TCSS 422 Final Report

Summary of Performance

Implementation of Producer Consumer Pairs

When implementing producer and consumer pairs, our group decided to treat them as a separate data structure. This structure would hold the two processes that were connected and two mutexs. These mutexs were used to detect whether deadlock occurred between the two processes. We decided to use a structure because there needed to be some way relate the data to each other. This was the easiest way we believed we could achieve this. Since the pairs were written in this manner, all we had to check for deadlock was to iterate through an array of these structures.

Implementation of starvation prevention

Implementing starvation detection was somewhat confusing. The initial problem with detecting starvation was figuring was quantifying it. It was hard to determine when a process should be considered starved. We decided that a process should be considered starved after a certain amount of time has passed since the process has been run. Our starvation boosting boosts processes all the way to position one. We decided to not go to zero because that priority is reserved for OS processes. The decided to calculate a boosted process by subtracting a boosted flag value by the priority of the process. This means that a process’s true priority is actually its priority – boosted.

Implementation of deadlock detection

Implementing deadlock detection was a fairly easy task. In order to detect the deadlock, a producer consumer pair struct was used to store references to the shared mutexs. After creating these, a thread was made to run in the background and check once and a while if one mutex had each reached one mutex. This will create deadlock because each process is trying to lock the mutex that is shared by its partner.

Implementation of mutexs and condition variables

Implementing the mutexs and condition variables was very difficult. However, it was not as difficult as it could have been because our group was now familiar with how pthreads work. Knowing how pthreads work was very helpful because we just used a similar interface to that. Using pthreads interface helped because it was something that we were familiar with. The familiarity allowed the group to have a clearer picture of what each method should do and how they relate to each other.