2/5/2023

PDC Assignment 1

Result Report

Fasih Abdullah i200432 CS-F

Table of Contents

[Task 1: 2](#_Toc126508846)

[Graphs of Results: 2](#_Toc126508847)

[Analysis: 2](#_Toc126508848)

[Task 2: 3](#_Toc126508849)

[Graph of results: 3](#_Toc126508850)

[Analysis: 4](#_Toc126508851)

# Task 1:

## Graphs of Results:

### Analysis:

As we can see from the graph above performance with use of Affinity is greater than the performance obtained without Affinity. The variation in the results per iteration is due to the array being populated by random values which affect the time for computation per iteration. The higher the Gflops the better the performance. Affinity allows a more parallel computation and hence increases overall performance. As we can see the best performance with affinity resulted in a total of 0.466 Gflops whereas without Affinity the maximum was 0.283 Gflops. Furthermore, the average Gflops with Affinity is 0.335 with variance 0.007256 as compared to 0.262 Gflops with 0.00108 variance. This show that in this case the performance advantage of Affinity is more than the overhead and with Affinity results seem to stay more consistent.

# Task 2:

## Graph of results:

### Analysis:

As we can the performance of block wise division of 2^2\*2^2 and 2^4\*2^4 show similar performance. However for this data size the best performance is by block division of 2^8\*2^8. The cyclic version also performs better than 2^2\*2^2 and 2^4\*2^4 but lags behind 2^8\*2^8 due to the large amount of thread switching overhead. The maximum Gflops attained in this scenario were 0.1822976. The mead for block wise division was 0.0325 with variance 0.000256 whereas for cylic these were 0.0479 with variance 0.004613. In conclusion the fastest method is block wise division into 2^8\*2^8 blocks in this scenario.