Chapter 20

Distributed DBMSs - Advanced Concepts

Transparencies

Chapter - Objectives

- Distributed transaction management.
- Distributed concurrency control.
- Distributed deadlock detection.

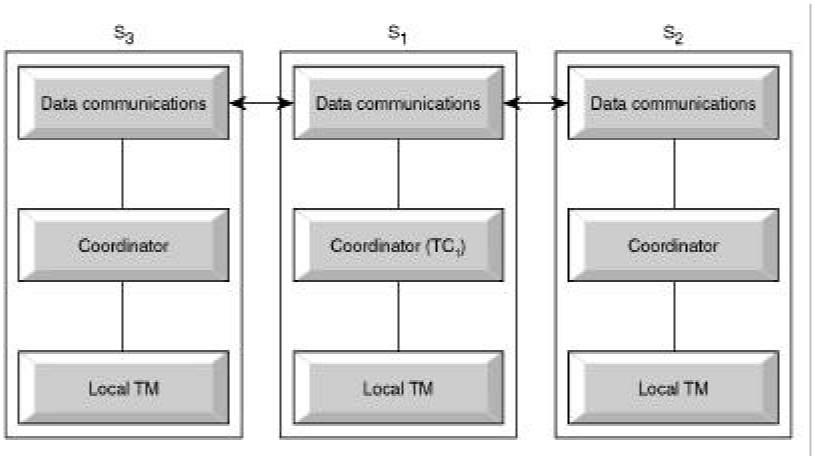
Distributed Transaction Management

- **Distributed transaction accesses data stored at more than one location.**
- **Divided into a number of *sub-transactions,* one for each site that has to be accessed, represented by an *agent.***
- **Indivisibility of distributed transaction is still fundamental to transaction concept.**
- **DDBMS must also ensure indivisibility of each sub-transaction.**

Distributed Transaction Management

- **Thus, DDBMS must ensure:**
  - **synchronization of subtransactions with other local transactions executing concurrently at a site;**
  - **synchronization of subtransactions with global transactions running simultaneously at same or different sites.**
- **Global transaction manager (transaction coordinator) at each site, to coordinate global and local transactions initiated at that site.**

Coordination of Distributed Transaction



Distributed Locking

- Look at four schemes:
  - Centralized locking
  - Primary Copy 2PL
  - Distributed 2PL
  - Majority Locking

Centralized Locking

- Single site that maintains all locking information.
- One lock manager for whole of DDBMS.
- Local transaction managers involved in global transaction request and release locks from lock manager.
- Or transaction coordinator can make all locking requests on behalf of local transaction managers.
- Advantage - easy to implement.
- Disadvantages - bottlenecks and lower reliability.

Primary Copy 2PL

- Lock managers distributed to a number of sites.
- Each lock manager responsible for managing locks for set of data items.
- For replicated data item, one copy is chosen as *primary copy*, others are *slave copies*
- Only need to write-lock primary copy of data item that is to be updated.
- Once primary copy has been updated, change can be propagated to slaves.

Primary Copy 2PL

- Disadvantages - deadlock handling is more complex due; still a degree of centralization in system.
- Advantages - lower communication costs and better performance than centralized 2PL.

Distributed 2PL

- Lock managers distributed to every site.
- Each lock manager responsible for locks for data at that site.
- If data not replicated, equivalent to primary copy 2PL.
- Otherwise, implements a Read-One-Write-All (ROWA) replica control protocol.

Distributed 2PL

- Using ROWA protocol:
  - Any copy of replicated item can be used for read.
  - All copies must be write-locked before item can be updated.
- Disadvantages - deadlock handling more complex; communication costs higher than primary copy 2PL.

Majority Locking

- Extension of distributed 2PL.
- To read or write data item replicated at *n* sites, sends a lock request to more than half the *n* sites where item is stored.
- Transaction cannot proceed until majority of locks obtained.
- Overly strong in case of read locks.

Distributed Deadlock

u More complicated if lock management is not centralized.

u Local Wait-for-Graph (LWFG) may not show existence of deadlock.

u May need to create GWFG, union of all LWFGs.

u Look at three schemes:

  – Centralized Deadlock Detection

  – Hierarchical Deadlock Detection

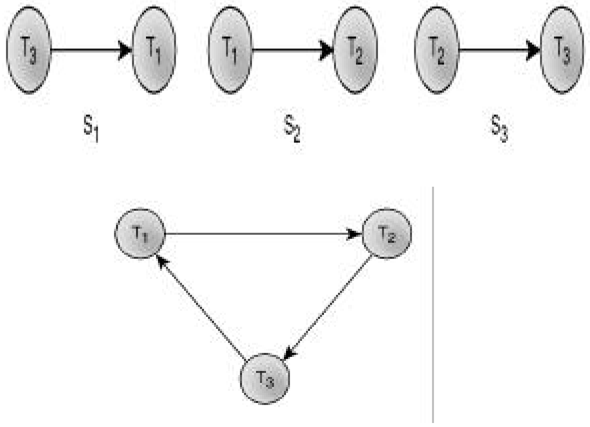
  – Distributed Deadlock Detection.

Example - Distributed Deadlock

- T_1 initiated at site S_1 and creating agent at S_2,
- T_2 initiated at site S_2 and creating agent at S_3,
- T_3 initiated at site S_3 and creating agent at S_1.

Time	S_1	S_2	S_3
t_1	read_lock(T_1, x_1)	write_lock(T_2, y_2)	read_lock(T_3, z_3)
t_2	write_lock(T_1, y_1)	write_lock(T_2, z_2)	
t_3	write_lock(T_3, x_1)	write_lock(T_1, y_2)	write_lock(T_2, z_3)

Example - Distributed Deadlock



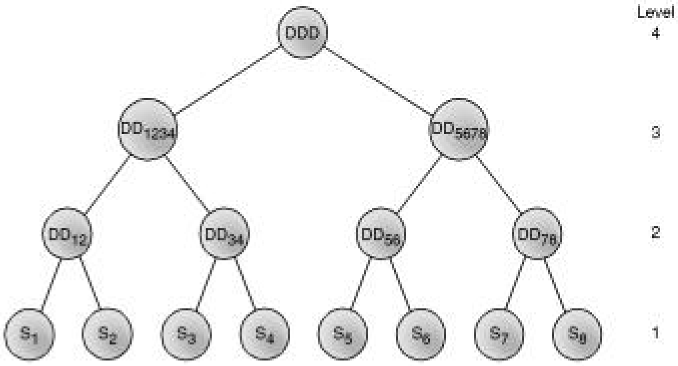
Centralized Deadlock Detection

- Single site appointed deadlock detection coordinator (DDC).
- DDC has responsibility of constructing and maintaining GWFG.
- If one or more cycles exist, DDC must break each cycle by selecting transactions to be rolled back and restarted.

Hierarchical Deadlock Detection

- Sites are organized into a hierarchy.
- Each site sends its LWFG to detection site above it in hierarchy.
- Reduces dependence on centralized detection site.

Hierarchical Deadlock Detection



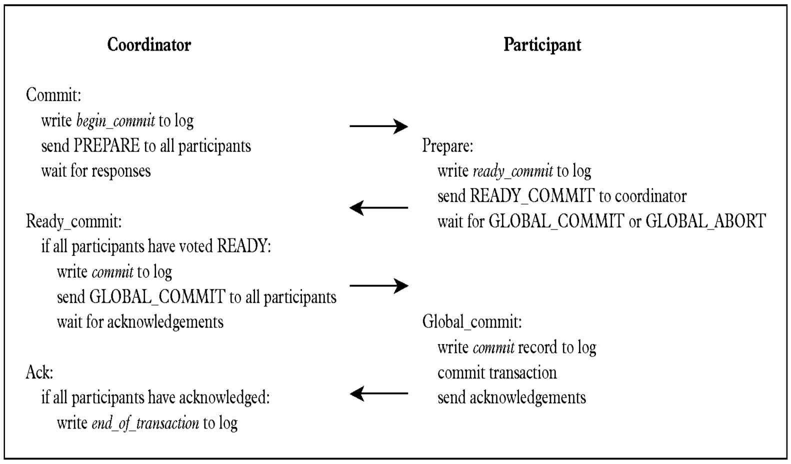
Two-Phase Commit (2PC)

- Two phases: a *voting phase* and a *decision phase*.
- Coordinator asks all participants whether they are prepared to commit transaction.
  - If one participant votes abort, or fails to respond within a timeout period, coordinator instructs all participants to abort transaction.
  - If all vote commit, coordinator instructs all participants to commit.
- All participants must adopt global decision .

Two-Phase Commit (2PC)

- If participant votes abort, free to abort transaction immediately
- If participant votes commit, must wait for coordinator to broadcast global-commit or global-abort message.
- Protocol assumes each site has its own local log and can rollback or commit transaction reliably.
- If participant fails to vote, abort is assumed.
- If participant gets no vote instruction from coordinator, can abort.

2PC Protocol for Participant Voting Commit



Coordinator

```
Commit:
   write begin_commit to log
   send PREPARE to all participants
   wait for responses

Ready_commit:
   if all participants have voted READY:
      write commit to log
      send GLOBAL_COMMIT to all participants
      wait for acknowledgements

Ack:
   if all participants have acknowledged:
      write end_of_transaction to log
```

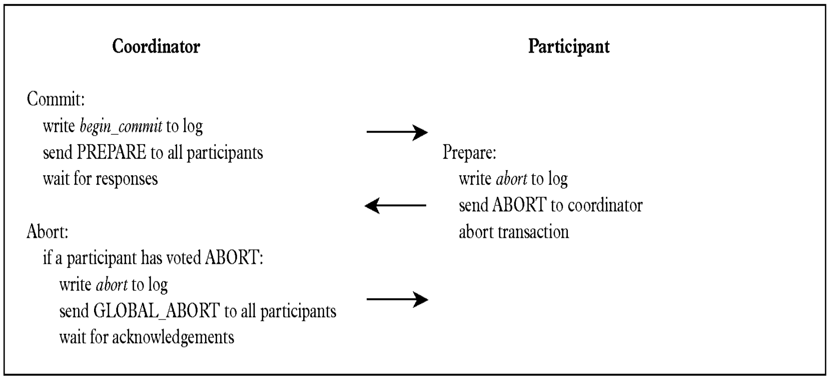
Participant

```
Prepare:
   write ready_commit to log
   send READY_COMMIT to coordinator
   wait for GLOBAL_COMMIT or GLOBAL_ABORT

Global_commit:
   write commit record to log
   commit transaction
   send acknowledgements
```

(a)

2PC Protocol for Participant Voting Abort



Coordinator

```
Commit:
   write begin_commit to log
   send PREPARE to all participants
   wait for responses

Abort:
   if a participant has voted ABORT:
      write abort to log
      send GLOBAL_ABORT to all participants
      wait for acknowledgements
```

Participant

```
Prepare:
   write abort to log
   send ABORT to coordinator
   abort transaction
```

(b)

Termination Protocols

- Invoked whenever a coordinator or participant fails to receive an expected message and times out.

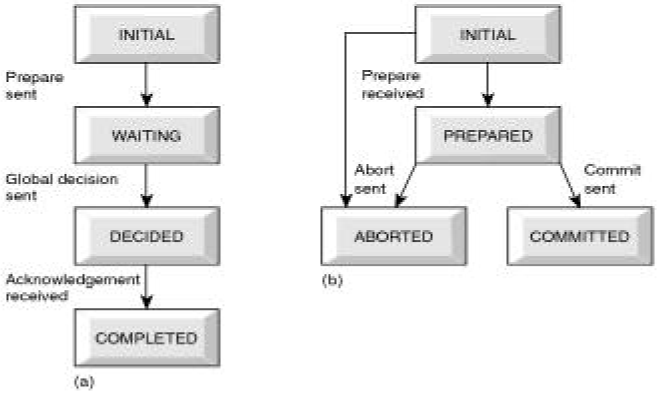
Coordinator

- Timeout in WAITING state
  - Globally abort the transaction.
- Timeout in DECIDED state
  - Send global decision again to sites that have not acknowledged.

Termination Protocols - Participant

- Simplest termination protocol is to leave participant blocked until communication with the coordinator is re-established. Alternatively:
- Timeout in INITIAL state
  - Unilaterally abort the transaction.
- Timeout in the PREPARED state
  - Without more information, participant blocked.
  - Could get decision from another participant .

State Transition Diagram for 2PC



Recovery Protocols

u Action to be taken by operational site in event of failure. Depends on what stage coordinator or participant had reached.

Coordinator Failure

u Failure in INITIAL state

– Recovery starts the commit procedure.

u Failure in WAITING state

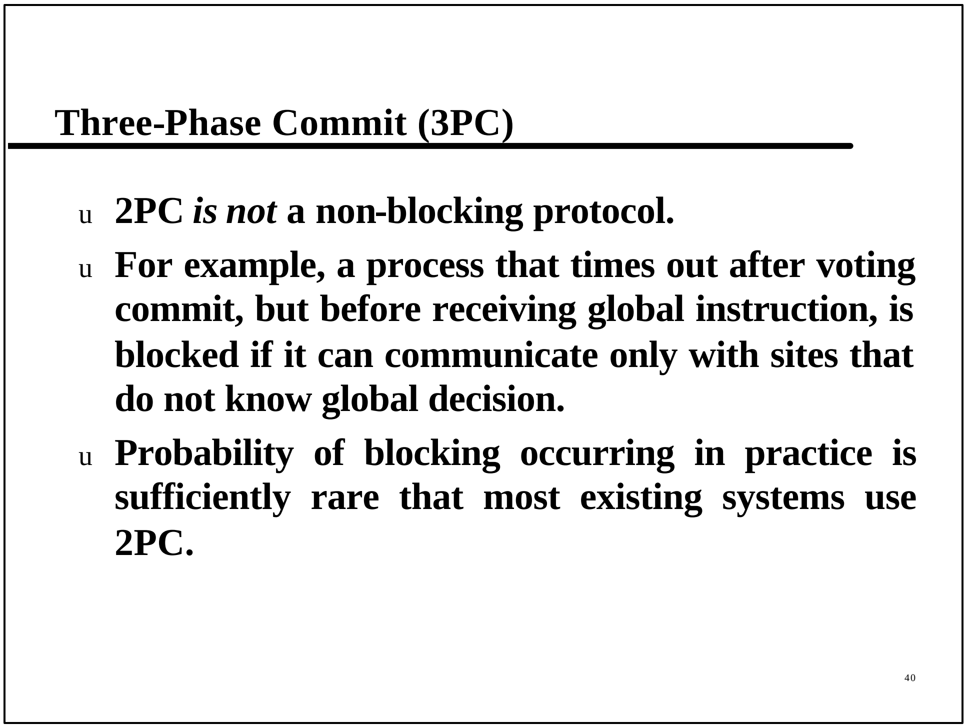
– Recovery restarts the commit procedure.

2PC - Coordinator Failure

- Failure in DECIDED state
  - On restart, if coordinator has received all acknowledgements, it can complete successfully. Otherwise, has to initiate termination protocol discussed above.

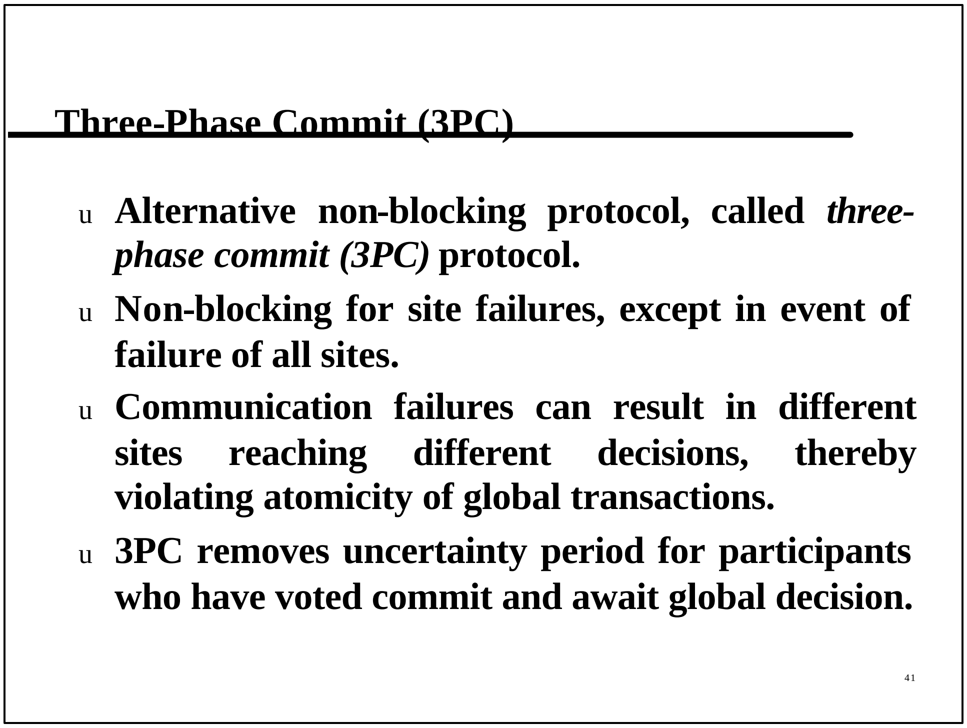
2PC - Participant Failure

- Objective to ensure that participant on restart performs same action as all other participants and that this restart can be performed independently.
- Failure in INITIAL state
  - Unilaterally abort the transaction.
- Failure in PREPARED state
  - Recovery via termination protocol above.
- Failure in ABORTED/COMMITTED states
  - On restart, no further action is necessary.



Three-Phase Commit (3PC)

- **2PC *is not* a non-blocking protocol.**
- **For example, a process that times out after voting commit, but before receiving global instruction, is blocked if it can communicate only with sites that do not know global decision.**
- **Probability of blocking occurring in practice is sufficiently rare that most existing systems use 2PC.**



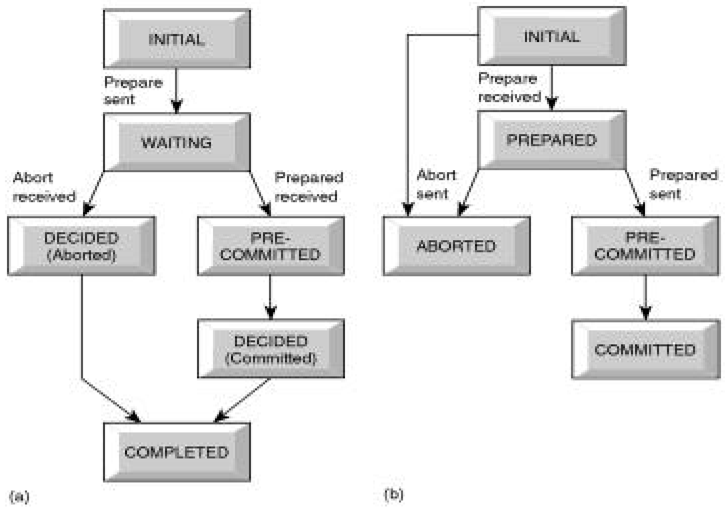
Three-Phase Commit (3PC)

- **Alternative non-blocking protocol, called *three-phase commit (3PC)* protocol.**
- **Non-blocking for site failures, except in event of failure of all sites.**
- **Communication failures can result in different sites reaching different decisions, thereby violating atomicity of global transactions.**
- **3PC removes uncertainty period for participants who have voted commit and await global decision.**

Three-Phase Commit (3PC)

- Introduces third phase, called *pre-commit,* between voting and global decision.
- On receiving all votes from participants, coordinator sends global pre-commit message.
- Participant who receives global pre-commit, knows all other participants have voted commit and that, in time, participant itself will definitely commit.

State Transition Diagram for 3PC


INITIAL
Prepare sent
WAITING
Abort received
Prepared received
DECIDED (Aborted)
PRE-COMMITTED
DECIDED (Committed)
COMPLETED
(a)
INITIAL
Prepare received
PREPARED
Abort sent
Prepared sent
ABORTED
PRE-COMMITTED
COMMITTED
(b)