Assignment 3 – Static loop scheduler

Name: Rajdeep S. Manjre ID: 801020258 Email:rmanjre@uncc.edu

Question: Report time and speedup across a range of precision, intensity, and synchronization mode. Use make test to test your code. Your code MUST pass the test before you can use make bench to start the PBS jobs. Once they are complete, plot the charts with make plot. Note: you must run these commands from within the static/ directory.
Answer: check in folder "plot" .

Question: Why do you think some measurements are so erratic?
Answer: In parallel computing parallelism depends on the operating system and also the node on which it is running. Thus, the threads are totally OS dependent and hence erratic. Also, the graph is taken for very less values which are far apart thus it cannot show where the glitch has occurred.

Question: Why is the speedup of low intensity runs with iteration-level synchronization the way it is?
Answer: For low intensities the cpu-cycles for computation are less thus the speedup when compared to sequential is not more. For lower intensities it works almost same as sequential with a constant speedup for all threads. Even if we add more threads it does not make much difference rather it may add overhead due to thread creations and thread joining where it waits for all the threads to sync.

Question: Compare the speedup of iteration-level synchronization to thread-level synchronization. Why is it that way?
Answer: In general the speedup in thread-level synchronization is much higher than the iteration-level synchronization. This is because in thread-level synchronization, the local variable gets updated for each thread and then the value is stored in global variable which is the total sum for value of 'n', while in iteration-level synchronization, each time the loop runs, global variable gets updated which increases the overhead and eventually increases the computation time which is more or less equal to sequential running code. Also, in iteration-level the lock is acquired to update the global variable which is a shared variable for all threads while in threads it is acquired only once per thread.