CSCI 234

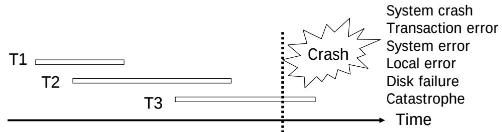
Design of Internet Protocols:
Database Recovery

George Blankenship

Outline

- Basic Principles
- Logging
- Logging algorithms
- Checkpoint
- Rollback algorithms

Why "Database Recovery Techniques"?



- ACID properties of Transaction

Database system should guarantee

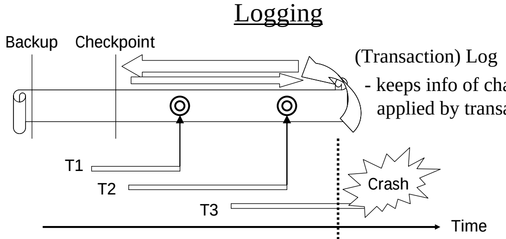
- Durability : Applied changes by transactions must not be lost. ~ T3

- Atomicity : Transactions can be aborted. ~ T1, T2

ACID

- **Atomicity** states that database modifications must follow an “all or nothing” rule If one part of the transaction fails, the entire transaction fails.
- **Consistency** states that only valid data will be written to the database. If, for some reason, a transaction is executed that violates the database’s consistency rules, the entire transaction will be rolled back and the database will be restored to a state consistent with those rules. On the other hand, if a transaction successfully executes, it will take the database from one state that is consistent with the rules to another state that is also consistent with the rules.
- **Isolation** requires that multiple transactions occurring at the same time not impact each other’s execution.
- **Durability** ensures that any transaction committed to the database will not be lost. Durability is ensured through the use of database backups and transaction logs that facilitate the restoration of committed transactions in spite of any subsequent software or hardware failures.

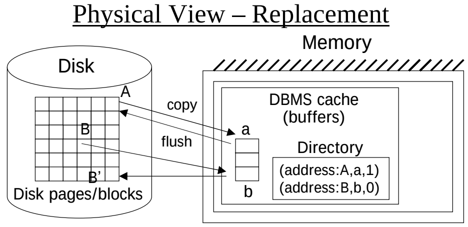
Logging



(Transaction) Log
- keeps info of changes applied by transactions

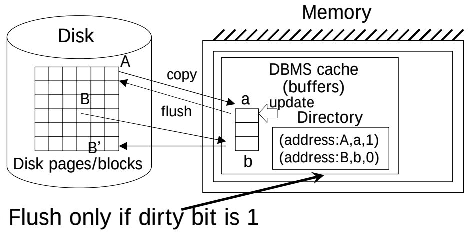
- Transaction Recovery
  - (Transaction) Log } Non-catastrophic failure
- Full DB Backup
  - Differential Backup
  - (Transaction) Log } Catastrophic failure

Physical View – Replacement



1) Check the directory whether in the cache
2) If none, copy from disk pages to the cache
3) For the copy, old buffers needs to be flushed from the cache to the disk pages

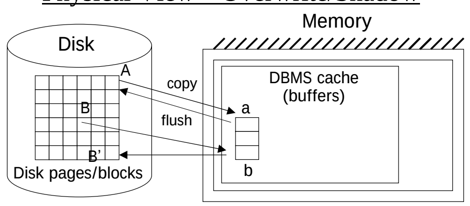
Physical View – Update



4) Flush only if dirty bit is 1

Dirty bit : (directory) indicates a change after copy to the cache
- 1 – updated in the cache
- 0 – not updated in the cache (no need to flush)

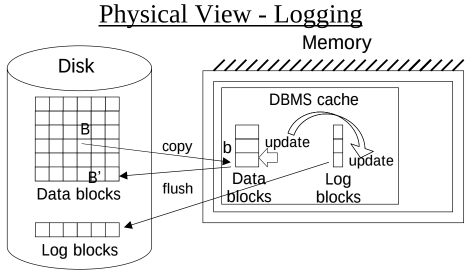
Physical View – Overwrite/Shadow



A-a : "in-place updating"
- when flushing, overwrite at the same location
- logging is required

B-b : "shadowing"
- logging is not required

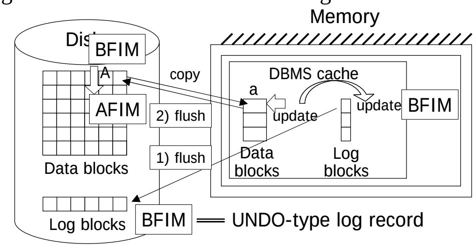
Physical View - Logging



(1) copy (from the disk to the cache)
(2) update the cached data, record it in the log
(3) flush the log and the data

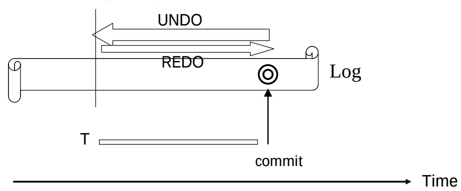
Write-Ahead Logging

- in-place updating → A log is necessary
  BFIM (BeFore IMage) – overwrite – AFIM (After IMage)
- WAL (Write-Ahead Logging)
  Log entries flushed before overwriting main data



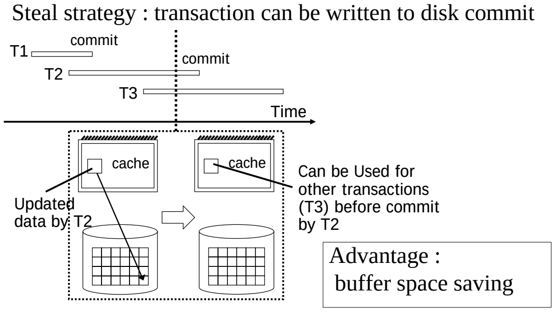
Write-Ahead Logging Protocol

- protocol requires UNDO and REDO
- BFIM cannot be overwritten by AFIM on disk until all UNDO-type log entries have been force-written to disk.
- The commit operation cannot be completed until all UNDO/REDO-type log have been force-written.



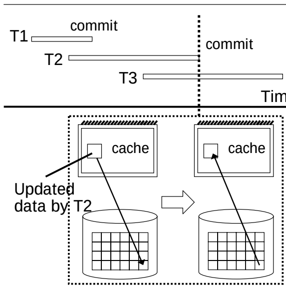
Steal Strategy

- Typical DB employs a steal/no-force strategy
- Steal strategy : transaction can be written to disk commit



No-Force Strategy

- No-Force strategy : a transaction need not to be written to disk immediately when it commits



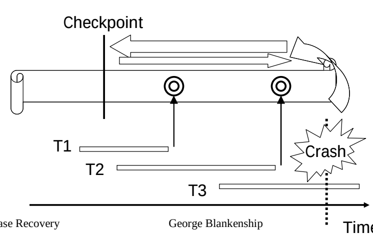
Advantage :
I/O operations saving

If T3 needs the same data, it must be copied again when T2 commits

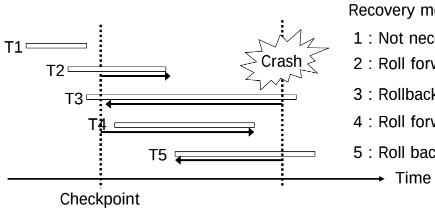
← Force strategy

Checkpoint

- All DMBS buffers modified are wrote out to disk.
- A record is written into the log. ([checkpoint])
- Periodically done (every n min. or every n transactions)



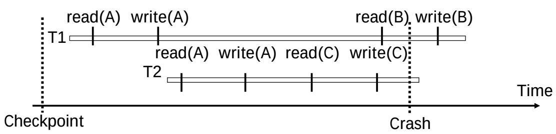
System Crash



Recovery method
1 : Not necessary
2 : Roll forward
3 : Rollback
4 : Roll forward
5 : Roll back

- transaction may be written on disk before it commits

Transaction Rollback



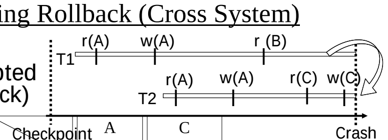
Name	Account
Mr.A	$10
Mr.B	$2,000
Mr.C	$30,000

T1 : A company pays salary to employees
- i) transfer $2,000 to Mr. A's account
- ii) transfer $2,500 to Mr. B's account …

T2 : Mr.A pays the monthly rent.
- i) withdraw $1,500 from Mr.A's account
- ii) transfer $1,500 to Mr.C's account

Database Recovery George Blankenship 16

Cascading Rollback (Cross System)



-T1 is interrupted (needs rollback)

System Log	A	C
[checkpoint]	$10	$30,000
[start_transaction, T1]		
[read_item, T1, A]	$10	
[write_item, T1, A, 10, 2010]	$2,010	
[start_transaction, T2]		
[read_item, T2, A]	$2,010	
[write_item, T2, A, 2010, 510]	$510	
[read_item, T1, B]		
[read_item, T2, C]		$30,000
[write_item, T2, C, 1500, 31500]		$31,500
~~ CRASH ~~~		

-T2 uses value modified by T1 (also needs rollback)

Database Recovery George Blankenship 17

Categorization of Recovery Algorithm

- Deferred update – the No-UNDO/REDO algorithm
- Immediate update – the UNDO/REDO algorithm

Database Recovery George Blankenship 18