Cpt S 411 Assignment Cover Sheet

(To be turned in along with each homework and program project submission)

Assignment # 4

For individual assignments:

Student name (Last, First): Cuevas, Jessica

For team projects:

List of all students (Last, First): Cuevas, Jessica

List of collaborative personnel (excluding team participants):
N/A

I[1] certify that I have listed above all the sources that I consulted regarding this assignment, and that I have not received or given any assistance that is contrary to the letter or the spirit of the collaboration guidelines for this assignment. I also certify that I have not referred to online solutions that may be available on the web or sought the help of other students outside the class, in preparing my solution. I attest that the solution is my own and if evidence is found to the contrary, I understand that I will be subject to the academic dishonesty policy as outlined in the course syllabus.

Please print your names.
Jessica Cuevas

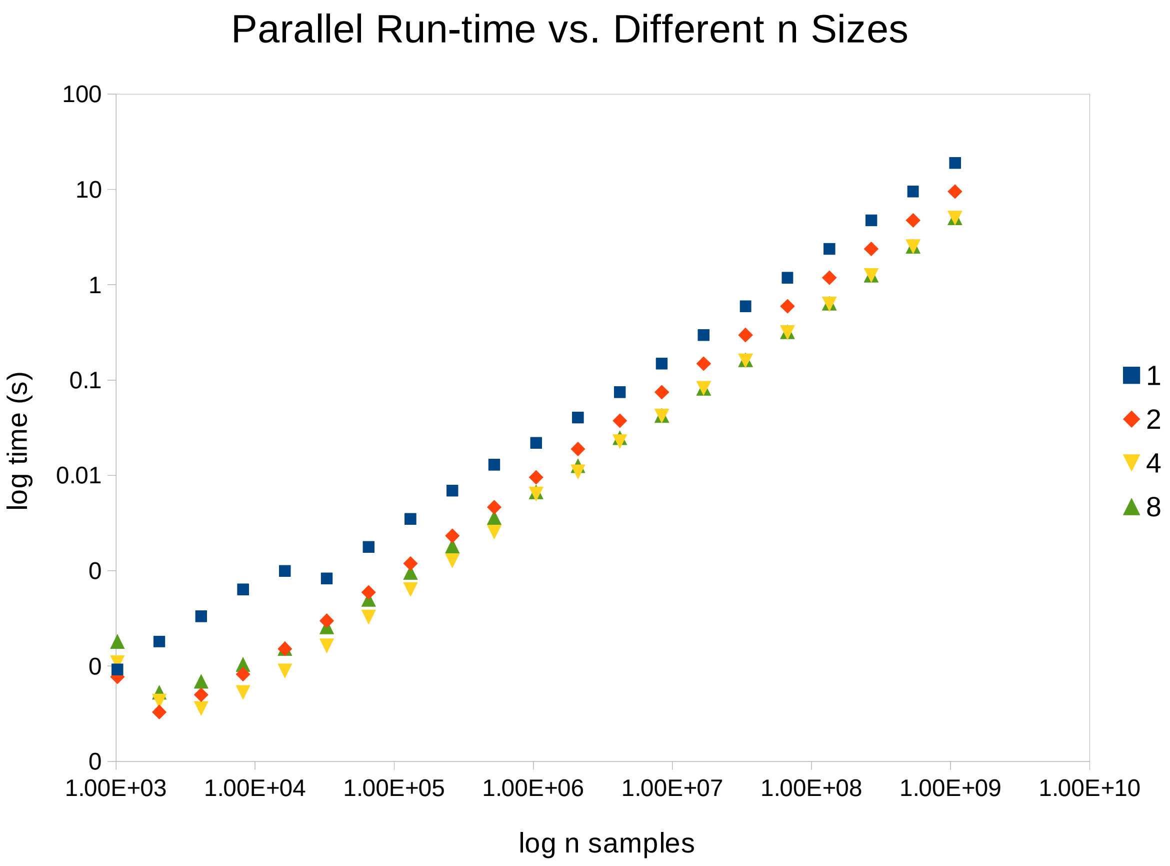
Assignment Project Participant(s):
Jessica Cuevas

Today's Date:
11/17/20

[1] If you worked as a team, then the word "I" includes yourself and your team members.

Programming Project 4 Report

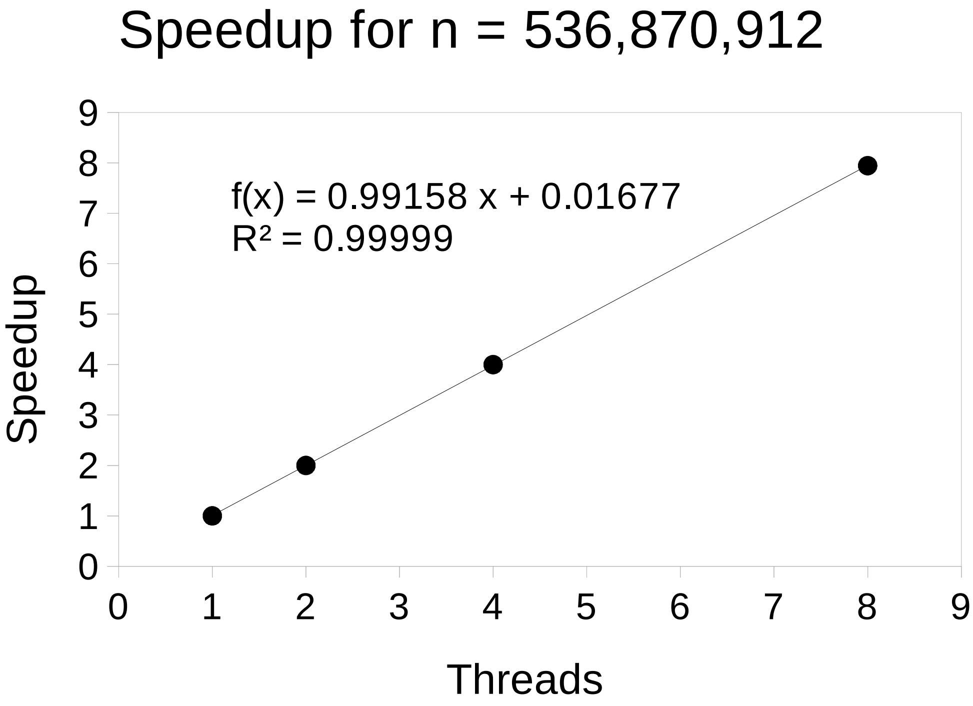
Table and Chart Results:



In this graph I had tested the run-time of various input sizes of n (ranging from 1024 - 1073741824) depending on the number of threads used. (p=1,2,4,8) . As observed above, as the number of threads increased the less time it took for the logarithm to estimate a value of PI.

Table 1. Parallel Run-time and Speedup (T(n,1) / T(n,p)))
(for a large input size of n)

Experiment 1			
threads	run-time	speedup	n
1	15.137943	1	536870912
2	7.570983	1.9994686291	536870912
4	3.78601	3.9983895975	536870912
8	1.905844	7.9429077091	536870912



The speedup of this large and ideal example of n, showed a ideal linear speedup. As threads increased the more efficient the algorithm.

Table 5. Precision Testing
(keep n/threads fixed, and increase threads)

Experiment 2				
threads	pi estimate	% error	n	pi
1	3.1415880919	0.0001452047	67108864	3.1415926536
2	3.1415880919	1.452047E-06	134217728	3.1415926536
4	3.1415872574	1.717665E-06	268435456	3.1415926536
8	3.1415872574	1.717665E-06	536870912	3.1415926536

I tested precision based on the percent error of the Pi estimation from the actual Pi value. As observed above the larger the size of n, the smaller the percent error. Though the algorithm is not ideal with relatively small n size (i.e. less than a million), it still showed an accurate estimate of Pi as n increased.