Concurrency control thus far

We learned how to enable concurrent read/write to a shared resource

- E.g., Locking [Mesa]
- If there are multiple resources, they are independent from the system’s perspective

Haven’t learned yet how to enable concurrent read/write to multiple related resources

- E.g., two related values in a DB
Strawmen solutions to related data

1. Lock everything
   • Too slow

2. Clients lock what they read/write (relationships hidden from the system)
   • Impossible to recover if they client crashes midway
Transaction

Unit of execution that read/writes one or more data items

- System is aware of work done as part of the transaction
- Goal is to make transactions atomic – all or nothing
Faults thus far

Replication faults
  • Replicas may disagree because of "natural" faults or attacks

But faults are not only about replication
  • Must minimize the probability of data corruption in non-replicated systems?
    • Related data items left in an inconsistent state is corruption
  • Also recall that handling replication faults assumed that each node has stable storage
Today’s and Monday’s papers

Concurrency control and recovery
  • Today: all data on one computer
  • Monday: data on multiple computers

The DB community has done the heavy lift here on these topics but the utility is not limited to databases
Over to Jieliang and Nick