

Homework #4

cpe 512

Kyle Ray

November 23, 2017

Contents

[Part 1 2](#_Toc499071840)

[Action 1.1 2](#_Toc499071841)

[Action 1.2 2](#_Toc499071842)

[Action 1.3 3](#_Toc499071843)

[Action 1.4 4](#_Toc499071844)

[Action 1.5 5](#_Toc499071845)

[Action 1.6 6](#_Toc499071846)

[Action 1.7 7](#_Toc499071847)

[Part 2 7](#_Toc499071848)

[Action 2.1 7](#_Toc499071849)

[Action 2.2 7](#_Toc499071850)

[Appendix 7](#_Toc499071851)

# Part 1

## Action 1.1

**Table 1: Laplace 2D OpenMP Run Times with NP = 1,2,4,8**



## Action 1.2

**Table 2: Laplace 2D OpenAcc Run Times Using Kernels**



The average run time has increased dramatically using only the “#pragma acc kernels” on the main loops.

****

The commented lines “Action 1.2 Change” are the portions of code that were changed for this action item.

## Action 1.3

**Table 3: Laplace 2D OpenAcc Run Times Using Loop Improvement**



This is a significant improvement in execution time compared to the OpenMP version even when using 8 CPU cores.

****

**The commented line “Action 1.3 Change” are the portions of code that were changed for this action item.**

## Action 1.4

**Table 4: Laplace 2D OpenAcc Run Times Loop Tuning**



**This is a very significant improvement over the OpenMP implementation even when using 8 CPU cores.**

****

## Action 1.5

**Table 5: Laplace 2D OpenAcc Run Times Parallel**



It seems that this is the slowest version yet, even slower than the OpenAcc kernels implementation.



TODO: Check and make sure that the “loop” is part of this problem. The action item in the homework assignment only states to use the acc parallel pragma but also to use the reduction. I’m not sure that you can have the reduction without the loop?

## Action 1.6

**Table 6: Laplace 2D OpenAcc Run Times Parallel Transfer Reduced**





## Action 1.7

# Part 2

## Action 2.1

## Action 2.2

# Appendix

Add anything else that might be pertinent to the assignment.