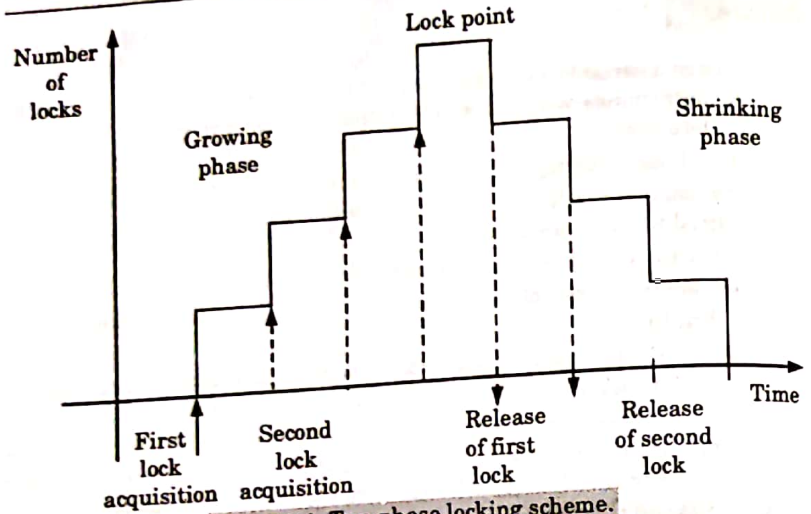


Fig. 5.5.1. Two phase locking scheme.

Strict two phase locking :

1. Under a strict execution, a transaction that needs to read or write an object must be delayed until other transactions that wrote the same object have committed or aborted.
2. To enforce this rule, any locks applied during the progress of transaction are held until the transaction either commit or aborts. This is called strict two phase locking.
3. Strict two phase locking eliminates cascaded aborts because transaction can read data objects modified by a transaction only after the transaction has completed.
4. However, strict two phase locking reduces concurrency as a transaction holds locks for a longer period than required for consistency.

Que 5.6. **How two phase locking is implemented in distributed database system ?**

Answer

Two phase locking can be implemented in a distributed database system in the following way :

1. A Data Manager (DM) at a site controls the locks associated with objects stored at that site.
2. A Transaction Manager (TM) communicates with the appropriate DM to lock or unlock a data object.

3. If a request for lock cannot be granted, the DM puts it on the waiting queue of the object.
4. When a lock on an object is released, one of the waiting requests for the lock on that object is granted.

Que 5.7. **Explain optimistic concurrency control.**

AKTU 2014-15, 2016-17; Marks 05

Answer

Optimistic concurrency control states that the conflicts among the transactions are rare in distributed database system. It is only an assumption so it is also called optimistic. In optimistic concurrency control scheme, each transaction goes through three phase :

1. **Working phase :** During this phase, each transaction has a tentative version of each of the objects that it updates. The use of tentative versions allows the transaction to abort either during the working phase or other validation phase. The rules for read/write are :
   a. **Read operation** is performed if the tentative version for that transaction already exists.
   b. **Write operation** record the new values of several concurrent transaction objects as tentative values which are invisible to other transactions.
2. **Validation phase :** When the close transaction request is received, the transaction is validated to establish whether or not its operations on objects conflicts with operations of other transaction on same objects.
3. **Update phase :** If the transaction is validated, all the changes recorded in its tentative versions are made permanent—read only transaction can commit immediately after passing validation.

Que 5.8. **Discuss the optimistic methods for distributed concurrency control. What are the different validation conditions for optimistic concurrency control ? Explain.**

AKTU 2015-16, Marks 10

Answer

Optimistic concurrency control : Refer Q. 5.7, Page 5–7B, Unit-5.

Validation condition for optimistic concurrency control : Let T_i and T_j be the two transactions. For a transaction T_j to be serializable with respect to an overlapping transaction T_i, their operations must confirm to the following rules/conditions :

1. T_i must not read objects written by T_j.
2. T_j must not read objects written by T_i.
3. T_i must not write objects written by T_j and T_j must not write objects written by T_i.

Que 5.9. **What are the different validation conditions for optimistic concurrency control ? How it effects the transaction in distributed system.**

AKTU 2018-19, Marks 10

Answer

Validation condition for optimistic concurrency control : Refer Q. 5.8, Page 5–7B, Unit-5.

Effects of validation conditions on transaction in distributed system :

1. If the validation conditions are successful, then the transaction can commit.
2. If the validation conditions fail, then some form of conflict resolution must be used and the current transaction will be aborted.
3. Rule 1 and 2 test whether there is a overlapping between the objects of pair of transaction T_j and T_i.
4. Rule 3 ensures that no two transactions can overlap in update phase.
5. Due to restriction on write operations no dirty read can occurs.

Que 5.10. **Write short notes on timestamp ordering transaction management.**

AKTU 2015-16, Marks 05

Answer

1. In distributed transaction, each coordinator issue globally unique timestamps.
2. A globally unique transaction timestamp is issued to the client by the first coordinator accessed by a transaction.
3. The transaction timestamp is passed to the coordinator at each server whose objects perform an operation in the transaction.
4. The servers of distributed transactions are jointly responsible for ensuring that they are performed in a serially equivalent manner.
5. A timestamp consists of a pair < local timestamp, server-id>.
6. The agreed ordering of pairs of timestamps is based on a comparison in which the server-id part is less significant.

7. The same ordering of transactions can be achieved at all the servers even if their local clocks are not synchronized.

Que 5.11. **Explain strict two phase locking with its rules.**

Answer

Strict two phase locking : Refer Q. 5.5, Page 5–5B, Unit-5.

The rules for the use of locks in a strict two phase locking implementation are as follows :

1. When an operation accesses an object within a transaction :
   a. If the object is not already locked, it is locked and the operation proceeds.
   b. If the object has the conflicting lock set by another transaction, the transaction wait until it is unlocked.
   c. If the object has the non-conflicting lock set by another transaction, the lock is shared and the operation proceeds.
   d. If the object has already been locked in the same transaction, the lock will be promoted if necessary and the operation proceeds.
2. When a transaction is committed or aborted, the server unlocks all objects it locked for the transaction.

These rules ensure strictness because the locks are held until a transaction has either committed or aborted.

Que 5.12. **Explain multiversion timestamp ordering protocol.**

Answer

1. In multiversion timestamp ordering, a list of old committed versions as well as tentative versions is kept for each object.
2. This list represents the history of the values of the object.
3. Each version has a read timestamp recording the largest timestamp of any transaction that has read from it in addition to a write timestamp.
4. Whenever a write operation is accepted, it is directed to a tentative version with the write timestamp of the transaction.
5. Whenever a read operation is carried out it is directed to the version with the largest write timestamp less than the transaction timestamp.
6. If the transaction timestamp is larger than the read timestamp of the version being used, the read timestamp of the version is set to the transaction timestamp.
7. When a read arrives late, it can be allowed to read from an old committed version, so there is no need to abort the read operations.

8. In multiversion timestamp ordering, read operations are always permitted although they may have to wait for earlier transaction to complete (either commit or abort), which ensures that executions are recoverable.

Que 5.13. **What are the advantages and drawback of multiversion timestamp ordering in comparison with the basic timestamp ordering ?**

AKTU 2014-15, Marks 10

Answer

Advantages of multiversion timestamp ordering :

1. It allows more concurrency in distributed system.
2. Improved system responsiveness by providing multiple versions.
3. Reduces the probability of conflicts transaction.
4. Read request never fails and is never made to wait.

Disadvantages of multiversion timestamp ordering :

1. Reading of a data item also requires the updating of the read timestamp field resulting in two potential disk accesses, rather than one.
2. The conflicts between transactions are resolved through rollbacks, rather than through waits.
3. It require huge amount of storage for storing multiple versions of data objects.
4. It does not ensure recoverability and cascadelessness.

PART-3

Distributed Transaction : Flat and Nested Transaction.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 5.14. **Explain distributed transaction.**

Answer

1. A distributed transaction is a database transaction in which two or more network hosts are involved.
2. These hosts provide transactional resources, while the transaction manager is responsible for creating and managing a global transaction that encompasses all operations against such resources.

3. Distributed transactions, as any other transactions, must have all four ACID (Atomicity, Consistency, Isolation and Durability) properties.
4. For distributed transactions, each computer (or nodes) acts as a local transaction manager.
5. If the transaction works at several computers, the transaction managers communicate with various other transaction managers by means of superior or subordinate relationships, which are accurate only for a specific transaction.

Que 5.15. **Explain distributed transactions. Discuss the functionality of flat and nested distributed transactions with example.**

AKTU 2018-19, Marks 10

OR

Write short note on flat and nested transaction.

AKTU 2016-17, Marks 7.5

Answer

Distributed transaction : Refer Q. 5.14, Page 5–10B, Unit-5.

Functionality of flat with example :

1. In a flat transaction, a client makes requests to more than one server.
2. A flat client transaction completes each of its requests before going on to the next one. Therefore, each transaction accesses server objects sequentially.
3. For example, in the Fig. 5.15.1, transaction *T* is a flat transaction that invokes operations on objects in servers *X*, *Y* and *Z*.

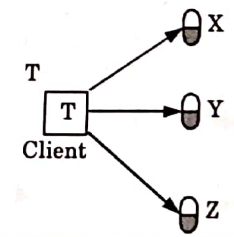


Fig. 5.15.1. Flat transactions.

Functionality of nested transaction with example : Refer Q. 5.2, Page 5–2B, Unit-5.

Que 5.16.

i. **What are the goals of distributed transaction ? Distinguish between flat and nested transaction along with its structure.**

ii. **Explain optimistic concurrency control.**

AKTU 2014-15, 2016-17; Marks 10

Answer

i. **Goals of distributed transaction :**

1. The goal of distributed transaction is to ensure that all of the objects managed by a server remain in a consistent state when they are accessed by multiple transactions and in the presence of server crashes.
2. To maintain ACID properties of transaction in distributed system.
3. To complete overall transaction occurring at different nodes.
4. To ensure the consistency of a set of shared data objects accessed by user at the time of failures and concurrent access.

Difference between flat and nested transaction :

S. No.	Flat transaction	Nested transaction
1.	A flat client transaction completes each of its requests before going on to the next one. Therefore, each transaction accesses server objects sequentially.	In a nested transaction, the top-level transaction can open subtransactions, and each subtransaction can open further subtransactions down to any depth of nesting.
2.	In the Fig. 5.16.1, transaction T is a flat transaction that invokes operation on objects in servers X, Y and Z.	In nested transaction as shown in Fig. 5.16.2, subtransactions at the same level can run concurrently, so T_1 and T_2 are concurrent, and as they invoke objects in different servers, they can run in parallel.

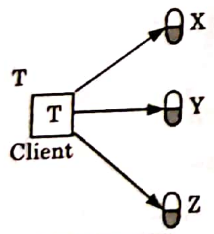


Fig. 5.16.1.

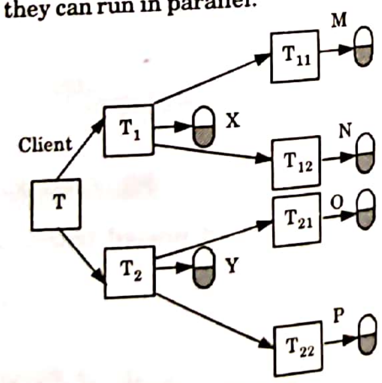


Fig. 5.16.2.

ii. **Optimistic concurrency control :** Refer Q. 5.7, Page 5–7B, Unit-5.

Que 5.17. **Draw a schematic diagram of the distributed transaction management model. Explain each component in brief.**

AKTU 2017-18, Marks 05

Answer

Transaction management model contain following three components :

1. **Transaction manager (TM) :**
   a. The transaction manager supervises the execution of a transaction.
   b. It intercepts and executes all the submitted transactions.
   c. TM interacts with the DM to carry out the execution of a transaction.
   d. TM assigns a timestamp to a transaction or issue requests to lock and unlock data objects on behalf of a user.
   e. TM acts as an interface between user and the database system.
2. **Scheduler :**
   a. Scheduler is used for enforcing concurrency control.
   b. It grants or releases locks on data objects as requested by a transaction.
3. **Data manager :**
   a. The data manager (DM) manages the database.
   b. It carries out the read-write requests issued by the TM on behalf of a transaction by operating them on the database.
   c. Thus, DM is an interface between scheduler and database.

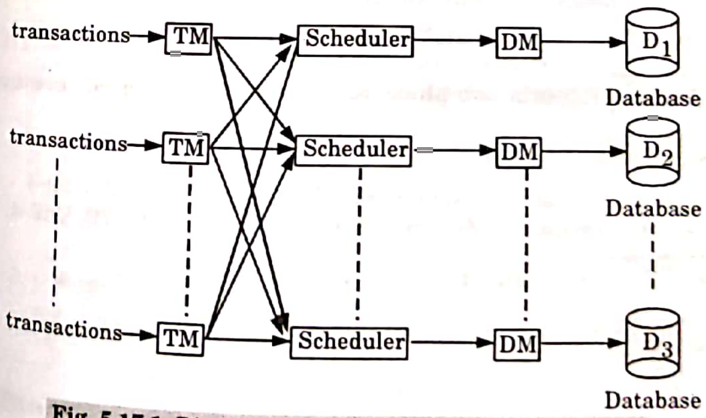


Fig. 5.17.1. Distributed transaction management model.

d. Execution of a transaction at the TM results in the execution of its actions at the DM.

e. So, the DM executes a stream of transaction actions, directed towards it by the TM.

PART-4

Atomic Commit Protocol, Concurrency Control in Distributed System.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 5.18. Write short note on atomic commit protocol.

Answer

1. Atomic commit protocol (ACP) is a protocol used by database manager to ensure that all the sub-transactions are consistently committed or aborted.
2. In this each server applies local concurrency control to its own object, which ensures that transaction are serialized locally as well as serialized globally.
3. When a distributed transaction comes to an end, either all the servers commit the transaction or abort the transaction.
4. There are two atomic commit protocol used in distributed database :
   a. Two phase commit protocol.
   b. Three phase commit protocol.

Que 5.19. Explain two phase and three phase commit protocol.

Answer

Two phase commit protocol : Refer Q. 4.15, Page 4–15B, Unit-4.
Three phase commit protocol : Refer Q. 4.17, Page 4–17B, Unit-4.

Que 5.20. Write short note on conflict resolution.

Answer

A conflict is resolved by taking one of the following actions :

1. **Wait :** The requesting transaction is made to wait until the conflicting transaction either completes or aborts.

2. **Restart :**
   a. Either the requesting transaction or the transaction it conflicts with is aborted and started afresh.
   b. Restarting is achieved by using one of the following primitives :
3. **Die :** The requesting transaction aborts and starts afresh.
4. **Wound :**
   a. The transaction in conflict with the requesting transaction is tagged as wounded and a message "wounded" is sent to all sites that the wounded transaction has visited.
   b. If the message is received before the wounded transaction has committed at a site, the concurrency control algorithm at that site initiates an abort of the wounded transaction, otherwise the message is ignored.
   c. If a wounded transaction is aborted, it is started again.
   d. The requesting transaction proceeds after the wounded transaction completes or aborts.

Que 5.21. What are the algorithms for conflict resolution in timestamp concurrency control ?

Answer

Following are the algorithms for conflict resolution in timestamps concurrency control :

1. **Wait-die algorithm :**
   a. The wait-die algorithm is a nonpreemptive algorithm because a requesting transaction never forces the transaction holding the requested data object to abort.
   b. Suppose requesting transaction T_1 is in conflict with a transaction T_2. If T_1 is older (*i.e.*, has a smaller timestamp), then T_1 waits, otherwise T_1 dies.
   c. The older transaction waits for the younger transaction if the younger has accessed the granule first.
   d. The younger transaction is aborted and restarted if it tries to access a granule after an older concurrent transaction.
2. **Wound-wait algorithm :**
   a. The wound-wait algorithm is a preemptive algorithm.
   b. Suppose a requesting transaction T_1 is in conflict with a transaction T_2. If T_1 is older, it wounds T_2, otherwise it watts.
   c. The older transaction preempts the younger by suspending it if the younger transaction tries to access a granule after an older concurrent transaction.

d. An older transaction will wait for a younger one to commit if the younger has accessed a granule that both want.

Que 5.22. **What are the advantages, problems and applications of optimistic concurrency control ?**

Answer

Advantages :

i. Optimistic concurrency control is very efficient when conflicts are rare. The occasional conflicts result in the transaction roll back.

ii. The rollback involves only the local copy of data. And thus no cascading rollback occurs.

Problems :

i. Conflicts are expensive to deal.

ii. Longer transactions are more likely to have conflicts and may be repeatedly rolled back because of conflicts with short transactions.

Applications :

i. Only suitable for environments where there are few conflicts and no long transactions.

ii. Acceptable for mostly read or query database systems that require very few update transactions.

Que 5.23. **Explain the schemes which conflicts in obtaining local locks ?**

Answer

Schemes which conflicts in obtaining local locks :

1. **Write-locks-all, read-locks-one :**
   a. In this scheme exclusive locks are acquired on all copies, while shared locks are acquired only on one arbitrary copy.
   b. A conflict is always detected, because a shared-exclusive conflict is detected at the site where the shared lock is required and exclusive-exclusive conflicts are detected at all sites.
2. **Majority locking :**
   a. Both shared and exclusive locks are requested at a majority of the copies of the data item.
   b. If two transactions are required to lock the same item, there is at least one copy of it where the conflict is discovered.
3. **Primary copy locking :** In primary copy locking, one copy of each data item is assigned the primary copy and all locks must be required at this copy so that conflicts are discovered at the site where the primary copy resides.

Que 5.24. **Explain conservative timestamp method in distributed system.**

Answer

The conservative timestamp method is based on the following rules :

1. Each transaction is executed at one site only and does not activate remote programs. It can only issue read or write requests to remote sites.
2. A site i must receive all the read requests from a different site j in timestamp order. Similarly, a site i must receive all the write requests from a different site j in timestamp order.
3. Assume that a site i has at least one buffered read and one buffered write operation from each other site of the network.
   a. **For a read operation R that arrives at site i :** If there is some write operation W buffered at site i such that $TS(R) > TS(W)$, then R is buffered until these writes are executed, otherwise R is executed.
   b. **For a write operation W that arrives at site i :** If there is some read operation R buffered at site i such that $TS(W) > TS(R)$, or there is some write operation W' buffered at site i such that $TS(W) > TS(W')$, then W is buffered until these operations are executed, otherwise W is executed.

PART-5

Distributed Deadlocks, Transaction Recovery.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 5.25. **Write short notes on wait for graph with example of distributed transaction.** **AKTU 2015-16, Marks 05**

Answer

1. A Wait-For Graph (WFG) is a graph where
   a. Each node represents a process.
   b. An edge, $P_i \rightarrow P_j$ means that P_i is blocked waiting for P_j to release a resource.

2. A system is deadlocked if and only if there is a directed cycle in the WFG.
3. In Distributed Database Systems (DDBS), users access the data objects of the database by executing transactions.
4. The data objects of a database can be viewed as resources that are acquired (through locking) and released (through unlocking) by transactions.
5. In DDBS a wait for graph is referred to as a transaction-wait-for graph (TWF graph).
6. In a TWF graph, nodes are transactions and there is a directed edge from node T_1 to node T_2 if T_1 is blocked and is waiting for T_2 to release some resource.
7. A system is deadlocked if and only if there is a directed cycle or a knot in its TWF graph.

Que 5.26. **What is phantom deadlock ? Describe the conditions for the occurrence of phantom deadlock.**

Answer

Phantom deadlock :

1. A deadlock that is 'detected' but is not really a deadlock is called a phantom deadlock.
2. In distributed deadlock detection, information about wait-for relationships between transactions is transmitted from one server to another.
3. If there is a deadlock, the necessary information will eventually be collected in one place and a cycle will be detected.
4. As this procedure will take some time, there is a chance that one of the transactions that hold a lock will meanwhile have released it, in that case the deadlock will no longer exist.
5. **For example :**
   a. Consider the case of a global deadlock detector that receives local wait-for graphs from servers X and Y, as shown in Fig. 5.26.1.
   b. Suppose that transaction U then releases an object at server X and requests the one held by V at server Y.
   c. Suppose also that the global detector receives server Y's local graph before server X's.
   d. In this case, it would detect a cycle $T \rightarrow U \rightarrow V \rightarrow T$, although the edge $T \rightarrow U$ no longer exists.

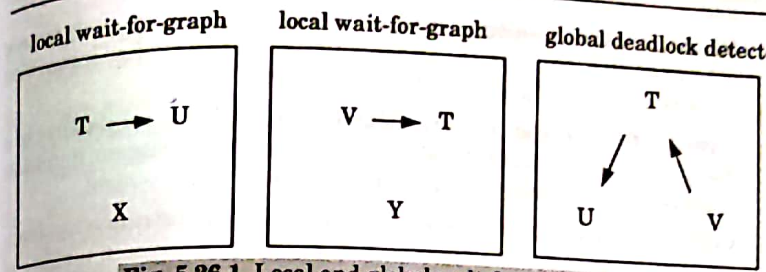


Fig. 5.26.1. Local and global wait-for-graphs.

6. A phantom deadlock could be detected if a waiting transaction in a deadlock cycle aborts during the deadlock detection procedure.
7. For example, if there is a cycle $T \rightarrow U \rightarrow V \rightarrow T$ and U aborts after the information concerning U has been collected, then the cycle has been broken already and there is no deadlock.

Necessary conditions for the occurrence of phantom deadlock are :

1. Presence of delay between two processes.
2. Notification of false global states by the communicating processes.
   a. If P_1 and P_2 are two processes executing on different nodes and a third node N_3 is testing for deadlock. Process P_1 holds resource R_1 and process P_2 holds resource R_2.
   b. Process P_1 releases resource R_1 and sends message M_1 to node N_3. Process P_1 then requests resource R_2 and sends message M_2 to node N_3.
   c. Process P_2 releases resource R_2 and sends message M_3 to node N_3. Finally, P_2 requests R_1 and sends message M_4 to node N_3.
   d. If there is network latency or delay, message M_2 and M_4 arrive at node N_3 before messages M_1 and M_3.
   e. Node N_3, which is testing deadlock, will detect a deadlock that does not exist. This results in occurrence of phantom deadlock.

Que 5.27. **Briefly explain the objectives of distributed transaction management.**

Answer

Following are the objectives of distributed transaction management :

1. **CPU and main memory utilization should be improved :** Most of the typical database applications spend much of their time waiting for I/O operations rather than on computations. To improve CPU and main memory utilization, a transaction manager should adopt specialized techniques.

2. **Response time should be minimized :** To improve the performance of transaction executions, the response time of each individual transaction must be considered and should be minimized.
3. **Availability should be maximized :** Although the availability in a distributed system is better than that in a centralized system, it must be maximized for transaction recovery and concurrency control.
4. **Communication cost should be minimized :** In distributed systems, an additional communication cost is incurred, because a number of message transfers are required between sites to control the execution of a global transaction. Preventative measures should be adopted by the transaction manager to minimize the communication cost.

Que 5.28. **Write short note on logging.**

Answer

1. In the logging technique, the recovery file represents a log containing the history of all the transactions performed by a server.
2. The history consists of values of objects, transaction status entries and intentions lists of transactions.
3. The order of the entries in the log reflects the order in which transactions have prepared, committed and aborted at that server.
4. During the normal operation of a server, its recovery manager is called whenever a transaction prepares to commit, commits or aborts a transaction.
5. When the server is prepared to commit a transaction, the recovery manager appends all the objects in its intentions list to the recovery file, followed by the current status of that transaction (prepared) together with its intentions list.
6. When a transaction is eventually committed or aborted, the recovery manager appends the corresponding status of the transaction to its recovery file.

Que 5.29. **Write short note on shadow versions.**

Answer

1. The shadow versions technique is an alternative way to organize a recovery file.
2. It uses a map to locate versions of the server's object in a file called a version store.
3. The map associates the identifiers of the server's versions in the version store.

4. The versions written by each transaction are shadows of the previous committed versions.
5. When a transaction is prepared to commit, any of the objects changed by the transaction are appended to the version store, leaving the corresponding committed versions unchanged.
6. When a transaction commits, a new map is made by copying the old map and entering the positions of the shadow versions.
7. To restore the objects when a server is replaced after a crash, its recovery manager reads the map and uses the information in the map to locate the objects in the version store.
8. The shadow version method provides faster recovery than logging because the positions of the current committed objects are recorded in the map, whereas recovery from a log requires searching throughout the log for objects.

PART-6

Replication : System Modal and Group Communication, Fault, Tolerant Services, Highly Available Services and Transaction With Replicated Data.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 5.30. **What is replication and replica manager ? Give the architectural model for replicated data.**

AKTU 2014-15, Marks 05

Answer

Replication :

1. Replication is the process of storing copies of data at more than one node.
2. Replication is a key for providing high availability and fault tolerance in distributed systems.
3. High availability means that all the users can access data after failure of one or more of the servers.
4. Fault tolerance is the property that enables a system to continue operating properly in the event of the failure.

Replica manager :

1. Replica manager is a subsystem that is responsible for managing the synchronization of replicas.
2. Replica refers to a single copy of the data in a system that employs replication.
3. Replica managers held various replicas and perform operations upon them.
4. A replica manager acts as a server in client-server environment.
5. A collection of replica managers provides service to client.

Architectural model for replicated data :

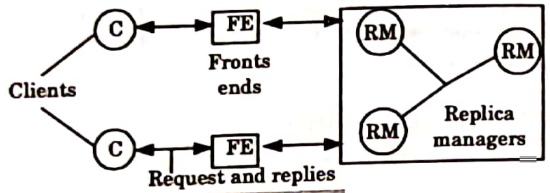


Fig. 5.30.1.

1. In this model, replicas are held by distinct replica managers.
2. This general model may be applied in a client-server environment, in which case a replica manager acts as a server.
3. It may be applied to an application. The application processes in that case acts as both clients and replica manager.
4. The general model of replica management is shown in Fig. 5.30.1.
5. A collection of replica managers provides service to clients.
6. The client sees a service that gives them access to objects which are replicated at the replica managers.
7. Each clients request a series of operations upon one or more of the objects.
8. An operation involves a combination of reads and updates to objects. Requested operations that involve no updates are called read-only requests requested operations that update an object are called update requests.
9. Each client's requests are handled by a component called front end. The role of the front end is to communicate by message passing with one or more of the replica manager.

Que 5.31. **Explain the following :**

i. **Gossip architecture**

ii. **Quorum consensus methods :**

AKTU 2014-15, Marks 10

OR

Explain the processing of queries and update operation in gossip services.

Answer

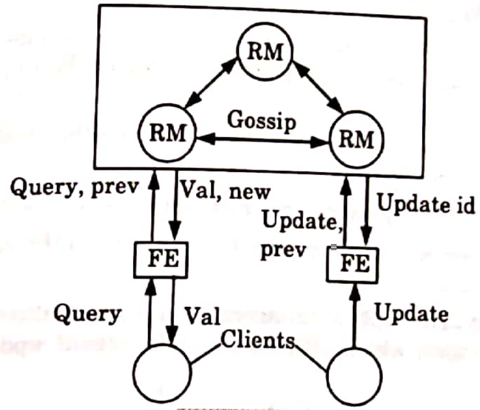


Fig. 5.31.1.

i. **Gossip architecture :**

1. Gossip architecture is a framework for implementing highly available services by replicating data close to the points where groups of clients need it.
2. Here the replica managers exchange 'gossip' messages periodically in order to convey the updates received from clients.
3. A gossip service provides two basic types of operation : queries (read-only operations) and updates (modify but do not read the state).
4. A key feature is that front ends send queries and updates to any replica manager that is available and can provide reasonable response times.
5. The system makes two guarantees :
   a. Each client obtains a consistent service over time.
   b. Relaxed consistency between replicas.

Processing of queries and update operations in gossip service :

a. **Request :**
   i. The front end normally sends requests to only a single replica manager at a time.
   ii. However, a front end will communicate with a different replica manager when the one it normally uses fails or becomes unreachable, or if the normal manager is heavily loaded.
   iii. Front ends, and thus clients, may be blocked on query operations.

iv. The default arrangement for update operation is to return to the client as soon as the operation has been passed to the front end; the front end then propagates the operation in the background.

b. **Update response :** If the request is an update then the replica manager replies as soon as it has received the update.

c. **Coordination :**

i. The replica manager that receives a request does not process it until it can apply the request according to the required ordering constraints.

ii. This may involve receiving updates from other replica managers, in gossip messages.

d. **Execution :** The replica manager executes the request.

e. **Query response :** If the request is a query then the replica manager replies at this point.

f. **Agreement :** The replica managers update one another by exchanging gossip messages, which contain the most recent updates they have received.

ii. **Quorum consensus methods :**

1. A quorum is a subgroup of replica managers whose size gives it the right to carry out operations.
2. In this scheme, an update operation on a logical object may be completed successfully by a subgroup of its group of replica managers.
3. The other members of the group will therefore have out-of-date copies of the object.
4. Versions numbers may be used to determine whether copies are up-to-date.
5. Each copy of an object has a version number, but only the copies that are up-to-date have the current version number.

Que 5.32. **Discuss the following in terms of distributed system :**

i. Sequential consistency

ii. Highly available services **AKTU 2018-19, Marks 10**

OR

Write short note on highly available services and sequentially consistency. **AKTU 2015-16, Marks 10**

Answer

i. **Sequential consistency :**

1. Sequential consistency is a strong safety property for concurrent systems.

2. A system is sequentially consistent if the result of any execution of the operations of all the processors is same as if they were executed in a sequential order, and the operations of each individual processor appear in this sequence in the order specified by its program.
3. It implies that operations appear to take place in some total order, and that order is consistent with the order of operations on each individual process.
4. Sequential consistency cannot be totally available in the event of a network partition, some or all nodes will be unable to make progress.

ii. **Highly available services :**

1. Availability of service means the percentage of time that a service is up.
2. Highly available service is the service whose availability is close to 100% with reasonable response time.
3. It may not conform to sequential consistency.
4. Gossip architecture is a framework for implementing highly available services by replicating data close to the points where groups of clients need it.

Que 5.33. **Describe the architecture of replicated transactions.**

Answer

Architectures for replicated transactions :

1. In this architecture, we assume that a front end sends client requests to one of the group of replica managers of a logical object.
2. In the primary copy approach, all front ends communicate with a distinguished 'primary' replica manager to perform an operation, and that replica manager keeps the backups up to date.
3. Front ends may communicate with any replica manager to perform an operation.
4. The replica manager that receives a request to perform an operation on a particular object local state responsible for getting the cooperation of the other replica managers in the group that have copies of that object.
5. Different replications schemes have different rules as to how many of the replica managers in a group are required for the successful completion of an operation.
6. In the read-one write-all scheme, a read request can be performed by a single replica manager, whereas a write request must be performed by all the replica managers in the group, as shown in Fig. 5.33.1.
7. Quorum consensus schemes are designed to reduce the number of replica managers that must perform update operations, but at the expense of increasing the number of replica managers required to perform read-only operations.

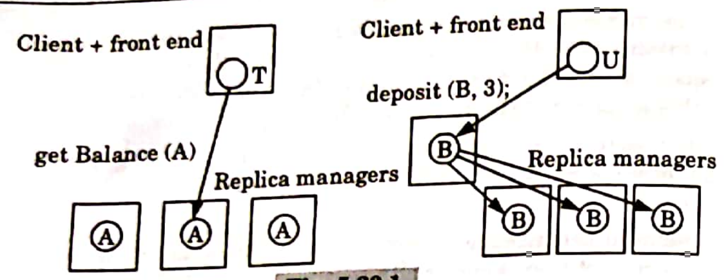


Fig. 5.33.1.

Que 5.34. **Explain the group communication in replicated data.**

Answer

Group communication :

1. Multicast communication is also known as group communication because process groups are the destinations of multicast messages.
2. Groups are useful for managing replicated data and in other systems where processes cooperate towards a common goal by receiving and processing the same set of multicast messages.
3. They are also useful where the group members independently consume one or more common streams of messages, such as messages carrying events to which the processes react independently.
4. A full implementation of group communication incorporates a group membership service to manage the dynamic membership of groups, in addition to multicast communication.
5. Multicast and group membership management are strongly interrelated.
6. Fig. 5.33.1 shows an open group, in which a process outside the group sends to the group without knowing the group's membership.

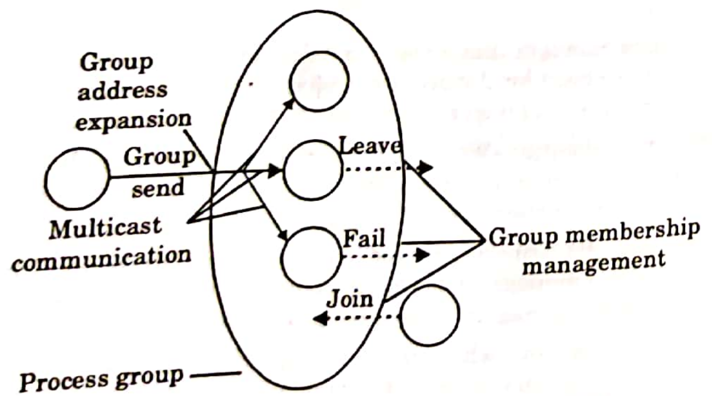


Fig. 5.34.1. Services provided for process groups.

7. The group communication services have to manage changes in the group's membership while multicasts take place concurrently.

Que 5.35. **Explain fault-tolerant services.**

Answer

1. Fault tolerant services are obtained by using replication.
2. By using multiple independent server replicas each managing replicated data help in designing a service which exhibits graceful degradation during partial failure and improve overall server performance.

Following are two main fault-tolerance service models :

1. **Passive (primary-backup) replication :**
   a. In a passive model of replication for fault tolerance, there is at any time a single primary replica manager and one or more secondary replica managers *i.e.*, 'backups' or 'slaves'.
   b. In the pure form of the model, front ends communicate only with the primary replica manager to obtain the service.
   c. The primary replica manager executes the operations and sends copies of the updated data to the backups.
   d. If the primary fails, one of the backups is promoted to act as the primary.
   e. The sequence of events when a client requests an operation to be performed is as follows :
      i. **Request :** The front end issues the request, containing a unique identifier, to the primary replica manager.
      ii. **Coordination :** The primary takes each request atomically, in the order in which it receives it. It checks the unique identifier, in case it has already executed the request and if so it simply re-sends the response.
      iii. **Execution :** The primary executes the request and stores the response.
      iv. **Agreement :** If the request is an update then the primary sends the updated state, the response and the unique identifier to all the backups. The backups send an acknowledgement.
      v. **Response :** The primary responds to the front end, which hands the response back to the client.

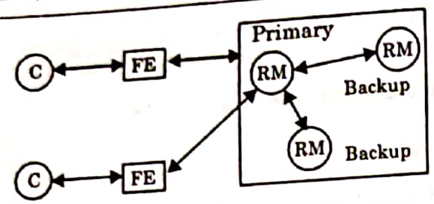


Fig. 5.35.1. The passive (primary-backup) model for fault tolerance.

2. **Active replication :**
   a. In the active model of replication for fault tolerance, the replica managers are state machines that play equivalent roles and are organized as a group.
   b. Front ends multicast their requests to the group of replica managers and all the replica managers process the request independently but identically and reply.
   c. If any replica manager crashes, then this need have no impact upon the performance of the service, since the remaining replica managers continue to respond in the normal way.
   d. Under active replication, the sequence of events when a client requests and operation to be performed is as follows :
      i. **Request :** The front end attaches a unique identifier to the request and multicasts it to the group of replica managers, using a totally ordered, reliable multicast primitive.
      ii. **Coordination :** The group communication system delivers the request to every correct replica manager in the same (total) order.

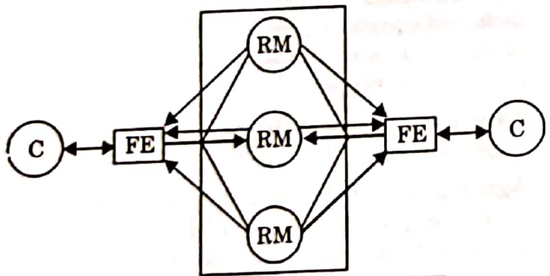


Fig. 5.35.2. Active replication.

      iii. **Execution :** Every replica manager executes the request.
      iv. **Agreement :** No agreement phase is needed, because of the multicast delivery semantics.
      v. **Response :** Each replica manager sends its response to the front end.

Que 5.36. **What are stub and skeleton and why are they needed in remote procedure calls ?**

AKTU 2017-18, Marks 10

Answer

Stub is a function which converts the function call into a network response and a network response into a function return.

Skeleton converts requests into function calls and function returns into network replies.

Need of stub and skeleton in Remote Procedure Call (RPC) :

1. RPC allows a local computer (client) to remotely call procedures on a different computer (server).
2. The client and server use different address spaces, so parameters used in a function (procedure) call have to be converted, otherwise the values of those parameters could not be used, because pointers to parameters in one computer's memory would point to different data on the other computer.
3. The client and server may also use different data representations, even for simple parameters.
4. Stubs perform the conversion of the parameters, so a remote procedure call looks like a local function call for the remote computer.
5. Stub libraries must be installed on both the client and server side.
6. A client stub is responsible for conversion of parameters used in a function call and deconversion of results passed from the server after execution of the function.
7. A server skeleton, the stub on the server side, is responsible for deconversion of parameters passed by the client and conversion of the results after the execution of the function.

Que 5.37. **What is the purpose of an Interface Definition Language ? Why does CORBA not just use the Java interface construct ?**

AKTU 2017-18, Marks 10

Answer

Purpose of Interface Definition Language (IDL) are :

1. To describe software component's Application Programming Interface (API).
2. To describe an interface in a language-independent way, enabling communication between software components that do not share one language, for example, between those written in C++ and those written in Java.

3. IDLs act as a bridge between the two different systems. For example, in case of RPC software the machines at either end of the link may be using different operating systems and computer languages.

Reasons for CORBA not just using the Java interface construct :

1. CORBA builds on the idea that all objects are remotable.
2. Objects can be accessed via their object reference. So, we pass object references in a call, not the object itself.
3. If we create our remote applications starting from CORBA IDL, then it will not pass objects by value.
4. However, we face the following problem :
   a. Java interfaces can define method invocations that include parameters that reference local objects.
   b. These references to local Java objects are only useful within a single virtual machine. So the ORB must copy these objects across Java virtual machines.

Que 5.38. **How does a server know that one of his remote objects provided by him is no longer used by clients and can be collected ? How does Java RMI handle this problem and what alternatives are there ?**

AKTU 2017-18, Marks 10

Answer

1. When a client first receives a reference to a remote object, a "referenced" message is sent to the server that is exporting the object.
2. Every subsequent reference within the client's local machine causes a reference counter to be incremented.
3. As a local reference is finalized, the reference count is decremented, and once the count goes to zero, an 'unreferenced' message is sent to the server.
4. Once the server has no more live references to an object and there are no local references, it is free to be finalized and garbage collected.
5. This condition tells a server that a remote object provided by him is no longer used by clients and can be collected.

RMI uses its distributed garbage collection feature to collect remote server objects that are no longer referenced by any client in the network.

Que 5.39. **De-activation is a technology used to preserve server resources where a server which provides remote objects to clients can de-activate those remote objects. Clients should not know about this. What must the server do to avoid surprises for the clients ?**

AKTU 2017-18, Marks 10

Answer

While using de-activation technologies to avoid surprises for the clients, server must do the following :

1. It must give the client permission to recreate (activate) the object again.
2. The remote objects must be available for a long period without any predetermined expiration time out.
3. The remote objects state must not be lost between individual invocations and must be available to all clients.
4. May provide remote objects whose lifetime is controlled by clients.

VERY IMPORTANT QUESTIONS

Following questions are very important. These questions may be asked in your SESSIONALS as well as UNIVERSITY EXAMINATION.

Q. 1. Explain transaction and its properties ?

Ans. Refer Q. 5.1.

Q. 2. Discuss 2PL and strict 2PL in context of distributed system.

Ans. Refer Q. 5.5.

Q. 3. Explain optimistic concurrency control.

Ans. Refer Q. 5.7.

Q. 4. Discuss the optimistic methods for distributed concurrency control. What are the different validation conditions for optimistic concurrency control ? Explain.

Ans. Refer Q. 5.8.

Q. 5. Write short notes on timestamp ordering transaction management.

Ans. Refer Q. 5.10.

Q. 6. What are the advantages and drawback of multiversion timestamp ordering in comparison with the basic timestamp ordering ?

Ans. Refer Q. 5.13.

Q. 7. Explain distributed transactions. Discuss the functionality of flat and nested distributed transactions with example.

Ans. Refer Q. 5.15.

Q. 8. Write short notes on wait for graph with example of distributed transaction.

Ans. Refer Q. 5.25.

Q. 9. What is replication and replica manager ? Give the architectural model for replicated data.

Ans. Refer Q. 5.30.

Q. 10. Discuss the following in terms of distributed system

i. Sequential consistency

ii. Highly available services

Ans. Refer Q. 5.32.

☺☺☺

1 UNIT

Characterization of Distributed System (2 Marks Questions)

1.1. Discuss the role of file system in distributed system.

AKTU 2015-16, Marks 02

Ans. **Following are the role of file system in distributed system :**

1. To manage files and folder across multiple machine.
2. To share information and files among users on a network in a controlled and authorized way.

1.2. List the goals of distributed systems.

AKTU 2015-16, Marks 02

Ans. **Goals of distributed system are :**

i. Improved system performance
ii. Resource sharing
iii. Improved reliability and availability
iv. Modular expendability

1.3. List out the main challenges of distributed systems.

AKTU 2016-17, Marks 02

Ans. **Following are the main challenges of distributed system :**

1. Security
2. Scalability
3. Failure handling
4. Concurrency
5. Transparency

1.4. What are logical clocks ? Why does a logical clock need to be implemented in distributed systems ?

AKTU 2016-17, Marks 02

Ans. A logical clock is a mechanism for capturing chronological and causal relationships in a distributed system. Distributed systems have no physically synchronous global clock, so a logical clock allows global ordering on events from different processes.

1.5. List out some issues in distributed file systems.

AKTU 2016-17, Marks 02

Ans. **Issues in distributed file system :**

1. Naming and name resolution
2. Cache consistency

3. Availability
4. Scalability
5. Semantics

1.6. What are the web challenges involved in distributed system ?

AKTU 2018-19, Marks 02

Ans. **Following are the web challenges involved in distributed system :**

1. Naming
2. Access control
3. Security
4. Performance management
5. Availability

1.7. Explain system model.

AKTU 2018-19, Marks 02

Ans. System model describe common properties and design choices for distributed system in a single descriptive model. System models are of two types :

i. Architectural models
ii. Fundamental models

1.8. Why would you design a system as a distributed system ? List some advantages of distributed systems.

AKTU 2017-18, Marks 02

Ans.

1. A distributed system is a group of individual computers working together and operating concurrently. This allows the whole system to keep working even if one or some of the computer fails.
2. The distributed system is scalable *i.e.*, the system can easily be altered to accommodate changes in the number of users and resources. Thus, a distributed system can be much larger and more powerful than combinations of stand-alone systems.
3. Due to the above reasons we would design a system as a distributed system.

Advantages of distributed systems :

i. Shareability
ii. Expandability
iii. Improved performance
iv. Improved reliability
v. Flexibility

1.9. Name the algorithm for causal ordering of messages.

Ans. **Algorithms for causal ordering of messages are :**

i. Birman-Schiper-Stephenson algorithm
ii. Schiper-Eggli-Sandoz algorithm

1.10. What is causal ordering of message ?

Ans. The casual ordering of message deals with the concept of maintaining same casual relationship that holds among "message send" event with corresponding "message receive" event.

1.11. Define global state.

Ans. The global state of a distributed computation is the set of local states of all individual processes involved in the computation and the state of the communication channels. The global state of the system is a collection of the local states of a processing system.

$$GS = \{LS_1, LS_2, LS_3 \ldots\ldots LS_N\}$$

where N is the number of sites in the system.

1.12. List three properties of distributed systems.

AKTU 2017-18, Marks 02

Ans. **Properties of distributed system :**

1. Resource sharing
2. Scalability
3. Fault tolerance

1.13. What is the advantage if your server side processing uses threads instead of a single process ?

AKTU 2017-18, Marks 02

Ans. **Advantages :**

1. Threads are created and destroyed much faster than process.
2. It is faster to switch between threads than to switch between processes.
3. Threads share data easily.

1.14. What is a proxy ? Give an example of where a proxy can be used.

AKTU 2017-18, Marks 02

Ans. A proxy (or proxy server) is a server that acts as a gateway between the user's computer and the internet. A proxy server verifies and forwards incoming client requests to other servers for further communication. Proxy servers can be used by a web browser or network in order to enhance privacy. For example, requests for a website made through a proxy may help to hide the client's IP from the web server.

1.15. What is the purpose of a firewall ?

AKTU 2017-18, Marks 02

Ans. **Purpose of firewall :**

1. To protect computers and private networks from internet threats.
2. Packet filtering.
3. To record all the information from packets that passes through it.

1.16. What were the reasons that middleware moved from distributed objects to distributed components ?

AKTU 2017-18, Marks 02

Ans. Following are the reasons that middleware moved from distributed objects to distributed components :

1. **Implicit dependencies :** The dependencies the object has on other objects in the distributed configuration.
2. **Interaction with the middleware :** Clean separation is needed between code related to operation in a middleware and code associated with the application.
3. **Lack of separation of distribution concerns :** The complexities of dealing with the distributed system services should be hidden wherever possible from the programmer.
4. **No support for deployment :** Middleware platforms should provide intrinsic support for deployment so that distributed software can be installed in the same way as software for a single machine.

1.17. Name two mechanisms that can be used to ensure performance in distributed systems.

AKTU 2017-18, Marks 02

Ans. Two mechanisms that can be used to ensure performance in distributed systems are :

1. Fault tolerance
2. Availability

1.18. What are the differences between a URL, URI and URN ?

AKTU 2017-18, Marks 02

Ans.

S. No.	URL	URI	URN
1.	A Uniform Resource Locator (URL) is a subset of the Uniform Resource Identifier (URI) that specifies where an identified resource is available and the mechanism for retrieving it.	Uniform Resource Identifier (URI) is a string of characters used to identify a name or a resource on the internet.	A Uniform Resource Name (URN) is a Uniform Resource Identifier (URI) that uses the URN scheme, and does not imply availability of the identified resource.
2.	A URL defines how the resource can be obtained.	A URI identifies a resource either by location, or a name, or both.	The URN defines something's identity, while the URL provides a location.

1.19. What is consistent cut and inconsistent cut ?

AKTU 2015-16, Marks 02

Ans. **Consistent cut :** A cut is called consistent, if for each event that it contains, it also includes all events all events causally ordered before it. Let a, b be two events in a distributed system. Then

$$(a \in \text{consistent cut } C) \wedge (B < a) \Rightarrow b \in C$$

Thus, for a message m, if the state following receive (m) belongs to a consistent cut, then the state following send (m) also must belong to cut.

Inconsistent cut : A cut C is inconsistent iff any of the following three basic conditions is true :

1. e is sending event of a message and e' is the receipt event of the same message $\wedge e' \in C \wedge e \notin C$.
2. e and e' are on events $\wedge e \leftrightarrow e' \wedge e' \in C \wedge e \notin C$.
3. e and e' are off events $\wedge e \leftrightarrow e' \wedge e' \in C \wedge e \notin C$.

1.20. What is termination detection problem ?

AKTU 2015-16, Marks 02

Ans. The termination detection problem involves detecting whether an ongoing distributed computation has finished all its activities. The termination detection problem arises when a distributed computation terminates implicitly, that is, once the computation finishes all its activities, no single process knows about the termination. Therefore a separate algorithm has to be run to detect termination of the computation.

1.21. Explain desirable features of a good message passing system.

AKTU 2015-16, Marks 02

Ans. **Following are the desirable features :**

i. Simplicity
ii. Uniform semantics
iii. Efficiency
iv. Atomicity
v. Ordered delivery
vi. Reliability
vii. Flexibility
viii. Security

☺☺☺

2 UNIT Distributed Mutual Exclusion (2 Marks Questions)

2.1. What do you mean by mutual exclusion in distributed system ? What are the requirements of a good mutual exclusion algorithm ? AKTU 2016-17, Marks 02

Ans. Mutual exclusion :

1. Mutual exclusion is a problem that arises if the process relies on a common resource that can be used only by one process at a time.
2. Concurrent access to shared resources is prevented.
3. Mutual exclusion algorithm guarantees that only one request accesses the critical section (CS) at a time.

Requirements of good mutual exclusion algorithm :

1. **Freedom from deadlocks :** Two or more sites should not endlessly wait for messages that will never arrive.
2. **Freedom from starvation :** A site should not be forced to wait indefinitely to execute CS *i.e.*, every requesting site should get an opportunity to execute CS in a finite time.
3. **Fairness :** Fairness dictates that requests must be executed in the order in which they arrive in the system.
4. **Fault tolerance :** A mutual exclusion algorithm is fault-tolerant if in the wake of a failure, it can recognize itself so that it continues to function without any (prolonged) disruptions.

2.2. Define deadlock detection in distributed systems. AKTU 2016-17, Marks 02

Ans. In deadlock detection, the system does not make any attempt to prevent deadlock but allows processes to request resources and wait for each other in uncontrolled manner.

2.3. What is distributed deadlock ? AKTU 2018-19, Marks 02

Ans. Deadlock is a situation in which a set of processes are blocked because each process is holding a resources and waiting for another resources acquired by some other process. The detection of deadlocks in a distributed DBMS is more complicated, because it involves several different sites. Thus, in a distributed DBMS it is necessary to draw a global wait-for graph (GWFG) for the entire system to detect a deadlock situation.

2.4. Explain token based algorithm. AKTU 2018-19, Marks 02

Ans.

1. In the token based algorithm, a unique token is shared among all sites. If sites possess the token then it is allowed to enter its CS (Critical Section).
2. Token based algorithms use sequence numbers instead of timestamps.
3. Every request for the token contains a sequence number and the sequence number of sites advances independently. A site increment its sequence number counter every time when it makes a request for the token.

2.5. What are the necessary conditions for deadlock ?

Ans. Necessary conditions for deadlock :

i. Mutual exclusion
ii. Hold and wait
iii. No preemption
iv. Circular wait

2.6. Define WFG.

Ans. WFG is wait-for graph. In distributed systems, the state of the system can be modeled by directed graph, called a wait-for graph (WFG). A system is deadlocked if and only if there exists a directed cycle or knot in the WFG.

2.7. What are the different classes of distributed deadlock detection algorithms ?

Ans. Following are different classes of distributed deadlock detection algorithms :

i. Path-pushing algorithm
ii. Edge-chasing algorithm
iii. Diffusion computation
iv. Global state detection

2.8. Explain edge chasing algorithm.

Ans. In edge chasing algorithms, special messages called the probes are circulated along the edges of wait-for graph to detect the cycle. When a blocked process receives the probe, it propagates the probe along its outgoing edges in wait-for graph.

2.9. Write a short note on hierarchical deadlock detection algorithm.

Ans. In this, sites are logically arranged in hierarchical fashion, and a site is responsible for detecting deadlocks involving only its children sites. It is of two types :

i. The Menasce-Muntz algorithm
ii. The Ho-Ramamoorthy algorithm

2.10. What are the types of token based algorithm ?

Ans. **Types of token based algorithm are :**

i. Suzuki-Kasami's broadcast algorithm.
ii. Singhal's algorithm

2.11. What are the types of non-token based algorithm ?

Ans. **Types of non-token based algorithms are :**

i. Lamport's algorithm
ii. Ricart-Agarwala algorithm
iii. Maekawa's algorithm

2.12. What are the performance matrices used to measure performance of mutual exclusion algorithm ?

Ans. **Performance matrices to measure performance of mutual exclusion algorithm are :**

i. Response time
ii. Synchronization delay
iii. Number of messages per critical section
iv. System throughput

2.13. Write are the design issues of Suzuki-Kasami algorithm ?

Ans. **Design issues of Suzuki-Kasami algorithm are :**

1. It distinguishes between outdated request messages and current request messages.
2. It determines which site has an outdated request for the critical section.

☺☺☺

3 UNIT

Agreement Protocols (2 Marks Questions)

3.1. What are distributed shared memory design issues ?

AKTU 2015-16, Marks 02

Ans. **Design issues in distributed shared memory are :**

i. Granularity
ii. Structure of shared memory space
iii. Memory coherence
iv. Data location and access
v. Block replacement policy
vi. Thrashing
vii. Heterogeneity

3.2. Why clocks need to be synchronized ?

AKTU 2015-16, Marks 02

Ans. In distributed systems, there is no global clock or common memory. Each processor has its own internal clock and these clocks can easily drift apart by several seconds per day, accumulating significant errors over time. This causes serious problem to applications that depend on synchronization of clock. So, to fix clock drift, clocks needs to be synchronized.

3.3. What do you mean by agreement protocol ?

AKTU 2016-17, 2018-19; Marks 02

Ans.

1. Process of sending and reaching the agreement to all sites is called agreement protocol.
2. In distributed system, the agreement protocols are very much useful for error free communication among various sites.
3. In distributed system, the chances of the faulty processors are more. The faulty processor may lead to wrong message communication, no response for a message etc.

4. The agreement protocols allow the non-faulty processors to reach a common agreement in the distributed system, whether there are other processors which are faulty or not.
5. The common agreement among the processors is taken through the agreement protocol.

3.4. State Byzantine agreement problem.

AKTU 2016-17, Marks 02

Ans.

1. In the Byzantine agreement problem, an arbitrarily chosen processor, called the source processor, broadcasts its initial value to all other processors.
2. A solution to the Byzantine agreement problem should meet the following objectives :
   i. **Agreement :** All non-faulty processors agree on the same value.
   ii. **Validity :** If the source processor is non-faulty, then the common agreed upon value by all non-faulty processors should be the initial value of the source.
   iii. **Termination :** Each non-faulty processor must eventually decide on a value.

3.5. Explain the concept of shared memory.

AKTU 2018-19, Marks 02

Ans.

1. Distributed shared memory (DSM) is a form of memory architecture where the (physically separate) memories can be addressed as one (logically shared) address space.
2. The shared memory model provides a virtual address space shared between all nodes.
3. DSM is primarily a tool for any distributed application in which individual shared data items can be accessed directly.

3.6. What are the various aspects for recognizing the agreement protocol ?

Ans. **Various aspects for recognizing the agreement protocols are :**

i. Synchronous and Asynchronous computations
ii. Model of processor failures
iii. Authenticated and non-authenticated messages
iv. Performance aspects

3.7. Give the two conditions that a clock synchronization algorithm should satisfy.

Ans. **Following are the two conditions that a clock synchronization algorithm should satisfy :**

i. At any time, the values of the clocks of all non-faulty processes must be approximately equal.
ii. There is a small bound on the amount by which the clock of a non-faulty process is changed during each resynchronization.

3.8. What are the services offered by a distributed file system ?

Ans. **Services offered by a distributed file system are :**

i. Storage service
ii. Flat-file service
iii. Name service

3.9. What are the requirements of a distributed file system ?

Ans. **Requirements of a distributed file system are :**

i. Transparency
ii. Concurrent file updates
iii. File replication
iv. Hardware and operating system heterogeneity
v. Fault tolerance
vi. Security
vii. Efficiency

3.10. List out the mechanisms for building Distributed File System (DSF).

Ans. **Mechanisms for building DFS :**

i. Mounting
ii. Caching
iii. Hints
iv. Bulk data transfer
v. Encryption

3.11. What are the algorithms for implementing distributed shared memory ?

Ans. **There are four basic algorithms to implement DSM systems :**

i. The central-server algorithm
ii. The migration algorithm
iii. The read-replication algorithm
iv. The full-replication algorithm

☺☺☺

4 UNIT

Failure Recovery in Distributed System (2 Marks Questions)

4.1. Compare and contrast static and dynamic vote protocol.

AKTU 2016-17, Marks 02

Ans.

S. No.	Static vote protocol	Dynamic vote protocol
1.	Static voting protocol is non-adaptive in nature.	Dynamic voting protocol is adaptive in nature.
2.	It can cause communication failure.	It prevents communication failure.
3.	It may or may not ensure availability.	It ensures availability.
4.	Selection time of replica does not depend on system state.	Selection time of replica depends on system state.

4.2. Define fault and failure. What are different approaches to fault tolerance ?

AKTU 2016-17, Marks 02

OR

Define fault and failure in distributed system.

AKTU 2018-19, Marks 02

Ans. **Faults :** A fault is an anomalous physical condition. The causes of a fault include design errors, manufacturing problems, damage fatigue or other deterioration, and external disturbances.

Failure : Failure of a system occurs when the system does not perform its services in the manner specified. An erroneous state of the system is a state which could lead to a system failure by a sequence of valid state transaction.

Different fault tolerance approaches :

1. **Replication :**
   a. Replication is the process of creating and maintaining multiple copies of data objects or processes on several nodes.
   b. Therefore, if failure on one node occurs then data will be accessible to the user from other node.
   c. Replication provides high data availability and performance.
2. **Checkpointing :**
   a. Fault tolerance can be achieved through checkpointing.
   b. Checkpointing means to periodically save the consistent state of the system in a reliable storage medium. Each such instance when a system is in the consistent state is called a checkpoint.
   c. Checkpointing is primarily used to avoid losing all the useful processing done before a fault has occurred.
   d. In case of a fault, checkpoint enables the execution of a program to be resumed from a previous consistent state rather than resuming the execution from the beginning.

4.3. What do you mean by commit protocol ?

AKTU 2018-19, Marks 02

Ans.
1. In distributed system commit protocols ensure the atomicity across the sites, *i.e.*, when a transaction executes at multiple sites it must either be committed at all the sites or aborted at all the sites.
2. The goal of commit protocols is to have all the concern participants agree either to commit or to abort a transaction.

4.4. What is error ?

Ans. An error is that part of the system state which differs from its intended value. An error is a manifestation of a fault in a system which could lead to a system failure.

4.5. Give classification of failures.

Ans. **Failures can be classified as :**

i. Process failure
ii. System failure
iii. Secondary storage failure
iv. Communication medium failure

4.6. Why one phase atomic commit protocol is inadequate ?

Ans. One phase atomic commit protocol is inadequate because when the client requests a commit, it does not allow a server to make a unilateral decision about a transaction.

4.7. What are the properties that influence ability of distributed file system to tolerate faults ?

Ans. **Properties are :**

i. Recoverability
ii. Availability
iii. Robustness

4.8. What are the approaches to implement resilient process ?

Ans. **Approaches to implement resilient process :**

i. Backup processes

ii. Replicated execution

4.9. Mention approaches to implement backward error recovery.

Ans. **Approaches to implement backward recovery :**

i. Operation based approach

ii. State based approach

4.10. Differentiate between local and global checkpointing.

AKTU 2015-16, Marks 02

Ans.

S. No.	Local checkpointing	Global checkpointing
1.	Local checkpoint is the state of the process at a given instance and the event of recording the state of a process is called local checkpoint.	Global checkpoint is the set of all local checkpoint, each from one process.

4.11. What is pessimistic logging ?

Ans. In pessimistic logging, an incoming message is logged before it is processed.

4.12. What is optimistic logging ?

Ans. In optimistic logging, processors continue to perform the computation and the messages received are stored in volatile storage, which are logged at certain intervals.

☺☺☺

5 UNIT

Transaction and Concurrency Control (2 Marks Questions)

5.1. What is atomic commit protocol ?

AKTU 2017-18, Marks 02

Ans. Atomic commit protocol (ACP) is a protocol used by database manager to ensure that all the sub-transactions are consistently committed or aborted. In this each server applies local concurrency control to its own object, which ensures that transaction are serialized locally as well as serialized globally.

5.2. How shadow versions are helpful in recovery ?

AKTU 2015-16, Marks 02

Ans. The shadow versions technique is an alternative way to organize a recovery file. It uses a map to locate versions of the server's objects in a file called a version store. To restore the objects when a server is replaced after a crash, its recovery manager reads the map and uses the information in the map to locate the objects in the version store.

5.3. What are the differences between a local call and a remote call ?

AKTU 2017-18, Marks 02

Ans.

S. No.	Local call	Remote call
1.	Local procedure call is faster.	Remote procedure call is slower.
2.	Procedure call cannot be executed on a remote machine.	With RPC the procedure call can be executed on a remote machine.
3.	No need of setup.	RPC's need to be set up before using them.

5.4. State timestamp ordering

Ans.
1. The timestamp ordering protocol is used to order the transactions based on their timestamps. The order of transaction is nothing but the ascending order of the transaction creation.
2. The priority of the older transaction is higher and it is executed first. To determine the timestamp of the transaction, the protocol uses system time or logical counter.
3. The timestamp ordering protocol also maintains the timestamp of last 'read' and 'write' operation on a data.

5.5. Explain the effect of replicated data in transactions.

AKTU 2018-19, Marks 02

Ans. Effect of replicated data in transaction :
1. It enhances the availability and performance of data.
2. It makes recovery more complicated.

5.6. What are the different validation conditions for optimistic concurrency control ?

AKTU 2016-17, Marks 02

Ans. Different validation conditions for optimistic concurrency control : Let T_i and T_j be the two transactions. For a transaction T_j to be serializable with respect to an overlapping transaction T_i, their operations must confirm to the following rules/conditions :
1. T_i must not read objects written by T_j.
2. T_j must not read objects written by T_i.
3. T_i must not write objects written by T_j and T_j must not write objects written by T_i.

5.7. What do you mean by transactions ?

Ans. A transaction is a sequence read, write or compute operation on any data object of a database.

5.8. What are the properties of transactions ?

Ans. Properties of transaction :
i. Atomicity
ii. Consistency
iii. Isolation
iv. Durability

5.9. What are the advantages of nested transaction ?

Ans. Advantages of nested transaction :
i. Subtransactions at one level may run concurrently with other subtransactions at the same level in the hierarchy. This can allow additional concurrency in a transaction.
ii. Subtransactions can commit or abort independently. In comparison with a single transaction a set of nested subtransaction is potentially more robust.

5.10. What do you mean by two phase locking ?

Ans. Two phase locking is a dynamic locking scheme in which a transaction requests a lock on a data item when it needs the data item. It has two phases :
i. Growing phase
ii. Shrinking phase

5.11. Write down the main categories of concurrency control.

Ans. Categories of concurrency control are :
i. Optimistic
ii. Pessimistic
iii. Semi-optimistic

5.12. What are main task of the group membership service ?

Ans. A group membership service has four main tasks which are as follows :
i. Providing an interface for group membership changes.
ii. Implementing a failure detector.
iii. Notifying members of group membership changes.
iv. Performing group address expansion.

5.13. Why two phase locking suffers from cascaded roll back ?

Ans. Two phase locking suffers from the problem of cascaded roll back because a transaction may be rolled back after it has released the locks on some data objects and other transactions have read those modified data objects.

5.14. What is static locking ?

Ans. In static locking, a transaction acquires locks on all the data objects it needs before executing any action on the data objects.

5.15. When a transaction is called well-formed ?

Ans. A transaction is well-formed if it :
i. Locks a data object before accessing it.
ii. Does not lock a data object more than once.
iii. Unlocks all the locked data objects before it completes.

5.16. What are the advantages of distributed database system ?

Ans. Advantages of distributed database system :
i. Sharing
ii. Higher system availability (reliability)
iii. Improved performance

5.17. Where distributed transactions can be used ?

AKTU 2015-16, Marks 02

Ans. Distributed transactions are implemented in systems based on web services. Usually these transactions utilize principles of compensating transactions, optimism and isolation without locking.

B. Tech.
(SEM VII) ODD SEMESTER THEORY EXAMINATION, 2014-15
DISTRIBUTED SYSTEMS

Time : 3 Hours **Max. Marks : 100**

Note : Attempt all question.

1. Attempt any **four** parts of the following : (4 × 5 = 20)

a. **What is a distributed system ? Describe the main characteristics of distributed system. Give two example of distributed system.**

Ans. Refer Q. 1.1, Page 1–2B, Unit-1.

b. **What are commit protocols ? Explain how two phase protocols respond to failure of participating site and failure of coordinator.**

Ans. Refer Q. 4.16, Page 4–16B, Unit-4.

c. **What do you mean by mutual exclusion in distributed system ? What are requirements of a good mutual exclusion algorithm ?**

Ans. Refer Q. 2.1, Page 2–2B, Unit-2.

d. **What are vector clocks ? Explain with the help of implementation rule of vector clocks, how they are implemented ? Give the advantages of vector clock over Lamport clock.**

Ans. Refer Q. 1.14, Page 1–14B, Unit-1.

e. **What is replication and replica manager ? Give the architectural model for replicated data.**

Ans. Refer Q. 5.30, Page 5–21B, Unit-5.

f. **What is distributed shared memory (DSM) ? Explain with diagram the architecture of distributed shared memory.**

Ans. Refer Q. 3.23, Page 3–24B, Unit-3.

2. Attempt any **four** parts : (4 × 5 = 20)

a. **Explain the following :**

i. **Gossip architecture**

ii. **Quorum consensus methods :**

Ans. Refer Q. 5.31, Page 5–22B, Unit-5.

b. **What do you mean by recovery in concurrent system ? Explain.**

Ans. Refer Q. 4.3, Page 4–4B, Unit-4.

c. **What is voting protocol ? Explain static voting and dynamic voting protocols.**

Ans. Refer Q. 4.18, Page 4–18B, Unit-4.

d. **Explain the Ricart-Agrawal algorithm for mutual exclusion. Mention the performance of this algorithm.**

Ans. Refer Q. 2.8, Page 2–7B, Unit-2.

e. **Define fault and failure. What are different approaches to fault-tolerance ? Explain.**

Ans. Refer Q. 4.13, Page 4–13B, Unit-4.

f. **Describe the following algorithm for implementing DSM :**

i. **The Migration Algorithm**

ii. **The Full-Replication Algorithm**

Ans. Refer Q. 3.26, Page 3–27B, Unit-3.

3. Attempt any **two** parts : (10 × 2 = 20)

a. i. **What are the goals of distributed transaction ? Distinguish between flat and nested transaction along with its structure.**

ii. **Explain optimistic concurrency control.**

Ans. Refer Q. 5.16, Page 5–11B, Unit-5.

b. **Define forward recovery and backward recovery. List advantages and disadvantages of forward recovery. Explain two approaches of backward-error recovery.**

Ans. Refer Q. 4.1, Page 4–2B, Unit-4.

c. **What are agreement protocols ? Explain Byzantine agreement problem, the consensus problem and interactive consistency problem. Describe Lamport-Shostak-Pease algorithm.**

Ans. Refer Q. 3.5, Page 3–5B, Unit-3.

4. Attempt any **two** parts : (10 × 2 = 20)

a. **What are the advantages and drawback of multiversion timestamp ordering in comparison with the basic timestamp ordering ?**

Ans. Refer Q. 5.13, Page 5–10B, Unit-5.

b. Write short note on :

i. Livelocks

ii. Domino effects

iii. Failure resilient processes

iv. Consistent checkpoints

Ans. Refer Q. 4.7, Page 4–7B, Unit-4.

c. i. Explain typical architecture of distributed file system. Give the mechanisms for building distributed file system.

ii. What is caching ? How is useful in DFS ?

Ans. Refer Q. 3.17, Page 3–17B, Unit-3.

5. Attempt any two parts : (10 × 2 = 20)

a. Give the deadlock handling strategies in distributed system ? What are the differences in centralized, distributed and hierarchical control organizations for distributed deadlock detection ?

Ans. Refer Q. 2.15, Page 2–14B, Unit-2.

b. Why is scalability an important feature in the design of distributed system ? Discuss some of the guiding principles for designing a scalable distributed system.

Ans. Refer Q. 1.7, Page 1–6B, Unit-1.

c. Distinguish between :

i. Resource deadlock and Communication deadlock.

Ans. Refer Q. 2.13, Page 2–12B, Unit-2.

ii. Token based and non-token based algorithm.

Ans. Refer Q. 2.5, Page 2–4B, Unit-2.

☺☺☺

B.Tech.
(SEM. VII) ODD SEMESTER THEORY EXAMINATION, 2015-16
DISTRIBUTED SYSTEMS

Time : 3 Hours **Max. Marks : 100**

SECTION – A

Note : Attempt **all** parts. All question carry equal marks. Write answer of each part in short. (2 × 10 = 20)

1. a. How shadow versions are helpful in recovery ?

Ans. Refer Q. 5.2, Page SQ–15B, Unit-5, Two Marks Questions.

b. Differentiate between local and global checkpointing.

Ans. Refer Q. 4.10, Page SQ–14B, Unit-4, Two Marks Questions.

c. Discuss the role of file system in distributed system.

Ans. Refer Q. 1.1, Page SQ–1B, Unit-1, Two Marks Questions.

d. What is consistent cut and inconsistent cut ?

Ans. Refer Q. 1.19, Page SQ–5B, Unit-1, Two Marks Questions.

e. Explain desirable features of a good message passing system.

Ans. Refer Q. 1.21, Page SQ–5B, Unit-1, Two Marks Questions.

f. What is termination detection problem ?

Ans. Refer Q. 1.20, Page SQ–5B, Unit-1, Two Marks Questions.

g. What are distributed shared memory design issues ?

Ans. Refer Q. 3.1, Page SQ–9B, Unit-3, Two Marks Questions.

h. Where distributed transactions can be used ?

Ans. Refer Q. 5.17, Page SQ–17B, Unit-5, Two Marks Questions.

i. Why clocks need to be synchronized ?

Ans. Refer Q. 3.2, Page SQ–9B, Unit-3, Two Marks Questions.

j. List the goals of distributed systems.

Ans. Refer Q. 1.2, Page SQ–1B, Unit-1, Two Marks Questions.

SECTION – B

Note : Attempt any five questions. All questions carry equal marks. **(10 × 5 = 50)**

2. Discuss at least three main issues that are relevant to the understanding of distributed fault tolerance system. Explain how that make it important.

Ans. Refer Q. 4.14, Page 4–14B, Unit-4.

3. Explain how the two phase commit protocol for nested transaction ensures that if the top level transactions commit, all the right descendents are committed or aborted ?

Ans. Refer Q. 5.3, Page 5–3B, Unit-5.

4. What are Lamport logical clocks ? List the important conditions to be satisfied by Lamport logical clocks. If *A* and *B* represent two distinct events in a process and if $A > B$ then $C(A) < C(B)$ but vice-versa not true. Justify the statement.

Ans. Refer Q. 1.12, Page 1–12B, Unit-1.

5. Caching is one of the techniques used to improve access to naming data. What are the benefits of caching and what assumptions must hold for it to be useful ?

Ans. Refer Q. 3.22, Page 3–22B, Unit-3.

6. What do you mean by backward and forward error recovery ? Discuss recovery in concurrent systems in detail.

Ans. Refer Q. 4.3, Page 4–4B, Unit-4.

7. What are distributed systems ? What are significant advantages, applications and limitations of distributed systems ? Explain with examples, what could be the impact of absence of global clock & shared memory.

Ans. Refer Q. 1.11, Page 1–11B, Unit-1.

8. What are agreement protocol ? Discuss the general system model where agreement protocols are used. Give the applications of agreement protocols.

Ans. Refer Q. 3.14, Page 3–14B, Unit-3.

9. Discuss the optimistic methods for distributed concurrency control. What are the different validation conditions for optimistic concurrency control ? Explain.

Ans. Refer Q. 5.8, Page 5–7B, Unit-5.

SECTION – C

Note : Attempt any two questions from this section. **(15 × 2 = 30)**

10. How distributed mutual exclusion is different from mutual exclusion in single-computer system ? Classify mutual exclusion algorithms. How the performance of mutual exclusion algorithms is measured ? Compare the performance of token and non-token based algorithms. How the Ricart- Agrawala algorithm optimize the Lamport's algorithm.

Ans. Refer Q. 2.10, Page 2–9B, Unit-2.

11. What do you mean by casual ordering of messages ? If process *P* sends two message m_1 and m_2 to another process *Q*, what problems may arise if the two messages are not received by recipient *Q*, in the order they were sent by process *P*. Develop an algorithm which guarantees the casual ordering of message in distributed system.

Ans. Refer Q. 1.15, Page 1–16B, Unit-1.

12. Discuss following terms in context of distributed systems :

a. 2PL and Strict 2PL.

Ans. Refer Q. 5.5, Page 5–5B, Unit-5.

b. Timestamp ordering for transaction management.

Ans. Refer Q. 5.10, Page 5–8B, Unit-5.

c. Highly available services.

Ans. Refer Q. 5.32, Page 5–24B, Unit-5.

d. Wait for graph with the example of distributed transaction.

Ans. Refer Q. 5.25, Page 5–17B, Unit-5.

e. Sequential consistency.

Ans. Refer Q. 5.32, Page 5–24B, Unit-5.

☺☺☺

B.Tech.
(SEM. VII) ODD SEMESTER THEORY EXAMINATION, 2016-17
DISTRIBUTED SYSTEM

Time : 3 Hours **Max. Marks : 100**

Section-A

1. **Answer all parts. All parts carry equal marks. Write answer of each part in short.** (2 × 10 = 20)

a. **List out the main challenges of distributed systems.**

Ans. Refer Q. 1.3, Page SQ–1B, Unit-1, Two Marks Questions.

b. **What are logical clocks ? Why does a logical clock need to be implemented in distributed systems ?**

Ans. Refer Q. 1.4, Page SQ–1B, Unit-1, Two Marks Questions.

c. **What do you mean by mutual exclusion in distributed system ? What are the requirements of a good mutual exclusion algorithm ?**

Ans. Refer Q. 2.1, Page SQ–6B, Unit-2, Two Marks Questions.

d. **Define deadlock detection in distributed systems.**

Ans. Refer Q. 2.2, Page SQ–6B, Unit-2, Two Marks Questions.

e. **List out some issues in distributed file systems.**

Ans. Refer Q. 1.5, Page SQ–1B, Unit-1, Two Marks Questions.

f. **State Byzantine agreement problem.**

Ans. Refer Q. 3.4, Page SQ–10B, Unit-3, Two Marks Questions.

g. **What do you mean by agreement protocol ?**

Ans. Refer Q. 3.3, Page SQ–9B, Unit-3, Two Marks Questions.

h. **Compare and contrast static and dynamic vote protocol.**

Ans. Refer Q. 4.1, Page SQ–12B, Unit-4, Two Marks Questions.

i. **Define fault and failure. What are different approaches to fault tolerance ?**

Ans. Refer Q. 4.2, Page SQ–12B, Unit-4, Two Marks Questions.

j. **What are the different validation conditions for optimistic concurrency control ?**

Ans. Refer Q. 5.6, Page SQ–16B, Unit-5, Two Marks Questions.

Section-B

Note : Attempt any five question from this section. (10 × 5 = 50)

2. i. **Discuss the limitations of Lamport's logical clock with suitable example.**

Ans. Refer Q. 1.13, Page 1–13B, Unit-1.

ii. **Give the Chandy-Lamport's global state recording algorithm.**

Ans. Refer Q. 1.18, Page 1–19B, Unit-1.

3. **Discuss causal ordering of messages. Give one algorithm which can order the messages according to causal dependencies.**

Ans. Refer Q. 1.15, Page 1–16B, Unit-1.

4. i. **Differentiate between token and non-token based algorithms.**

Ans. Refer Q. 2.5, Page 2–4B, Unit-2.

ii. **What are the deadlock handling strategies in distributed file system ? What is control organization for distributed deadlock detection ? Discuss an algorithm which can remove phantom deadlock.**

Ans. Refer Q. 2.14, Page 2–13B, Unit-2.

5. **What are agreement protocols ? Explain Byzantine agreement problem, the consensus problem and interactive consistency problem.**

Ans. Refer Q. 3.4, Page 3–4B, Unit-3.

6. **Describe in detail :**

a. **Dynamic voting protocols**

b. **Method to obtain consistent set of checkpoint**

Ans. Refer Q. 4.18, Page 4–18B, Unit-4.

7. **Define forward recovery and backward recovery. List advantages and disadvantages of forward recovery. Explain two approaches of backward error recovery.**

Ans. Refer Q. 4.1, Page 4–2B, Unit-4.

8. **Explain design issues in distributed shared memory and also write algorithm for implementation of shared memory.**

Ans. Refer Q. 3.25, Page 3–26B, Unit-3.

9. i. What are the goals of distributed transaction ? Distinguish between flat and nested transaction along with its structure.

ii. Explain optimistic concurrency control.

Ans. Refer Q. 5.16, Page 5–11B, Unit-5.

Section-C

Note : Attempt any two questions from this section. (15 × 2 = 30)

10. Describe Lamport-Shostak-Pease algorithm. How does vector clock overcome the disadvantages of Lamport clock ? Explain with an example.

Ans. Refer Q. 3.6, Page 3–6B, Unit-3.

11. Discuss the following :

a. Performance metric for distributed mutual exclusion algorithms

Ans. Refer Q. 2.9, Page 2–8B, Unit-2.

b. Obermarck's path-pushing algorithm

Ans. Refer Q. 2.16, Page 2–14B, Unit-2.

12. Write short note on :

a. Flat and nested transaction

Ans. Refer Q. 5.15, Page 5–11B, Unit-5.

b. 2PL and Strict 2PL

Ans. Refer Q. 5.5, Page 5–5B, Unit-5.

☺☺☺

B. Tech.
(SEM. VII) EVEN SEMESTER THEORY EXAMINATION, 2017-18
DISTRIBUTED SYSTEM

Time : 3 Hours **Max. Marks : 100**

Note : 1. Attempt all Sections. Assume missing data, if any.

SECTION-A

1. Attempt all questions in brief. **(2 × 10 = 20)**

a. Why would you design a system as a distributed system ? List some advantages of distributed systems.

Ans. Refer Q. 1.8, Page SQ–2B, Unit-1, Two Marks Questions.

b. List three properties of distributed systems.

Ans. Refer Q. 1.12, Page SQ–3B, Unit-1, Two Marks Questions.

c. What is the advantage if your server side processing uses threads instead of a single process ?

Ans. Refer Q. 1.13, Page SQ–3B, Unit-1, Two Marks Questions.

d. What is a proxy ? Give an example of where a proxy can be used.

Ans. Refer Q. 1.14, Page SQ–3B, Unit-1, Two Marks Questions.

e. What are the differences between a local call and a remote call ?

Ans. Refer Q. 5.7, Page SQ–15B, Unit-5, Two Marks Questions.

f. What is the purpose of a firewall ?

Ans. Refer Q. 1.15, Page SQ–3B, Unit-1, Two Marks Questions.

g. What were the reasons that middleware moved from distributed objects to distributed components ?

Ans. Refer Q. 1.16, Page SQ–3B, Unit-1, Two Marks Questions.

h. Name two mechanisms that can be used to ensure performance in distributed systems.

Ans. Refer Q. 1.17, Page SQ–4B, Unit-1, Two Marks Questions.

i. What are the differences between a URL, URI and URN ?

Ans. Refer Q. 1.18, Page SQ–

j. What is atomic commit protocol ?

Ans. Refer Q. 5.5, Page SQ–15B, Unit-5, Two Marks Questions.

SECTION-B

2. Attempt any **three** of the following : (10 × 3 = 30)

a. What is distributed transparency ? Explain the different types of distributed transparencies.

Ans. Refer Q. 1.4, Page 1–4B, Unit-1.

b. Explain shared memory architecture and distributed memory architecture.

Ans. Refer Q. 3.21, Page 3–21B, Unit-3.

c. Describe Byzantine agreement problem, and explain its solution. Show that Byzantine agreement cannot always be reached among four processors if two processors are faulty.

Ans. Refer Q. 3.9, Page 3–9B, Unit-3.

d. Discuss the major issue in designing a distributed system.

Ans. Refer Q. 1.6, Page 1–5B, Unit-1.

e. What is mutual exclusion ? Describe the requirements of mutual exclusion in distributed system. Is mutual exclusion problem more complex in distributed system than single computer system ? Justify your answer.

Ans. Refer Q. 2.3, Page 2–3B, Unit-2.

SECTION-C

3. Attempt any **two** parts of the following : (5 × 2 = 10)

a. Draw a schematic diagram of the distributed transaction management model. Explain each component in brief.

Ans. Refer Q. 5.17, Page 5–13B, Unit-5.

b. Describe three phase commit protocol. How three phase commit protocol is different than two phase commit protocol ?

Ans. Refer Q. 4.17, Page 4–17B, Unit-4.

c. Write and explain various issues that must be addressed in design and implementation of distributed file system.

Ans. Refer Q. 3.19, Page 3–19B, Unit-3.

4. Attempt any **two** parts of the following : (5 × 2 = 10)

a. What is cache ? Discuss read operation with cache and write operation with cache.

Ans. Refer Q. 3.18, Page 3–18B, Unit-3.

b. Explain naming in distributed system. What is flat naming and structured naming ?

Ans. Refer Q. 3.20, Page 3–20B, Unit-3.

c. Classify the deadlock detection algorithms. Describe the path-pushing deadlock detection algorithm.

Ans. Refer Q. 2.16, Page 2–14B, Unit-2.

5. Attempt any **one** parts of the following : (10 × 1 = 10)

a. What are stub and skeleton and why are they needed in remote procedure calls ?

Ans. Refer Q. 5.36, Page 5–29B, Unit-5.

b. What is the purpose of an Interface Definition Language ? Why does CORBA not just use the Java interface construct ?

Ans. Refer Q. 5.37, Page 5–29B, Unit-5.

6. Attempt any **one** part of the following : (10 × 1 = 10)

a. How the distributed computing system is better than parallel processing system ? Explain.

Ans. Refer Q. 1.3, Page 1–3B, Unit-1.

b. What is termination detection in distributed system ? Explain any algorithm for termination detection.

Ans. Refer Q. 1.19, Page 1–20B, Unit-1.

7. Attempt any **one** part of the following : (10 × 1 = 10)

a. How does a server know that one of his remote objects provided by him is no longer used by clients and can be collected ? How does Java RMI handle this problem and what alternatives are there ?

Ans. Refer Q. 5.38, Page 5–30B, Unit-5.

b. De-activation is a technology used to preserve server resources where a server which provides remote objects to clients can de-activate those remote objects. Clients should not know about this. What must the server do to avoid surprises for the clients ?

Ans. Refer Q. 5.39, Page 5–30B, Unit-5.

☺☺☺

B. Tech.
(SEM. VII) ODD SEMESTER THEORY EXAMINATION, 2018-19
DISTRIBUTED SYSTEM

Time : 3 Hours **Max. Marks : 100**

Note : 1. Attempt all Sections. If require any missing data; then choose suitably.

SECTION-A

1. Attempt all questions in brief. **(2 × 10 = 20)**

a. **What are the web challenges involved in distributed system ?**

Ans. Refer Q. 1.6, Page SQ–2B, Unit-1, Two Marks Questions.

b. **Explain system model.**

Ans. Refer Q. 1.7, Page SQ–2B, Unit-1, Two Marks Questions.

c. **What is distributed deadlock ?**

Ans. Refer Q. 2.3, Page SQ–6B, Unit-2, Two Marks Questions.

d. **What do you mean by commit protocol ?**

Ans. Refer Q. 4.3, Page SQ–13B, Unit-4, Two Marks Questions.

e. **State timestamp ordering.**

Ans. Refer Q. 5.4, Page SQ–15B, Unit-5, Two Marks Questions.

f. **Explain the concept of shared memory.**

Ans. Refer Q. 3.5, Page SQ–10B, Unit-3, Two Marks Questions.

g. **Define fault and failure in distributed system.**

Ans. Refer Q. 4.2, Page SQ–12B, Unit-4, Two Marks Questions.

h. **Explain token based algorithm.**

Ans. Refer Q. 2.4, Page SQ–7B, Unit-2, Two Marks Questions.

i. **What do you mean by agreement protocol ?**

Ans. Refer Q. 3.3, Page SQ–9B, Unit-3, Two Marks Questions.

j. **Explain the effect of replicated data in transactions.**

Ans. Refer Q. 5.5, Page SQ–16B, Unit-5, Two Marks Questions.

SECTION-B

2. Attempt any three of the following : **(10 × 3 = 30)**

a. **State the classification of distributed mutual exclusion. What is requirement of mutual exclusion theorem ?**

Ans. Refer Q. 2.1, Page 2–2B, Unit-2.

b. **What do you understand by Byzantine agreement problem ?**

Ans. Refer Q. 3.7, Page 3–7B, Unit-3.

c. **Give the design issues in distributed shared memory. State the algorithm for implementation of distributed shared memory.**

Ans. Refer Q. 3.25, Page 3–26B, Unit-3.

d. **Explain the limitations of distributed system with example.**

Ans. Refer Q. 1.10, Page 1–10B, Unit-1.

e. **Define forward and backward recovery. Also list the advantages and disadvantages of both.**

Ans. Refer Q. 4.2, Page 4–3B, Unit-4.

SECTION-C

3. Attempt any one part of the following : **(10 × 1 = 10)**

a. **What is token based algorithm and non-token based algorithm in distributed system ? Explain with example.**

Ans. Refer Q. 2.4, Page 2–4B, Unit-2.

b. **What are distributed systems ? What are significant advantages and applications of distributed systems ?**

Ans. Refer Q. 1.2, Page 1–3B, Unit-1.

4. Attempt any one part of the following : **(10 × 1 = 10)**

a. **What are Lamport logical clocks ? List the important conditions to be satisfied by Lamport logical clocks. Discuss the limitations of Lamport logical clocks.**

Ans. Refer Q. 1.13, Page 1–13B, Unit-1.

b. **Explain the mechanism for building distributed file system also explain the design issues in distributed shared memory.**

Ans. Refer Q. 3.24, Page 3–25B, Unit-3.

5. Attempt any one part of the following : (10 × 1 = 10)

a. **How distributed mutual exclusion is different of mutual exclusion in single computer system ? How the performance of mutual exclusion algorithm is measured ?**

Ans. Refer Q. 2.9, Page 2–8B, Unit-2.

b. **Discuss the following in terms of distributed system**

i. **Sequential consistency**

ii. **Highly available services**

Ans. Refer Q. 5.32, Page 5–24B, Unit-5.

6. Attempt any one part of the following : (10 × 1 = 10)

a. **Explain typical architecture of distributed file system. State the algorithm for implementation of distributed shared memory.**

Ans. Refer Q. 3.25, Page 3–26B, Unit-3.

b. **What is Byzantine agreement problem ? Provide the solution to Byzantine agreement problem.**

Ans. Refer Q. 3.7, Page 3–7B, Unit-3.

7. Attempt any one part of the following : (10 × 1 = 10)

a. **What are the different validation conditions for optimistic concurrency control ? How it effects the transaction in distributed system.**

Ans. Refer Q. 5.9, Page 5–8B, Unit-5.

b. **Explain distributed transactions. Discuss the functionality of flat and nested distributed transactions with example.**

Ans. Refer Q. 5.15, Page 5–11B, Unit-5.

☺☺☺

B. Tech.
(SEM. VII) ODD SEMESTER THEORY EXAMINATION, 2019-20
DISTRIBUTED SYSTEM

Time : 3 Hours **Max. Marks : 70**

Note : Attempt **all** sections. If require any missing data; then choose suitably.

SECTION-A

1. Attempt **all** questions in brief. (2 × 7 = 14)

a. **Where can distributed transactions be used ?**

Ans. Refer Q. 5.17, Page SQ–17B, Unit-5, Two Marks Questions.

b. **Define fault and failure.**

Ans. Refer Q. 4.2, Page SQ–12B, Unit-4, Two Marks Questions.

c. **Define global state and consistent global state.**

Ans. **Global state :** Refer Q. 1.11, Page SQ–3B, Unit-1, Two Marks Questions.

Consistent global state : In a consistent global state, for every received message a corresponding send event is recorded in the global state.

d. **Define causal order and total order.**

Ans. **Casual order :** Refer Q. 1.10, Page SQ–2B, Unit-1, Two Marks Questions.

Total order : A total order is a binary relation that defines an order for every element in some set.

e. **Explain token-based algorithm.**

Ans. Refer Q. 2.4, Page SQ–7B, Unit-2, Two Marks Questions.

f. **State Byzantine agreement problem.**

Ans. Refer Q. 3.4, Page SQ–10B, Unit-3, Two Marks Questions.

g. **Define transparency. List various types of transparencies in distributed systems.**

Ans. Refer Q. 1.4, Page 1–4B, Unit-1.

SECTION-B

2. Attempt any three of the following : (7 × 3 = 21)

a. **Describe Lamport-Shostak-Pease algorithm. How does vector clock overcome the disadvantages of Lamport clock ? Explain with an example.**

Ans. Refer Q. 3.6, Page 3–6B, Unit-3.

b. **Give the deadlock handling strategies in distributed systems. What are the differences in centralized, distributed and hierarchical control organizations for distributed deadlock detection ?**

Ans. Refer Q. 2.15, Page 2–14B, Unit-2.

c. **Write short note on any one of the following :**
i. **Flat and Nested transaction**
ii. **Timestamp ordering for transaction management**

Ans.
i. Refer Q. 5.15, Page 5–11B, Unit-5.
ii. Refer Q. 5.10, Page 5–8B, Unit-5.

d. **Define forward and backward recovery. List advantages and disadvantages of forward recovery. Explain two approaches of backward error recovery.**

Ans. Refer Q. 4.1, Page 4–2B, Unit-4.

e. **What is agreement protocol ? Discuss the general system model where agreement protocols are used. Give the applications of agreement protocols.**

Ans. Refer Q. 3.14, Page 3–14B, Unit-3.

SECTION-C

3. Attempt any one part of the following : (7 × 1 = 7)

a. **What are vector clocks ? Explain how vector clocks are implemented using implementation rule of vector clocks ? Give the advantages of vector clock over Lamport clock.**

Ans. Refer Q. 1.14, Page 1–14B, Unit-1.

b. **What are distributed systems ? What are significant advantages, applications and limitations of distributed systems ? Explain with examples, what could be the impact of absence of global clock and shared memory ?**

Ans. Refer Q. 1.11, Page 1–11B, Unit-1.

4. Attempt any one part of the following : (7 × 1 = 7)

a. **What is Mutual exclusion ? Describe the requirements of mutual exclusion in distributed system. Is mutual exclusion problem more complex in distributed system than single computer system ? Justify your answer.**

Ans. Refer Q. 2.3, Page 2–3B, Unit-2.

b. **What do you mean by deadlock avoidance ? Explain in brief. Describe edge chasing deadlock detection algorithm.**

Ans. **Deadlock avoidance :** Refer Q. 2.12, Page 2–10B, Unit-2.
Edge chasing algorithm : Refer Q. 2.14, Page 2–13B, Unit-2.

5. Attempt any one part of the following : (7 × 1 = 7)

a. **Describe mechanism for building distributed file system. Explain data access actions in distributed file system.**

Ans. **Mechanism of building distributed file system :** Refer Q. 3.16, Page 3–15B, Unit-3.

Data access actions :

1. A request by a process to access a data block is presented to the local cache (client cache) of the machine (client) on which the process is running.

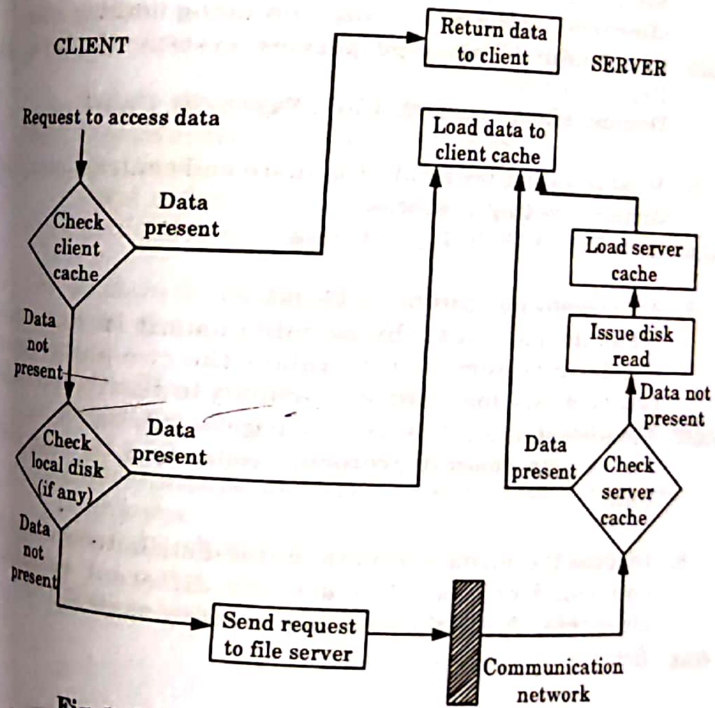


Fig. 1. Data access action in distributed file system.

2. If the block is not in the cache, then the local disk, if present, is checked for the presence of the data block.

3. If the block is present, then the request is satisfied and the block is loaded into the client cache.
4. It the block is not stored locally, then the request is passed on to the appropriate file server (as determined by the name server).
5. The server checks its own cache for the presence of the data block before issuing a disk I/O request.
6. The data block is transferred to the client cache in any case and loaded to the server cache if it was missing in the server cache.

b. Discuss the architecture of distributed shared memory and various design issues related to this memory.

Ans. **Architecture of distributed shared memory :** Refer Q. 3.23, Page 3–24B, Unit-3.
Design issues related to distributed shared memory : Refer Q. 3.24, Page 3–25B, Unit-3.

6. Attempt any one part of the following : (7 × 1 = 7)

a. What is checkpoint in message passing system ? Show that when checkpoints are taken after every $K(K > 1)$ message sent, the recovery mechanism suffers from domino effect. Assume that a process takes a checkpoint immediately after sending the K^{th} message but doing nothing else.

Ans. **Checkpoint in message passing system :** Refer Q. 4.4, Page 4–5B, Unit-4.
Domino effect : Refer Q. 4.7(ii), Page 4–7B, Unit-4.

b. What is voting protocol ? Compare and contrast static and dynamic voting protocol.

Ans. Refer Q. 4.18, Page 4–18B, Unit-4.

7. Attempt any one part of the following : (7 × 1 = 7)

a. What do you mean by atomic commit in distributed database system ? Also explain the two-phase commit protocol used for realizing atomicity in distributed system.

Ans. **Atomic commit :** Refer Q. 5.18, Page 5–14B, Unit-5.
Two phase commit protocol : Refer Q. 4.15, Page 4–15B, Unit-4.

b. Discuss the optimistic methods for distributed concurrency control. Explain what are the different validations conditions for optimistic concurrency control ?

Ans. Refer Q. 5.8, Page 5–7B, Unit-5.

☺☺☺

B. Tech.
(SEM. VII) ODD SEMESTER THEORY EXAMINATION, 2020-21
DISTRIBUTED SYSTEM

Time : 3 Hours **Max. Marks : 70**

Note : Attempt **all** sections. If require any missing data; then choose suitably.

SECTION A

1. Attempt **all** questions in brief. (2 × 7 = 14)

a. Define replication.

Ans. Refer Q. 4.13, Page 4–13B, Unit-4.

b. Explain locks.

Ans. In distributed systems, lock is a mechanism that allows only one of the innumerable nodes (or process) to access and modify a resource or data that is being shared commonly to prevent execution of same task twice and also maintain data integrity.

c. Define causal and total ordering.

Ans. **Causal ordering :** Refer Q. 1.10, Page SQ–2B, Unit-1, Two Marks Questions.
Total ordering : System is said to have total order if causal relationship among all events in the system, can be established.

d. Explain limitation of distributed system.

Ans. Refer Q. 1.10, Page 1–10B, Unit-1.

e. Describe distributed deadlock.

Ans. Refer Q. 2.3, Page SQ–6B, Unit-2, Two Marks Questions.

f. Differentiate between resource and communication deadlocks.

Ans. Refer Q. 2.13, Page 2–12B, Unit-2.

g. Differentiate between Backward and Forward recovery.

Ans.

S. No.	Forward recovery	Backward recovery
1.	In the forward recovery state, if transaction is failure then it removes the error and continues the process.	In the backward recovery, if there is any error occurred then it restarts that process.
2.	It removes the error and then going forward so it confirms that the error does not happens again.	It does not remove the error and then going forward, also it does not confirm that the error does not happen again.
3.	It is complex to implement.	It is easy to implement.

SECTION B

2. Attempt any **three** of the following : **(7 × 3 = 21)**

a. Explain distributed system. What are the basic features of distributed system in detail ?

Ans. Refer Q. 1.1, Page 1–2B, Unit-1.

b. Define the architectural models of distributed system.

Ans. Refer Q. 1.8, Page 1–7B, Unit-1.

c. Describe distributed mutual exclusion. What are the requirements of distributed mutual exclusion theorems ?

Ans. Refer Q. 2.1, Page 2–2B, Unit-2.

d. Discuss atomic commit in distributed database system with example.

Ans. Refer Q. 3.13, Page 3–13B, Unit-3.

e. Describe the issues in fault tolerance in detail.

Ans. Refer Q. 4.14, Page 4–14B, Unit-4.

SECTION C

3. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. What is logical clock ? Explain Lamport's clock with suitable example.

Ans. **Logical clock :** A logical clock is a mechanism for capturing chronological and causal relationships in a distributed system. They refer to implementing a protocol on all machines within a distributed system, so that the machines are able to maintain consistent ordering of events within some virtual time span.

Lamport's clock : Refer Q. 1.12, Page 1–12B, Unit-1.

Implementation Rules [IR] :

- **[IR1] :** If $a \rightarrow b$ ['a' happened before 'b' within the same process] then, $C_i(b) = C_i(a) + d$.
- **[IR2] :** $C_j = \max(C_j, t_m + d)$ [If there's more number of processes, then t_m = value of $C_i(a)$, C_j = max value between C_j and $t_m + d$].

For Example :

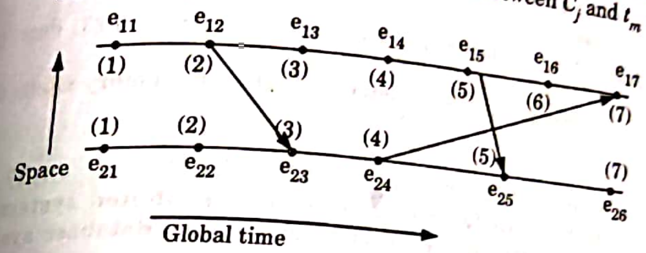


Fig. 1.

- Take the starting value as 1, since it is the 1st event and there is no incoming value at the starting point :
- $e_{11} = 1$
- $e_{21} = 1$
- The value of the next point will go on increasing by d (d = 1), if there is no incoming value *i.e.*, to follow [IR1].
- $e_{12} = e_{11} + d = 1 + 1 = 2$
- $e_{13} = e_{12} + d = 2 + 1 = 3$
- $e_{14} = e_{13} + d = 3 + 1 = 4$
- $e_{15} = e_{14} + d = 4 + 1 = 5$
- $e_{16} = e_{15} + d = 5 + 1 = 6$
- $e_{22} = e_{21} + d = 1 + 1 = 2$
- $e_{24} = e_{23} + d = 3 + 1 = 4$
- $e_{26} = e_{25} + d = 6 + 1 = 7$
- When there will be incoming value, then follow [IR2] *i.e.*, take the maximum value between C_j and $T_m + d$.
- $e_{17} = \max(7, 5) = 7$, [$e_{16} + d = 6 + 1 = 7$, $e_{24} + d = 4 + 1 = 5$, maximum among 7 and 5 is 7].
- $e_{23} = \max(3, 3) = 3$, [$e_{22} + d = 2 + 1 = 3$, $e_{12} + d = 2 + 1 = 3$, maximum among 3 and 3 is 3].
- $e_{25} = \max(5, 6) = 6$, [$e_{24} + 1 = 4 + 1 = 5$, $e_{15} + d = 5 + 1 = 6$, maximum among 5 and 6 is 6].

b. Describe vector logical clock with suitable example.

Ans. Refer Q. 1.14, Page 1–14B, Unit-1.

4. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. Explain any Token based algorithm in detail.

Ans. Refer Q. 2.6, Page 2–5B, Unit-2.

b. Explain the performance metric for distributed mutual exclusion algorithms in detail.

Ans. Refer Q. 2.9, Page 2–8B, Unit-2.

5. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. Briefly explain the classification of agreement problem.

Ans. Refer Q. 3.4, Page 3–4B, Unit-3.

b. Define distributed shared memory. What are the design issues in distributed shared memory ?

Ans. **Distributed shared memory :** Refer Q. 3.23, Page 3–24B, Unit-3.

Design issues in distributed shared memory : Refer Q. 3.24, Page 3–25B, Unit-3.

6. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. Explain failure recovery in distributed systems. Also explain the recovery in distributed database systems in detail.

Ans. Refer Q. 4.1, Page 4–2B, Unit-4.

b. Describe the following :

i. Consistent checkpoints.

ii. Voting protocols.

Ans.

i. Refer Q. 4.5, Page 4–6B, Unit-4.

ii. Refer Q. 4.18, Page 4–18B, Unit-4.

7. Attempt any **one** part of the following : **(7 × 1 = 7)**

a. Discuss the concurrency control in distributed transactions in detail.

Ans. **Concurrency control in distributed transaction :** Concurrency controlling techniques ensure that multiple transactions are executed simultaneously while maintaining the ACID properties of the transactions and serializability in the schedules.

1. **Locking :** Refer Q. 5.4, Page 5–4B, Unit-5.
2. **Two phase locking :** Refer Q. 5.5, Page 5–5B, Unit-5.
3. **Optimistic concurrency control :** Refer Q. 5.7, Page 5–7B, Unit-5.
4. **Timestamp ordering :** Refer Q. 5.10, Page 5–8B, Unit-5.
5. **Multiversion timestamp ordering protocol :** Refer Q. 5.12, Page 5–9B, Unit-5.

b. Explain the following :

i. Highly available services.

ii. Fault-tolerant services.

Ans.

i. Refer Q. 5.32, Page 5–24B, Unit-5.

ii. Refer Q. 5.35, Page 5–27B, Unit-5.

☺☺☺