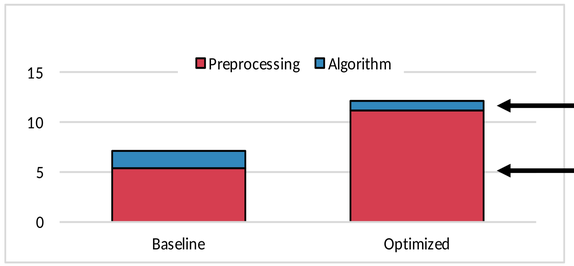
Data Integration Tasks on Heterogeneous Systems Using OpenCL

Clayton Faber* Anthony Cabrera* Orondé Booker* Gabe Maayan† Roger Chamberlain*
*Washington Univ. in St. Louis †Rensselaer Polytechnic Inst.

MOTIVATION

An often overlooked pain point of big data applications is the need of data transformation as a pre-processing step. DIBS Benchmark [1] characterizes these applications. We utilize OpenCL to implement a subset of the apps in DIBS on three separate platforms and evaluate their performance. We also evaluate some implementation tuning parameters that do not affect the functionality but instead the performance of these applications.



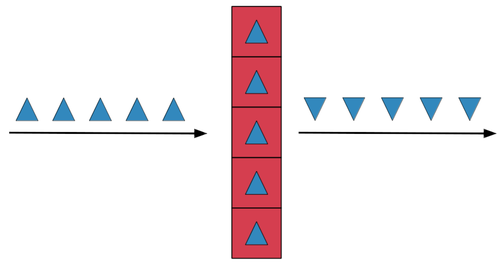
Execution time of breadth first search on twitter data. [2]

CHARACTERISTICS

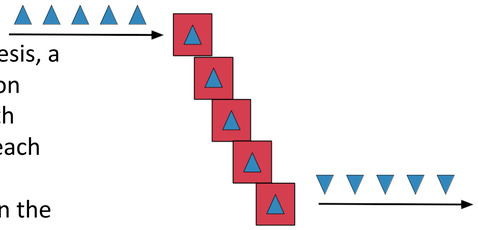
- The data integration apps in DIBS have an overall appearance of a streaming application, however, there is potential for contention in individual applications
- Fix → Float and FASTA → 2Bit could be considered embarrassingly parallel.
- GoTrack → CSV and IDX3→TIFF are more nuanced as each data record is made of multiple elements that can be dependent on other elements.
- Applications that require more finesse in their operation can create problems for certain types of parallel architectures where barriers are of great concern.
- A work item may be a block of data records or a single data record depending on the application.

PLATFORMS

Three separate architecture platforms (CPU/GPU/FPGA) are used to evaluate OpenCL's performance when performing data integration tasks. Both the GPU and CPU Implement some form of multi-work item execution while the FPGA implements a deeply pipelined execution style using a single work item.



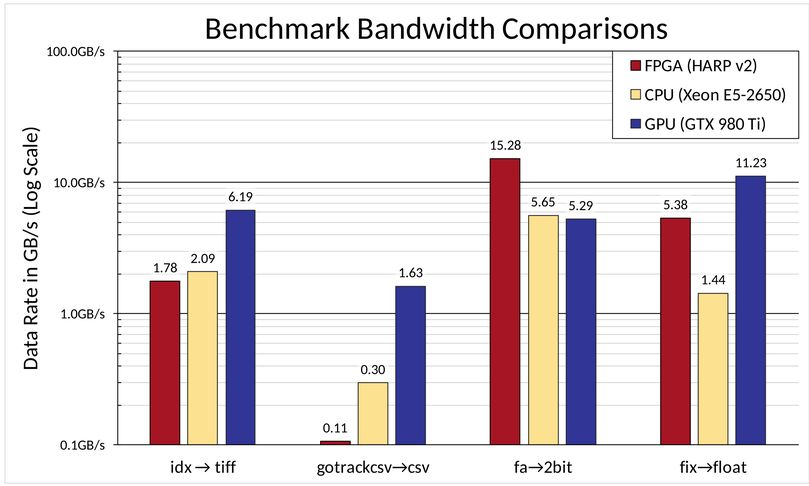
In multi-work item execution some type of SIMD unit is implemented in hardware and multiple work items are executed in sequence.

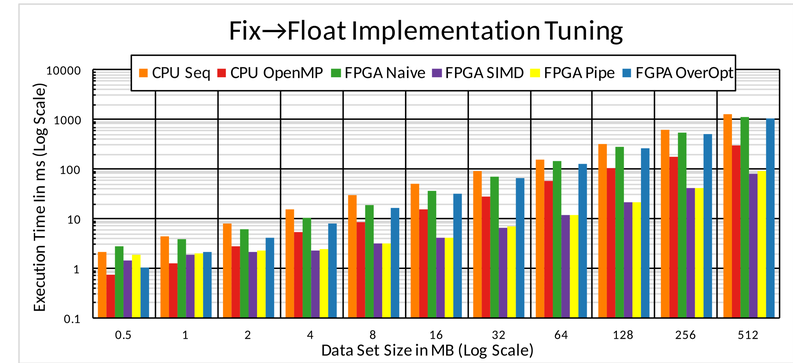


Through high level synthesis, a single work item execution style can be realized. With specialized hardware in each stage each clock cycle produces an output when the pipeline is full.

CPU: Xeon E5-2650 @ 2.0 GHz – AWS dedicated instance, SVM memory
GPU: Nvidia GTX 980 Ti – OpenCL 1.2 (Using cl_mem)
FPGA: HARP v2 (Arria 10 in socket w/ Cache coherent bus), SVM memory

RESULTS





DISCUSSION

- Although we see promising speedups compared to the reported DIBS paper bandwidth, performance gains are not universal across the board. The CSV parser (GoTrack → CSV) is the worst offender.
- The FPGA and CPU actually beat out the GPU implementation when performing the FASTA → 2Bit transformation.
- We attempted to make adjustments to the Fix → Float application when implementing and found that trying to turn on all optimizations netted us worse performance.
- Further work will include implementing different styles of execution on all platforms to see if performance improves, particularly on the CSV parser.

[1] Cabrera, Faber, et al. 2018. DIBS: A Data Integration Benchmark Suite. ICPE '18.
[2] Malicevic, Lepers et al. 2017. Everything you always wanted to know about multicore graph processing but were afraid to ask. USENIX ATC '17

This work was funded in part by NSF grants CNS-1205721, CNS-1527510, CCF-1527692, and CNS-176350
Special thanks to Intel for the Hardware Accelerator Research Program

International Workshop on OpenCL, 2019. Boston, MA