CS 4348.004 Operating Systems Concepts Max Lichter
Spring 2024
Summary
Due: 4/6/24

Overview:
The purpose of the doctor's office simulation that I made for Project 2 was to emulate how patients, nurses, receptionists, and doctors would interact in a typical doctor's office (using semaphores). Numerous issues were looked at during the development. process, with the main perpetrator being semaphores for synchronization. Thread management and data structures for patient and doctor interactions were also thoroughly looked at. Semaphores were initialized at the beginning of the simulation to control access to critical areas like patient registration with the receptionist and doctor interactions with patients. After the patient entered the waiting area and registered with the receptionist, they were duo'd with a doctor. Nurses assisted with patient transfers to doctors' offices, where doctors learned about the patient's symptoms, gave advice, then finally, released the patients when the visit was over.

Difficulties Encountered:
One challenge that came up from the development. was ensuring that multiple threads were properly synchronized. It was VERY very important to carefully handle the semaphores and thread interactions to make sure that the scheduled patient arrivals, registrations, and doctor visits were all in the correct order. Even though it was hard, debugging concurrency problems like race situations and deadlocks inside my program taught me so many important lessons about concurrent programming. I feel as if I gained a plethora of important information in this area. Aside from the many concurrency and synchronization issues that I encountered, maximizing the consumption of resources and minimizing the amount of bottlenecks inside the simulation was a consistent and considerable problem. Cautiously adjusting thread interactions and resource allocation procedures was extremely crucial to minimize the wait times of patients while also distributing the burden among doctors, nurses, and receptionists. It was hard to figure out the best way to schedule while managing resource allocation tactics to max out throughput without overloading parts of my code. Overcoming these obstacles was difficult, but I found an ideal balance between efficiency and timing inside the emulation through algorithmic improvements, performance deep-diving, and continuous simulation parameter adjusting.

Lessons Learned:
The overall process of making this simulation taught me a ton about concurrent programming and modeling a system. During the project, very foundational abilities such as managing shared resources, organizing thread activities, and knowing the little things about semaphore usage were picked up. Debugging this concurrency project also improved my problem-solving skills and highlighted the importance of thorough testing and validation in concurrent programs. I feel as If I learned SO much about semaphores and am ready to tackle any problem detailing them.

Results:
Patients coming through the doctor's office—everything from the arrival to receptionist consultation to the doctor's office and departure were precisely copied by the simulation. Receptionists, nurses, and doctors "worked together" totally to skillfully register their patients, walk them to their corresponding doctors, hear the symptoms of their patients, and provide advice to their patients.

CS 4348.004 Operating Systems Concepts Max Lichter
Spring 2024
Summary
Due: 4/6/24

To sum it all up, the simulation project was a very nice tool that provided me with a thorough understanding of concurrent programming ideas and program design strategies. The lessons that I learned showcase the importance of having strong synchronization tools while also being able to have testing procedures.