Names: Aditya Sharma Pranav Angiya Janarthanan
Professor: maria.striki@rutgers.edu
Course: 16:332:516:01 Cloud Computing and Big Data
Term: Fall 2024
Due Date: Oct 30, 2024 11:59 PM EDT

Final Project Proposal

Group Number:

1

Names and Emails:

Aditya Sharma - as4108@scarletmail.rutgers.edu
Pranav Angiya Janarthanan - pa446@scarletmail.rutgers.edu

Names: Aditya Sharma Pranav Angiya Janarthanan
Professor: maria.striki@rutgers.edu
Course: 16:332:516:01 Cloud Computing and Big Data
Term: Fall 2024
Due Date: Oct 30, 2024 11:59 PM EDT

Assignment:

> Towards this end, for your proposal, you need to write a few paragraphs to describe your final project idea, your motivation, the method you are going to follow, the tools you will be using, and an analysis of how much time each of your major steps takes. Your final project should be worth of 5-6 weeks of work.

Proposal:

The primary goal of this project is to evaluate and compare the efficiency of a gravity simulation algorithm implemented using two different computing frameworks: PySpark and CUDA. Gravity simulators, which model gravitational interactions between celestial bodies, involve significant computational demands due to the need for iterative pairwise force calculations and position updates. This computational load makes them ideal candidates for exploring performance trade-offs in distributed computing versus GPU-accelerated processing.

CUDA, a GPU computing platform, provides parallel processing capabilities to accelerate the arithmetic-heavy nature of gravitational force calculations. At the same time, PySpark offers distributed processing across multiple cores, enabling efficient data partitioning and task parallelization. By comparing the performance of PySpark and CUDA implementations on Google Colab, this project aims to highlight differences in speed, resource utilization, and scalability, contributing valuable insights for selecting computing frameworks for scientific simulations.

Tools and Technologies:

- **Google Colab** as the development platform.
- **PySpark** for distributed data processing and task parallelism.
- **CUDA** for GPU-accelerated computation.
- Built-in Colab profiling tools to monitor and compare memory and CPU/GPU utilization.

Implementation Using PySpark:

- Implement the gravity simulation algorithm in PySpark.
- Each partition will calculate gravitational forces for a subset of bodies, using transformations such as map and reduction to compute forces and positions iteratively.
- Measure the time taken for each timestep and record resource usage.

Implementation Using CUDA:

- Implement the same gravity simulator in CUDA through the use of Numba library to perform both vectorization capabilities as well as CUDA Kernels for non-Ufunction applications.
- Write CUDA kernels to execute computations for gravitational forces and position updates simultaneously for large numbers of bodies
- Measure computation time per timestep and record GPU resource usage.

Comparison and Analysis:

- Collect and analyze both implementations' computation time, memory usage, and scalability metrics.
- Compare the results, examining how the distributed computing model of PySpark fares against the GPU acceleration of CUDA.
- Identify strengths and weaknesses of each framework for gravitational simulations, focusing on areas such as computation speed and resource optimization.

Timeline of Major Steps:

- **Week 1**:
    - Set up a Gravity Simulation environment for both CUDA and Spark to allow for a fair comparison between both applications.
- **Weeks 2 - 4:**
    - Aditya Sharma : Implement the PySpark version of the simulator.
    - Pranav Angiya Janarthanan : Implement the CUDA version of the simulator.
- **Weeks 5 - 6:**
    - Compare results, analyze findings, fine-tune results, and finalize the project report with insights on the relative efficiencies of PySpark and CUDA.