DREXEL UNIVERSITY
Electrical and
Computer Engineering
College of Engineering

Drexel University

Electrical and Computer Engineering Dept.

ECEC-413

Assignment 5:
Jacobi Solver OpenMP

Minjae Park

John Truong

Professor Naga Kandasamy

Date: 05/15/2020

Jacobi Solver with OpenMP:

OpenMP implementation of Jacobi Solver utilized pragma for synchronization between iterations and for loop parallelism. The pragma was initialized before each main loop to handle openMP. The code was parallized by splitting the operations between rows and columns between the threads. Since the convergence factor needed to be calculated for the entire matrix, a mutex lock was declared for protected writing between threads. There are 2 fork points in this program: Initialization of the X matrix and the Jacobi Iteration for updating the X matrix. The initialization step, which copies matrix B to X, was parallelized by creating a parallel OPM to the function jacobi_setup. After all the threads join the main thread, thread arguments are updated with the global ssd variable, mutex addresses and threads are created for Jacobi iteration. The created threads run through the iteration of calculating portions of the X matrix. Once the threads exit, the main thread calculates the convergence factor mse and checks whether the program should exit the while loop. At the end of the main function, data structures are freed before exiting.

Results:Speed Up:

Matrix Size	Single Thread	4 Threads	8 Threads	16 Threads	32 Threads
512x512	3.793	1.52	1.288	1.182	1.716
1024x1024	28.918	8.635	6.25	5.966	5.284
2048x2048	247.909	62.548	32.218	31.742	30.488
4096x4097	1957.529	504.908	263.669	246.632	232.783

Table 1: Jacobi Solver Data on Xunil-05

Matrix Size	Single Thread	4 Threads	8 Threads	16 Threads	32 Threads
512x512		249.54%	294.49%	320.90%	221.04%
1024x1024		334.89%	462.69%	484.71%	547.27%
2048x2048		396.35%	769.47%	781.01%	813.14%
4096x4096		387.70%	742.42%	793.70%	840.92%

Table 1: Speedup of Jacobi Solver