NOVA SCHOOL OF SCIENCE & TECHNOLOGY
DEPARTAMENTO DE INFORMÁTICA

Concurrent Programming with Semaphores

LAB 6

Introduction and Lab objective

The objective of this lab includes comparing process creation/termination with pthread creation/termination and also using the *Pthreads* library for multi-threaded file copy using the producer-consumer strategy. This strategy will try to dispatch read and write system calls in parallel.

Comparing process creation/termination with thread creation/termination

Perform the following experiments where the time taken by different system and library calls related with process management is measured.

1. **fork + wait system calls** – evaluate the time to create and terminate a new process. Remember the timing.c from Lab 01. In the function do_something just fork a new process and wait for it; note that the child process should exit immediately.

2. **pthread_create + pthread_join calls** – now evaluate the time to execute and terminate a new thread. In the do_something create a new thread and join to it. The function run by the new thread just needs to return.

Multithreaded file copy

The goal of this work is to implement a file copy (e.g. similarly to Lab 01) using a file reader thread and a file writer thread. A producer-consumer strategy will allow those threads to exchange the file contents so that the *reader thread* reads data from a file and sends it to the *writer thread* that writes it to the new file.

To implement your solution, the main program is in mtfcopy.c, and is similar to the fcopy.c from Lab 01. Also, there is a module (blockqueue.c) as a starting point to your multithreaded implementation of a queue like the one from Lab 05. This queue is based on a circular buffer of 1KBytes data blocks.

- Implement the concurrency control using in-memory unnamed semaphores or named semaphores (note that in MacOS there is only named semaphores).
- Implement a new version of blockqueue.c using conditional variables and mutexs as in Lab 05.

Bibliography

[1] Sections about Concurrency of the recommended book, “Operating Systems: Three Easy Pieces Remzi H. Arpaci-Dusseau and Andrea C. Arpaci-Dusseau”

http://pages.cs.wisc.edu/~remzi/OSTEP/threads-cv.pdf

http://pages.cs.wisc.edu/~remzi/OSTEP/threads-sema.pdf

[2] Slides from theoretical classes (in CLIP)