Rollback Recovery in Message Passing Systems

By Philip Lee

I. Overview

- Rollback Recovery
  1. What it is and why we need it
  2. Factors behind algorithms
- Checkpoint-Based Rollback Recovery
  1. Uncordinated Checkpointing
  2. Coordinated Checkpointing
  3. Communication-induced Checkpointing
- Log-Based Rollback Recovery
  1. Pessimistic Logging
  2. Optimistic Logging
  3. Causal Logging
- Summary

II. Rollback Recovery

- Why do we need efficient recovery in message passing systems?
  1. Costly messaging can be avoided
  2. All data is not computed repeatedly
  3. The algorithms are suited for different situations
- Factors we need to consider
  1. Stable Storage
  2. Input/Output protocols
  3. Garbage collection
  4. Consistency

Consistency

Fig. 2. An example of a consistent and inconsistent state.

III. Checkpoint-Based Rollback Recovery

- Basic Strategy
  1. Create checkpoints on each process.
  2. During a failure, rollback to the last checkpoint.
  3. Check the consistency among other processes.
  4. Rollback further if inconsistent (i.e. dependency on a variable).

Stable Storage

Fig. 4. Message logging for deterministic replay.
Uncoordinated Checkpointing (cont’d)

- Disadvantages
  1. Domino effect possibility
  2. Useless checkpoints
  3. Garbage collection of useless checkpoints
  4. Not suitable for frequent output because of necessary global coordination

Coordinated Checkpointing

Non-Blocking Checkpoint Coordination

- Advantages
  1. Simplifies recovery.
  2. Avoids the domino effect.
  3. Only one permanent set of states in stable storage.

- Disadvantages
  1. However, there exists high latency for interaction with the outside world (global checkpoint necessary).

Advantages vs. Disadvantages

- Other Coordinated Protocols

  - Checkpointing with Synchronized Clocks
    1. One process takes a checkpoint.
    2. Wait for max_clock_deviation_time + max_failure_detection_time.
    3. All errors will be reported within the allotted wait time.

  - Checkpointing and Communication Reliability
    1. Consider a scenario:
     a) p sends a message before taking a checkpoint.
     b) q takes a checkpoint before receiving its.
     c) a failure forces a rollback-recovery to these two checkpoints.
    2. Solution: all in-transit messages are saved by the destinations as part of the recorded state.

  - Minimal Checkpoint Coordination
    1. Two-phase protocol.
    2. The process is unable to send messages until the checkpoint is completed successfully, but it can still receive messages.

Communication-Induced Checkpointing

Fig. 9. Z-paths Z cycles.
CIC Protocols

- Model-based Protocols
  1. A model to detect possible Z-formations is created using some heuristic.
  2. Force checkpoints to break these formations.
  3. Disadvantages
     a) Multiple processes may force independent checkpoints for the same Z-formation.
     b) Checkpoints may be forced for Z-formations that never emerge.

- Index-based Protocols
  1. Timestamps are used with two requirements:
     a) Given two checkpoints \( c_i, m_i \) and \( c_j, n_j \), then \( ts(c_j, n_j) \geq ts(c_i, m_i) \).
     b) Consecutive local checkpoints of a process have increasing timestamps.
  2. These timestamps are used to determine if there are any Z-formations detected or beginning to emerge.
  3. Forced checkpoints are made to avoid such circumstances.

Advantages and Disadvantages

- Advantages
  1. Avoids domino effect
  2. Does not require checkpoints to be coordinated

- Disadvantages
  1. Mentioned for model-based protocols
  2. Depends on when checkpoint-decision is made

Checkpointing Implementations

- Concurrent Checkpointing
- Incremental Checkpointing
- System-level vs. User-level
- Compiler Decisions
- Checkpoint Heuristics Research

IV. Log-Based Rollback Recovery

- Basic Strategy
  1. Founded on the idea that process execution can be modeled as a sequence of deterministic state intervals, each starting with the execution of a nondeterministic event. ([Strøm and Yemm 1985]).
  2. Keep track of nondeterministic events as determinants on stable storage
     a) System Calls
     b) Asynchronous signals
  3. Create checkpoints
  4. Rollback to consistent checkpoints upon failure to reduce recovery work
  5. Use determinants to replay all failure-free operations before the failure

The No-Orphan Condition

- Process \( p \) is an orphan when it does not fail, but its state cannot be recovered because of missing dependent information.
- Formal definition
  a) Formally stated as:
  \( \forall c : \neg(S(c) \Rightarrow D(c) \subseteq L(c)) \)

Pessimistic Logging

- Fig. 4: Message logging for deterministic replay.
- Fig. 10: Pessimistic logging.
Optimizing Pessimistic Logging

- Reducing Performance Overhead
  1. Special hardware for atomic logging and/or stable storage.
  2. Sender Based Message Logging (SBML) protocol

- Relaxing Logged Atmocity
  1. Instead of $\forall e: \neg S(e) \Rightarrow |D(e)| > 0$, we have $\forall e: \neg S(e) \Rightarrow |D(e)| \leq 1$.

Advantages and Disadvantages

- Advantages
  1. Processes communicate with outside world without special protocol.
  2. Processes are replayed from their last checkpoints.
  3. Recovery is restricted only to the processes that fail.
  4. Garbage collection is easy to manage.

- Disadvantage
  1. Performance penalty on synchronous logging

Optimistic Logging

![Optimistic Logging Diagram]

Fig. 11. Optimistic logging.

Optimistic Logging (cont’d)

- Synchronous vs. Asynchronous
  1. Synchronous recovery requires that all processes use of dependency and logged information to compute the maximum recoverable state.
  a) Dependency Tracking: Direct and Transitive
  2. Asynchronous recovery begins with the failed process broadcasting a rollback
  3. Exponential Rollbacks (Figure 12)

![Optimistic Logging (cont’d) Diagram]

Fig. 12. Exponential rollbacks.

Causal Logging

![Causal Logging Diagram]

Fig. 13. Causal logging (a) Maximum recoverable states, and (b) precedence graph of $P_a$ at state $X$.

Causal Logging (cont’d)

- Tracking Causality
  1. Piggybacking of determinants onto messages
  2. In failure-free operation, the local precedence graph is piggybacked.
  3. The piggybacked data can be huge if several graphs are sent within one message. However, the only differences between a process’ graph and the graph it receives from another process is the data of its own that was sent, and the data that was sent to it from that other process.
  4. Family Based Logging (FBL)
Advantages and Disadvantages

• Advantages
  1. It avoids synchronous access to stable storage unless for output commit.
  2. It allows each process to commit output independently.
  3. The rollback of a failed process goes to the last set of checkpoints.
  4. Deterministic log of each process protects it from other failed processes.

• Disadvantages
  1. More complex recovery protocol

Log-Based Implementations

• Message Logging Maintenance
  1. Three sources of overhead
  2. Sender vs. Receiver logging

• Messages with Coordinated Checkpointing
  1. Simpler garbage collection and recovery.
  2. Fast output commit

Summary

<table>
<thead>
<tr>
<th>Message Logging</th>
<th>Coordinated Checkpointing</th>
<th>Both</th>
<th>Continuous Garbage Collection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recoverable</td>
<td>Recoverable</td>
<td>No</td>
<td>Recoverable</td>
</tr>
<tr>
<td>Stable</td>
<td>Stable</td>
<td>No</td>
<td>Stable</td>
</tr>
<tr>
<td>Communication</td>
<td>Communication</td>
<td>No</td>
<td>Communication</td>
</tr>
<tr>
<td>Storage</td>
<td>Storage</td>
<td>No</td>
<td>Storage</td>
</tr>
</tbody>
</table>

Summary (cont’d)

• Trends
  1. Commercial message passing uses pessimistic logging because of simpler recovery.
  2. Stable storage access is a major overhead factor.
  3. Message logging systems have less pessimistic protocols for committing output and logging input that is not possible in checkpoint-only systems.
  4. The complexity of nondeterminism can be supported for message logging systems.

• Implementation Factors to Consider
  1. Communication Protocols
     a) Location-Independent Identities and Redirection
     b) Reliable Channel Protocols
  2. Stable Storage
     a) Technological changes