Brian Parks
Dr. Jones
CSE 5441 Lab 2
March 11, 2016

Lab 2 Disposable and Persistent Pthreads

This report will cover notable differences in runtimes between the serialized version of the program and the parallelized version. Also, two different pthread approaches were used; disposable threads and persistent threads. These two approaches will be compared as well as the number of threads for each approach and how the number of threads affected the run time. First a listing of the results:

Lab 1 Serial Results for testgrid_400_12206

testgrid_400_12206

****

Dissipation converged in 75269 iterations.
Max DSV : 0.095762
Min DSV : 0.086186
Affect rate : 0.100000
Epsilon : 0.100000
Elapsed time (clock) in seconds : 1089.869995
Elapsed time (time) in seconds : 1090.266467
Elapsed time (chrono) in seconds : 1090.27
real 18m10.328s
user 18m9.926s
sys 0m0.006s

Lab 2 Results: Disposable Threads for testgrid_400_12206

****

Dissipation converged in 75269 iterations.
Max DSV : 0.095762
Min DSV : 0.086186
Affect rate : 0.100000
Epsilon : 0.100000
Thread Count: 2
Elapsed time (clock) in seconds : 3628.270020
Elapsed time (time) in seconds : 2084.917912
Elapsed time (chrono) in seconds : 2084.92

real 34m45.027s
user 60m23.193s
sys 0m5.171s

****

Dissipation converged in 75269 iterations.
Max DSV : 0.095762
Min DSV : 0.086186
Affect rate : 0.100000
Epsilon : 0.100000
Thread Count: 4
Elapsed time (clock) in seconds : 3516.080078
Elapsed time (time) in seconds : 1282.966978
Elapsed time (chrono) in seconds : 1282.97

real 21m23.075s
user 58m27.818s
sys 0m8.379s

Lab 2 Results: Disposable Threads for testgrid_400_12206 (cont)

****

Dissipation converged in 75269 iterations.
Max DSV : 0.095762
Min DSV : 0.086186
Affect rate : 0.100000
Epsilon : 0.100000
Thread Count: 8
Elapsed time (clock) in seconds : 3569.619873
Elapsed time (time) in seconds : 886.176981
Elapsed time (chrono) in seconds : 886.177

real 14m46.285s
user 59m11.140s
sys 0m18.583s

****

Dissipation converged in 75269 iterations.
Max DSV : 0.095762
Min DSV : 0.086186
Affect rate : 0.100000
Epsilon : 0.100000
Thread Count: 16
Elapsed time (clock) in seconds : 3791.800049
Elapsed time (time) in seconds : 862.988468
Elapsed time (chrono) in seconds : 862.988

real 14m23.097s
user 62m40.558s
sys 0m31.349s

Lab 2 Results: Disposable Threads for testgrid_400_12206 (cont)

****

Dissipation converged in 75269 iterations.
Max DSV : 0.095762
Min DSV : 0.086186
Affect rate : 0.100000
Epsilon : 0.100000
Thread Count: 24
Elapsed time (clock) in seconds : 3707.450195
Elapsed time (time) in seconds : 769.405017
Elapsed time (chrono) in seconds : 769.405

real 12m49.513s
user 60m57.001s
sys 0m50.555s

****

Dissipation converged in 75269 iterations.
Max DSV : 0.095762
Min DSV : 0.086186
Affect rate : 0.100000
Epsilon : 0.100000
Thread Count: 36
Elapsed time (clock) in seconds : 3799.739990
Elapsed time (time) in seconds : 815.457354
Elapsed time (chrono) in seconds : 815.457

real 13m35.566s
user 61m55.506s
sys 1m24.342s

Lab 2 Results: Persistent Threads for testgrid_400_12206

****

Dissipation converged in 75269 iterations.

Max DSV : 0.095762

Min DSV : 0.086186

Affect rate : 0.100000

Epsilon : 0.100000

Thread Count: 2

Elapsed time (clock) in seconds : 3894.760010

Elapsed time (time) in seconds : 2386.064105

Elapsed time (chrono) in seconds : 2386.06

real 39m46.176s

user 64m52.715s

sys 0m2.158s

****

Dissipation converged in 75269 iterations.

Max DSV : 0.095762

Min DSV : 0.086186

Affect rate : 0.100000

Epsilon : 0.100000

Thread Count: 4

Elapsed time (clock) in seconds : 3411.120117

Elapsed time (time) in seconds : 1167.571214

Elapsed time (chrono) in seconds : 1167.57

real 19m27.682s

user 56m45.919s

sys 0m5.309s

Lab 2 Results: Persistent Threads for testgrid_400_12206 (cont)

****

Dissipation converged in 75269 iterations.
Max DSV : 0.095762
Min DSV : 0.086186
Affect rate : 0.100000
Epsilon : 0.100000
Thread Count: 8
Elapsed time (clock) in seconds : 4012.120117
Elapsed time (time) in seconds : 939.771667
Elapsed time (chrono) in seconds : 939.772

real 15m39.884s
user 66m38.534s
sys 0m13.693s

****

Dissipation converged in 75269 iterations.
Max DSV : 0.095762
Min DSV : 0.086186
Affect rate : 0.100000
Epsilon : 0.100000
Thread Count: 16
Elapsed time (clock) in seconds : 4149.569824
Elapsed time (time) in seconds : 827.295231
Elapsed time (chrono) in seconds : 827.295

real 13m47.407s
user 68m32.287s
sys 0m37.391s

Lab 2 Results: Persistent Threads for testgrid_400_12206 (cont)

****

Dissipation converged in 75269 iterations.
Max DSV : 0.095762
Min DSV : 0.086186
Affect rate : 0.100000
Epsilon : 0.100000
Thread Count: 24
Elapsed time (clock) in seconds : 3662.010010
Elapsed time (time) in seconds : 684.236610
Elapsed time (chrono) in seconds : 684.237

****

real 11m24.348s
user 60m9.211s
sys 0m52.906s

****

Dissipation converged in 75269 iterations.
Max DSV : 0.095762
Min DSV : 0.086186
Affect rate : 0.100000
Epsilon : 0.100000
Thread Count: 36
Elapsed time (clock) in seconds : 3734.870117
Elapsed time (time) in seconds : 735.415011
Elapsed time (chrono) in seconds : 735.415

real 12m15.525s
user 60m58.145s
sys 1m16.833s

Analysis

Timing Results: Serial vs Parallel

First I will compare the results of the serial version vs the best runtime out of the parallel versions. The best timing results from the parallel versions was 684.237 seconds and the time from the serial version was 1090.27 seconds. Comparing the two times, the parallel version had about a 37% run time performance increase over the serial version or about 6 minutes. A performance increase was expected since allowing multiple boxes to be calculated at once should have improved run time. The worst run time actually came for the paralyzed version which was significantly slower than the serial (at 2 threads). I believes this is because of cpu scheduling overhead, overhead from creating the threads, and the structural changes that were made to program to parallelize it.

Timing Results: Most Effective Number of Threads

For both persistent and disposable threads, the number of threads that yielded the best runtime was 24. When thinking about this question initially, I thought that the number of threads closest the number number of processors on the system would be optimal. Since Nodes on Oakley have 12 processors per node, one would think that the optimal amount of threads would be close to the number of processors but this was not the case. Having 12 threads per Node should reduce the scheduling overhead on the cpu and have a better run time than a large number of threads. I believe that I failed to consider hyper threading per CPU, which means that there could be multiple threads running on one CPU at the same time and thus the number of threads per CPU might be 2 or more which might explain why 24 is the optimal number (2 threads per CPU with 12 CPUs).

Timing Results: Most Effective Parallel Version

Disregarding some discrepancies, overall the persistent thread version of the program out preformed the disposable thread in runtime in almost all cases. I believe this is because the overhead of creating threads every literation in the convergence loop lead to the additional run time that was seen. For example, the 24 thread version of both the persistent and disposable program was the fastest performing version with a runtime of 684.237 seconds for the persistent version and 769.405 seconds for the disposable version. The overhead of recreating the threads each time lead to around an 11% increase in runtime for the disposable version. Overall, I believe the persistent threaded version was more effective.

Timing Results: Expectations

I did expect the the run time of the parallel version of the program to have a better runtime than the serial version however I thought that a 37% increase in runtime was not that great considering the work that

went into restructuring the code. I expected the runtime of the persistent version to be better that the disposable version, which was the case, because of the additional overhead of creating the threads every iteration in the convergence loop. The optimal number of threads turned out to be 24 which was not what I was expecting. I believe that I did not take into account hyper threading on the CPUs which might have lead to a better run time than 8 or 16 threads.

Time Results: Unexpected Anomalies

There were two anomalies in my timing results. It was expected that the persistent version of the program would have a better run time than that of the disposable version however there were two cases in which it was worse; 2 threads and 8 threads. In all other cases, the persistent version preformed significantly better (around a 10% increase in runtime).

TimeResults: Best timing method

Going into lab 2, I expected that the best timing method would be something other than time(clock) which was correct. Since time clock deals with CPU clock cycles this lead to anomalous results in the paralyzed versions. The best timing method which I used to compare each result was chrono. I think that a three digit precision time is best for readability and making quick comparisons. Also, since chrono uses points in time as a time as a reference, it would provide the most understandable results comparing to the other time functions used.

Additional notes:

When creating the persistent version of the program, I was having an issue in where the persistent version was vastly slower than that of the serial and disposable version. Since my initial expectation was that the persistent version should have been faster, I went to Dr. Jones' office hours to resolve the issue. The issue was that, in each thread, I was checking the convergence condition which was calling a function called isConverged(); Inside isConverged(), every thread would look through the entire set of boxes and calculate (sometimes suboptimal) min and max DSVs. This was corrected by moving the logic of checking the convergence condition partially into the loop and only updating the min and max DSVs in the master thread (which was thread id == 0). After completing this fix, my persistent version had results that were more consistent with what was initially assumed.