Proyecto de Grado

Instalación, Implementación y uso de las librerías boost y su aplicación en recorrido de grafos para la supercomputadora Apolo

Autor:

Sergio Andrés Monsalve Castañeda
Código: 200410061010
smonsal3@eafit.edu.co
semonsalve@gmail.com

Asesor:

Juan Guillermo Lalinde Pulido
jlalinde@eafit.edu.co
2619500 ext 9588

25 de septiembre de 2013

Índice general

Resumen

aca iria el abstract

Agradecimientos

Pineda, Silva, Mateo, Alejandro, Jaime, Juan Guillermo

Capítulo 1

Introduccion

This document seeks to explain the activities undertaken as part of the research group of Professor Marisol Koslowski during the Summer Undergraduate Research Fellowship, at the same time, serve as a background document for those who continue working on related projects pf the research group.

1.1. Project Goals

In this project Jaime Perez and Sergio Monsalve were assined to work with professor Marisol Koslowski research group. In the begining of SURF we had a series of nanohub trainings, and later we were introduced to the rest of the team, as the meetigns went by we got some Activities Asigned to us. (Jaime Perez and Me) in the begging we work on the same activities, and later on the summer we distribute the task between us.

1.1.1. Project Context and Background

In the first days a contextualizations were necessary, research topic related papers were given to us.

1.1.2. Voronoi structure generator

This was one of the first task that we worked on, we stopped in the 4th week after realized that this was no longer necessary.

1.1.3. Input Code Improvement

This improvement had to standardize the way the variables were entered to code without the need to recompile. We seemed unnecessary for two people work on this task so that only Jaime followed working on it.

1.1.4. Verification Cases

Due to the number of people who manipulate and constantly change the code and the number of outputs that the code provides, is necessary to have verification system that allows ensure that the code still works after being modified.

1.1.5. Code Optimizations

Different approaches were taken into account, since the modification of the numerical methods to rewrite expressions that allowed optimize operations, thereby obtaining better performance.

- Algorithmic Optimizations
- Numerical Methods Improvement
- Expression Rewriting

1.1.6. Diferent Numerical Methods implementations

Of all the possible optimizations, the largest effect is usually the modification of the algorithm, in this case the use of a numerical method with less number of iterations and significantly greater accuracy increases quality and

reduces code complexity, leading to an model closer to the analytical solution with less processing resource consumption.

This task was not completed, i had unexpected behavior on the code and could not make the code to converge.

1.1.7. Plot Convergency and Results from virtual evolv function

This depended on the previous task therefore was not completed.

1.1.8. Matrix Operations Optimizations

Left for future work for time reasons.

Capítulo 2

Marco de Referencia

- Grafos
- Super Computadores
- tipos de Problemas

2.1. Results

Although computing capacity continues to grow, every year there are bigger and faster machines, the execution speed is increasing but the effect of writing efficient code will always be significant.

After analyzing the code implemented we can conclude that with good programming practices, and small modifications in details in how the code is writen, a great difference at the time of execution may be achievable, most when such changes are added together.

The clarity and simplicity in the code when translating the equivalent mathematical model can make the code be inefficient, but excessive expression level optimization would make the code difficult understanding and modification. Therefore, a balance between the two is necessary for a higher quality

code.

Use the syntactic sugar that the programming languages offer to us can help to reduce the loss of clarity while maintaining a balance between readability and efficiency. [2] [3] [1]

Introducción

metaprogramación metaprogramación en C++

2.2. MPI

2.3. Boost Graph Library

(BGL) - http://www.boost.org/ build mpi build GraphParallel build serilization

Boost and Eclipse configuración

2.3.1. Parallel Boost Graph Library (Parallel BGL)

2.4. Virtualización de un cluster

[imagen de apolo]

2.4.1. Instalacion de Boost en Apolo

2.4.2. makefile

Cómo ejecutar el script de installación

que se debe modificar en el makefile[4]

instalar rpm de boost: Cuando se formatea o reinicia máquina Cuando se instala normalmente.

2.5. ejemplo

ejemplo Programa Corriendo en Apolo

Capítulo 3

Glosario

HPC: (High Performance Computing) Computación de alto desempeño.

Bibliografía

[1] Boost C++ Libraries.

[2] The Boost and Graph Library. *The Boost Graph Library.*

[3] George Em Karniadakis and Robert M Kirby Ii. in C ++ and MPI A seamless approach to parallel algorithms and their implementation.

[4] Kurt Wall. *Programación en Linux con ejemplos.* Prentice-Hall, 2000.

.1. Appendix 1: Code Repository

Para una copia del codigo utilizado dirigirse a: `https://github.com/smonsalve/tesis.git`

.1. Appendix 2: Future Work

Recomendations for future work.e points given in the last weeks of work)