Recovery Management

CISC437/637, Lecture #19

Ben Carterette

ACID Review

Atomicity – all actions in the transaction happen or none do

Consistency – each transaction starts with consistent database and ends with consistent database

Isolation – each transaction is executed independently of others

Durability – after a transaction commits, its effects persist

- Concurrency control/locking ensures consistency and isolation
- The **recovery manager** ensures atomicity and durability

Assumptions

- Assume the database is using concurrency control
  - Specifically, assume strict two-phase locking
  - (Transactions do not release exclusive locks until after committing)
- Assume updates happen in place
  - Updated data is overwritten or deleted on disk; no redundancy in storage
  - (Production database would usually use some storage redundancy)
- Assume writing a page to disk is atomic
  - The system won't crash while a write is in progress
  - (Production database would not assume this)

Data Access

- Up to now, we have assumed that data access happens via disk access only
- In reality, disk accesses are used to transfer data to memory
  - Data may persist in memory after transaction accessing it completes
  - The buffer manager decides when to move data from memory back to disk
- **Physical pages** are pages stored on disk ("permanently")
- **Buffer pages** are pages temporarily in main memory

Data Access

- Pages move from disk to memory via two operations:
  - Input(B) transfers a physical page B to memory
  - Output(B) transfers a buffer page B to disk, replacing the corresponding physical page

- Each transaction has its own memory space for local copies of the data it needs
  - For transaction Ti accessing data item O, we'll refer to its local copy as Oi

Data Access

- Transactions transfer data from the buffer to their private workspace
  - Read(O) assigns the O in memory to Oi in the local workspace
  - Write(O) assigns the value of Oi in the local workspace to the O in memory
  - If the physical page in which O lives is not in memory, an Input() may have to happen first

- Transactions only operate on the data in the buffer
  - A buffer manager determines when to perform input()s and output()s
  - Output()s may not happen immediately after write()s

