Joshua Landron
CSS430
Program 3

Contents

Algorithm descriptions

Each of the 2 parts altered the operation of the Kernal in threadOS. The 2 main changes were implementations of a more effective method of synchronization that the use of spin locks. The first main change was to allow the Kernel to place the children of threads, which are themselves threads, into a queue marked for the parent thread. This queue served to hold executing threads in order such that a parent thread must wait on its child threads to finish.

The second part of this homework was to add this queue functionality to I/O operations. These operations stored waiting processes in queues for processes requesting I/O and finishing I/O operations. The threads awaiting execution are placed into a queue whenever the disk they are trying to execute is not ready for the process.

The two new classes implemented are SyncQueue and QueueNode. These classes are setup as follows.

SyncQueue{

    Array of QueueNodes

    Default PID, default number of conditions

    dequeueAndWakup method

    enqueueAndSleep method

}

QueueNode{

    Vector of Integers (to hold tids)

    Sleep method

    Wakeup method

}

Each of these classes enables the functionality of both parts 1 and 2.

Discussion of results

Part 1 of this assignment was to update the Kernal.java to use queues for process synchronization in the form of assigning parent threads a queue to have its child threads placed into. Each queue used by the kernel uses a QueueNode object that is has a queue internally to hold child thread IDs.

The first test conducted was to ensure that processes do not try to execute in the wrong order. To test this I launched ThreadOS, launched my Shell from P0 and executed 2 rounds of Test2. As expected, each process waited for its child processes to execute fully before proceeding. An example of this Shell output is shown in Figure 2 as run in the Linux lab.

Part 2 of this assignment was to Alter the kernel again so that Rawread, Rawwrite, and Sync all use another queue to store processes that are being slowed by I/O. These processes were initially using a spin lock where they would just wait while doing nothing until they could run. This meant that they were wasting CPU clock cycles and slowing progress of other programs. Order to remedy this, each of these commands checked the virtual Disk to see if the requesting process could run, if it could not the process was added to a queue to wait to be notified that the disk was available. When the disk becomes available a thread is notified so that that thread can execute. Figures 2 and 3 show the output of Test3 running 3 threads each for processes doing computations and I/O. This update to the Kernel improved performance by about 24%. This improvement is across all I/O operations. Figure 4 shows a comparison of ThreadOS running 4 tests with and without this improvement. The tests were read, write, read+write, and read+write+heavy computation. Each of these tests show about a 25% improvement in execution time after the Kernel was updated to perform this way.

Output

Figure 1: Part 1 Test2 shell output

```
[jlandron@hermes01 src]$ java Boot
threadOS ver 1.0:
Type ? for help
threadOS: a new thread (thread=Thread[Thread-3,5,main] tid=0 pid=-1)
-->l Shell
l Shell
Shell Started.
Type 'exit' to close the shell.

threadOS: a new thread (thread=Thread[Thread-5,5,main] tid=1 pid=0)
Shell[1]% Test2 ; Test2
threadOS: a new thread (thread=Thread[Thread-7,5,main] tid=2 pid=1)
threadOS: a new thread (thread=Thread[Thread-9,5,main] tid=3 pid=2)
threadOS: a new thread (thread=Thread[Thread-11,5,main] tid=4 pid=2)
threadOS: a new thread (thread=Thread[Thread-13,5,main] tid=5 pid=2)
threadOS: a new thread (thread=Thread[Thread-15,5,main] tid=6 pid=2)
threadOS: a new thread (thread=Thread[Thread-17,5,main] tid=7 pid=2)
Thread[a]: response time = 498 turnaround time = 4000 execution time = 3502
Thread[b]: response time = 998 turnaround time = 4600 execution time = 3602
Thread[c]: response time = 1498 turnaround time = 5201 execution time = 3703
Thread[d]: response time = 1999 turnaround time = 5801 execution time = 3802
Thread[e]: response time = 2499 turnaround time = 6502 execution time = 4003
Test2 finished
threadOS: a new thread (thread=Thread[Thread-19,5,main] tid=8 pid=1)
threadOS: a new thread (thread=Thread[Thread-21,5,main] tid=9 pid=8)
threadOS: a new thread (thread=Thread[Thread-23,5,main] tid=10 pid=8)
threadOS: a new thread (thread=Thread[Thread-25,5,main] tid=11 pid=8)
threadOS: a new thread (thread=Thread[Thread-27,5,main] tid=12 pid=8)
threadOS: a new thread (thread=Thread[Thread-29,5,main] tid=13 pid=8)
Thread[a]: response time = 500 turnaround time = 4001 execution time = 3501
Thread[b]: response time = 1000 turnaround time = 4602 execution time = 3602
Thread[c]: response time = 1500 turnaround time = 5202 execution time = 3702
Thread[d]: response time = 1999 turnaround time = 5801 execution time = 3802
Thread[e]: response time = 2500 turnaround time = 6502 execution time = 4002
Test2 finished
Shell[2]% exit
 Exiting
-->q
q
[jlandron@hermes01 src]$
```

Figure 2: Part 1 Test3 output

```
[jlandron@hermes01 src]$ javac -deprecation *.java
Note: Some input files use unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
[jlandron@hermes01 src]$ java Boot
threadOS ver 1.0:
Type ? for help
threadOS: a new thread (thread=Thread[Thread-3,2,main] tid=0 pid=-1)
-->l Test3 3
l Test3 3
threadOS: a new thread (thread=Thread[Thread-5,2,main] tid=1 pid=0)
threadOS: a new thread (thread=Thread[Thread-7,2,main] tid=2 pid=1)
threadOS: a new thread (thread=Thread[Thread-9,2,main] tid=3 pid=1)
threadOS: a new thread (thread=Thread[Thread-11,2,main] tid=4 pid=1)
threadOS: a new thread (thread=Thread[Thread-13,2,main] tid=5 pid=1)
threadOS: a new thread (thread=Thread[Thread-15,2,main] tid=6 pid=1)
threadOS: a new thread (thread=Thread[Thread-17,2,main] tid=7 pid=1)
computation finished
computation finished
computation finished
Read Finished
Read Finished
Read Finished
Write Finished
disk finished
Write Finished
disk finished
Write Finished
disk finished
elapsed time = 165795 msec.
-->q
q
[jlandron@hermes01 src]$
```

Figure 3: Part 2 Test3 results

```
[jlandron@hermes01 src]$ javac -deprecation *.java
Note: Some input files use unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
[jlandron@hermes01 src]$ java Boot
threadOS ver 1.0:
Type ? for help
threadOS: a new thread (thread=Thread[Thread-3,2,main] tid=0 pid=-1)
-->l Test3 3
l Test3 3
threadOS: a new thread (thread=Thread[Thread-5,2,main] tid=1 pid=0)
threadOS: a new thread (thread=Thread[Thread-7,2,main] tid=2 pid=1)
threadOS: a new thread (thread=Thread[Thread-9,2,main] tid=3 pid=1)
threadOS: a new thread (thread=Thread[Thread-11,2,main] tid=4 pid=1)
threadOS: a new thread (thread=Thread[Thread-13,2,main] tid=5 pid=1)
threadOS: a new thread (thread=Thread[Thread-15,2,main] tid=6 pid=1)
threadOS: a new thread (thread=Thread[Thread-17,2,main] tid=7 pid=1)
computation finished
computation finished
computation finished
Read Finished
Write Finished
disk finished
Read Finished
Write Finished
disk finished
Read Finished
Write Finished
disk finished
elapsed time = 126796 msec.
-->
```

Figure 4: All test variations compared