HEP-CCE Portable Parallelization Strategies

Matti Kortelainen
CSAID Roadmap meeting
9 November 2023

HEP-CCE

Introduction

- Following slides show the outcome and remaining plans of the "Portable Parallelization Strategies" thrust of the HEP-CCE "phase 1"
  - Based on a review DOE had in July
- FNAL contributors
  - Meghna Bhattacharya
  - Matti Kortelainen
  - Martin Kwok
  - Alexei Strelchenko
  - Oliver Gutsche (technical lead)

Software and Hardware Support Matrix in 2019

	CUDA	Kokkos	SYCL	HIP	OpenMP	alpaka	std::par
NVIDIA GPU			*codeplay*	*hipcc*			*nvc++*
AMD GPU			*hipSYCL*	*hipcc*			
Intel GPU			*oneAPI*				*oneAPI:dpl*
x86 CPU			*oneAPI*				*gcc*
FPGA							

Fermilab

Software and Hardware Support Matrix in 2023

	CUDA	Kokkos	SYCL	HIP	OpenMP	alpaka	std::par
NVIDIA GPU				*hipcc*	*nvc++ LLVM, Cray GCC, XL*		*nvc++*
AMD GPU			*openSYCL intel/llvm*	*hipcc*	*AOMP LLVM Cray*		
Intel GPU			*oneAPI intel/llvm*	CHIP-SPV: *early prototype*	*Intel OneAPI compiler*	*prototype*	*oneapi::dpl*
x86 CPU			*oneAPI intel/llvm openSYCL*	*via HIP-CPU Runtime*	*nvc++ LLVM, CCE, GCC, XL*		
FPGA				*via Xilinx Runtime*	*prototype compilers (OpenArc, Intel, etc.)*	*protytype via SYCL*	

HEP Testbeds

- FastCaloSim
  - ATLAS parameterized LAr calorimeter simulation
  - 3 simple kernels (large workspace reset, main simulation, stream compaction)
  - 1-D and 2-D jagged arrays
  - small data transfer d->h at end of each event
- Patatrack
  - CMS pixel detector reconstruction
  - 40 kernels of varying complexity and lengths (many are short)
  - good test for latency, concurrency, asynchronous execution, memory pools
- Wirecell Toolkit
  - LArTPC signal simulation
  - 3 kernels: rasterization, scatter-add, FFT convolution
- P2r: CMS "propagate-to-R" track reconstruction in a single kernel

HEP Testbeds

- FastCaloSim
  - ATLAS parameterized LAr calorimeter simulation
  - 3 simple kernels (large workspace reset, main simulation, stream compaction)
  - 1-D and 2-D jagged arrays
  - small data transfer d->h at end of each event
- Patatrack
  - CMS pixel detector reconstruction
  - 40 kernels of varying complexity and lengths (many are short)
  - good test for latency, concurrency, asynchronous execution, memory pools
- Wirecell Toolkit
  - LArTPC signal simulation
  - 3 kernels: rasterization, scatter-add, FFT convolution
- P2r: CMS "propagate-to-R" track reconstruction in a single kernel

Testbed Completion Status

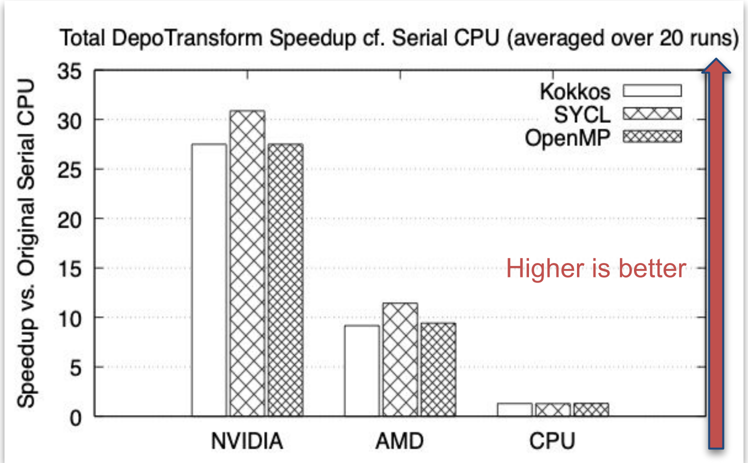
	Kokkos	SYCL	OpenMP	Alpaka	std::par
Patatrack	Done	Done*	WIP	Done*	Done compiler bugs
Wirecell	Done	Done	Done	no	Done
FastCaloSim	Done	Done	Done	Done	Done
P2R	done	Done	OpenACC	Done	Done

Metrics

- Ease of Learning
- Code conversion
  - From CPU to GPU and between different APIs
- Extent of modifications to existing code
  - Control of main, threading/execution model
- Extent of modifications to the Data Model
- Extent of modifications to the build system
- Hardware Mapping
- Feature Availability
- Interoperability
  - Interaction with external libraries, thread pools, C++ standards
- Address needs of large and small workflows
- Long term sustainability and code stability
- Compilation time
- Run time/Performance
- Ease of Debugging
- Aesthetics

Wirecell Toolkit

- Three major steps of LArTPC simulation
  - **Rasterization**: depositions ⟶ patches (small 2D array, ~20×20)
    - # depo ~100k for cosmic ray event
  - **Scatter adding**: patches ⟶ grid (large 2D array, ~10k×10k)
- Summary
  - Restructured the code to expose more parallelism
  - Wrappers to use optimized vendor libraries
  - Ported to **CUDA (partial)**, **Kokkos**, **SYCL**, **OpenMP** and **std::par** implementations
  - Developed a stand-alone testing framework (without LArSoft dependence)
  - Validated and benchmarked Kokkos, SYCL and OpenMP implementations; Achieved similar performance with different portability layers.



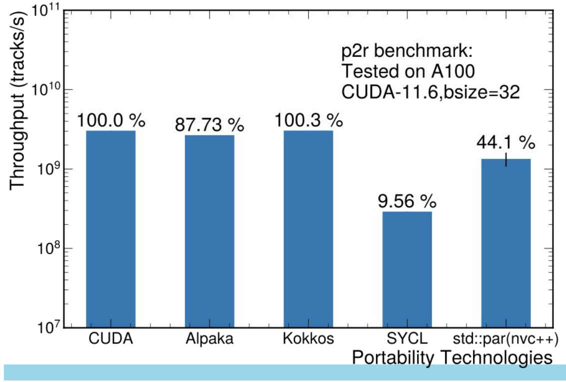
Speedup in DepoTransform compared to original CPU on NVIDIA V100, AMD Radeon Pro VII, and AMD Ryzen 24-core CPU with Kokkos, SYCL and OpenMP

Fermilab

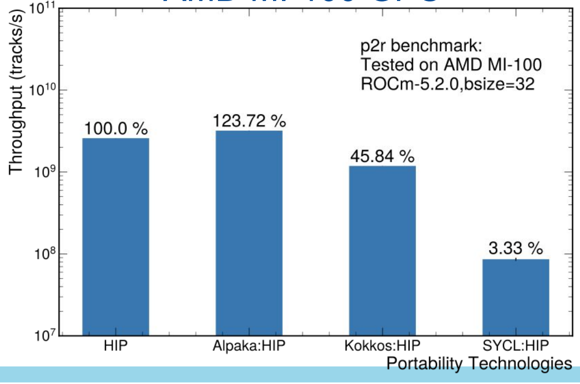
P2r summary

- p2r is a relatively lightweight benchmark
  - Performs core math of track reconstruction (track propagation and Kalman updates)
  - Easy to port
  - Easy to experiment with features/data-layout
- Lessons learned:
  - Alpaka/Kokkos give close-to-native performance in NVIDIA/AMD GPUs and CPU
  - SYCL/std::par performance are significantly behind despite sharing very similar implementation

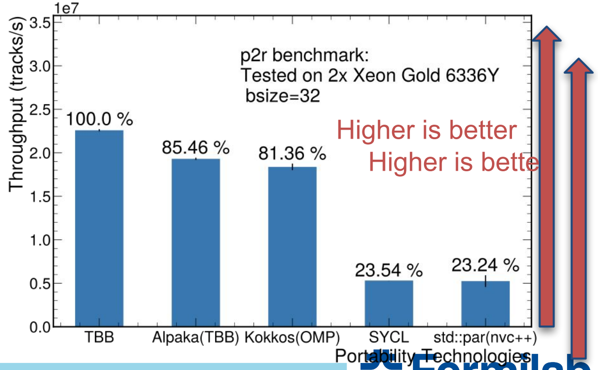
NVIDIA A100 GPU



AMD MI-100 GPU

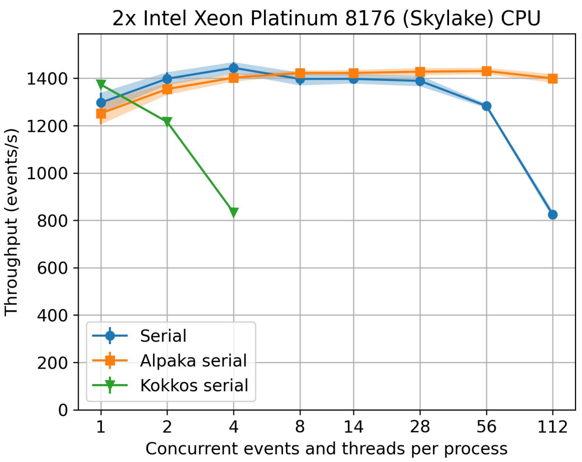


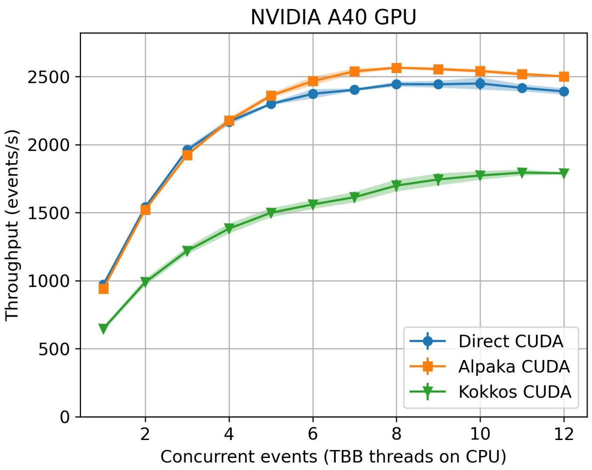
Intel Xeon Gold 6336Y CPU

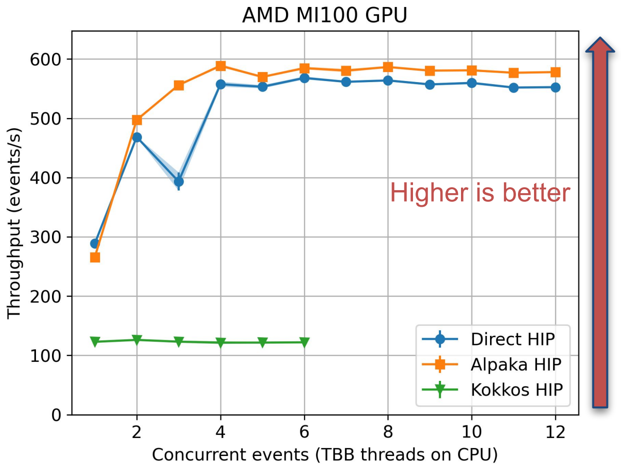


Patatrack summary

- Most complex use case: CMS pixel detector reconstruction from raw data to pixel tracks and vertices, multithreaded mock framework and build system
  - closest approximation of integration in an experiment framework without actually doing it
  - 40 kernels divided in 5 “framework modules” using rich set of CUDA features







Patatrack summary

- Lessons learned
  - Best performance on CPU, and NVIDIA and AMD GPUs with Alpaka
  - Kokkos currently difficult to work with in a concurrent application, overheads
  - SYCL (Intel oneAPI implementation): compilation problems, overheads
  - std::par: compilation problems, crashes, leads to many more kernels (expect poor performance)
  - OpenMP Target offload: compilation problems, data movement is concern

Impacts on HEP

- Final stage of CCE/PPS is reporting back to experiments
  - Not yet there, but have made multiple interim reports
  - ACAT, CHEP, HSF, IRIS-HEP
- Already affected what HEP experiments are doing in both short and medium term
  - CMS's choice of Alpaka
  - ATLAS Run4 milestones
  - DUNE signal processing kernels on GPUs
  - Contributions to Snowmass21 process
- Continue to interact with HEP experiments as we close out phase 1 of CCE
  - Focussed meetings and workshops with stakeholder experiments
  - Continued engagement with broader HEP community

Plans for 2023

- We have completed all major development work.
  - uniform benchmarking on standard platforms
  - investigating outstanding bugs with new compiler versions
- Remainder of year will be devoted to publishing and presenting results.
  - individual reports for each testbed
  - cross-cutting reports for portability layers
  - reporting back to stakeholder experiments
    - general meetings
    - focussed workshops
  - outreach to wider HEP and computing community
    - ~~Supercomputing 2023 BOF "Beyond Portability: How is your science domain coping with the heterogeneous era of HPC?"~~
    - Conferences
    - HSF, IRIS-HEP, SWIFT

Conclusions

- There is no "one size fits all" solution. Different applications will have different optimal solutions.
  - We have identified pain points with each portability layer
  - Allows us to offer useful guidance to experiments depending on their use cases
- Both software and hardware continue to rapidly evolve.
  - Need to monitor GPU ecosystems and update recommendations as needed
  - Emergence of C++ language level standards in the next 5 years may be a game changer
- See increasing use of ML solutions for previously purely "algorithmic" tasks and vendor shifts to ML optimized hardware.
  - Will expose a whole new set of issues