|epcc|

Message Passing Programming Coursework Assignment

Exam number B136013

November 14, 2018

Contents

1 Introduction **3**

2 Project Specification **3**
2.1 Description 3

3 Analysis **3**
3.1 Tools 3
3.2 MPP API 3
3.3 Design 3
3.4 MPI Scatter 4
3.5 MPI Gather 4

4 Evaluation **4**
4.1 Correctness 4
4.2 Performance 4

5 Conclusion **4**

1 Introduction

The project solves an image processing problem. It uses a two-dimensional domain decomposition in order to split the workload to the active processes. To achieve this we use MPI API for process communication. This approach arise a variety of challenges that need to be addressed, such as the communication, decomposition, boundary conditions and halo swaps.

2 Project Specification

2.1 Description

Boundary conditions: horizontal->sawtooth vertical->periodic Terminate condition: D parameter

3 Analysis

3.1 Tools

For the development of this project the used programming language is C due to its performance and for low level calculations. In addition, gnu make was used for the build phase and python to compare the output for testing reasons.

3.2 MPP API

Tools from MPP (derived types, non blocking communication, virtual topologies)

3.3 Design

brief description of the design and implementation of your MPI program bypassed buff to edge

3.4 MPI Scatter

3.5 MPI Gather

4 Evaluation

Platform: backend of cirrus (.pbs), compiled using -O3 Build, Run and Submit job: in Readme.md

4.1 Correctness

Test: parallel code produces the same output as the serial

how long the code needs to run in order to give a reasonable assessment of its performance and/or correctness It is not necessary to run all and be correct

4.2 Performance

Timing: exclude I/O from timing The timing starts before the scatter and ends after the gather Graphs: demonstrate that the performance of the code improves as you increase the number of processes. processes Input variety: performance change input(problem) size and see how it scales Speedup of strong scaling: performance metrics speed up vs ideal (linear) speedup average time per iteration

5 Conclusion

In conclusion the decision about scheduling selection and number of threads that need to be deployed is not an obvious choice. It always depends on the problem and how the work load is distributed between iterations. Analysis and measurements using different configuration options will guide the developer on what is the best approach of solving a problem effiently.