Failure Model

- System crash
  - CPU halts
  - Memory lost
  - Data on disk is OK
- Everything else

Recover from System Crash

- Data recovery
- Recovery incomplete transactions
  - Removed changes made by un-committed transaction – Undo
  - Reapply changes made by committed transactions – Redo

Logging

- Log
  - A sequence of log records
  - Append only
  - Undo, Redo, Undo/Redo
- Common log records
  - <START T>
  - <COMMIT T>
  - <ABORT T>

Undo Logging

- Log records
  - <T,X,v>
- Rules
  - U₁: Write <T,X,v> before writing new value of X to disk
  - U₂: Write <COMMIT T> after writing all new values to disk
- Flush log

Undo Logging Example

1. READ(A,t)
2. t = t + 2
3. WRITE(A, t)
4. READ(B,t)
5. OUTPUT(A)
6. OUTPUT(B)
T1: Read (A,t); t ← t×2
    Write (A,t);
    Read (B,t); t ← t×2
    Write (B,t);
    Output (A);
    Output (B); failure!

More about Undo Recovery
- System failure during recovery
  - Undo recovery is idempotent
- Checkpointing
  - Simple checkpointing
    - \texttt{<CKPT>}

Nonquiescent Checkpointing
- Write \texttt{<START CKPT(T1,...,T_k)>} and \texttt{flush log}
  - \(T_1,...,T_k\) are active transactions, e.g.
    transactions that have not completed
  - Wait until \(T_1,...,T_k\) commit or abort
  - Write \texttt{<END CKPT>} and \texttt{flush log}

Redo Logging
- Log records
  - \texttt{<T,X,v>}
- Rules
  - R1: write \texttt{<T,X,v>} and \texttt{<COMMIT T>}
    before writing new value of \(X\) to disk

Redo Logging Example
- Write log record
  - \texttt{<START T>}
  - \texttt{<COMMIT T>}
  - \texttt{<ABORT T>}
  - \texttt{<T,X,v>}
- Flush log
  - (1) \quad \texttt{READ(A,t)}
  - (2) \quad t = t\times2
  - (3) \quad \texttt{WRITE(A, t)}
  - (4) \quad \texttt{READ(B,t)}
  - (5) \quad \texttt{OUTPUT(A)}
  - (6) \quad \texttt{OUTPUT(B)}
Redo Recovery

- Scan the complete log, and re-apply changes made by transactions that have a <COMMIT T> record
- Implementation issues
  - Redo order
    - latest to earliest??
    - earliest to latest??

Checkpointing a Redo Log

- Write <START CKPT(T1,...,Tn)> and flush log
- ??
- Write <END CKPT> and flush log

Redo Checkpointing Example

```plaintext
<START T1>
...
<START T2>
...
<START T3>  there is a <END CKPT> in log??
...  there is no <END CKPT> in log??
<COMMIT T2>
...
<START CKPT(??)>  ??
<END CKPT>
...
```

Undo vs. Redo

- Undo: I/O may be too frequent
- Redo: I/O may be too infrequent
- Both: may lead to contradictory requirements on buffer management

Undo/Redo Logging

- Log records
  - <T,X,Y,W>
- Rules
  - UR1: write <T,X,Y,W> before writing new value of X to disk

Undo/Redo Recovery

- Redo all the committed transactions
- Undo all the incomplete transactions
Checkpointing an Undo/Redo Log

- Write <START CKPT(T_1, ..., T_n)>
- Flush all dirty pages
- Write <END CKPT>

Undo/Redo Checkpointing Example

- <START T1>
- ...
- <START T2>
- ...
- <START T3>
- ...
- <COMMIT T2>
- ...
- <START CKPT(??)>
- ??
- <END CKPT>
- ...