Computing Platform Benchmark

By

Boonyarit Changaival

King Mongkut's University of Technology Thonburi (KMUTT)

Introducing Myself!

- A summer student at CERN this year
- Worked in ALICE O2 project
  - GPU benchmarking for ITS Cluster Finder
- Carry on this summer project to be a Master Thesis
  - Computing Platform Benchmark with two advisors
    - Prof. Tiranee Achalakul, KMUTT
    - Mr. Sylvain Chapeland, ALICE O2, CERN
  - Study platforms through various implementations (CUDA, C, OpenCL) of ALICE applications

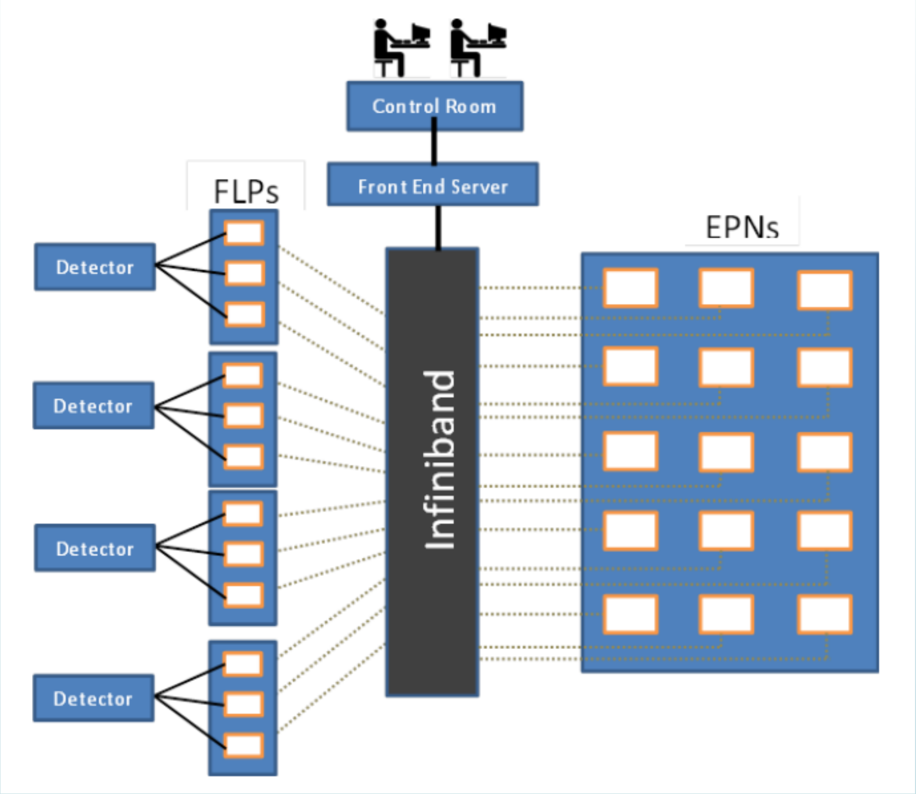
Outline

- ALICE Upgrade
- Why Platform Benchmarking?
- Survey Discussion
- Example Applications
- Evaluation Method
- Initial Result
- Conclusion

ALICE Upgrade

- Expected to be installed in 2018
- What's new?
  - Improve the read-out rate
    - Peak at 1TB/S
  - Improve Impact parameter resolution
  - Improve tracking efficiency by increasing granularity
  - Improve the computing system
    - Processing data online

Upgraded System Architecture



Upgraded System Architecture

- First Level Processor (FLP)
  - connected to the receiver at the detector
  - grouping and aggregating each collision of particle inside the ring (Reducing data)
- Event Processing Node (EPN)
  - For calculation and reconstruction for physic experiment
  - Receive processed data from FLP

Why benchmarking?

- To find out which platform produce the highest throughput for ALICE applications
- Each platform will have its own implementation for optimum result
- The end result will be used to suggest the suitable platform for each ALICE application type

Targeted Accelerators

- Graphic Processing Unit (GPU)
  - High performance per cost and energy efficiency
  - Had been accepted and used widely to accelerate scientific application
- Many Integrated Core (MIC)
  - Fewer processors than GPU, but each is more powerful
  - Highly portable (compare to CUDA&OpenCL)
- Accelerated Processing Unit (APU)
  - CPU+GPU on the same chip
  - GPU can access CPU memory directly
  - Consume low energy

Project Objectives

- To study the potential performance of each accelerators for ALICE applications
- To study factor(s) that affect the performance of applications on each accelerators
- To study the performance of OpenCL on all targeted accelerators
- To study the tradeoffs between each accelerator

Questions

- The result should answer these questions.
  - What is the performance overhead in OpenCL and CUDA? Does it worth the portability tradeoff?
  - Which accelerator produces the best result with OpenCL implementations?
  - Which accelerators should be suggested to be integrated in the upgraded ALICE system?

Survey Discussion

- Several previous works had been done
  - “A CPU, GPU, FPGA System for X-ray Image Processing using High-speed Scientific Cameras” (Binotto et al., 2013)
  - “Accelerating Geospatial Applications on Hybrid Architectures” (Lai et al., 2013)
  - “MIC Acceleration of Short-Range Molecular Dynamics Simulations” (Wu et al, 2010)
  - Face detection, Ocean Surface simulation, Dwarfs and the likes

Survey Discussion

- Yet, they are not quite connected with ALICE Application
  - Different Data Format
  - Different Algorithms and problem specifications
- To optimize the result, better work with the real problem definitions

Application Categories

- Categorized into 3 category
  - Data Intensive
  - Computing Intensive
  - Communication Intensive
- Communication intensive applications are not presented in ALICE
  - Only Data Intensive and Computing Intensive will be focused

Data Intensive

- High dependency between each element in the data
- Data is needed to be accessed and updated multiple times
- Example
  - ITS Cluster Finder
    - Put particles into groups
    - Calculate the "Center of Gravity" of the cluster
    - Discard coordinates and use only CG to represent the cluster

Computing Intensive

- Most of the work is computation
- Little to none dependency between elements
- Sometimes, Embarrassingly parallel can be used
- Example
  - TPC Track Identification
    - Using Hough Transform to identify track
    - True computing intensive application
    - Highly Parallelizable

Design of Experiment

- Responses
  - Throughput
  - Scalability
- Control Factor: Type of platform, Languages
  - GPU (CUDA and OpenCL)
  - MIC (C and OpenCL)
  - APU (OpenCL)
- Blocking Factor: Application Category
  - Data Intensive and Computing Intensive

Design of Experiment

- Experiment Plan
  - Throughput Analysis

Application category / Accelerator	Data Intensive		Computing Intensive	
GPU	C CUDA	OpenCL	C CUDA	OpenCL
MIC	C, OpenMP	OpenCL	C,OpenMP	OpenCL
APU	OpenCL	-	OpenCL	-

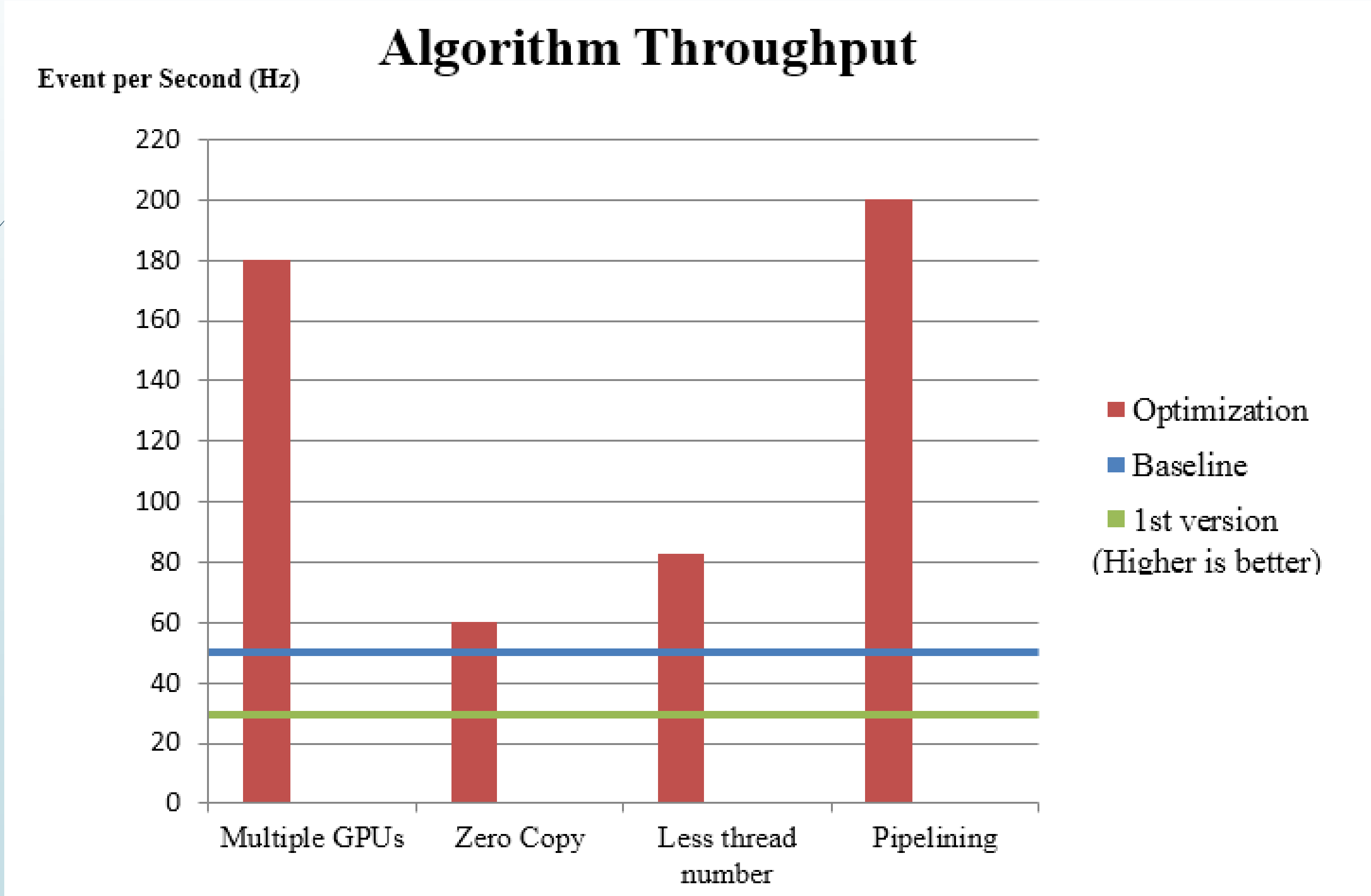
  - Scalability Analysis
    - Vary the thread numbers
    - Plot the Throughput against Thread Numbers
    - The trend in the graph will determine the scalability

Evaluation

- Throughput
  - Set the baseline performance
    - Using the CPU result
  - Speed up from the baseline is computed
  - Determine the most suitable accelerator from the highest throughput
- Scalability
  - Fixed input size with varied thread numbers
  - Varied input size and fixed thread numbers
  - Throughput should be on the rise when thread number is increased
  - Maintain the peak performance when input size is increased

Initial Result

- ITS Cluster Finder on Tesla K20xm



Initial Result

- OpenCL implementation of ITS Cluster Finder was completed
- Showed similar results as CUDA
- APU and MIC is not yet tested
- Next is to improve it with the pipeline method

Discussion

- High dependency made it hard to work efficiently on GPU
- GPU provide very little synchronization in Kernel
- Not in the GPU specialties: Only load, compare and store
- Data Intensive should perform better on MIC (from speculation)
- Data Intensive can then be separate into two
  - With dependency and No dependency

Expected Milestone

- January, 2015
  - Optimize CUDA and OpenCL implementation of Cluster Finder
  - C Implementation for Cluster Finder to be tested on MIC
  - Study the TPC Track Identification problem definition and design
- February, 2015
  - Complete all implementations of TPC Track Identification
  - Acquire more examples for implementation

Conclusion

- ALICE Upgrade calls for a high performance computing system
  - Cope with the higher read-out rate
  - Online processing
- Accelerators are aimed to be integrated to increase the throughput
- Benchmark is done to suggest the most suitable platform
  - Using ALICE applications to benchmark
  - GPU, MIC and APU