CS301-HPC Lab 3

Assignment 3

Deadline: April 5, 2022

Part 1 -

Learn to use the following script for data collection and analysis, and perform the Matrix multiplication assignment given below:

https://github.com/letshpcorg/letshpcsample/tree/master/collect_data_without_perf

Only Part-2 has to be reported in the lab-assignment-report

Part2 -

Matrix multiplication (MM) using three implementations (code provided for 1st implementation, implement the other two similarly):

A. Matrix multiplication outermost loop parallelization (https://github.com/letshpcorg/letshpcsample/tree/master/collect_data_without_perf/all_codes/201401114-201401114-matrix_multiplication-outermost)

B. Matrix multiplication middle loop parallelization

C. Matrix multiplication innermost loop parallelization

Collect the data for the following experiments, plot it and perform the necessary analysis/interpretation (do it on the local LAB 207 machine)

First note down the serial time for matrix multiplication (for problem sizes 64, 128, 256, 512 and 1024). Prepare a table showing Problem size/ serial run-time/ memory footprint (total array size in MB)

Two times can be collected using the scripts provided to you. Do the following analysis using Algorithmic time. You may also collect the End-to-end time and report about it.

• Algorithmic (ALG) time: This includes only the time for the core algorithm (memory access and operations), and does not include any time for the I/O part or other pre-processing.

• End-to-end (E2E) time: This counts the entire run-time of the program. In addition to ALG time,

this includes the time taken for I/O (reading the input test-case files and writing the output data files as well) as well as any other pre/post-processing that is not a part of the core algorithm.

1. Compare the execution time (ALG) of three parallel MM implementations (default scheduling using 4 threads) for problem sizes 64, 128, 256, 512 and 1024. Are the run-times same for three different implementations, if not why the run-times are different for different implementations - explain clearly and quantitatively.

2. Analyze the execution time on different threads using the Vtune amplifier and report about the load balance aspect - for all the three implementations.

3. Explore whether load imbalance can be improved by choosing a better work size (chunk size, scheduling etc. and Report about it.

4.Perform and report about memory access analysis using VTune amplifier for the three different implementations for the experiment performed above (serial number 1). https://www.intel.com/content/www/us/en/develop/documentation/vtune-help/top/analyze-performance/microarchitecture-analysis-group/memory-access-analysis.html