ACS Theory Assignment 1

Kai Arne S. Myklebust, Silvan Adrian

Handed in: November 29, 2018

Contents

1 Question 1: Techniques for Performance

1.1 Question 1.1

Concurrency may improve latency, but one must be careful because of locking and correctness. While using concurrency two processes may need to write on the same element and then one process needs to be locked and latency may get worse.

1.2 Question 1.2

Batching is when you run multiple requests at once. One example is billing in a credit card company, where they run a monthly billing cycle. All data gets collected at the end of the month and sent at once.
Dallying is when you wait until you have some requests accumulated and then run them. One example is where a request comes in too overwrite a disk block, but waits for more requests that may overwrite the same disk block. Then the first request would not be needed anymore.
Batching may improve latency and throughput, but dallying typically incurs a latency penalty because it waits for more requests.

1.3 Question 1.3

Yes, because it makes one optimized path for common requests, where it tries to eliminate the need for reads and writes in lower level memory. But only goes deeper when needed.

2 Question 2: Fundamental Abstractions

2.1 Question 2.1

2.2 Question 2.2

2.3 Question 2.3

2.4 Question 2.4

3 Question 3: Serializability & Locking

3.1 Precedence graph

3.1.1 Schedule 1

Yes it is conflict-serializable, because there is no cycle in the precedence graph.

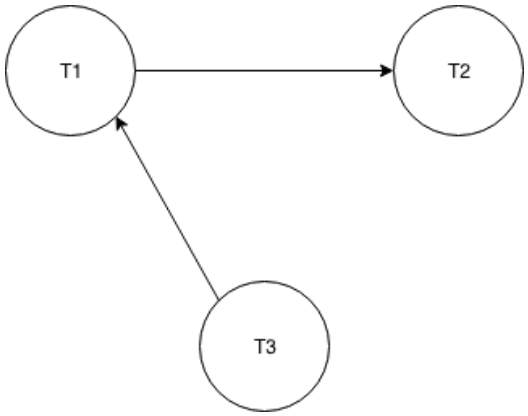

Figure 1: Precedence graph of schedule 1

3.1.2 Schedule 2

Yes, since the precedence graph has no precedences and therefore also no cycle.

3.2 Strict 2PL scheduler

3.2.1 Schedule 1

Transaction 2 has a write on X but there is already a shared lock on X in T1, that's why it has to abort.

Figure 2: Schedule 1, strict 2PL

3.2.2 Schedule 2

Yes it can be generated by strict 2PL scheduler, see figure.

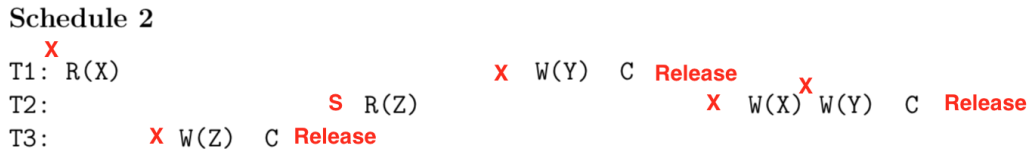

Figure 3: Schedule 2, strict 2PL

4 Question 4: Optimistic Concurrency Control