Program 1 Report

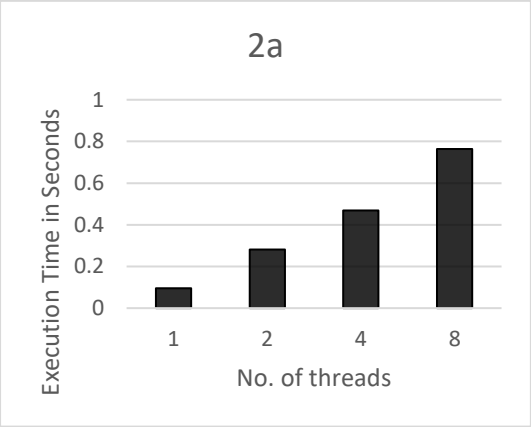
Karthick Narayanan Srinivasa Raghavan ksriniv@ncsu.edu
Atit Shetty akshetty@ncsu.edu

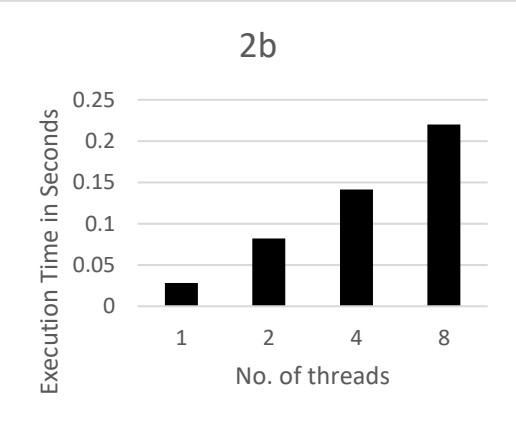
2. d.

Execution times:

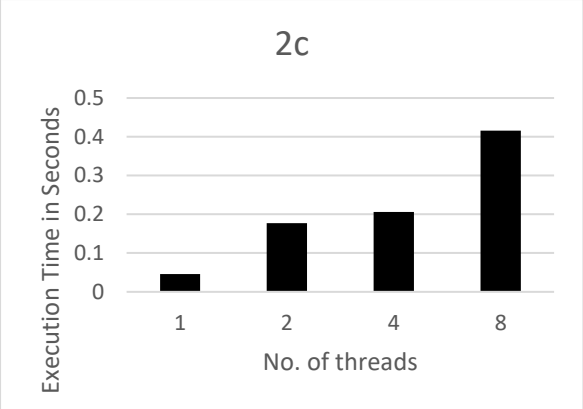
Question	No of threads	Loop Exec Time (Seconds)
2a	1	0.09464
	2	0.281893
	4	0.469183
	8	0.763086
2b	1	0.028244
	2	0.082167
	4	0.141278
	8	0.219967
2c	1	0.045587
	2	0.177226
	4	0.206077
	8	0.415643

* The timings were generated from cluster.





Program 1



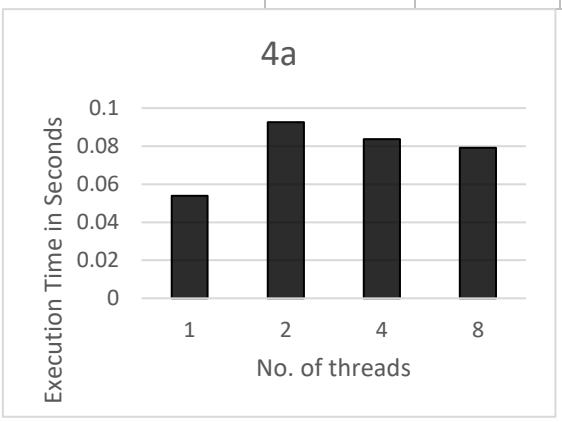
We noticed that **locks** had the highest overhead. The execution time increased with number of threads.
Critical was an improvement over locks as the execution times were much better.
However, **atomic** achieved the best results with least execution time.

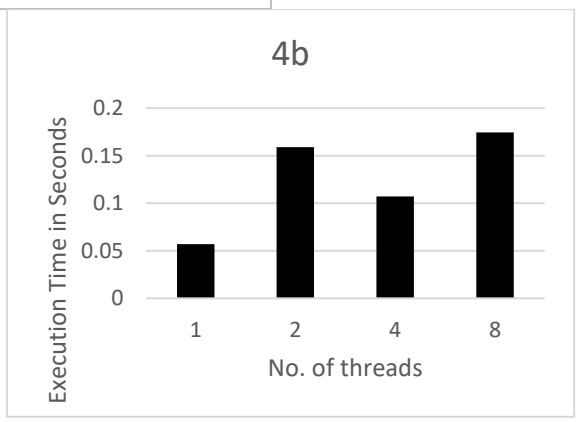
Program 1

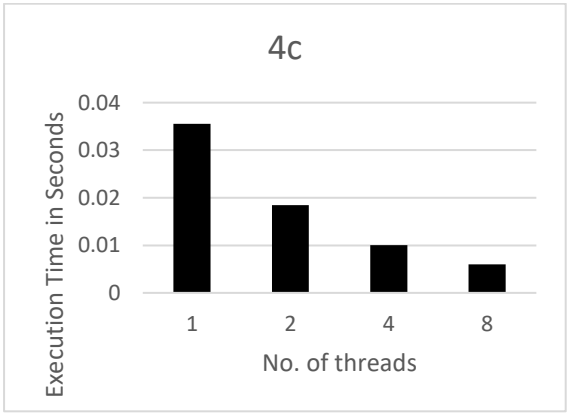
4. 1.

Execution times:

Question	No of threads	Loop Exec Time (Seconds)
4a	1	0.053865
	2	0.092681
	4	0.083709
	8	0.079216
4b	1	0.056868
	2	0.159103
	4	0.107
	8	0.17436
4c	1	0.035519
	2	0.018444
	4	0.010033
	8	0.006011







4.2.

Changes in value of PI corresponding to changes in values of numPartitions.

Value of numPartitions	Value of PI
30000	
10000	3.1415
2400	3.141605
1200	3.141603
600	3.141711
300	3.142133
150	3.141689
100	3.1428
50	3.144
25	3.1616
15	3.18222
10	3.16000

- We noticed that as numPartitions were decreased from 1200 to 15, the value of PI was diverging from expected value.
- It then converged slightly at 10.
- As the value of numPartitions was increased from 1200 to 10000, there was more convergence in the value of PI.
- At 3000, the cluster didn't execute the program.

4.3.

- Static is the default scheduling used for iterations in OpenMP.
  In this method of scheduling each thread is statically assigned number of iterations that it should execute
- Another method of scheduling is dynamic scheduling.
  In this method of scheduling, each thread is assigned an iteration at run time. When a thread has finished an execution and is idle, it can be assigned new thread.
- Dynamic scheduling can help balance uneven workload, by splitting work between threads at runtime.
- However, due to this the overhead associated with dynamic scheduling is more than static scheduling.
- Chunks can be used to mitigate this issue. A thread is dynamically assigned chunks of iterations to perform. When it completes the operation, it is again assigned a new chunk dynamically, thus improving the performance and not increasing the overheads.

Program 1

In this experiment, we found ways to parallelize loops while calculating value of PI.

In 4.b, we parallelized only j loop. To do this we had to include an atomic section. This helped to decrease the time for execution, but as threads increased, so did the bottle neck due to atomic operations.

In 4.a, we parallelized only i loop. The performance was better than the above case, considering we had to also include an atomic section in the loop.

In 4.c, we parallelized both the i and j loops. This greatly increased the performance than the above two cases.

We realized that critical sections, will indeed cause execution time to increase in a parallel process.