Distributed Mandelbrot set & DES Brute-force Algorithm on the "*Cerberus"* Beowulf Cluster (Case Study)

Jeff Becker, Richard Hayes, Charles Singleton
{j_becker4, r_hayes, c_singleton2}@post.massbay.edu
Department of Computer Science
Massachusetts Bay Community College, Wellesley Hills, MA

Faculty Advisor
Giuseppe Sena
gsena@massbay.edu
Department of Computer Science
Massachusetts Bay Community College, Wellesley Hills, MA

We used the "***Cerberus***" Beowulf Cluster to develop two applications as a case study:

- Distributed Mandelbrot set [1].
- Distributed DES [2] Brute-Force Algorithm.

Applications follow a Master-Slave paradigm, and are written in **C** and **Open MPI** [3]. We analyzed the performance and visualized the output using GUIs developed in **Python**.

During the Summer & Fall of 2012, the faculty at MassBay Community College worked on the design and implementation of the "***Cerberus***" Beowulf cluster [4]. ***Cerberus*** is an 8-node dual-core Linux cluster with **CUDA** [5] support. This project is based on a professional development grant [6] with the idea of creating labs and microlabs to introduce parallelism and distributed systems concepts into the CS curricula. ***Cerberus*** is one of the few Beowulf clusters built at a community college in the US and probably second in Massachusetts.

Fractals are an example of an "*embarrassingly parallel problem*". There is no dependency between the data. We implemented a distributed Mandelbrot set fractal.

We also developed a distributed DES brute-force algorithm. Once a slave node finishes its work, it sends the result back to the master. If there is more work to do, the master sends another group of keys to the slave node.

We developed GUIs for the two **MPI** applications written in **Python** using libraries like **TKinter**. Communication with the master node (**C/MPI**) uses the **TCP** protocol. The UI tier receives events serialized as **JSON** [7] objects from the **MPI** tier. Serializing events

in JSON was chosen because of Python's support for marshaling Python Dictionaries and Lists in the standard library. That will allow us to expand the GUIs when a new type of visualization schema is required.

OPTIONAL REFERENCES

[1] Benoit B. Mandelbrot. **The Fractal Geometry of Nature**. W.H.Freeman & Co Ltd. 1983.

[2] U.S Department of Commerce, National Bureau of Standards. **Data Encryption Standard**. Federal Information Processing Standards Publications, No. 46. January 1977.

[3] Richard L. Graham, Galen M. Shipman, Brian W. Barrett, Ralph H. Castain, George Bosilca, Andrew Lumsdaine. **Open MPI: A High-Performance, Heterogeneous MPI**. In Proceedings, Fifth International Workshop on Algorithms, Models & Tools for Parallel Computing on Heterogeneous Networks. Barcelona, Spain. September 2006.

[4] Robert W. Lucke. **Building Clustered Linux Systems**. Prentice Hall PTR. 2005.

[5] David B. Kirk, Wen-mei W. Hwu. **Programming Massively Parallel Processor: A Hands-on Approach.** Morgan Kaufmann & NVIDEA. Second Edition. 2013.

[6] Giuseppe A. Sena. ***Developing Labs and Microlabs for Adding Parallelism and Distributed Computing into Computer Science Curricula***. Professional Development Grants, Massachusetts Bay Community College, Computer Science Department. September 2012.

[7] tutorialspoint. **JSON Tutorial**. http://www.tutorialspoint.com/json/index.htm