Info

• Quiz 5 average is 24.63 / 30.

• Today
  • Optimistic Concurrency Control
  • Two-Phase Commit
  • Three-Phase Commit

• Quiz next Tuesday
Serializability

- Two operations **conflict** if they operate on the same data time and at least one of them is a write.

- Two **histories are equivalent** if they contain the same transactions and they order conflicting operations (of non-aborted transactions) in the same way.

- A history is **serializable** if it is equivalent to some serial history.
• Are the following histories serializable?

(a) \( r_1(x) \ w_2(x) \ c_2 \ w_3(y) \ c_3 \ r_1(y) \ c_1 \)

(b) \( r_1(z) \ r_2(y) \ w_1(y) \ w_3(x) \ c_3 \ w_1(x) \ c_1 \ r_2(x) \ c_2 \)
Consider the following input to a 2PL scheduler:

\[ r_1(z) \quad r_2(y) \quad w_1(y) \quad w_3(x) \quad c_3 \quad w_1(x) \quad c_1 \quad r_2(x) \quad c_2 \]

What will the schedule output be?
Optimistic Concurrency Control

• 2PL is an example of pessimistic concurrency control.
  • If conflicts are rare, locks may be obtained unnecessarily.

• Alternate approach: optimistic concurrency control (OCC).
  • No locks
  • Scheduler aborts transactions to maintain serializability.

• Based on Kung and Johnson, 1981
OCC Phases

- Every transaction is processed in 3 phases:
  - 1. **Read Phase**: data items are read from database, and all writes are done in local workspace
  - 2. **Validation phase**: When transaction is ready to commit, the scheduler tests whether transaction execution is correct in terms of serializability. If not, transaction is aborted. Otherwise, go to phase 3.
  - 3. **Write phase**: Local writes are applied to the database.

- Assumption: Validation and Write phases are executed as single critical section.
Definitions

- **RS(t_i):** Read set of transaction \( t_i \).
  Set of data items read by transaction \( t_i \).

- **WS(t_i):** Write set of transaction \( t_i \).
  Set of data items written by transaction \( t_i \).
COMMIT PROTOCOLS

P.A. Bernstein, V. Hadzilacos, and N. Goodman, Concurrency Control and Recovery in Database Systems. Chapter 7
The Commit Problem

- \( n \) processes
- Every process has an initial value, either 1 (commit) or 0 (abort).
- Want all processes to reach the same decision (commit or abort).

- **Crash-recovery failure model**: processes may fail, but they will eventually recover.

- Synchronous system.

- No message loss
Requirements

• **Agreement:**
  - No two processes decide on different values.

• **Validity:**
  - If any process starts with “abort”, then “abort” is the only possible decision value.
  - If all processes start with “commit” and there are no failures, then “commit” is the only possible decision value.

• **Weak Termination:**
  - If at any point, all failures are recovered, and there are no failures for sufficiently long, then all processes will eventually decide.
Two-Phase Commit Algorithm

- Two-phase commit is the most commonly used commit protocol.
- On process is the **coordinator**.
- The others are **participants**.
- Coordinator collects votes, makes decision, and sends decision to participants.
Phase 1: Voting

1. Coordinator sends a “vote request” message to all participants.
Phase 1: Voting

1. Coordinator sends a “vote request” message to all participants.
2. Participants respond with vote, “commit” or “abort”.

Diagram:
- C (Coordinator)
- P_1 (P1) with “commit”
- P_2 (P2) with “commit”
- P_3 (P3) with “commit”
- P_4 (P4) with “abort”
Phase 2: Commit

1. If **all** participants voted “commit”, coordinator sends “commit” response. Otherwise, coordinator sends “abort” response.
2. When participant receives response, it decides “commit” or “abort”.

![Diagram](image-url)
Analysis of Two-Phase Commit

- If no failures
  - Does Two-Phase Commit satisfy Agreement?
    - Yes, because single decision value is determined by coordinator
  - Does Two-Phase Commit satisfy Validity?
    - Yes, because coordinator determines decision value based on votes from all processes.
  - Does Two-Phase Commit satisfy Termination?
    - Yes, all processes decide at by completion of 2\textsuperscript{nd} phase.

- What happens when there are failures?
Analysis of Two-Phase Commit

- Message complexity (without failures):
  - Voting phase takes $2n$ messages.
    - $n$ messages: coordinator sends request to participants.
    - $n$ messages: participants send vote to coordinator.
  - Commit phase takes $n$ messages.
    - $n$ messages: coordinator sends decision to participants.
Failures in Two-Phase Commit: Voting

- If coordinator fails before sending “request” messages to a participant, participant detects failure and decides abort.
Failures in Voting Phase

- If participant fails before sending vote, coordinator detects failure, and sends “abort” to all participants.
Failures in Two Phase Commit

- If **coordinator fails** before sending decision to all processes, a process cannot decide to commit or abort unilaterally.
  - They need to collaborate.
  - This collaboration is called a **termination protocol**.
Simplest Termination Protocol

- Simplest termination protocol – wait for coordinator to come back online.

- Participants may be blocked unnecessarily.
  - If one process has already decided, can just ask that process for its decision.
Cooperative Termination Protocol

• On detecting coordinator failure, any process p that is uncertain (hasn’t decided commit or abort) initiates protocol – sends decision request to all processes.

• When a process q receives the decision request:
  • (1) If q has has already decided, send decision .
  • (2) If q has not yet voted, q aborts and sends abort
  • (3) If q has voted, but has not decided, q cannot help p.

• If p finds a q that can execute (1) or (2), then p can decide.
• Otherwise, p is blocked until such a q is found or the coordinator comes back online.
Analysis of Two-Phase Commit

- Two-phase commit with cooperative termination protocol satisfies agreement, validity, and termination.
- But it is a blocking protocol.