Name: Joe Kuczek
Section: E
University ID: 011405426

Project 2 Report

Summary:

10pts

I learned how to build a multi-threaded program using the pthread library. I also learned about the complexities of dealing with mutexes and the differences between fine-grain locking and course-grain locking. I also spent some time considering pthread condition variables, even though I didn't end up using them in my project. I also went back and had to relearn some C, there are many tedious intricacies that go along with it. Overall, I learned a lot from this project.

One thing to note in my program is that I am processing each transaction (account ID, value) as a separate request in the output file. This means that if you type "TRANS 1 100 2 100" there will be two records in the output file for each account. I know this isn't what the specification states, I apologize but I didn't see this part of the specification until right before. At the time, I also figured this solution would be much more efficient, since we are only locking an account if we are going to write to it, not if we plan on writing to it.

Part II:

6.2:

5pts Average for each program:

Ignored – Test script not working

6.3:

3.2.1:

3pts Which technique was faster - coarse or fine grained locking?

Fine-grain locking

3pts Why was this technique faster?

This is because we don't have to lock the entire bank each time we perform a transaction. The bank becomes locked and we can't take advantage of parallelism.

3pts Are there any instances where the other technique would be faster?

Yes, if there are multiple threads that need to perform similar operations on a large component. One example would be if we are performing an ALTER TABLE statement on a database in SQL. It would be more efficient to put one course grain lock on the entire table as opposed to putting a fine-grain lock on every single row of the table.

3pts What would happen to the performance if a lock was used for every 10 accounts? Why?

You would probably see a performance between the course-grain locking and fine-grain locking. This is because we aren't locking every single account, but we aren't locking all the accounts at once. We are doing something in between, so we would expect the performance to be in between as well, probably slightly worse than fine-grain locking.

3pts What is the optimal locking granularity (fine, coarse, or medium)?

It depends on the program you are creating. Fine grain works better if smaller components need to be accessed by many different threads. Course grain locking works better if there are large components being passed around threads.