Shiny Patel CSE 5441 Lab 2 Report

## Timing Results

Serial version of program on testgrid_400_12206 with epsilon and affect rate = .1

| posix clock | clock ticks | time | real | user | sys |
|---|---|---|---|---|---|
| 63954065.00 | 63850000 | 64 | 63.97 | 63.83 | 0.03 |

#iterations: 75197, max DSV: 0.086671 and min DSV: 0.078004

Parallel versions:

*Disposable* threads on testgrid_400_12206 with epsilon and affect rate = .1

| #threads | posix clock | clock ticks | time | real | user | sys |
|---|---|---|---|---|---|---|
| 2 | 47521028.00 | 76260000 | 47 | 47.63 | 72.56 | 3.71 |
| 8 | 46694213.00 | 94490000 | 47 | 46.71 | 78.69 | 15.81 |
| 16 | 73580036.00 | 145090000 | 74 | 73.59 | 88.19 | 56.91 |
| 32 | 305715125.00 | 389580000 | 305 | 305.73 | 102.30 | 287.29 |

#iterations: 75197, max DSV: 0.086671 and min DSV: 0.078004 for every run

*Persistent* threads on testgrid_400_12206 with epsilon and affect rate = .1

| #threads | posix clock | clock ticks | time | real | user | sys |
|---|---|---|---|---|---|---|
| 2 | 40623443.00 | 67820000 | 41 | 40.64 | 66.07 | 1.76 |
| 8 | 41373636.00 | 85280000 | 41 | 41.39 | 75.38 | 9.91 |
| 16 | 61756259.00 | 99110000 | 62 | 61.77 | 80.01 | 19.11 |
| 32 | 82449914.00 | 107380000 | 82 | 82.46 | 81.34 | 26.05 |

#iterations: 75197, max DSV: 0.086671 and min DSV: 0.078004 for every run

## Q&A

- **Did this program perform better sequentially or in parallel?**
  On comparing the user and sys times of the serial vs the parallel versions (both disposable and persistent) it is clear that the program performs better sequentially. This is due to the lack of overhead from threads in the serial version.

| Version | user | sys |
|---|---|---|
| Serial | 63.83 | 0.03 |
| Persistent (with 2 threads) | 66.07 | 1.76 |
| Disposable (with 2 threads) | 72.56 | 3.71 |

  On comparing the real time of the serial vs the parallel versions (both disposable and persistent) it is clear that the program performs better in parallel version (and persistent version amongst the parallel versions).

| Version | real |
|---|---|
| Serial | 63.97 |
| Persistent (with 2 threads) | 40.64 |
| Disposable (with 8 threads) | 72.56 |

The real, user and sys times for disposable and persistent versions are provided for runs which provide the best timing results.

- **Which number of threads was most effective?**
  2 threads were the most effective in both disposable and persistent versions. As seen from the timing results, all the 3 times (real, user and sys) increase with the increase in the number of threads (except for the real time in disposable version run with 8 threads (47.63 s), which is only slightly better than the run with 2 threads (46.71 s))

- **Which parallel version (disposable or persistent) was most effective?**
  Persistent version was more effective and consumes less time (real, user and sys) for all runs (i.e. with 2, 8, 16 and 32 threads) than its equivalent disposable version. This is because compared to the disposable version, persistent version does not have the overhead of creating new threads (and thus does not consume resources) during every iteration of the convergence loop.

- **How did your results match or conflict with your expectations?**
  The results match with my expectations. I was expecting the persistent version to perform better than the serial version in terms of real time due to parallelization and the serial version to perform better than the parallel version in terms of user and sys times due to its lack thread overhead.

- **Were there any unexpected anomalies in the timing information collected?**
  The real time of the programs may differ significantly from those provided in the timing results as it depends on the workload of the CPUs as well as which node the program is assigned to on logging into stdlinux.

- **Which timing methods seem best for parallel programs? How does this compare with your expectations?**
  The real time may fluctuate significantly on stdlinux but the user and sys times more or less remain the same. So, they are the best timing methods for parallel programs and the timing results support my expectations.

NOTE: All time, real, user and sys times are in seconds.