Benchmarking A State-of-the-art GPU Simulator

Northeastern University
College of Engineering

Yifan Sun, Ph.D. student, Northeastern University
Xin Li, REU Participant Student, Bunker Hill Community College
David Kaeli, Electrical and Computer Engineering, Northeastern University

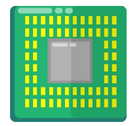
Grant #1559894

Abstract

Akita is a multi-GPU simulator developed with Google's Go programming language. Akita uses machine learning workloads to demonstrate how an accelerator can impact performance. Akita uses reinforcement learning to schedule events within the simulator to collect performance data. This poster presents performance data on the Akita simulator.

GPU

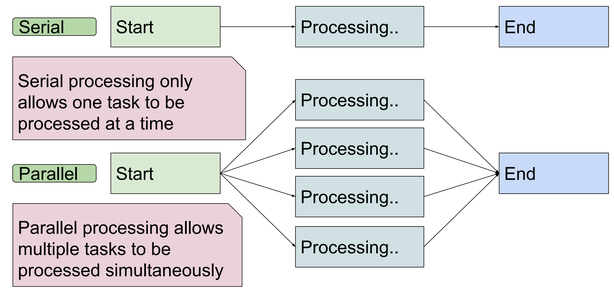
Serves as an accelerator - executes a lot of concurrent tasks efficiently



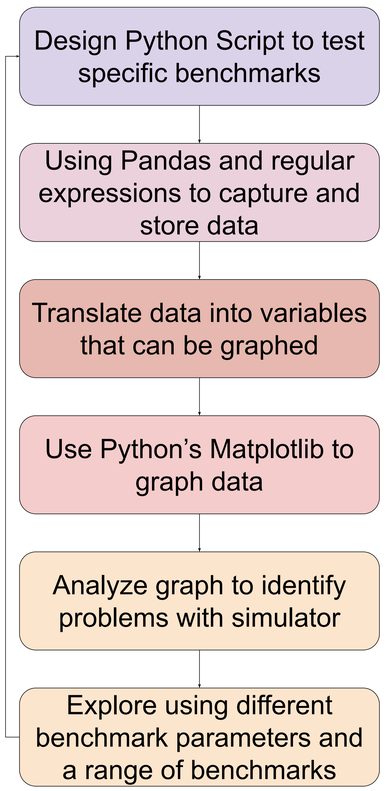
CPU

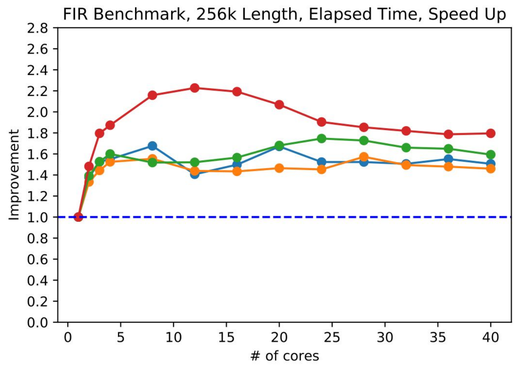
A general purpose multi-core device that can manage work and tasks across the system

Parallel Computing



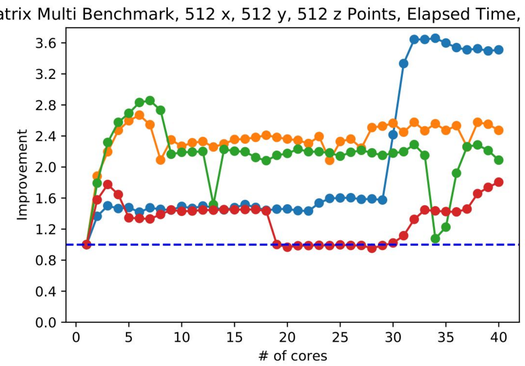
Process





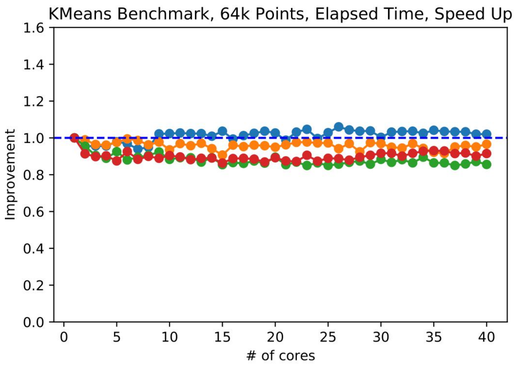
FIR Benchmark

- Benchmark shows that initial performance is good
- The best speedup is achieved at 2 cores
- We achieve peak performance at ~8 cores
- Adding additional cores does not improve performance
- 4 GPUs seem to be sweet spot



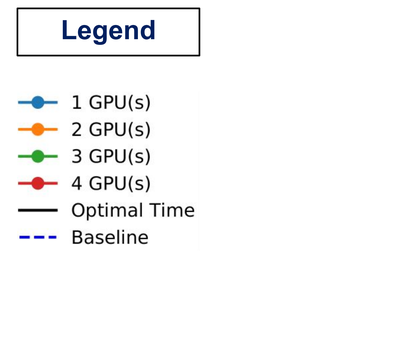
Matrix Multiplication Benchmark

- Benchmark performance is not uniform
- Scalability (i.e., increasing the number of GPUs) is not observed



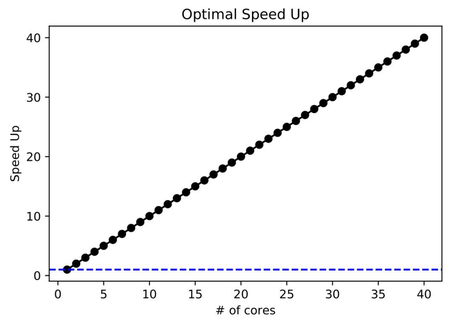
K-means Benchmark

- Benchmark scales linearly
- The performance in this benchmark are almost all negative when running parallely
- Only single GPU has a positive speed up



- *The **elapsed time** is the amount of time needed to complete the benchmark*
- *The **speedup** is a performance metric, calculated by taking the design's execution time and dividing by the baseline design's execution time*
- *We scale the number of cores used in each design, and compare against a single-core design*
- *The more cores available for parallel processing, the faster the predicted performance should be. Any test after core 1 is performed in parallel.*

Intended Performance



Python

- Pandas & Regular Expression to capture data
- NumPy & MatPlotLib Graph and analyze data

Go

- The language Akita is built on

Conclusions

- GPU simulator is not presently achieving peak parallel performance
- Many bottlenecks are present in the simulator's parallel engine
- In some benchmarks, initial performance is promising, but degrades as we scale

Future Work

- Develop a Python-based capability to better interface/execute graph and list data automatically
- Improve Akita's engine to tune performance
- Implement different types of machine learning algorithms

Acknowledgements
Yifan Sun - PhD Student, David Kaeli - Principal Investigator,
Claire Duggan - Director of Center for STEM Education, Camara Johnson - REU D3/POWER Program Coordinator