Jeffrey Cocklin

2-28-17

CSCE311 Project 3 Constant Time Scheduling

I started my solution by replacing the ready queue, with two arrays of GenericLists active array, and expired array. The result of this changed required a cascade of changes to previous methods defined in the class, along with the inclusion of some new ones. The first method do\_create now appends new threads to the expired queue with a default priority of four, to the queue in the fourth position of the expired array. Next the do\_kill method had to be altered to ensure that the thread to be killed was removed from both the active and expired arrays. The do\_Suspend method also had its remove from ready commands, replaced with remove from active and expired arrays.

In addition to adding the commands to remove from both active and expired arrays when a thread has Thread Ready status, the do\_Resume method now includes instructions to raise the priority of a thread if it is not greater than the highest priority zero. This is accomplished by simply decrementing the thread’s priority value by one. The thread is then placed into the expired array, into the queue that matches its new priority.

The do\_dispatch method now removes the preempted thread from both active and expired arrays, and if the preempted thread exceeded its time quantum and its priority value is less than nine the thread’s priority is decreased. This is handled by incrementing the thread’s priority value. Moreover, do\_dispatch now includes branching conditions based on whether the active, and expired arrays are empty. In the case that the active array is not empty, threads with the highest priority will be dispatched from the corresponding priority queue and SUCCESS is return as usual. In the case that the active array is empty but the expired array is full, the active array now swaps values with the expired array; threads are selected and dispatched and success is returned. Now only when both arrays are empty is the PTBR set to null and FAILURE is returned. Furthermore, in both cases in which threads are successfully dispatched, a timer is set to the value of the appropriate time quantum, which is determined by a thread’s priority value before returning SUCCESS. In this program threads with priority values less that the default priority value, are set with a quantum of 40 and threads with priority values greater than the default priority value are set with a quantum of 20.

Finally, my solution features two new methods arrayEmpty, and selectThread. The arrayEmpty method simply checks an Array of Generic Lists, to see if all the lists are empty. In the case that all the lists are empty the Boolean value true is returned, else false is returned. The select Thread method takes no arguments, and returns the next thread with the highest priority from the active array. All the previously stated changes were made to project two, along with the addition of a few Global constants for style, to create the Constant time scheduling algorithm for project three.