Outline

- Introduction
- Background
- Distributed DBMS Architecture
- Distributed Database Design
- Distributed Query Processing
- Transaction Management
  - Commit/Termination protocols – 3PC
- Building Distributed Database Systems (RAID)
- Mobile Database Systems
- Privacy, Trust, and Authentication
- Peer to Peer Systems
Useful References

- Textbook *Principles of Distributed Database Systems*, Chapter 12.5
Three-Phase Commit

- 3PC is non-blocking.
- A commit protocols is non-blocking iff
  - it is synchronous within one state transition, and
  - its state transition diagram contains
    - no state which is “adjacent” to both a commit and an abort state, and
    - no non-committable state which is “adjacent” to a commit state
- Adjacent: possible to go from one state to another with a single state transition
- Committable: all sites have voted to commit a transaction
  - e.g.: COMMIT state
State Transitions in 3PC

Coordinator

INITIAL

WAIT

ABORT

PRE-COMMIT

COMMIT

Commit command
Prepare

Vote-abort
Global-abort

Participants

INITIAL

READY

ABORT

PRE-COMMIT

COMMIT

Prepare
Vote-commit

Prepare
Vote-abort

Global-abort
Ack

Prepared-to-commit
Ready-to-commit

Global commit
Ack
Communication Structure (see book)

Phase 1
ready? yes/no

Phase 2
pre-commit/pre-abort? yes/no

Phase 3
commit/abort, ack
Formalism for Commit Protocols

\[ < Q, \Sigma_I, \Sigma_0, \delta, V, A, C > \]

- **Q**: Finite set of states
- **\Sigma_I**: Messages addressed to the site
- **\Sigma_0**: Messages sent by the site
- **\delta**: \((Q, \Sigma^*) \rightarrow (Q, \Sigma^*)\)
- **\forall i \in Q**: Initial state
- **A \subseteq Q**: Abort states
- **C \subseteq Q**: Commit states
Formalism for Commit Protocols

Properties:
1. \( A \cap C = \emptyset \)
2. \( V_i \notin A \) and \( V_i \notin C \)

Protocols are non-deterministic:
- Sites make local decisions.
- Messages can arrive in any order.
Global State Definition

- Global state vector containing the states of the local protocols.
- Outstanding messages in the network
- A global state transition occurs whenever a local state transition occurs at a participating site.
- Exactly one global transition occurs for each local transition.
Global State Graph

Global state is inconsistent if its state vector contains both a commit and abort state.
Two states are potentially concurrent if there exists a reachable global state that contains both local states.

Concurrency set of $s$ is set of all local states that are potentially concurrent with it. $C(s)$

$$C(w_1) = \{ q_2, a_2, w_2 \}$$

The sender set for $s$,

$$S(s) = \{ t/t \text{ sends message } m \& m \in M \}$$

where $M$ be the set of messages that are received by $s$.

t is a local state.
States of Various States in the Commit Protocol

- Global state inconsistent if it contains both
  - local commit state
  - local abort state
- Final state if
  - All local states are final
- Terminal state if:
  - there exists an immediately reachable successor state
  - $\Rightarrow$ deadlock
- Committable state (local) if:
  - all sites have voted yes on committing the transaction
- Otherwise, non-committable
An Example when Only a Single Site Remains Operational

- This site can safely abort the transaction if and only if the concurrency set for its local state does not contain a commit state.

- This site can safely commit only if
  - Its local state must be “committable”
  - And the concurrency set for its state must not contain an abort state.

- A blocking situation arises when
  - The concurrency set for the local state contains both a commit and an abort state
  - Or the site is in a “noncommittable” state and the concurrency set for that state contains a commit state
    - The site can not commit because it can not infer that all sites have voted yes on committing
    - It can not abort because another site may have committed the transaction before crashing.

- These observations imply the simple but powerful result in the next slide.
**Fundamental Non-blocking Theorem**

- **Definition**: protocol is synchronous within one state transition if one site never leads another site by more than one state transition.

- **Theorem** Fundamental non-blocking: A protocol is non-blocking iff
  - There exists no local state $s$ such that $C(s) = A$ (abort) and $C(s) = C$ (commit)
  - And there exists no non-committable state $s$ such that $C(s) = C$ (commit)

- **Lemma**: A protocol that is synchronous within one state transition is non-blocking iff:
  - No local state is adjacent to both a commit & an abort state
  - No non-committable state is adjacent to a commit state
Three-Phase Commit Protocol

Site I (co-ordinator)

\[
\begin{align*}
\text{q}_1 & \xrightarrow{\text{request}} \text{w}_1 \\
\text{w}_1 & \xrightarrow{\text{yes}_1 \cdots \text{yes}_n, \text{prepare}_1 \cdots \text{prepare}_n} \text{p}_1 \\
\text{p}_1 & \xrightarrow{\text{ack}_1 \cdots \text{ack}_n} \text{c}_1 \\
\text{c}_1 & \xrightarrow{\text{commit}_1 \cdots \text{commit}_n}
\end{align*}
\]

Site i (i = 2,3,…n) (slave)

\[
\begin{align*}
\text{q}_i & \xrightarrow{\text{xact}_i} \text{w}_i \\
\text{w}_i & \xrightarrow{\text{yes}_i, \text{prepare}_i} \text{p}_i \\
\text{p}_i & \xrightarrow{\text{ack}_i} \text{c}_i \\
\text{c}_i & \xrightarrow{\text{commit}_i, \text{no}_i}
\end{align*}
\]