***HW7 - Particle Interaction Report***

***parallel algorithm implementation***

Marcelo Andrade Rodrigues d’Almeida

Computer Science Department

University of Pittsburgh

Pittsburgh, USA

maa261@pitt.edu

*Abstract* — This report has the purpose of describe the parallel implementation of n-body problem (particle iteration) in C programing language with MPI library, as well, analyze speedup and efficiency between parallel and serial implementations

Keywords—particle iteration; n-body problem; parallel;

# Introduction

Part II describes the parallelization strategy. Part III and IV, speedup and efficiency analysis, respectively. Part V will discuss possible performance bottlenecks

# Paralellization Strategy

## The strategy is to first use Scatter to divide the work among the Threads. Each of them can pass/receive the particle information to another Thread. Then all of them can compute the collision between pair of particles. Repeat this passing/receiving - computation process until all pair computations are processed (a particle is collided with each any other of particle set). Then one can send the data to original owner to merge all the processed colisions and compute the actual value. At the end, all information is returned to the Thread that divided the work, using Gather function. Isend and Irecv are used to avoid deadlocks.

# Speedup Analysis

Speedup: S = TS(n)/TP(n), n = 1,000,000

All execution information in Seconds (TS(n), TP(n))

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TS(n)\TP(n) | 326.44 | 157.44 | 77.34 | 38.07 | 18.52 |
| 14,396 | 44.1 | 91.44 | 186.15 | 378.14 | 777.48 |

Speedup (n = 1,000,000)

Speedup: S = TS(n)/TP(n), n = 100,000

All execution information in Seconds (TS(n), TP(n))

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TS(n)\TP(n) | 380.85 | 185.83 | 91.06 | 43.34 | 21.00 |
| 14,396 | 37.80 | 77.47 | 158.10 | 332.15 | 685.41 |

Speedup (n = 100,000)

Speedup: S = TS(n)/TP(n), n = 10,000

All execution information in Seconds (TS(n), TP(n))

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TS(n)\TP(n) | 439.43 | 213.88 | 104.55 | 49.43 | 24.26 |
| 14,396 | 32.76 | 67.31 | 137.7 | 291.27 | 593.34 |

Speedup (n = 10,000)

# Efficiency Analysis

Efficiency: E = S / P, P = {63,127,255,511,1023}

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| S | 44.1 | 91.44 | 186.15 | 378.14 | 777.48 |
| P | 63 | 127 | 255 | 511 | 1023 |
| E | 0.70 | 0.72 | 0.73 | 0.74 | 0.76 |

Efficiency (n = 1,000,000)

Efficiency: E = S / P, P = {63,127,255,511,1023}

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| S | 37.80 | 77.47 | 158.10 | 332.15 | 685.41 |
| P | 63 | 127 | 255 | 511 | 1023 |
| E | 0.60 | 0.61 | 0.62 | 0.65 | 0.67 |

Efficiency (n = 100,000)

Efficiency: E = S / P, P = {63,127,255,511,1023}

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| S | 32.76 | 67.31 | 137.7 | 291.27 | 593.34 |
| P | 63 | 127 | 255 | 511 | 1023 |
| E | 0.52 | 0.53 | 0.54 | 0.57 | 0.58 |

Efficiency (n = 10,000)

# Performance Bottlenecks

The Performance Bottlenecks can be found on parts of code that cannot be parallelized (strictly serial program), like per example, read input and write output from/to a file.

# Code Snippet

Below, there is the part of code that describes the strategy utilized:





