



Innovative Algorithms

David Lange and Heather Gray



Major Activities and Goals

Developing tracking algorithms for HL-LHC

Determining charged-particle trajectories (“tracking”) requires most reconstruction CPU

- Develop more efficient algorithms
- Develop more performant algorithms

Hardware accelerators are the way forward to speed up and reduce infrastructure cost

- Use of hardware accelerators for tracking
- ML on accelerators in realistic HEP apps

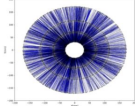




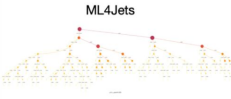
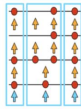
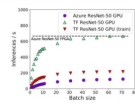
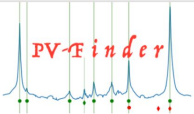
Re-engineering algorithms for hardware accelerators

Exploiting major advances in machine learning (ML)

Capitalize on industry and data science techniques and tools

- Investigate new HEP applications of ML
- Apply new ML techniques to HEP

Current Projects

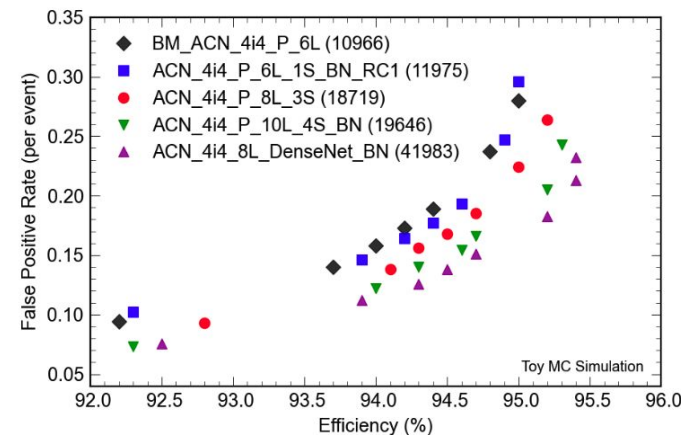
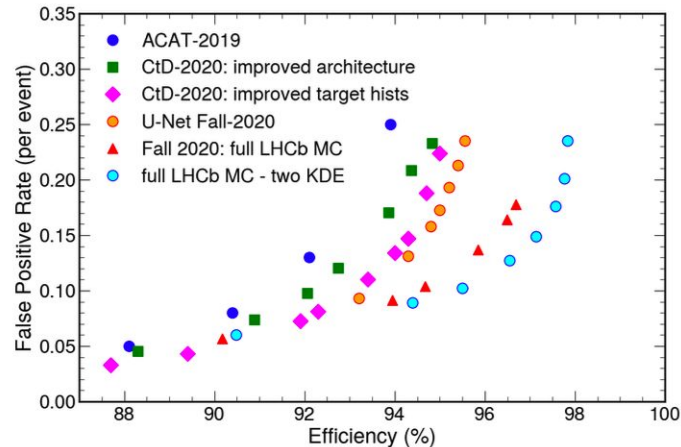
 <p>Accelerated GNN Tracking</p> <p>accel-gnn-tracking More information</p>	 <p>ACTS</p> <p>Development of experiment-independent, inherently parallel track reconstruction. More information</p>	 <p>exploratory-ml</p> <p>Analysis Reinterpretation More information</p>	 <p>FastPID</p> <p>Fast PID simulation for LHCb More information</p>
 <p>GPU Trigger Project</p> <p>Allen: a GPU trigger for LHCb More information</p>	 <p>ML4Jets</p> <p>Machine Learning for jets</p> <p>Machine learning for jets More information</p>	 <p>mkFit</p> <p>Modernizing Kalman filter tracking for CMS More information</p>	 <p>ML on FPGAs</p> <p>Fast inference of deep neural networks on FPGAs More information</p>
 <p>PV-Finder</p> <p>CNNs to find primary vertices More information</p>	<p>One goal for this retreat: update our webpages!</p>		

PV Finder

We extended our work on Pv-finder.

- we systematically studied performance for variations on the original architecture and developed a new (U-Net) architecture;
- we developed a new set of kernel density estimators (KDEs) that enable better performance *and* which a deep neural network (DNN) can learn;
- we extended our studies to use full LHCb Monte Carlo in addition to the “toy Monte Carlo” we used for initial studies;
- we developed a DNN to read in track parameters and produce KDEs which can be used by the original DNN to find primary vertices.

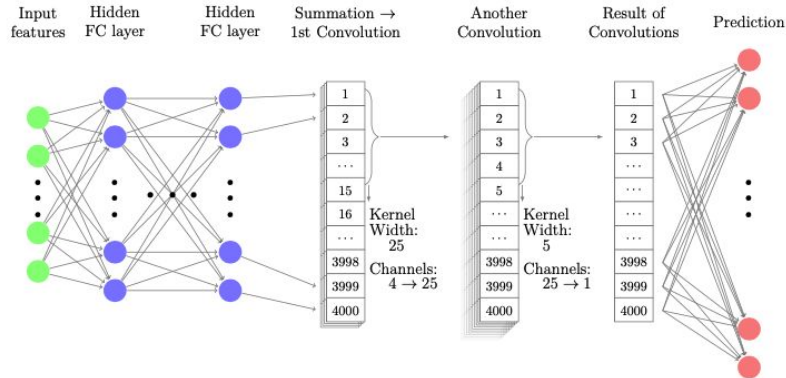
PI: Mike Sokoloff



PV Finder: Plans for Year 4

Our primary goal in Year 4 is to develop a fully performant DNN that starts with track information and finds primary vertices and instantiate it in both the GPU framework for Hlt1 and the CPU framework for Hlt2.

- improve the tracks-to-KDE DNN algorithm developed this past year;
- build a GPU inference engine that works in the Allen framework;
- develop an algorithm that associate's individual tracks with found primary vertices.

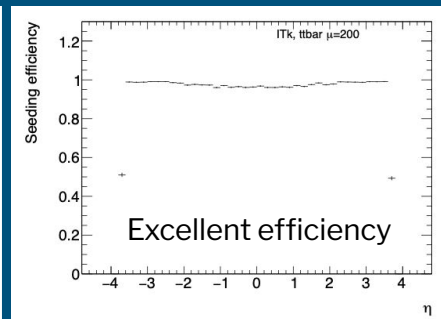
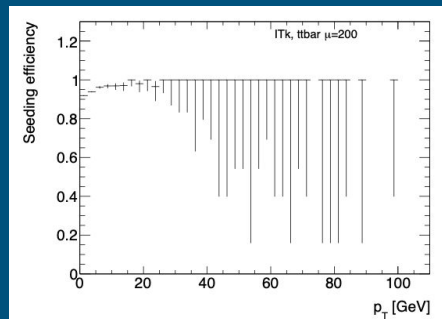
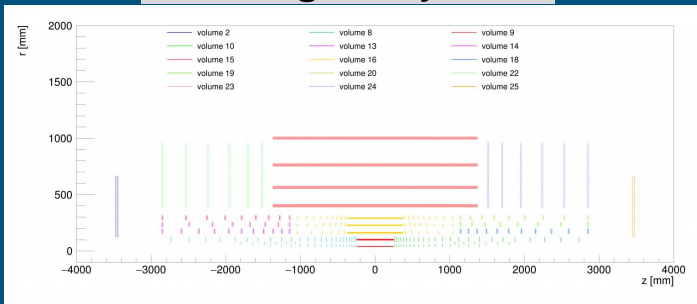


Explore PVFinder
for ATLAS (w.
Tompkins)

ACTS: Seeding on ATLAS ITk

PIs: Heather Gray and
Lauren Tompkins

ATLAS ITk geometry in ACTS



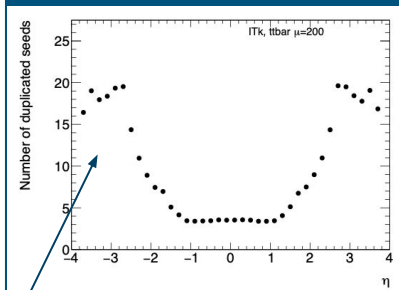
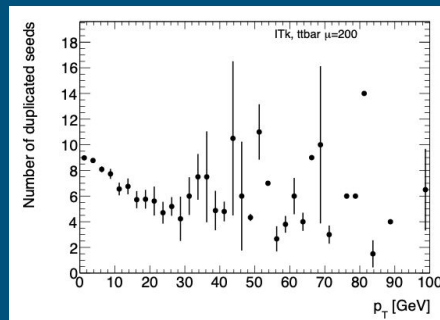
$ttbar$ events with $\mu = 200$

Preliminary results for seeding performance ($pt > 500$ MeV)

- 92% efficiency
- 65% fake rate

Two papers in advanced preparation (expected to be submitted this spring)

- A Common Track Software (ACTS)
- A GPU based Kalman Filter for High-Energy Physics Experiments



Further tuning required to reduce duplicates in the endcaps

ACTS: Improving Seeding & Vertexing

Layer Linking for Pixel Seed Finding

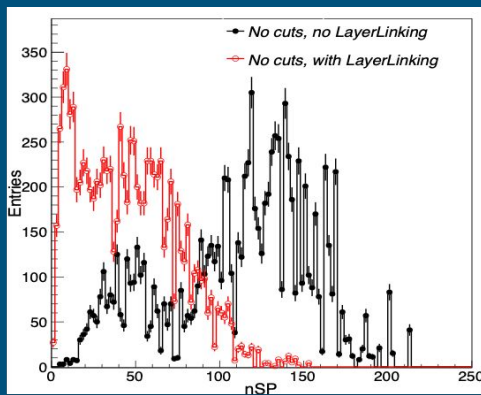
- Pixel layers are linked together based on probability learned from simulation
- In the absence of any other requirements, layer links reduce combinatorics by 50%
- Applying cuts to seeds reduces impact. Currently exploring application to more layers to maximize impact

Integrated ACTS vertexing algorithms into Athena:

- *Gaussian Track Density Seed Finder*
- *Full Billoir Vertex Fitter*
- *Iterative Vertex Finder*

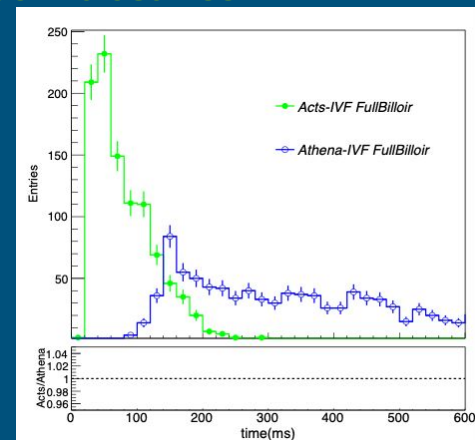
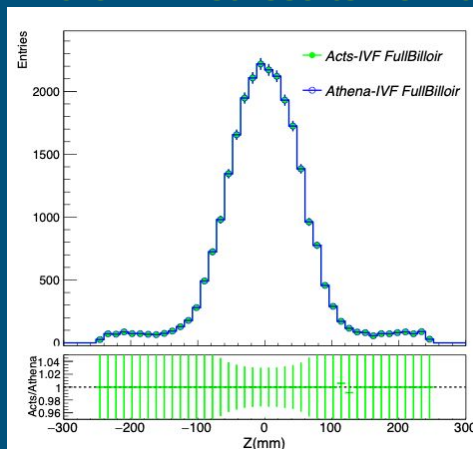
ACTS version shows same physics results but with a significant CPU time reduction

ACTS Athena Tracking Integration Task Force Launched in March ... first results from track fit last week



Technique from
D. Emelianov
and TrackML
Challenge

nSP: no. of top and bottom SpacePoints searched for potential seeds for a given middle SpacePoint



ACTS: Evolutionary Algorithms automatically optimize track seeding parameters

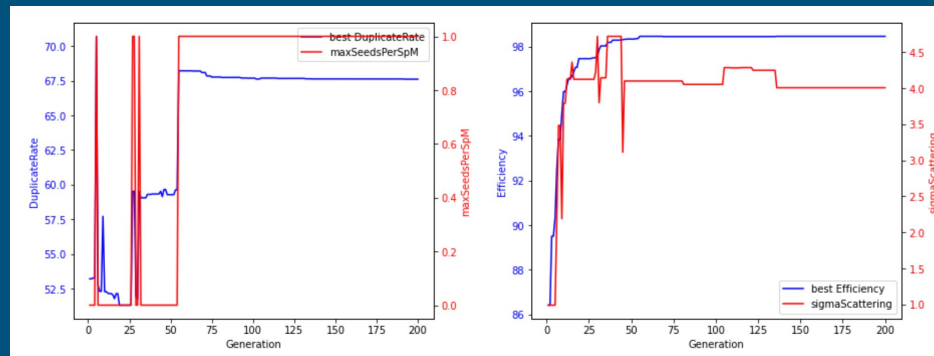
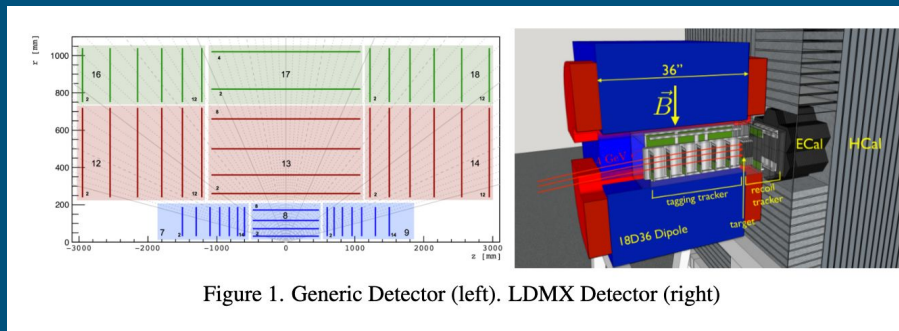
Tracking often requires extensive hand tuning of parameters

Developed strategy based on evolutionary algorithms (EA) to optimize track seeding parameters automatically.

Tested on ACTS generic detector (collider) and LDMX detector (fixed target).

- Improved efficiencies, lowered duplicate and fake rates in both cases
- EA worked best compared to hand tuning or automated parameter scan
- Plan to apply to ITK geometry

Paper accepted for vCHEP2021



Work of IRIS-HEP fellow & undergraduate Peter Chatain

ACTS Plans

- Core is moving rapidly towards deployment and commissioning over the next year (partnership with USATLAS)
 - Tuning ACTS tracking algorithms for ITk
 - ACTS tracking integration into athena
- R&D Directions
 - Tracking on accelerators
 - End-to-end tracking on GPUs with tracc (w/ USATLAS and HEP-CCE)
 - Machine-learning for tracking
 - Evolutionary algorithms for algorithmic tuning
 - Developing a library of various implementations to facilitate comparisons
 - Layer-linking for pixel seed finding
 - spotify bucket algorithm (fellow project)
 - Test LHCb PV-finder for ACTS generated ATLAS geometry, possibly port to ACTS (collab with M.Sokoloff, DIANA-HEP fellow)
 - ACTS exploration and testing for other experiments continues
 - LDMX, Belle-2, sPhenix, muon collider, EIC?

GPU trigger for LHCb: Allen

- In the next run, LHCb will have triggerless readout
- First-stage software trigger must run at 30 MHz
- Massive parallelization required → GPUs well suited
- Allen allows for an arbitrary sequence of configurable algorithms to be processed on a GPU
- Run trigger over $\mathcal{O}(1000)$ events in parallel
- Single exchange of input/output data with the CPU
- GPU/CPU memory allocation controlled by custom memory manager
- Allen adopted as the baseline implementation for Run 3 first-level trigger at LHCb
- The IRIS-HEP part of the project now primarily focussed on the monitoring element of Allen
- Monitoring of trigger rates, features of selected events and, where possible, features of events that aren't selected

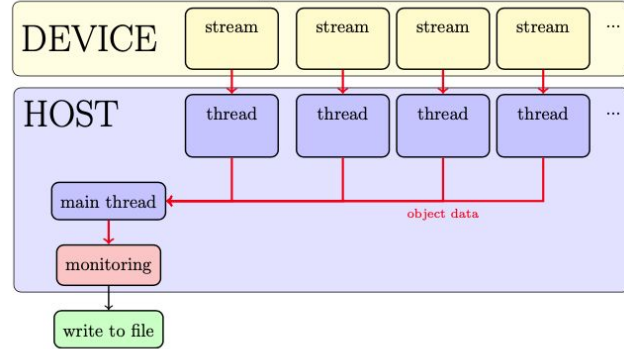
Selection of recent progress:

- Introduced functionality to split “data slices” of events from multiple data-taking runs into separate blocks allowing run change configuration to be performed between runs
 - Unit tests to test this feature in nightly builds
- Initial work towards moving monitoring task from dedicated thread to individual host algorithms...

Allen monitoring goal for Run 3

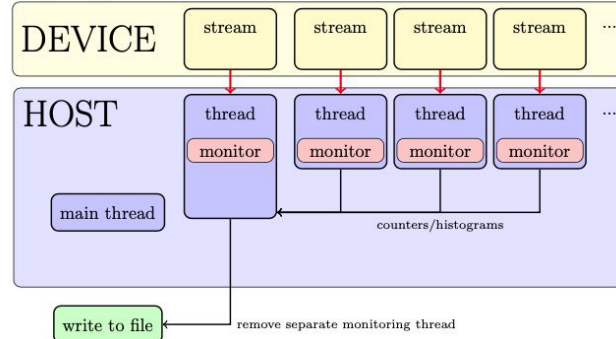
Remove dedicated monitoring thread
Thread-safe counters/histograms filled from individual threads

Current monitoring flow:



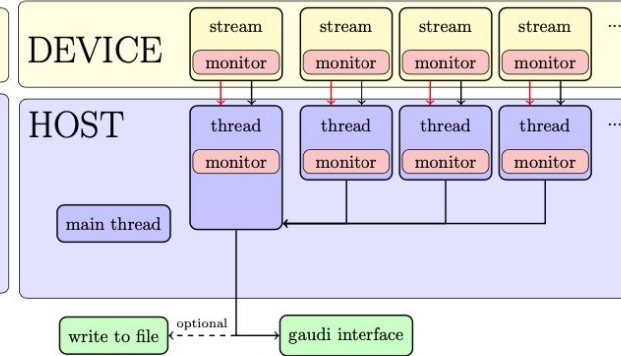
- Monitoring performed in dedicated thread
- Monitored quantities not stored in Allen's decision outputs must be copied to host and passed to monitoring thread

In progress:



- Monitoring functionality to be added to the base class for host algorithms to allow any host algorithm to define monitoring objects
- Extending this to also have monitoring in device algorithms will reduce the volume of data to be copied back from the device to the host
- For the start of Run 3 data taking, Allen must also output monitoring objects to LHCb's common gaudi interface for monitoring

Run 3 target:



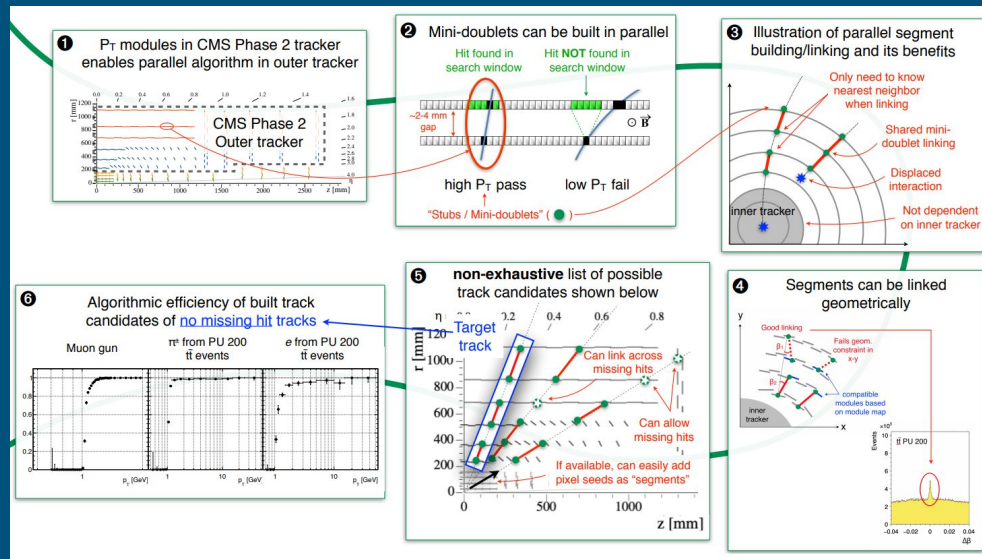
Parallelized & Vectorized KF Tracking (mkFit)

- Extended infrastructure to support multiple iterations, including hit masking.
 - Triggered a major re-structuring of hit format to be identical to that used in CMSSW, thereby minimizing reformatting overhead.
- Work on the CMSSW side to:
 - Configure mkFit
 - Accept tracks built by mkFit, fit and store in track collection
 - Run standard CMS track validation module
 - Performance results to be included in HLT TDR (to be submitted to LHCC in summer)
- Aiming for Run 3 tracking configurations using mkFit into next major CMSSW release cycle (12.X)

PIs: Peter Wittich,
Avi Yagil

Line Segment Tracking – LST

- Novel tracking algorithm designed and developed targeting GPUs.
 - Based on Phase-2 CMS tracker geometry.
 - Achieved good efficiency for PU200
- Current work focused on reducing linking combinatorics (redundancy in track building)
- Development on CPU, “mirrored” to GPU
 - Common framework and repository for both
 - Common validation suite, ~identical results (up to precision)
- Working to reduce memory footprint on GPU
- Testing various schemes (e.g. unified vs explicit allocation as well as nested parallelism, vs grid allocation)
- Technique also included in HLT TDR for HL-LHC



Plans for years 4 & 5

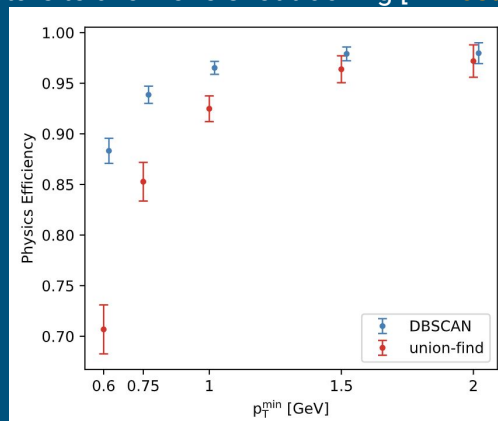
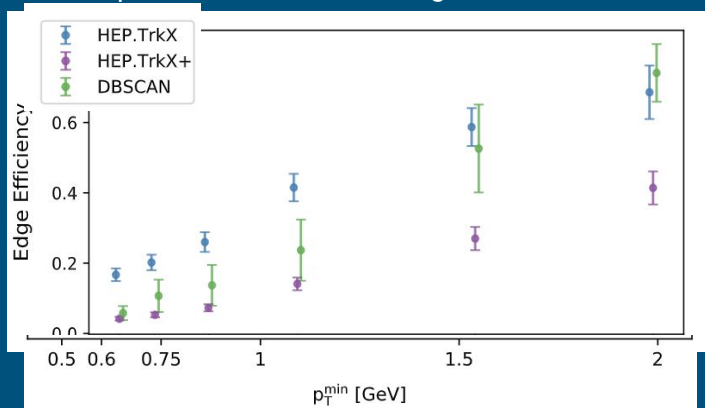
- mkFit Run3 related:
 - Run 3 integration / testing / deployment
 - Cleanup and consolidate implementation “shortcuts” physics tuning
- mkFit Phase2 related:
 - Adjust to phase-2 geometry, test, and adjust as needed
 - Usage of min-doublet?
 - Usage of patatracks as seeds?
- SLT related:
 - Continue development with emphasis on timing and memory footprint
 - Deployment into CMSSW and testing performance in-situ

For both: keep an eye on HW (e.g. AMD) as well as SW developments (e.g. One API by Intel)

GNN Tracking: Y3 Accomplishments

- Data Augmentation:
 - Added end-caps to graphs using TrackML dataset (other projects are barrel only) [UUC]
 - Added edge features to Interaction Network (IN) [Princeton] and edge classifier network [UUC]
 - Demonstrated improved performance (faster convergence, higher accuracy) using simple edge parameters (ϕ , η , d_r , d_R)
 - Hough Transform accumulator was most useful edge feature for edge classifier network but at very high computational cost
- Graph Construction
 - Converted ExaTrkX graph construction code from Pytorch to Pytorch Geometric [UUC]
 - Standardized code to measure graph construction edge efficiency and truth efficiency [Princeton]
 - Implemented DBScan-based graph construction in Cartesian and eta-phi space (achieved improved graph efficiency) [Princeton]
- Network Architecture Improvements
 - Achieved 99.7% edge classification efficiency and 85-94% tracking efficiency with $\sim 6,000$ learnable parameter IN (substantially smaller than previous architectures) [Princeton]
 - Implemented an instance segmentation based architecture to allow one-shot tracking [Princeton]

PIs: Peter Elmer,
Mark Neubauer

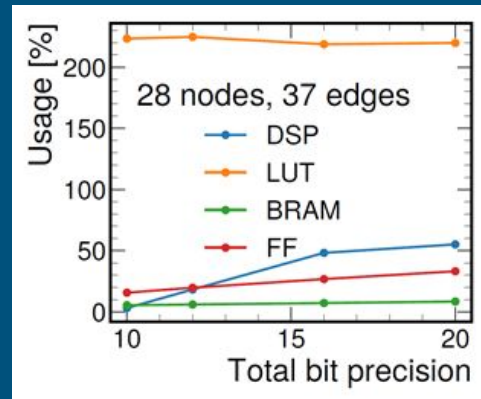
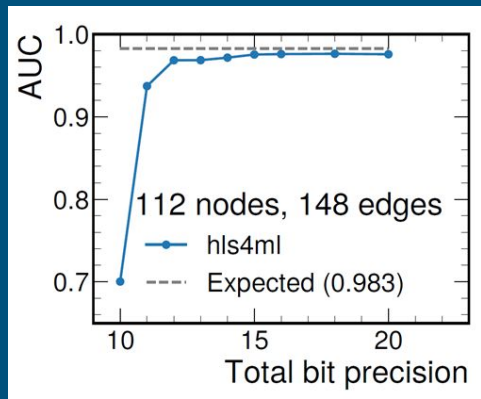
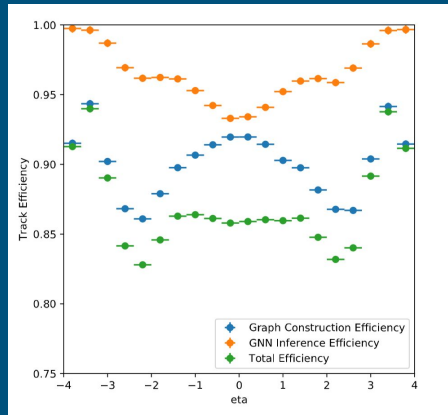
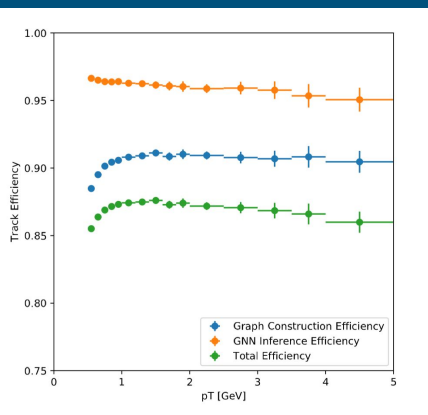


Left: graph construction edge efficiency (true edges/all edges) for range of p_T for 3 different algorithms

Right: track finding efficiency (using DBScan and UnionFind) for the IN architecture (found track counted if >50% of hits are from the same particle and >50% of main particle's truth hits are included)

Y3 Accomplishments Cont

- Tracking efficiency measurements
 - Defined 3 tracking efficiency working points (clustering, ExaTrk (see previous slide) and full match) [Princeton, UIUC]
 - Measured efficiency of UnionFind and DBScan track finding for IN and edge classifier [Princeton, UIUC]
- FPGA-based GNN Acceleration
 - Implemented first GNN in HLS for particle physics using [hls4ml](#) [UCSD]
 - Small graphs (pt>2GeV, 1/16 graph), 5 ns clock periods, good performance at <12,6> bit precision
 - Implementing user-friendly front-end code into hls4ml (Pytorch Geometric GNNs) [UCSD]
 - Implemented OpenCL-based GNN inference, data transfer latency is a large bottleneck [Princeton]
- Publications and (Selected) Talks
 - Papers: “[Charged particle tracking via edge-classifying interaction networks](#)” (vCHEP 2021), “[Instance Segmentation for One-Shot Conformal Tracking at the LHC](#)” and “[Accelerated Charged Particle Tracking with GNNs on FPGAs](#)” (NeurIPS ML4PS 2020)
 - Talks: [FastML for Science](#), [Accelerated AI for Big-Data Experiments](#), [IML Conference](#), [ML4SYS GNN Workshop Keynote](#)



Left: tracking efficiency with UnionFind for edge-classifier network vs pT (far left) and eta (center left)

Right: AUC (center right) and resource utilization breakdown (far right) for hls4ml implementation of IN on FPGA

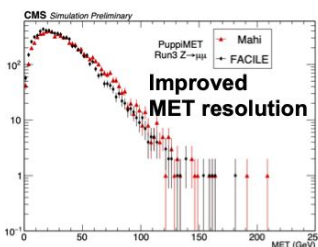
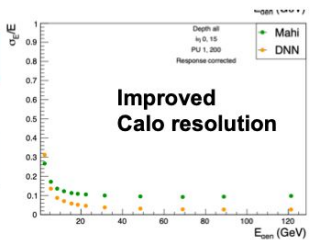
Vision for Y4 and Y5

- Planned improvements to FPGA implementations (Y4)
 - Scale to larger graph size [Princeton, UCSD]
 - Optimize matrix multiplication kernels in OpenCL [Princeton]
 - Explore pruning and QAT to reduce model size [UCSD]
 - Implement graph construction on FPGA to bypass data transfer bottleneck [Princeton]
- Efficiency comparisons to current tracking algorithms (Y4) [Princeton, UIUC, UCSD]
 - Measure inference time on different computing systems
 - Define comparable tracking efficiency working point (possibly consider parameter resolution)
- Evaluate GNN architectures with experimental data (Y4, Y5)
 - CMS data test in collaboration with TrackingPOG (organizing hackathon) [Princeton]
 - ATLAS data test in collaboration with ExaTrkX [UIUC]
 - Emulate real tracking environment for hardware acceleration [UCSD]
- Explore GNN architecture extensions and improvements (Y4, Y5)
 - Enforce invariance/equivariance to expected symmetries in learned GNN convolutions [Princeton]
 - Test different localization bounding shapes and optimize tracking branch for instance segmentation network [Princeton]
- Extend and improve ML Particle Flow algorithm [UCSD] (Y4, Y5)
 - Analyze graph structure towards interpretable/explainable model
 - Accelerate inference with co-processor as a service

HCal with GPU Trigger

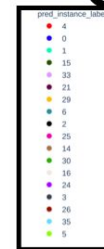
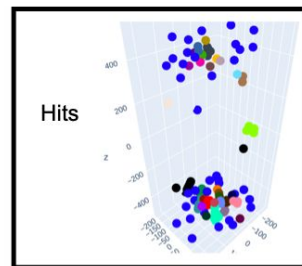
Current Highlights

ALGO	Throughput
MAHI CPU	60ms/evt
HCal (FACILE) GPU	0.45 ms/evt
HCal(FACILE) FPGA	0.10 ms/evt



- Algorithm gives similar/better performance to default algo
 - We are aiming to go through the scheme of full integration
 - Not necessarily trying to become the default algorithm
- Most recent status update:
 - Sonic at ML Forum April 7th

Going Beyond



Cluster labeling

We see this as a major advancement for ATLAS/CMS

- Ultimate goal is to go from raw inputs to cluster
 - 3D+timing clustering performed in one NN
 - preliminary 3D cluster algorithm with S.C. Hsu
 - ▶ Additionally working with members of MIT EECS
 - New NN architecture explicitly designed for fast inference
- Working to merge information into a single algorithm

PI: Phil Harris

ML4Jets: Y3 Accomplishments

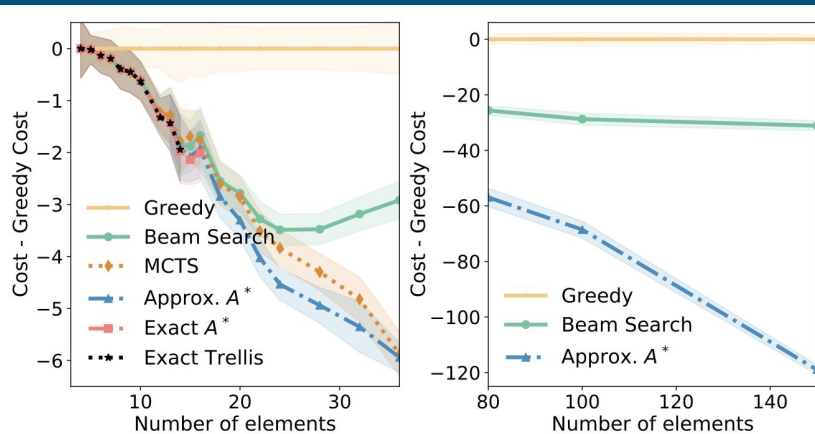
Pi: Kyle Cranmer

Reframed various problems in jet physics in terms of hierarchical clustering.

Partnered with researchers at NIST, UMass Amherst, Google Research on probabilistic treatment of hierarchical clustering.

- Data structures and algorithms that allow us to efficiently search through search spaces as large as 10^{300}
- Papers at accepted AISTATS, NeurIPS, & vCHEP. Submitted to UAI & snowmass
- Software implementations (python)
- Fellows project to implement in Julia

1. arXiv:2104.07061 [pdf, other] [cs.LG](#) [cs.DS](#) [physics.data-an](#) [stat.ML](#)
Exact and Approximate Hierarchical Clustering Using A*
Authors: Craig S. Greenberg, Sebastian Macaluso, Nicholas Monath, Avinava Dubey, Patrick Flaherty, Manzil Zaheer, Amr Ahmed, Kyle Cranmer, Andrew McCallum
Abstract: Hierarchical clustering is a critical task in numerous domains. Many approaches are based on heuristics and the properties of the resulting clusterings are studied post hoc. However, in several applications, there is a natural cost function that can be used to characterize the quality of the clustering. In those cases, hierarchical clustering can be seen as a combinatorial optimization problem. To... [More](#)
Submitted: 14 April, 2021; **originally announced:** April 2021.
Comments: 30 pages, 9 figures
2. arXiv:2011.08191 [pdf, other] [cs.AI](#) [cs.LG](#) [hep-ph](#)
Hierarchical clustering in particle physics through reinforcement learning
Authors: Johann Brehmer, Sebastian Macaluso, Duccio Pappadopulo, Kyle Cranmer
Abstract: Particle physics experiments often require the reconstruction of decay patterns through a hierarchical clustering of the observed final-state particles. We show that this task can be phrased as a Markov Decision Process and adapt reinforcement learning algorithms to solve it. In particular, we show that Monte-Carlo Tree Search guided by a neural policy can construct high-quality hierarchical clust... [More](#)
Submitted: 18 December, 2020; v1 submitted 16 November, 2020; **originally announced:** November 2020.
Comments: Accepted at the Machine Learning and the Physical Sciences workshop at NeurIPS 2020
3. arXiv:2002.11661 [pdf, other] [cs.DS](#) [cs.LG](#) [physics.data-an](#) [stat.ML](#)
Data Structures & Algorithms for Exact Inference in Hierarchical Clustering
Authors: Craig S. Greenberg, Sebastian Macaluso, Nicholas Monath, Ji-Ah Lee, Patrick Flaherty, Kyle Cranmer, Andrew McGregor, Andrew McCallum
Abstract: Hierarchical clustering is a fundamental task often used to discover meaningful structures in data, such as phylogenetic trees, taxonomies of concepts, subtypes of cancer, and cascades of particle decays in particle physics. Typically approximate algorithms are used for inference due to the combinatorial number of possible hierarchical clusterings. In contrast to existing methods, we present novel... [More](#)
Submitted: 22 October, 2020; v1 submitted 26 February, 2020; **originally announced:** February 2020.
Comments: 27 pages, 12 figures



ML4Jets: Vision for Y4/Y5 Impact

showmass:

Other grants that have mainly supported Sebastian end in August 2021. Request to extend to Dec. 21 (or longer) at 100%

Would like to use Y3 developments to demonstrate that it is possible to fit parameter of parton shower model summing over all possible jet clusterings weighted appropriately.

Develop “parameterized trellis” connecting to VAE like models for trees

Explore integration of algorithms with FastJet or other Jet clustering tools

Longer term (who?):

- Extend algorithms to 2->3 splitting to aid in ML/PS matching bottleneck for high jet multiplicity event generation
- demonstrate use of probabilistic programming to efficiently populate tails of phase space (eg. QCD faking a boosted jet tagger)

Emerging Computational Techniques for Jet Physics

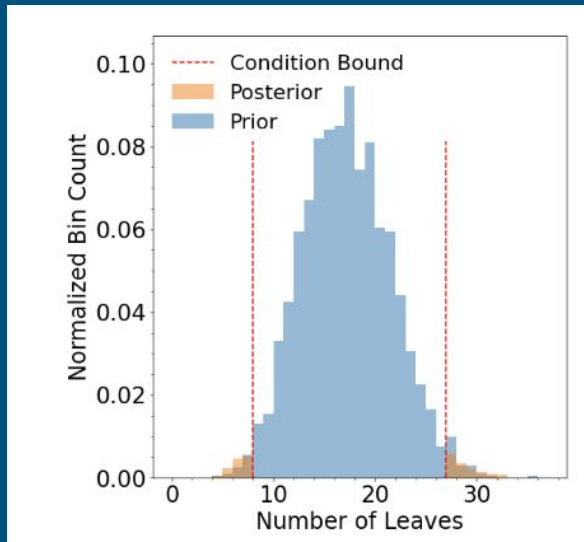
Sebastian Macaluso, Kyle Cranmer, Matthew Drnevich, Johann Brehmer (*New York University*);
Duccio Pappadopulo (*N.A.*); Atılım Güneş Baydin (*Oxford*); Matthew Schwartz (*Harvard*)

vCHEP:

Reframing Jet Physics with New Computational Methods

Kyle Cranmer¹, Matthew Drnevich¹, Sebastian Macaluso^{1,*}, and Duccio Pappadopulo

¹New York University



Proof of principle: using prob. prog. to populate tails

Milestones

Innovative Algorithms	Status	On Schedule	Due Date	Year	Completion Date
G4.1: Demonstrate that the Allen monitoring suite is deployable in the Allen multi-t...	Future	On-Time	◆ Sep 1	2021	
G4.2: Benchmark physics and technical performance of PVFinder with respect to ...	In Progress	On-Time	◆ Mar 1	2021	
G4.3: Demonstrate physics and technical performance of graph-based tracking pi...	Future	On-Time	◆ Sep 1	2021	
G4.4: Demonstrate physics and technical performance of calorimetric reconstructi...	Future	On-Time	◆ Sep 1	2021	
G4.5: Demonstrate the performance of sparse hierarchical trellis algorithm compa...	Done	On-Time	◆ Jun 1	2021	
G4.6: Demonstrate segment-linking approach to track finding using the CMS HL-L...	Future	On-Time	◆ Sep 1	2021	
G4.7: Adapt mkFit to perform track finding the CMS HL-LHC tracker geometry and...	Future	On-Time	◆ Sep 1	2021	
G4.8: Demonstrate the physics and technical performance of the ACTS reconstru...	Future	On-Time	◆ Sep 1	2021	
G4.9: At least two LHC experiments use Institute developed algorithms in their pro...	Future	On-Time	◆ Jun 1	2022	
G4.10: Institute developed tracking algorithms are used to produce HL-LHC simul...	Future	On-Time	◆ Mar 1	2023	
G4.11: Demonstrate physics and technical performance of at least three Institute t...	Future	On-Time	◆ Mar 1	2023	
		! N/A	◆ Mar 1 - 1		-

Many of our milestones are due during the next 6 months

Backup

LST - Algorithm Overview

- Load hits
- Make mini-doublets (MD)
- Make segments (LS)
 - Import pixels (pLS)
- Create tracklets
 - Triplets (T3)
 - Quadruplets (T4)
 - Quintuplets (T5)
- Create track candidates
 - pT3, pT4
 - T5
- [MTV-like Validation]

