PROJECT PROGRESS REPORT

Team: Jessica Lynch and Andrew Arnopoulos

Project Title: Performance measurement/Performance model for concurrent Client-Server program.

Overview and summary:

The goal of our project is to determine which of the three contrasting methods below provide the fastest running time under various circumstances by testing the performance of messaging between client and server.

1. Shared memory client-server program
2. TCP client-server program
3. Client-server program using named pipes

To vary the circumstances under which these implementations were run and tested, we modified the following settings: number of threads, number of concurrent clients, number of cores, and message load. We also faced the following concurrency issues: dealing with multiple clients sending and receiving data, ensuring a fair and starvation free solution in terms of both readers and writers, and maintaining data integrity for multiple writers. As our platform/environment for development and testing, we used a Windows 2-core laptop and CSEL server (8-core) for performance measurement (i.e. elra-01.cs.colorado.edu). The programming languages, APIs, standards, etc. used in our project included Java, C, JNI (Java Native Interface), TCP and POSIX.

Progress and Completion Timeline:

- Week of 2/16: Proposal submitted; delegated initial tasks and commenced implementation of TCP client-server program in Java.
- Week of 2/23: Completed TCP client-server program.

Progress and Completion Timeline (continued):

- Week of 3/2: Delegated tasks to begin writing the shared memory client-server program. Commenced implementation of this method.
- Week of 3/9: Finished writing shared memory queue in C. Learned JNI. Commenced implementation of JNI link to shared memory queue.
- Week of 3/16: Learned named pipe with Java method and commenced implementation of this method.
- Week of 3/23: Finished writing client program using named pipes. Designed test suite (not yet coded up).
- Week of 3/30: All three client-server programs were mostly completed by checkpoint. JNI implementation for shared memory client-server program will be completed by April 5 end-of-day, which will complete the shared memory implementation.

Details of Our Progress:

We completed the following three implementations of a client-server program since the submission of our project proposal:

1. Shared memory client-server program in C, which involves implementation of a shared memory library (written in C) linked to by JNI, uses POSIX shared memory through queueing where memory is shared globally among all processes, and is locked using POSIX Pthread locks.
2. TCP client-server program written in Java, which uses TCP for the transport of data and queueing for local shared memory among multiple threads, and is locked using Pthread mutex locks.
3. Client-server program in Java using named pipes, which uses java pipes to provide communication between the client and the server, and is locked using standard Java locks.

Details of Our Progress (continued):

We also gave our client-server program a purpose by writing a module that serves as a translator and uses each implementation of client-server to translate single words and up to 250 words. It is a simple design which doesn't handle punctuation and can only translate word-by-word. An existing concern, however, is that adding this feature will take away from the primary goal of our project since we are now no longer dealing with straight queueing. The translator functionality naturally adds a data conversion aspect which will likely slow our performance testing and could slow it significantly as settings are maximized during the experiments.

Certain challenges we faced with the above implementations included not allotting enough time for implementing the JNI for the shared memory client-server program in C. We will therefore have it implemented prior to the week of April 6. Initial testing has shown slow run times likely due to the translator functionality we added.

Timeline illustrating the work needing to be done until project deadline:

- Week of 4/6: Enhance and finish test suite, and commence testing of implementations. Start and finish collecting and analyzing performance measurements. Start assembling submission with source code and project report.
- Week of 4/13: Complete submission with source code and project report. Start presentation.
- Week of 4/20: Finish and rehearse presentation.
- Week of 4/27: ***April 27 Project Submission deadline*** (submit source code, examples, project report). Present project to class this week.