Lecture 5: TM – Lazy Implementations

- Topics: TM design (TCC) with lazy conflict detection and lazy versioning, intro to eager conflict detection

Design Space

- Data Versioning
  - Eager: based on an undo log
  - Lazy: based on a write buffer
    Typically, versioning is done in cache;
    The above two are variants that handle overflow

- Conflict Detection
  - Optimistic detection: check for conflicts at commit time (proceed optimistically thru transaction)
  - Pessimistic detection: every read/write checks for conflicts (so you can abort quickly)

Basic Implementation – Lazy, Lazy

- Writes can be cached (can't be written to memory) – if the block needs to be evicted, flag an overflow (abort transaction for now) – on an abort, invalidate the written cache lines

- Keep track of read-set and write-set (bits in the cache) for each transaction

- When another transaction commits, compare its write set with your own read set – a match causes an abort

- At transaction end, express intent to commit, broadcast write-set (transactions can commit in parallel if their write-sets do not intersect)