New directions in computing

Advanced Graduate Lectures on
Practical Tools, Applications and Techniques in HEP
Part-II

Jun 15 – 19, 2026
RAL, Visitors Centre

Brij Kishor Jashal
Rutherford Appleton Laboratory, Oxford

brij@cern.ch

Overview:

Part-I

- LHC Computing and scale
- Landscape of research software projects .
- Advancements in platforms and architectures.
- Languages and software engineering.

Break: Questions and discussion

Part-II

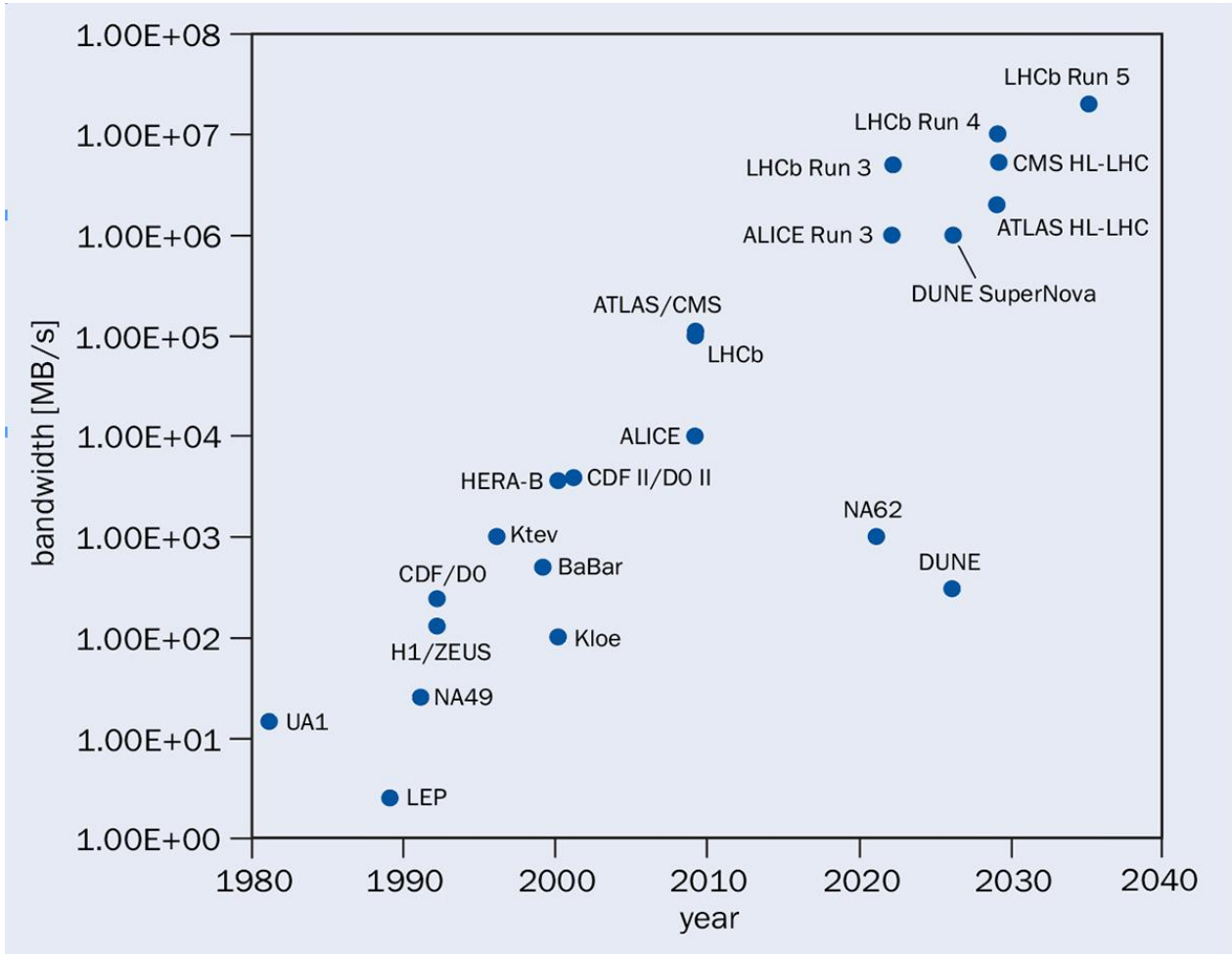
- Real-Time analysis (online)
- GPU programming.
- AI and future

End: Questions and discussion

Triggers and Real-Time (online) analysis

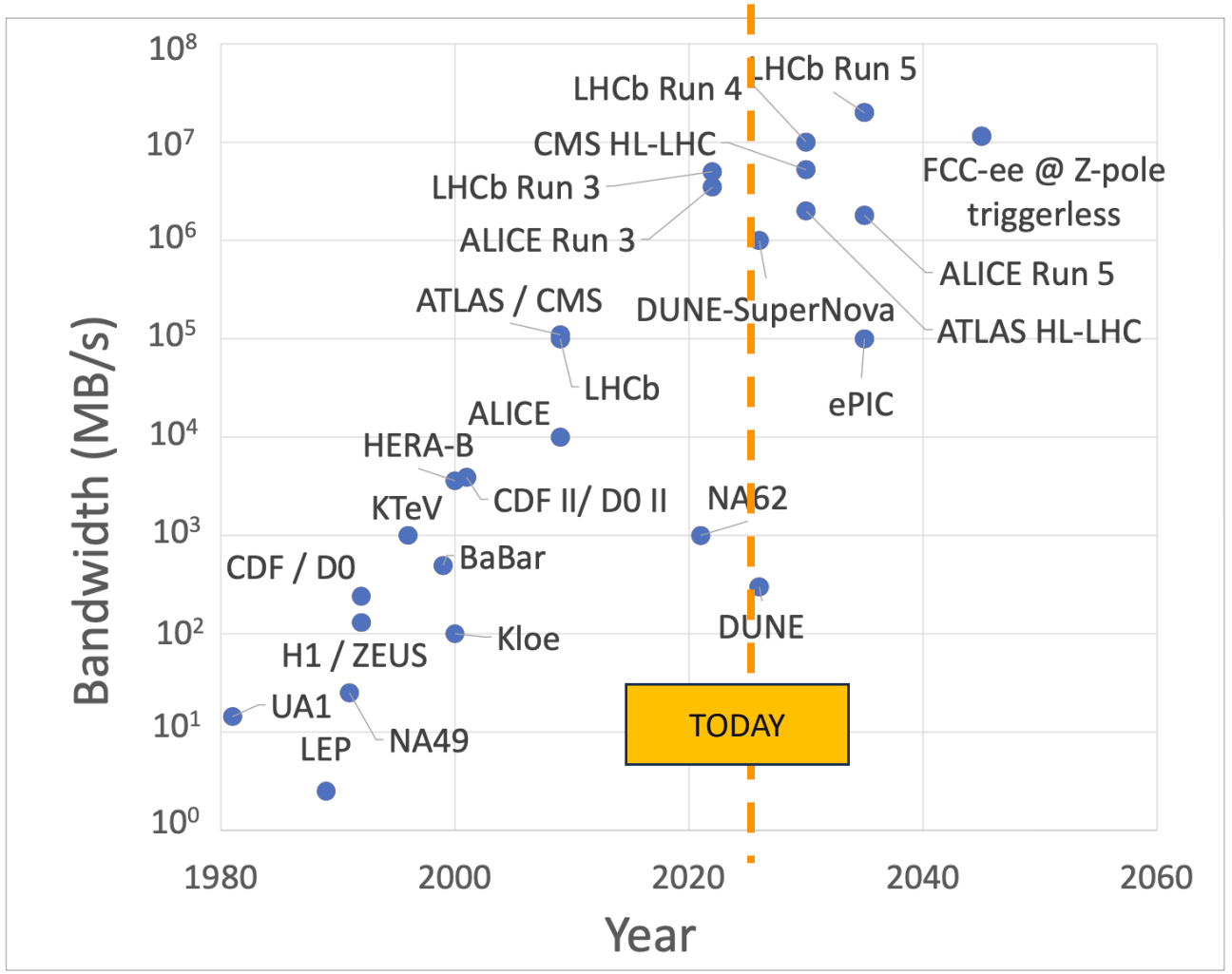
Real-Time Analysis (online)

Ref: CERN Courier Mar 2023



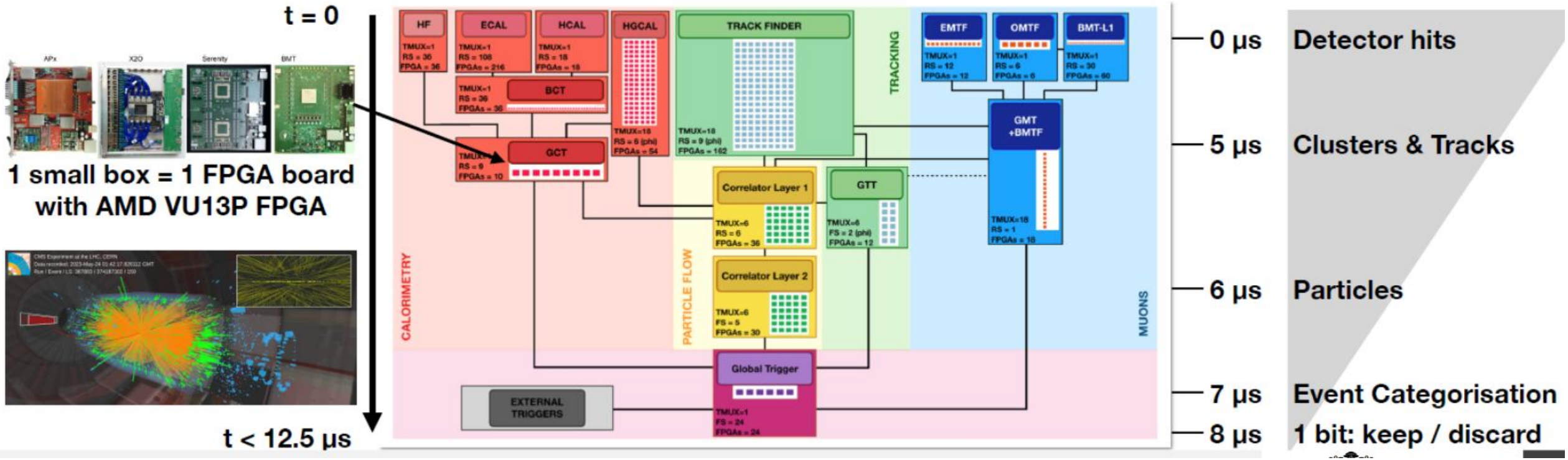
Real-Time Analysis (online)

Ref: CERN Courier Mar 2023



Real-Time Analysis (online): CMS Trigger Run3

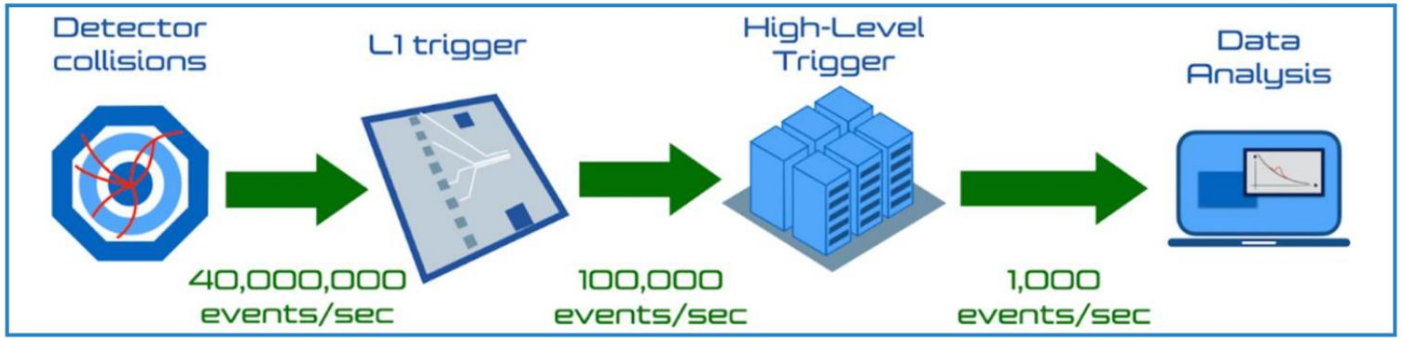
CHEP24



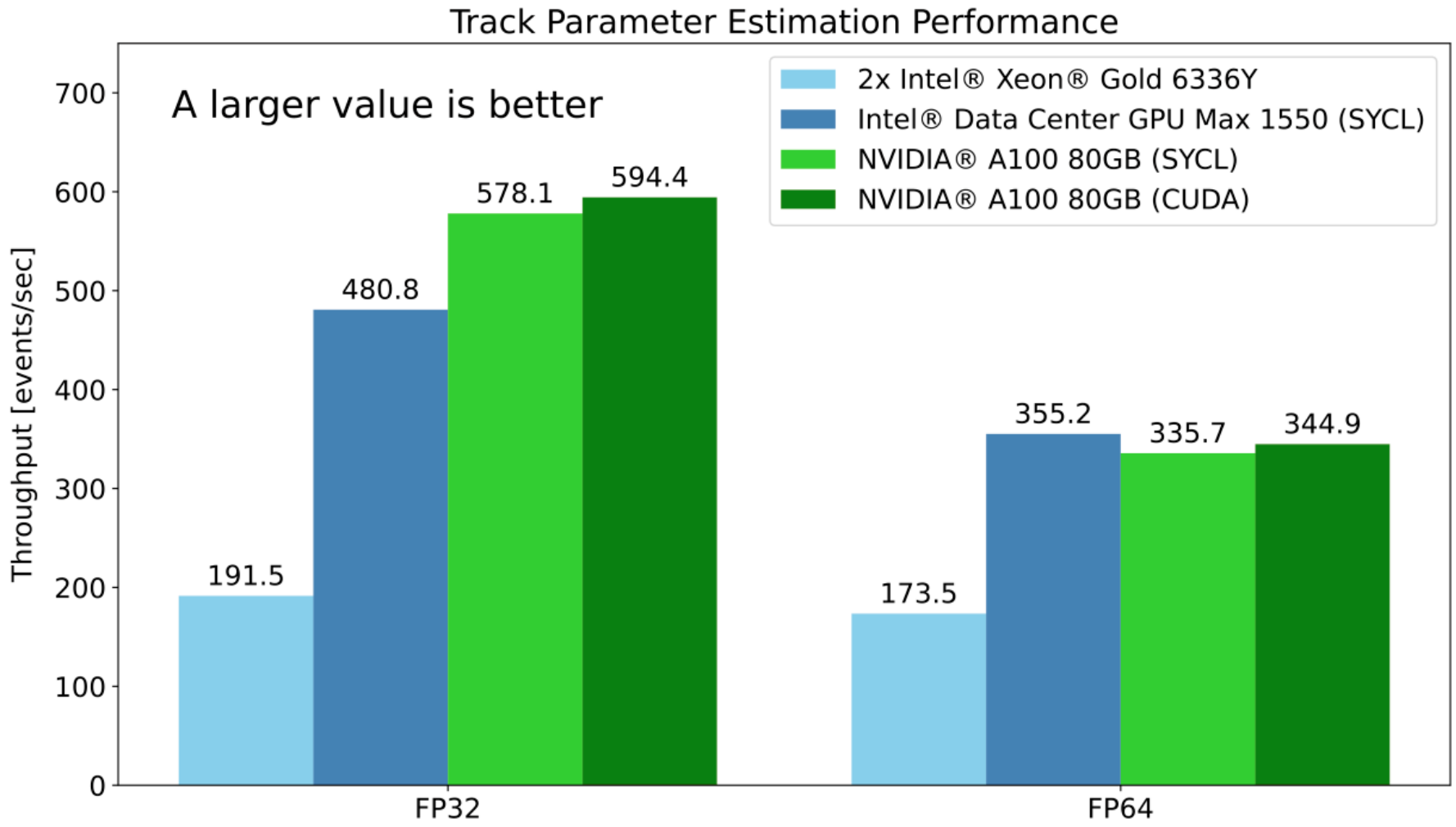
Each small box is one Xilinx Ultrascale+ FPGA

These have NNs and/or BDTs inside

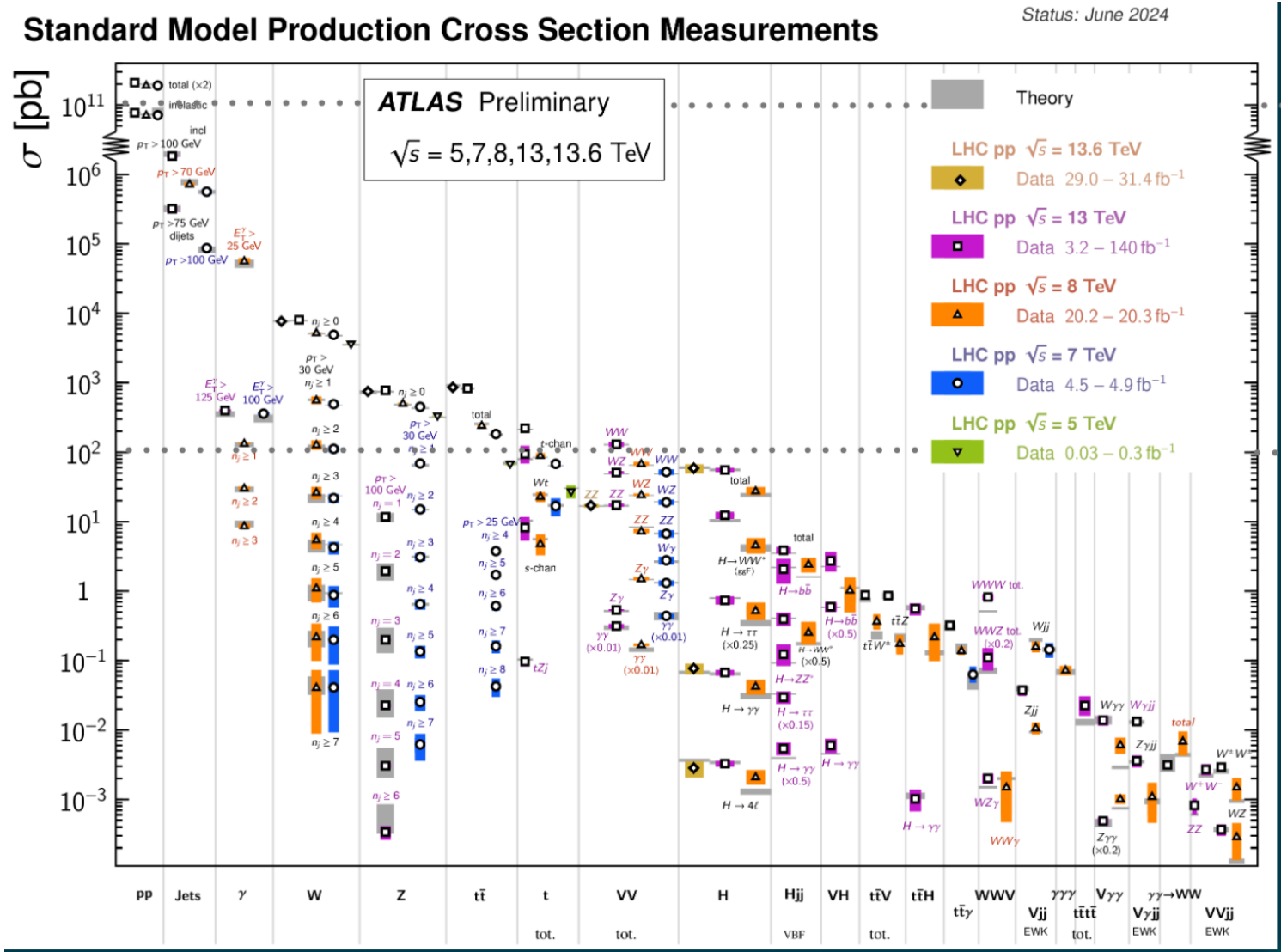
hls4ml Conifer



Real-Time Analysis (online): ATLAS ACTS



Real-Time Analysis (online): ATLAS



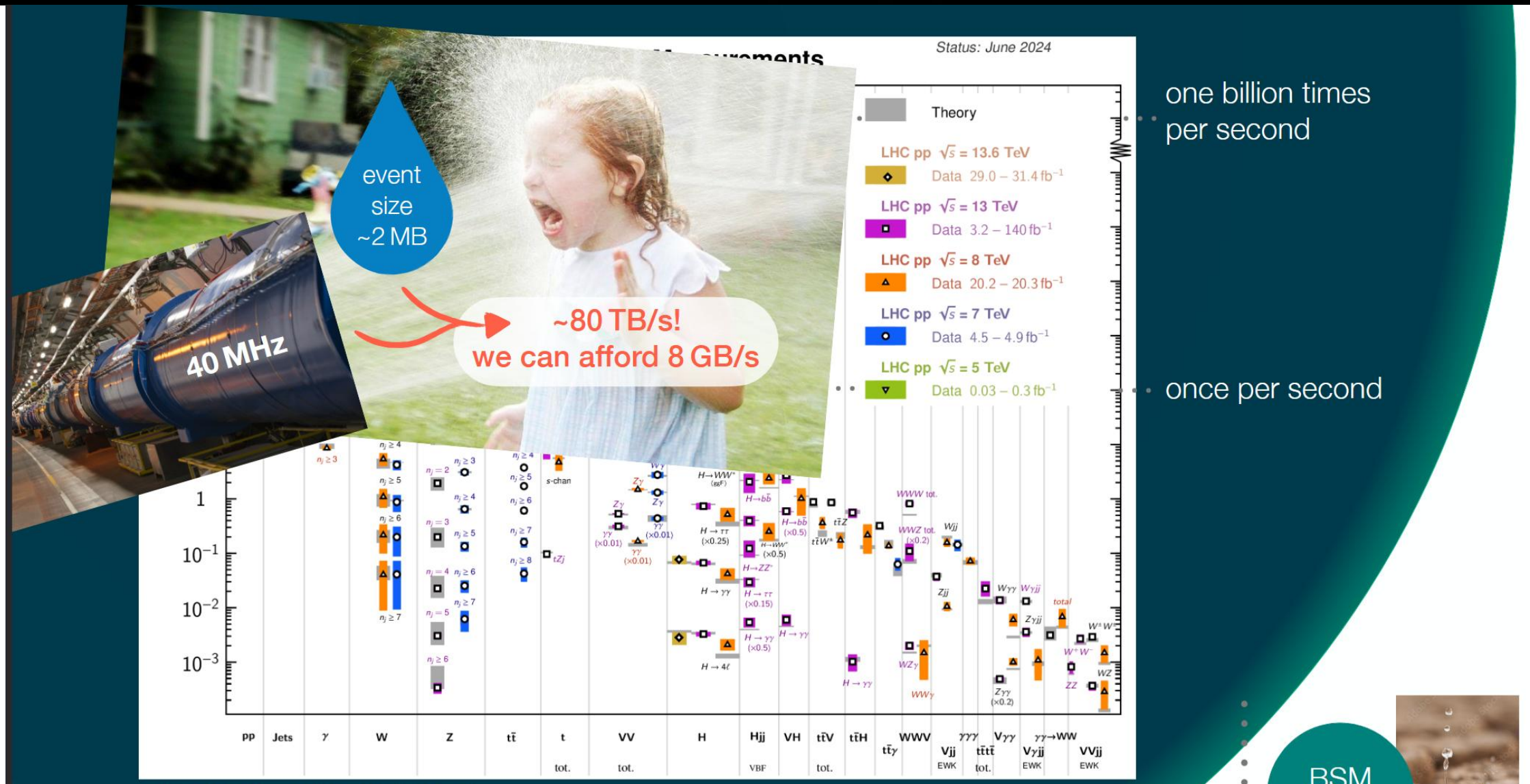
AtlasPublic/StandardModelPublicResults

one billion times per second

once per second

BSM Physics?

Real-Time Analysis (online): ATLAS

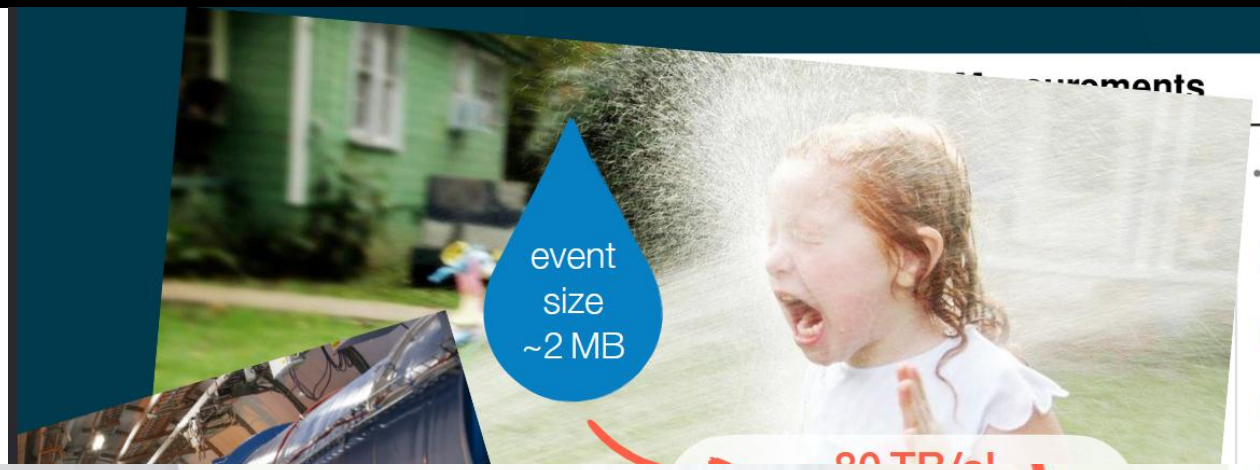


one billion times per second

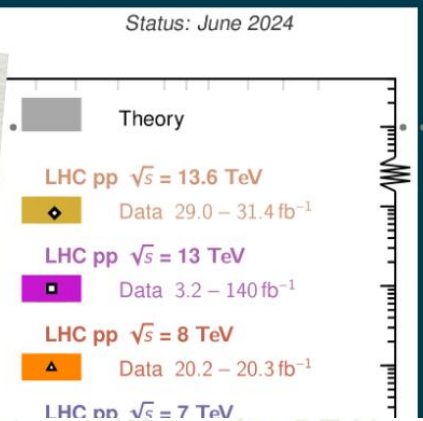
once per second

BSM Physics?

Real-Time Analysis (online): ATLAS



event size ~2 MB

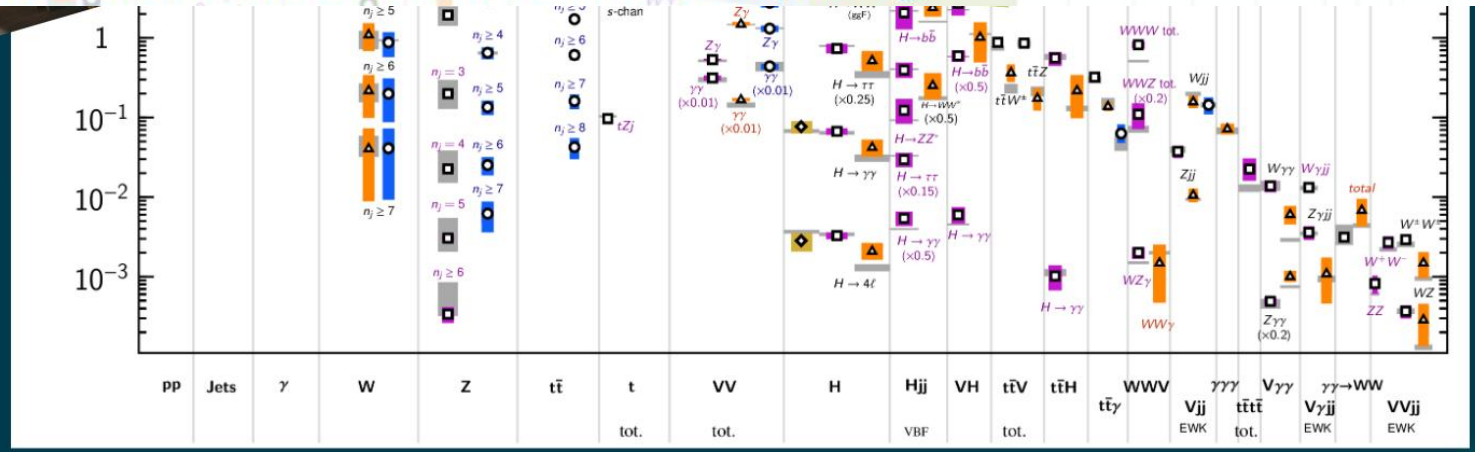


one billion times per second

rare and interesting physics

We can't deal with all of that and we want more of this

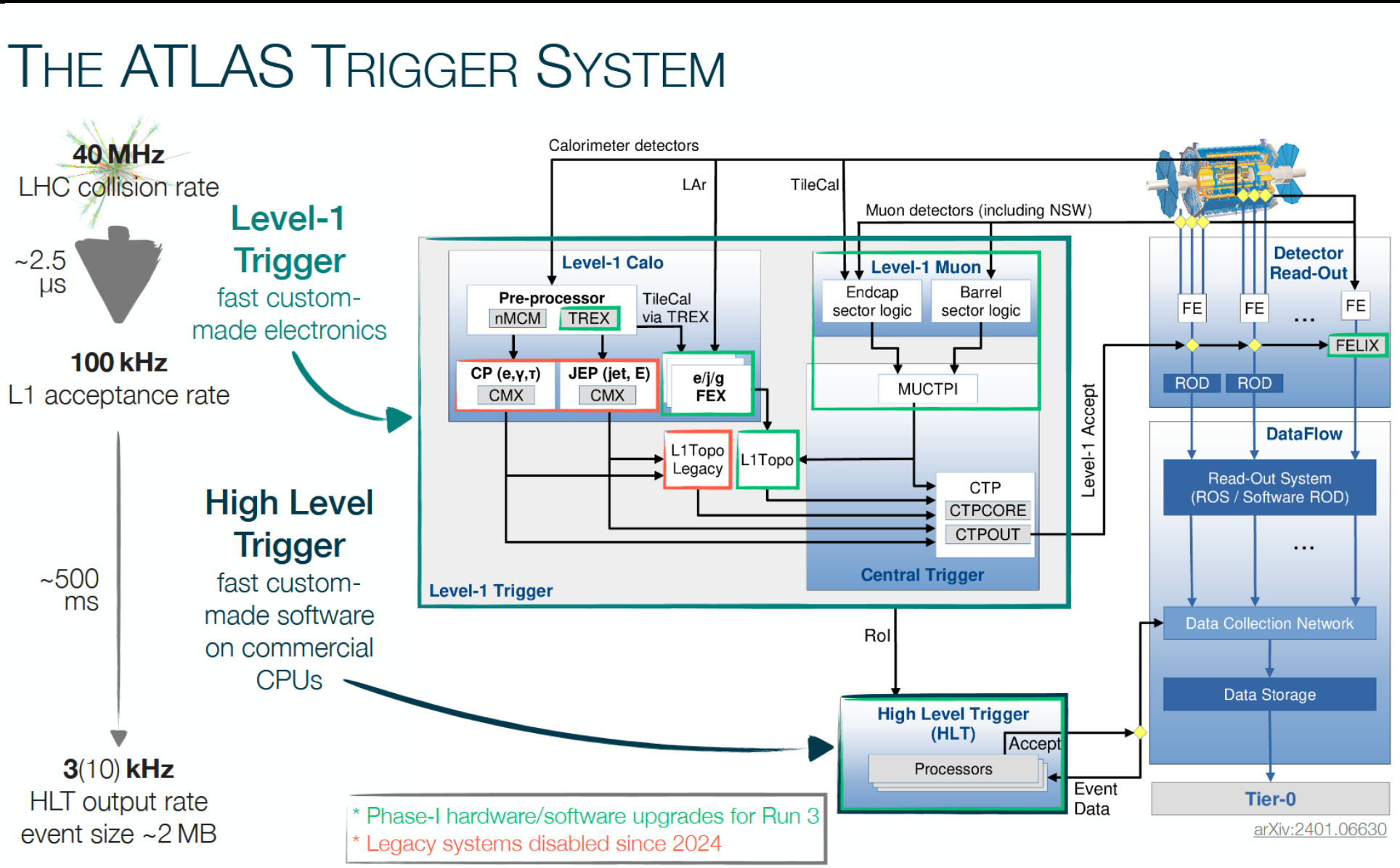
too much for detector read-out and offline computing



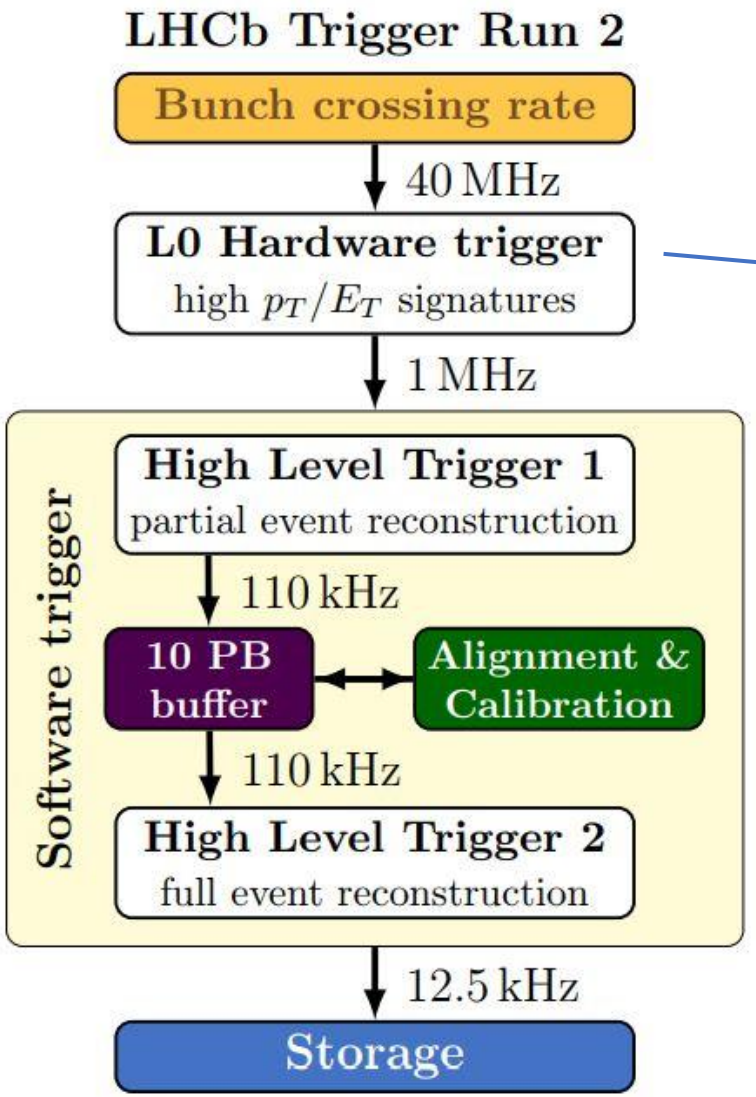
AtlasPublic/StandardModelPublicResults

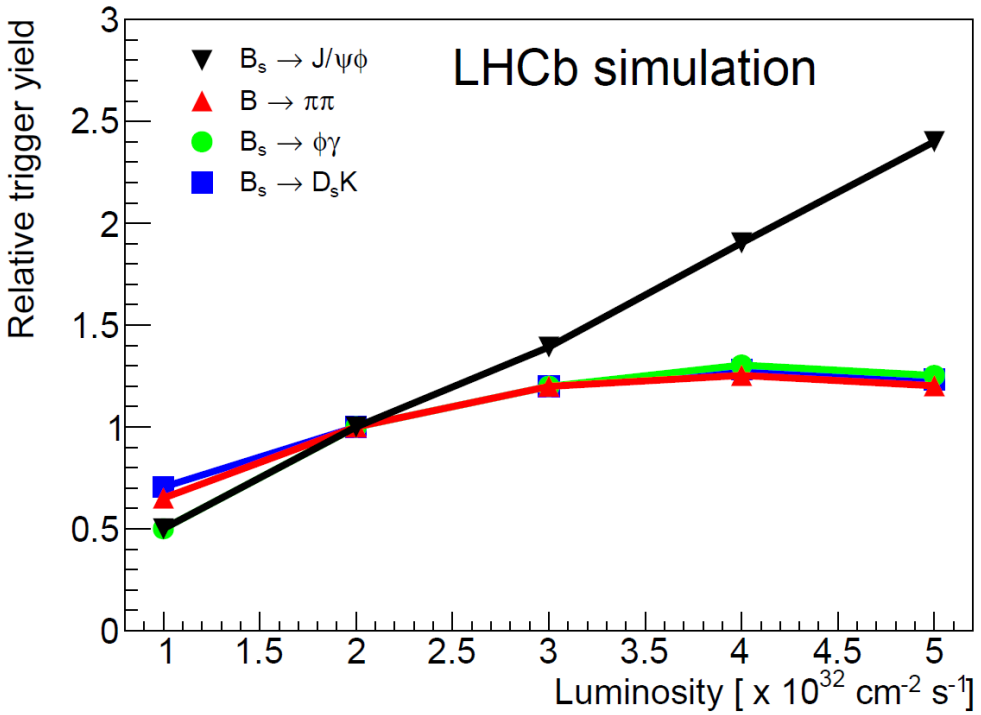
BSM Physics?

Real-Time Analysis (online): ATLAS



Real-Time Analysis (online): LHCb





- At high luminosities, hardware trigger cannot cope with the event rate and starts rejecting interesting events.
- Need to study properties of events in real time of collision.

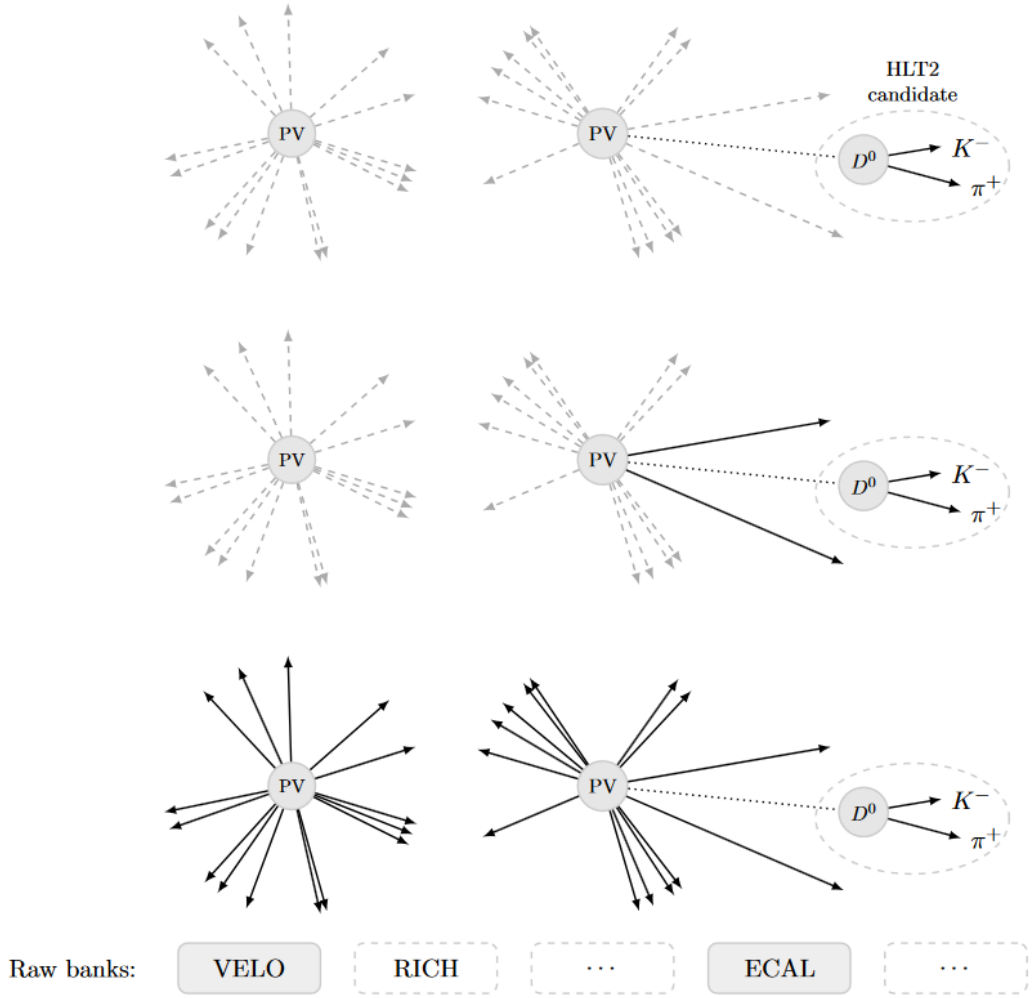
Real-Time Analysis (online): LHCb

What to keep and what to discard ?

we have to be also very careful with what we dismiss..

Real-Time Analysis (online): LHCb

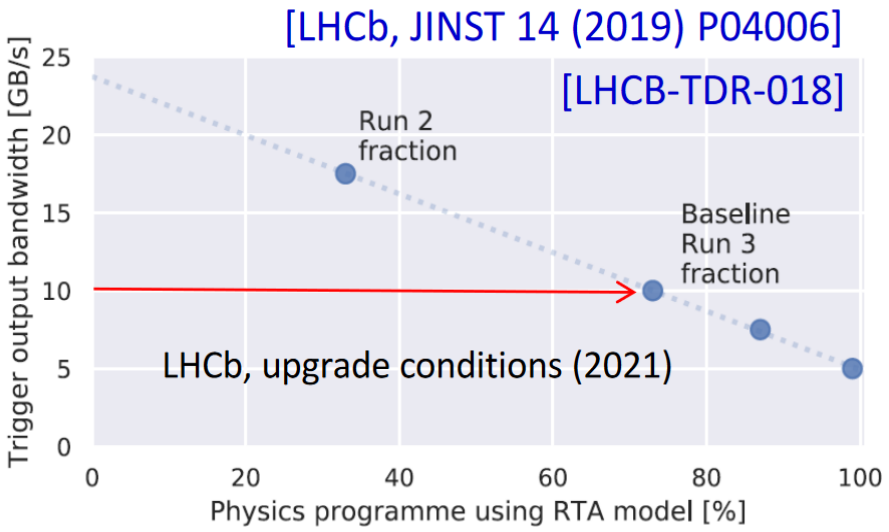
Bandwidth [GB/s] ~ Trigger output rate [kHz] x Average event size [MB]



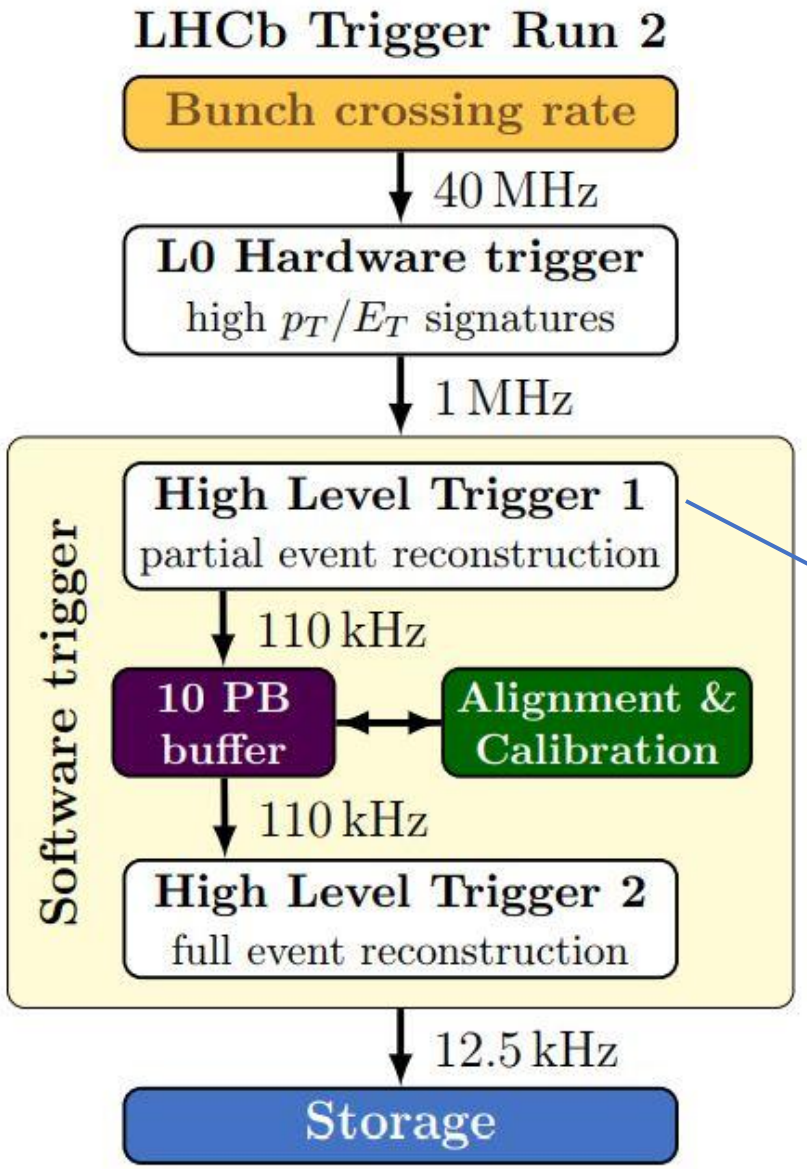
- Need to **reduce the event size**

Instead of raw data from the detector, store only the relevant information of interesting events.

- Need to reconstruct and **analyse** the events to select them in **real time.**



Real-Time Analysis (online): LHCb



LHCb Run2 HLT1 trigger configuration

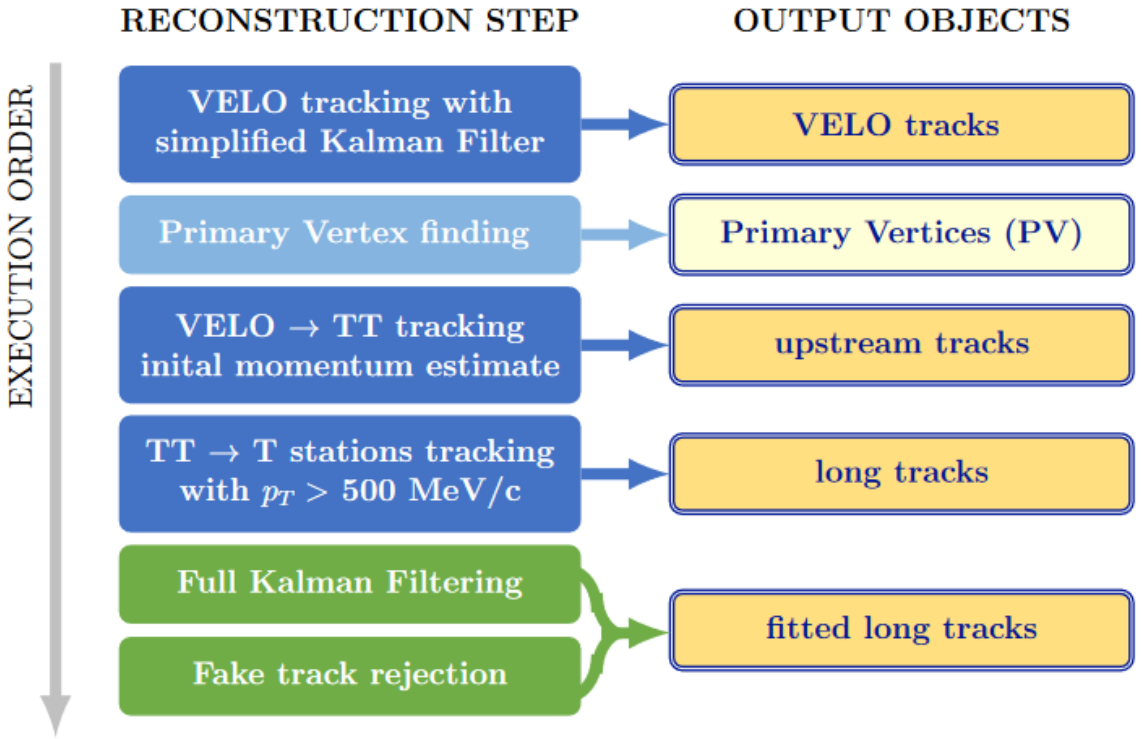
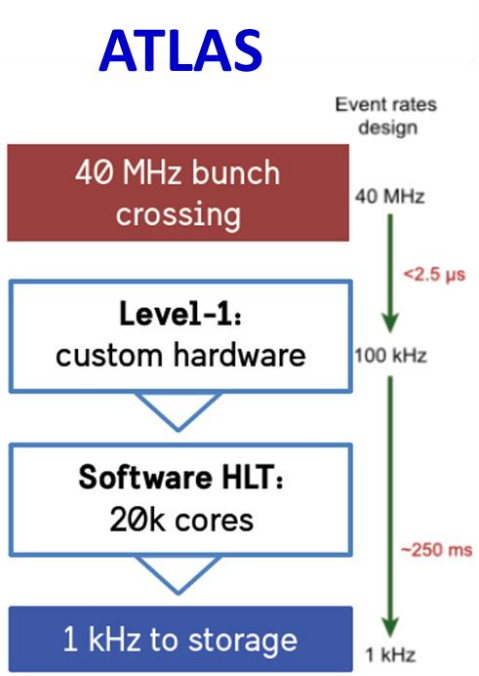
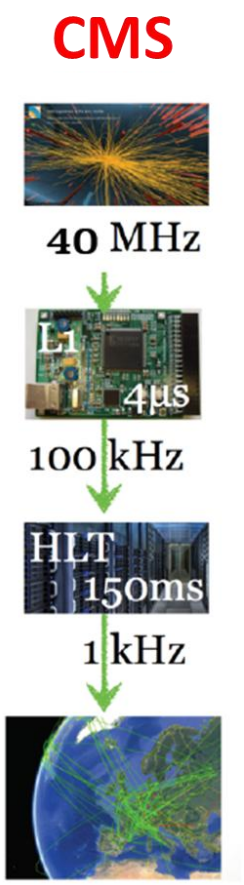


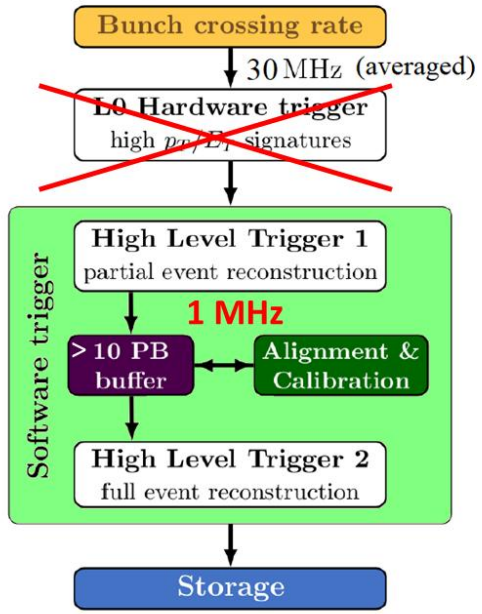
Figure 4: Sketch of the HLT1 track and vertex reconstruction.

Real-Time Analysis (online): LHCb

The trigger systems:







- At the end of Run2 (2018) HLT2 was processing ~68 events /s on a single node.
  - i.e in order to process 30 million events / s , we would have needed about 4,41,176 such nodes (CPU)
    ~ $ 80 M
  - Total budget of LHCb Upgrade-I for Run3 (including everything)
    ~ $ 58 M

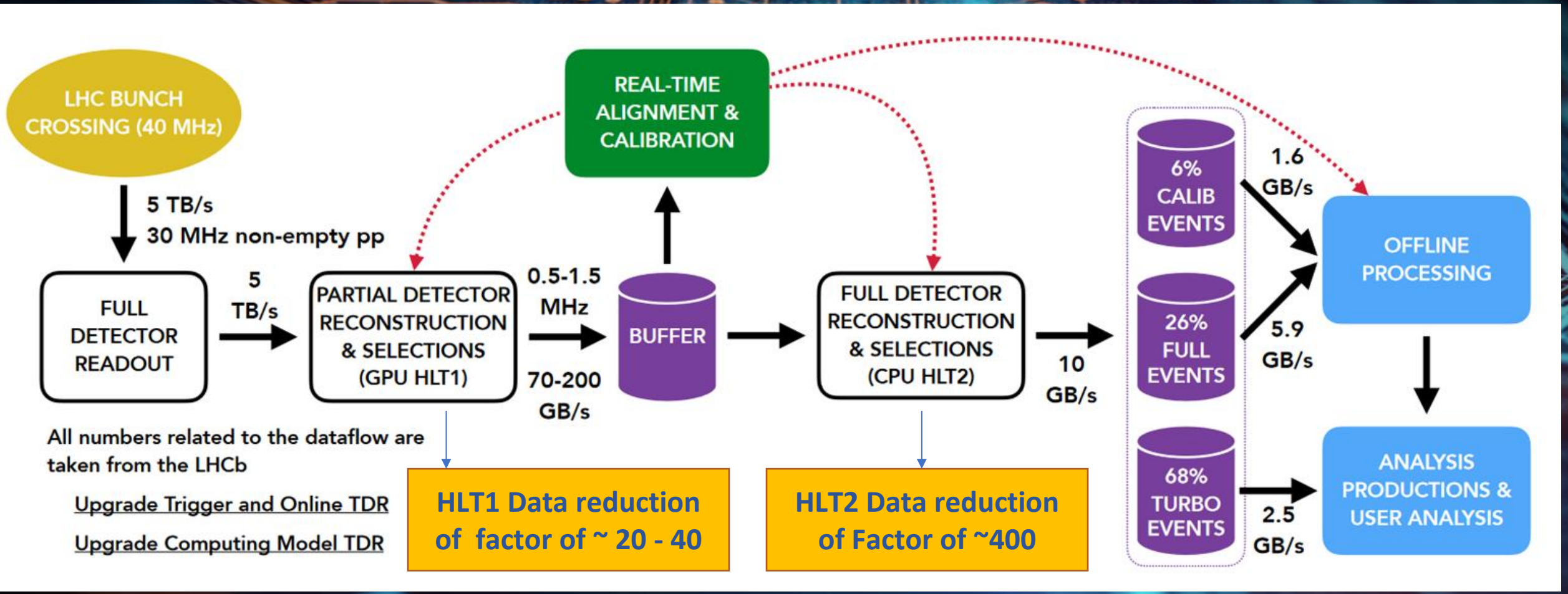
Bandwidth [GB/s] ~ Trigger output rate [kHz] x Average event size [MB]

~ 1 GB/s

1 kHz (ATLAS & CMS)
12.5 kHz (LHCb)

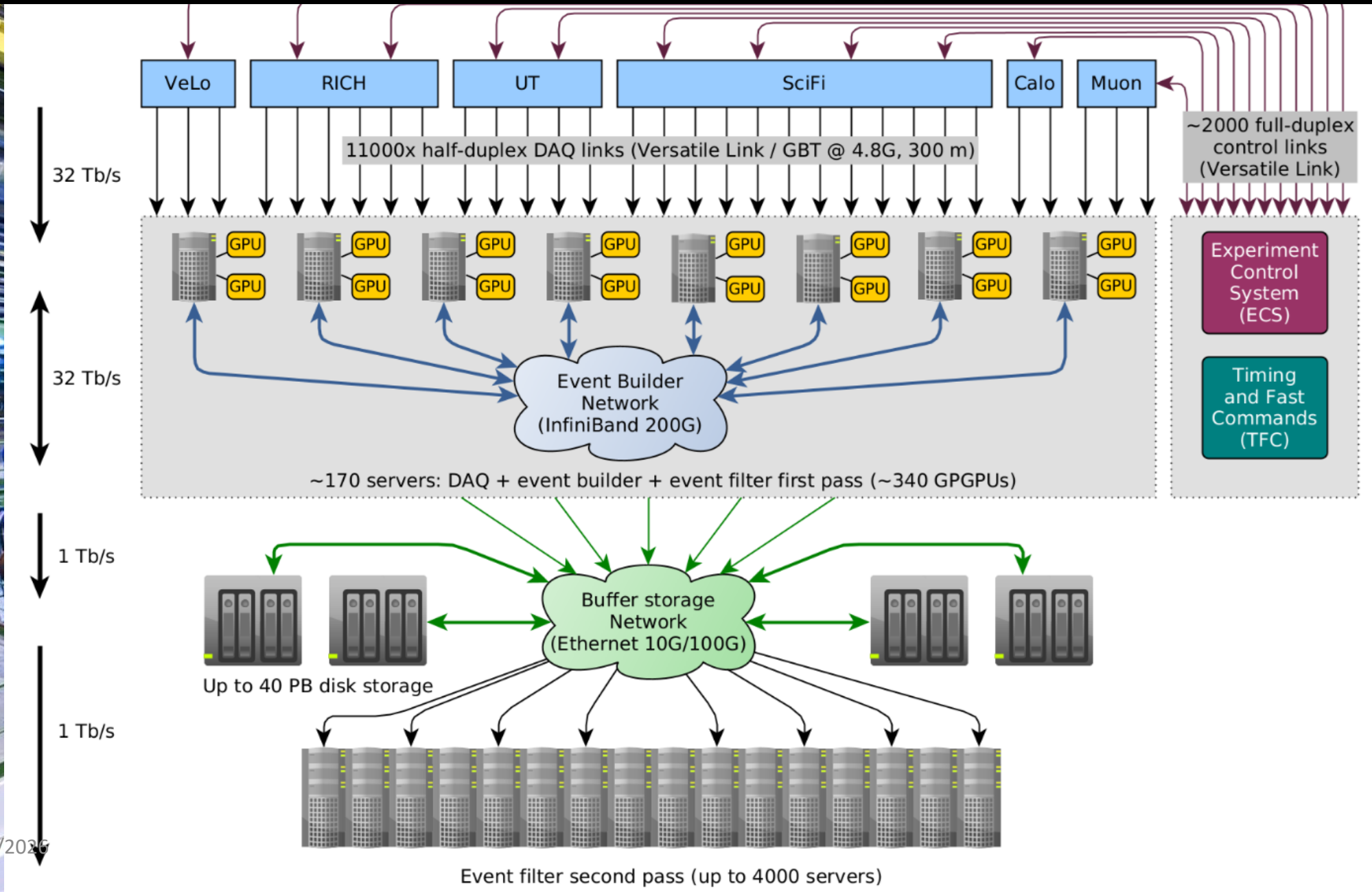
Raw event data size
~1 MB (ATLAS and CMS)
~0.1 MB (LHCb)

Real-Time Analysis (online): LHCb



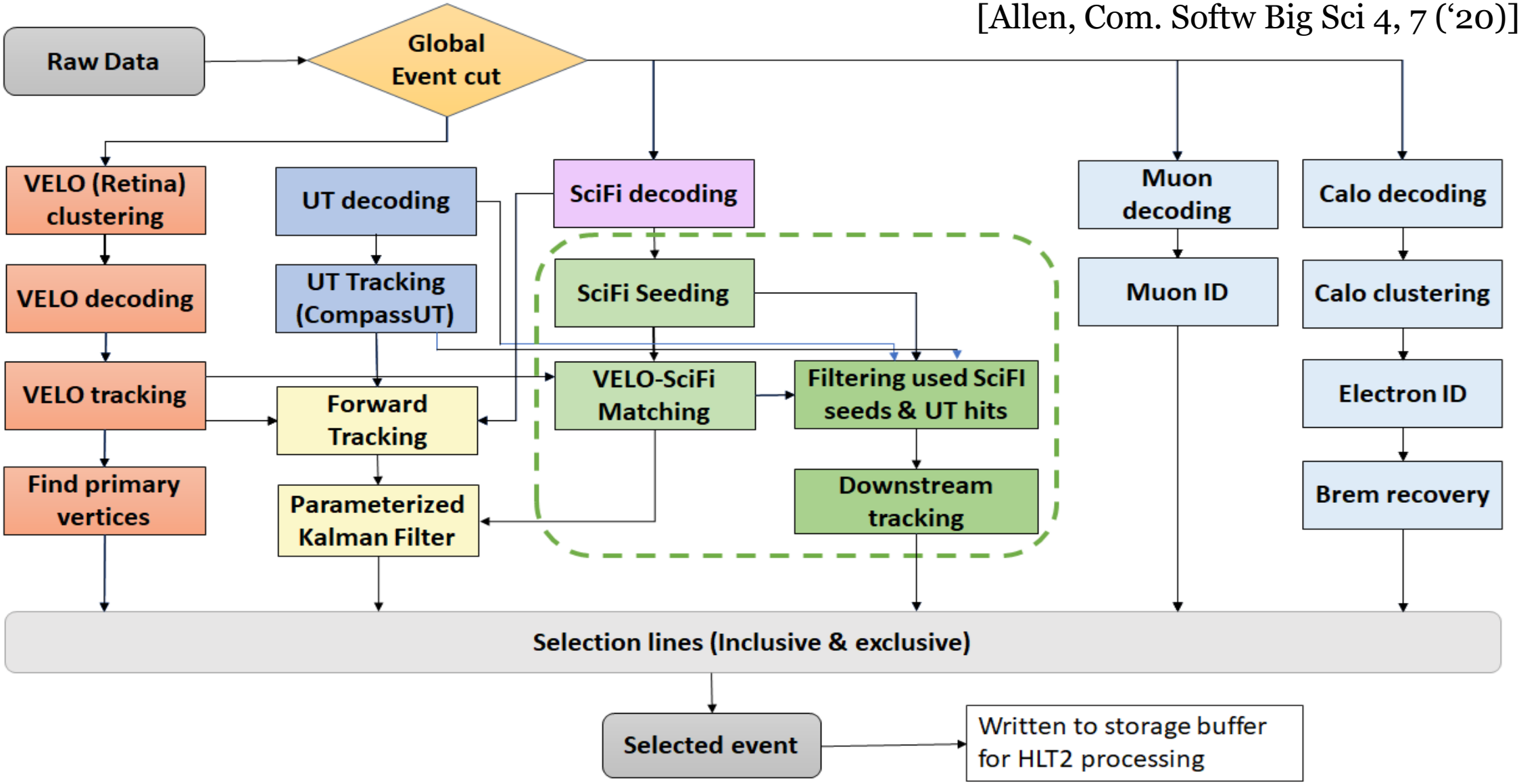
HLT1: Partial reconstruction of charged particle trajectories and few simple selection lines
HLT2: Full reconstruction and selection based on different decay chains and signatures
Must reduce without the loss in fidelity

Real-Time Analysis (online): LHCb

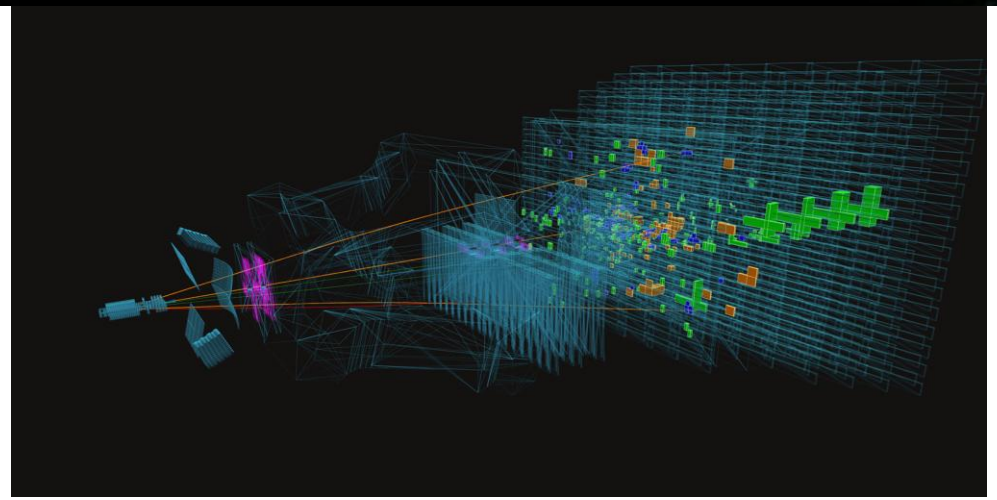


Real-Time Analysis (online): LHCb algorithms: HLT1 sequence (Run-3)

[Allen, Com. Softw Big Sci 4, 7 ('20)]



Real-Time Analysis (online): Tracking



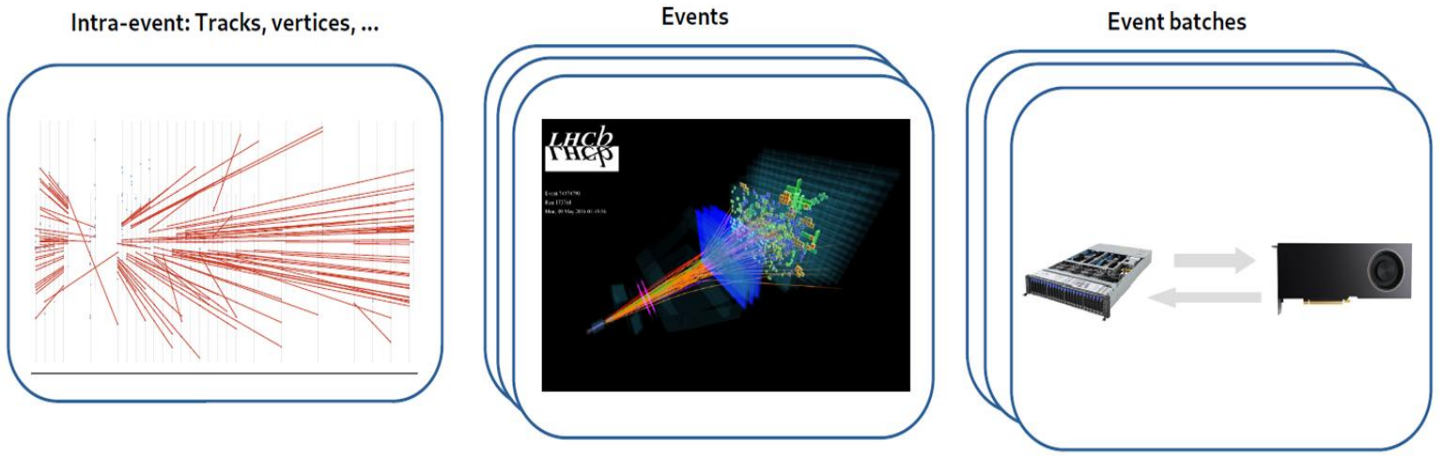
Track reconstruction (Tracking)

What

- Tracking deals with converting the signal from a subdetector (hits, clusters…) into a trajectory.
- Roughly speaking, two phases: pattern recognition and track fitting

Why

- We need to reconstruct trajectories of particles in our detector to:
  - Build vertices, measure decay topologies;
  - Measure momenta → measure invariant masses, angular variables (so… do physics).



Real-Time Analysis (online): Tracking

Computing and Software for Big Science (2020) 4:7
https://doi.org/10.1007/s41781-020-00039-7

ORIGINAL ARTICLE

Allen: A High-Level Trigger on GPUs

R. Aaij[1] · J. Albrecht[2] · M. Belous[3,4] · P. Billoir[5] · T. Boett
D. H. Cámpora Pérez[1,8] · A. Casais Vidal[7] · D. C. Craik[6]
B. Jashal[11] · N. Kazeev[3,4] · D. Martínez Santos[7] · F. Pisa
M. Rangel[16] · F. Reiss[5] · C. Sánchez Mayordomo[11] · R.
X. Vilasís Cardona[18] · M. Williams[6]

Received: 18 December 2019 / Accepted: 3 April 2020 / Published onl

Abstract
We describe a fully GPU-based implementation of the fir
taking in 2021. We demonstrate that our implementatio
LHCb detector and perform a wide variety of pattern r
particles, finding proton–proton collision points, identif
vertices of long-lived particles. We further demonstrate
GPU cards, that it is not I/O bound, and can be operated
high-throughput GPU trigger proposed for a HEP exper

Keywords GPU · Real-time data selection · Trigger · LH

A Downstream and vertexing algorithm for Long Lived Particles (LLP) selection at the first High level trigger (HLT1) of LHCb

V. Kholoimov[1], B. Kishor Jashal[1,2], A. Oyanguren[1], V. Svintozelskyi[1] and J. Zhuo[1]

[1]Instituto de Física Corpuscular (IFIC), University of Valencia- CSIC, Valencia, Spain.
[2]Rutherford Appleton Laboratory (RAL), Oxford, United Kingdom.

Abstract
A new algorithm has been developed at LHCb which is able to r
placed vertices in real time at the first level of the trigger (HLT1)
Tracker (UT) and the Scintillator Fiber detector (SciFi) of LHC
inside the Allen framework. In addition to an optimized strateg
(NN) implementation to increase the track efficiency and reduce t
throughput and limited time budget. Besides serving to reconstr
the Standard Model, the Downstream algorithm and the associa
largely increase the LHCb physics potential for detecting long-li

Keywords: LHCb, HLT1, GPUs, downstream, LLPs

1 Introduction

The LHCb forward spectrometer is one of the main detectors at the Large Hadron Collider (LHC) accelerator at CERN, with the primary purpose of searching for new physics through studies of CP-violation and heavy-flavour hadron decays. It has been operating during its Run 1 (2011-2012) and Run 2 (2015-2018) periods with very high performance, recording an integrated luminosity of 9 fb^{-1} at center-of-mass energies of 7, 8, and 13 TeV and delivering a plethora of accurate physics results and new particles discoveries. One of the main issues concerning the present Run 3 was that, even if many physics results are statistically limited,

new full-softw
egy was necess
Level Trigger
has been very
lect 10 fb^{-1} at
the computing
runs the first s
partial particl
large hit occup
by two orders
Xeon E5-2630
ing the reconst
algorithm call
in this work, v
scheme based

frontiers | Frontiers in Big Data

TYPE Original Research
PUBLISHED 07 November 2022
DOI 10.3389/fdata.2022.1008737

10.3389/fdata.2022.1008737

Check for updates

OPEN ACCESS

EDITED BY
Michael Spannowsky,
Durham University, United Kingdom

REVIEWED BY
Letizia Lusito,
Inside Climate Service, Italy
Alexander Kish,
Fermi National Accelerator Laboratory (DOE), United States

*CORRESPONDENCE
Arantza Oyanguren
arantza.oyanguren@ific.uv.es

SPECIALTY SECTION
This article was submitted to Big Data and AI in High Energy Physics, a section of the journal Frontiers in Big Data

RECEIVED 01 August 2022
ACCEPTED 03 October 2022
PUBLISHED 07 November 2022

CITATION
Calefice L, Hennequin A, Henry L, Jashal B, Mendoza D, Oyanguren A, Sanderswood I, Sierra CV, Zhuo J and part of LHCb-RTA Collaboration (2022) Effect of the high-level trigger for detecting long-lived particles at LHCb. *Front. Big Data* 5:1008737. doi: 10.3389/fdata.2022.1008737

Effect of the high-level trigger for detecting long-lived particles at LHCb

Lukas Calefice[1,2], Arthur Hennequin[3], Louis Henry[4], Brij Jashal[4], Diego Mendoza[5], Arantza Oyanguren[5*], Izaac Sanderswood[5], Carlos Vázquez Sierra[4], Jiahui Zhuo[5] and part of LHCb-RTA Collaboration

[1]Laboratoire de Physique Nucléaire et de Hautes Energies, Sorbonne Université, CNRS/IN2P3, Paris, France, [2]Fakultät Physik, Technical University Dortmund, Dortmund, Germany, [3]Laboratory for Nuclear Science, Massachusetts Institute of Technology, Cambridge, MA, United States, [4]European Organization for Nuclear Research (CERN), Geneva, Switzerland, [5]Instituto de Física Corpuscular (IFIC), Consejo Superior de Investigaciones Cientificas, University of Valencia, Valencia, Spain

Long-lived particles (LLPs) show up in many extensions of the Standard Model, but they are challenging to search for with current detectors, due to their very displaced vertices. This study evaluated the ability of the trigger algorithms used in the Large Hadron Collider beauty (LHCb) experiment to detect long-lived particles and attempted to adapt them to enhance the sensitivity of this experiment to undiscovered long-lived particles. A model with a Higgs portal to a dark sector is tested, and the sensitivity reach is discussed. In the LHCb tracking system, the farthest tracking station from the collision point is the scintillating fiber tracker, the SciFi detector. One of the challenges in the track reconstruction is to deal with the large amount of and combinatorics of hits in the LHCb detector. A dedicated algorithm has been developed to cope with the large data output. When fully implemented, this algorithm would greatly increase the available statistics for any long-lived particle search in the forward region and would additionally improve the sensitivity of analyses dealing with Standard Model particles of large lifetime, such as K^0_S or Λ^0 hadrons.

CERN-THESIS-2023-249

VNIVERSITAT DE VALÈNCIA

Facultat de Física
Departament de Física Atòmica, Molecular i Nuclear
Institut de Física Corpuscular (UV-CSIC)

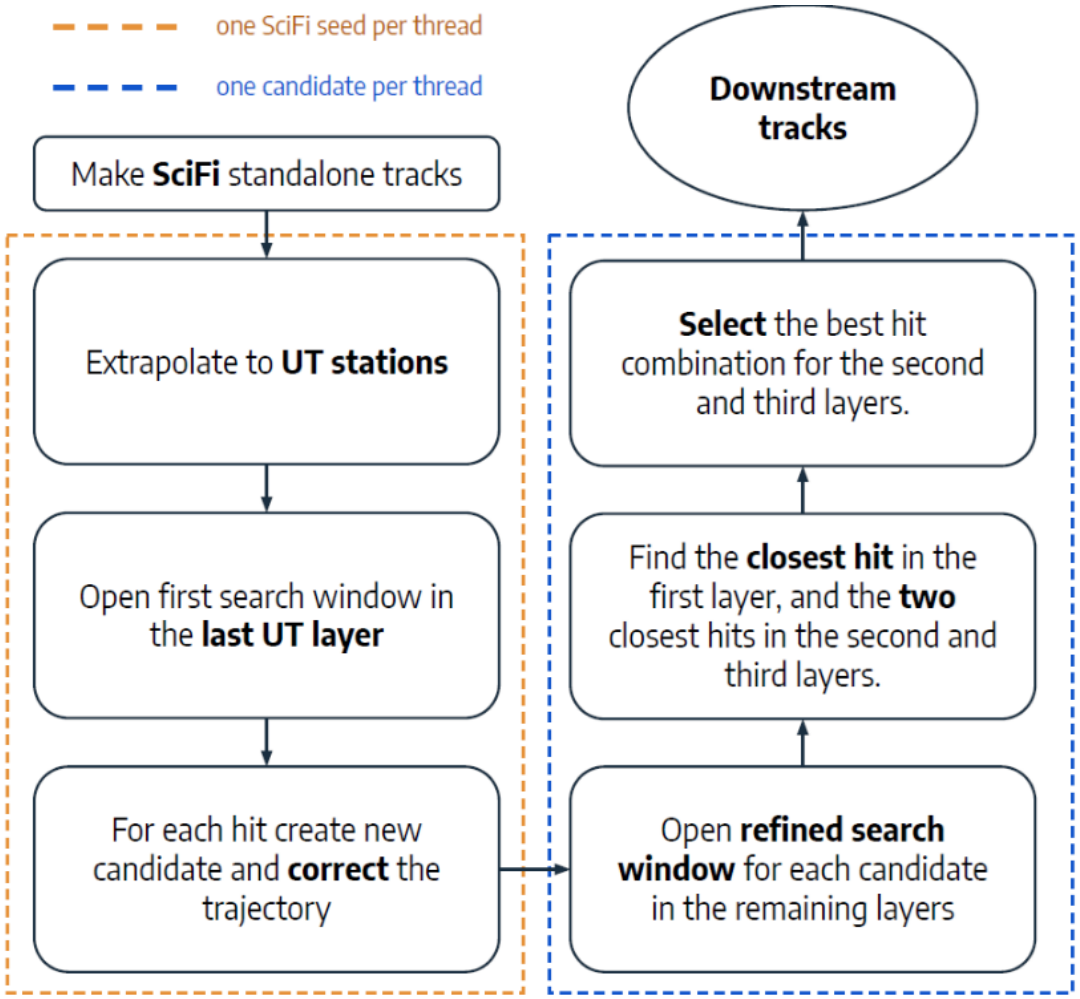
ng new discoveries:
nt of advanced HLT1
r detection of long-lived
ticles at LHCb

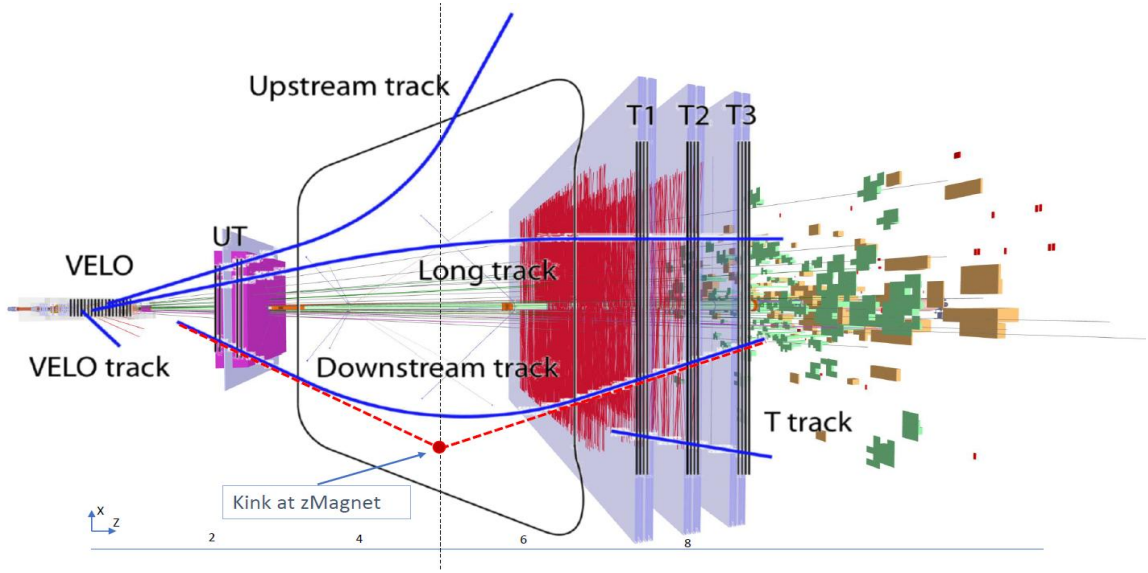
j Kishor Jashal

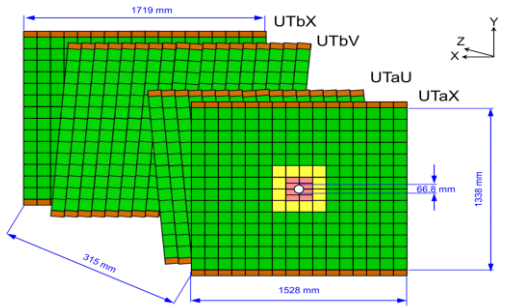
Directed by:
zazu de Oyanguren Campos

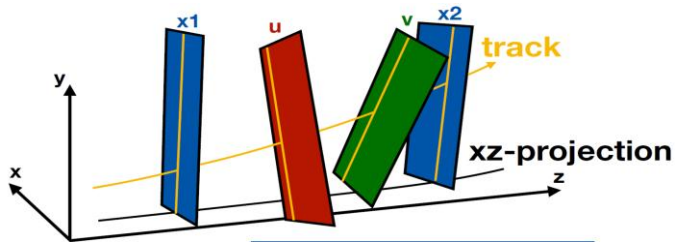
PhD thesis
octorado en Física
Valencia, España
November, 2023

Real-Time Analysis (online): Tracking









Real-Time Analysis (online): Tracking

Algorithm is divided into 3 main kernel functions:

Kernel function 1:
128 SciFi seeds per thread block

- Filtering used SciFi seeds
- For each input SciFi seed, extrapolate to last x layer (UTbX)
- Store up to 10 best candidates
- Update slope of each candidate using magnet point and hit position.

Kernel function 2:
256 candidates per thread block

- Add hits from rest of the UT layers
- Find best combination of U/V hits
- Compute the scores based on distance b/w extrapolation and real UT hit positions

Kernel function 3:
256 candidates per thread block

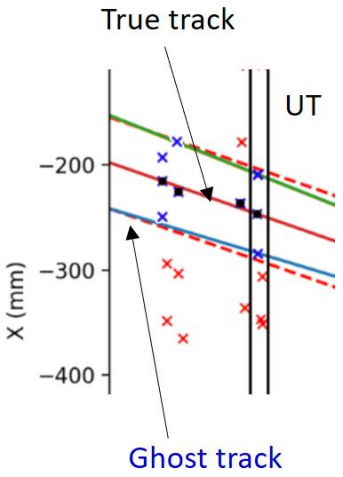
- Find best candidate based on the scores from previous function
- Check for hit duplication
- Perform ghost killing

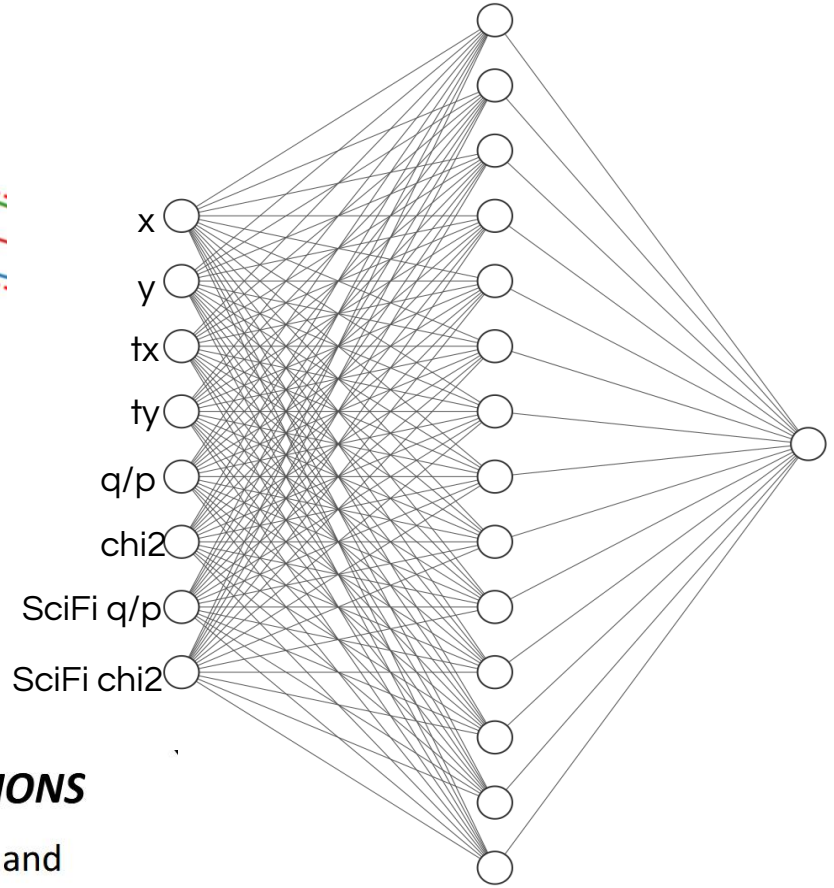
Prepare output:

- Copy hits and tracks to output (compact SOA container)
- Create standard multi-event viewer

Real-Time Analysis (online): Tracking

- A single hidden (14 nodes) layer fully connected NN
- It utilizes **8 variables** as input:
  - *Downstream* track state ($\mathbf{x,y,t_x,t_y,q/p,}\ \boldsymbol{\chi^2}$)
  - SciFi track properties ($\mathbf{q/p,}\ \boldsymbol{\chi_y^2}$)
- The model was trained using $B_s \to \phi\phi$ events.
- In order to boost speed, certain C++/CUDA tricks are applied, such as using **static structs**, employing **fast math functions**, and **unwinding for-loops**.





STATIC STRUCTURES

Fixing the number of input variables and the number of neurons (compile-time optimizations, registers vs global memory).

```
namespace DownstreamGhostKiller {
  namespace Model {
    constexpr unsigned num_node = 14;
    constexpr unsigned num_input = 8;
```

LOOP UNROLLING

Expanding for-loops using the NVCC-specific #pragma unroll directive (replicates the loop body multiple times).

```
// First layer
DownstreamHelpers::unwind<0, Model::num_node>([&](int i) {
  DownstreamHelpers::unwind<0, Model::num_input>([&](int j) {
    h1[i] += input[j] * Model::weights1[i][j];
  });
  h1[i] = ActivateFunction::relu(h1[i] + Model::bias1[i]);
});
```

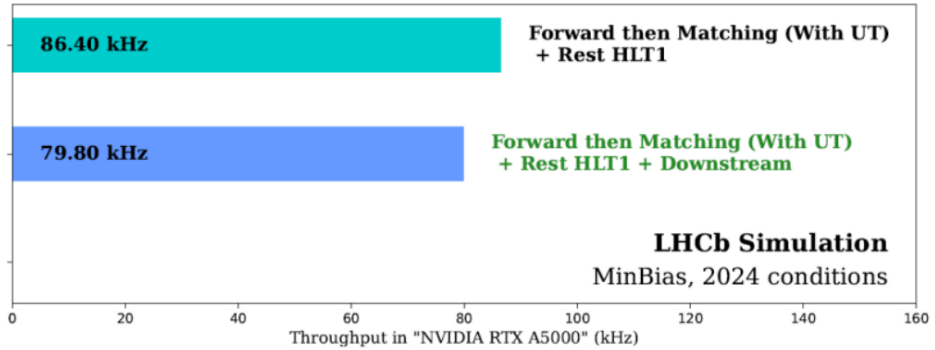
FAST MATH FUNCTIONS

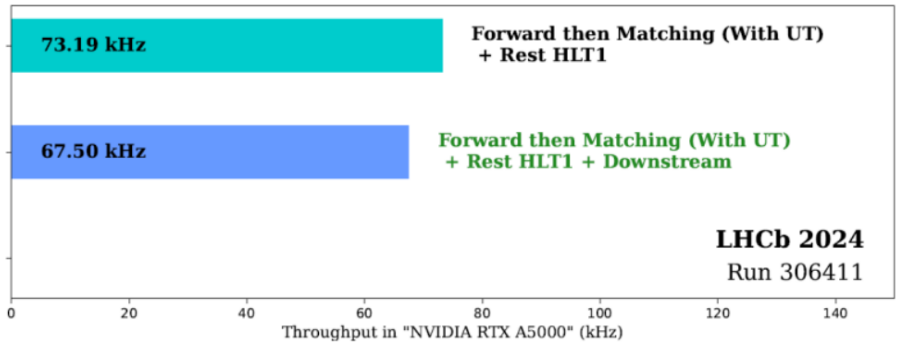
Such as `_fdividef` and `_expf`, to accelerate floating-point operations.

```
...
namespace ActivateFunction {
  // rectified linear unit
  __device__ inline float relu(const float x) {
    return x > 0 ? x : 0;
  }
  // sigmoid
  __device__ inline float sigmoid(const float x) {
    return __fdividef(1.0f, 1.0f + __expf(-x));
  }
} // namespace ActivateFunction
```

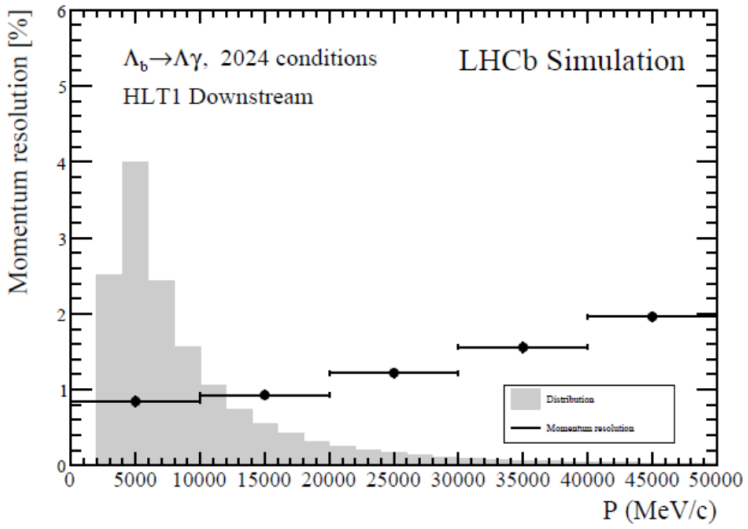
Real-Time Analysis (online): Tracking

- **Throughput:** -5 kHz, very fast algorithm! [RTX A5000 card]

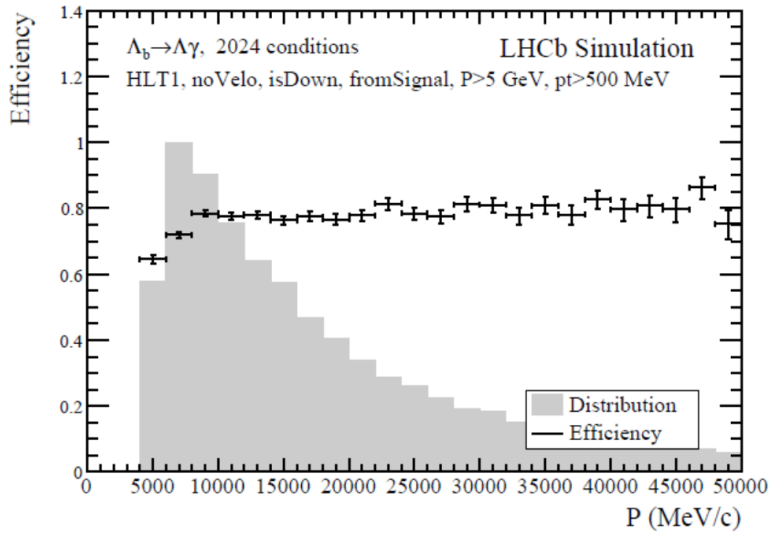




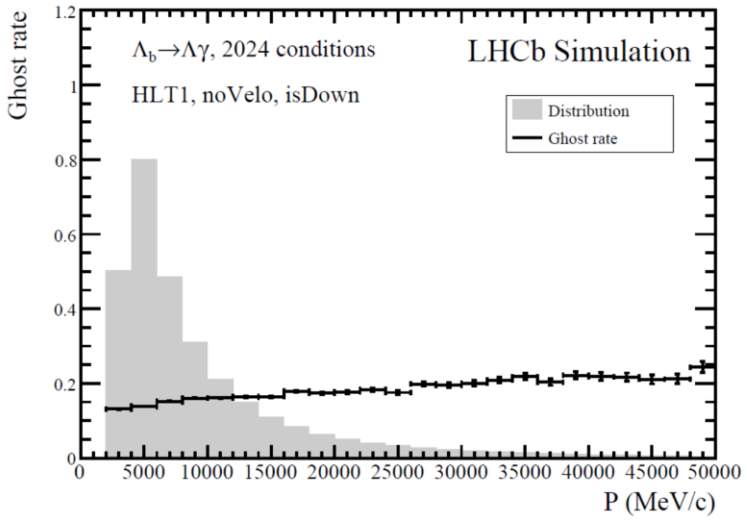
Track momentum resolution: 1-2 %



Efficiency: ~ 80 %



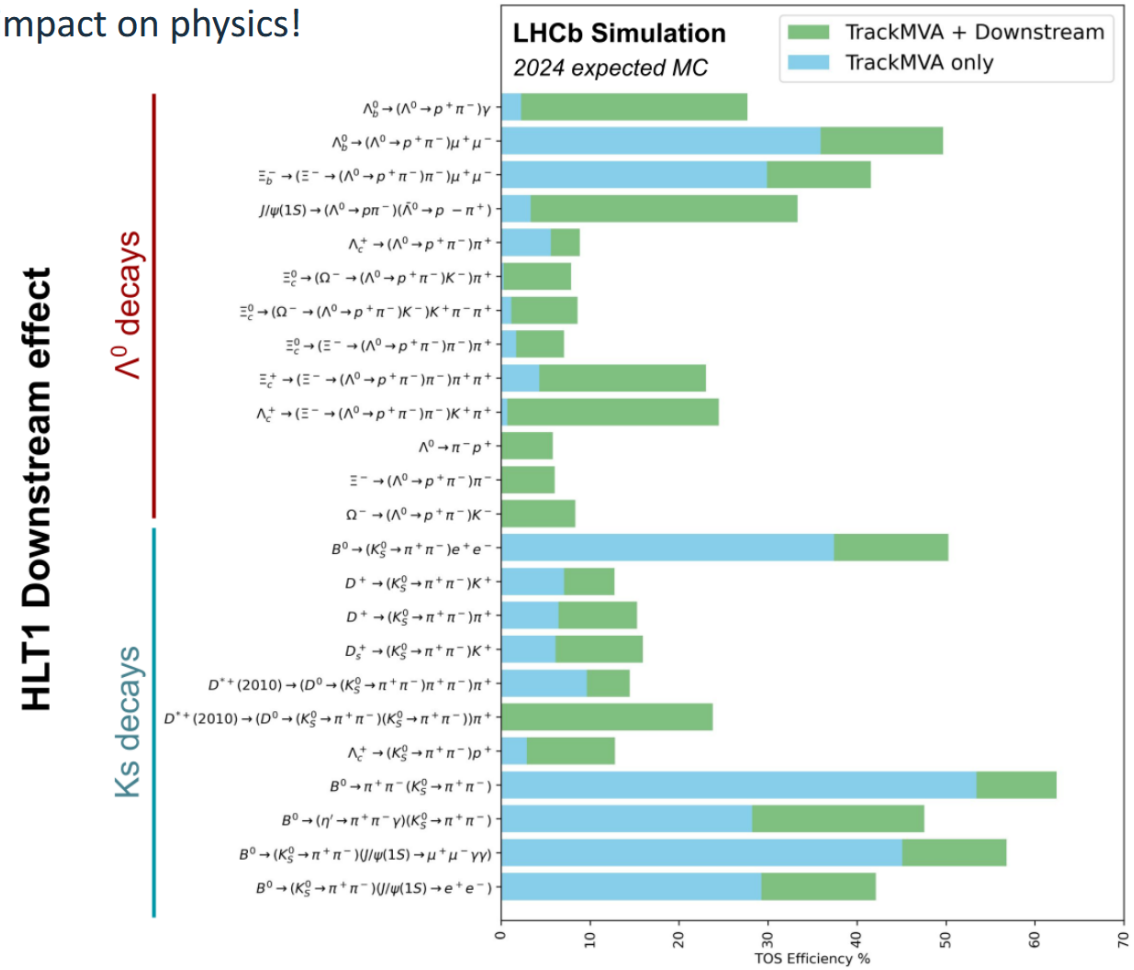
Ghosts < 20% !

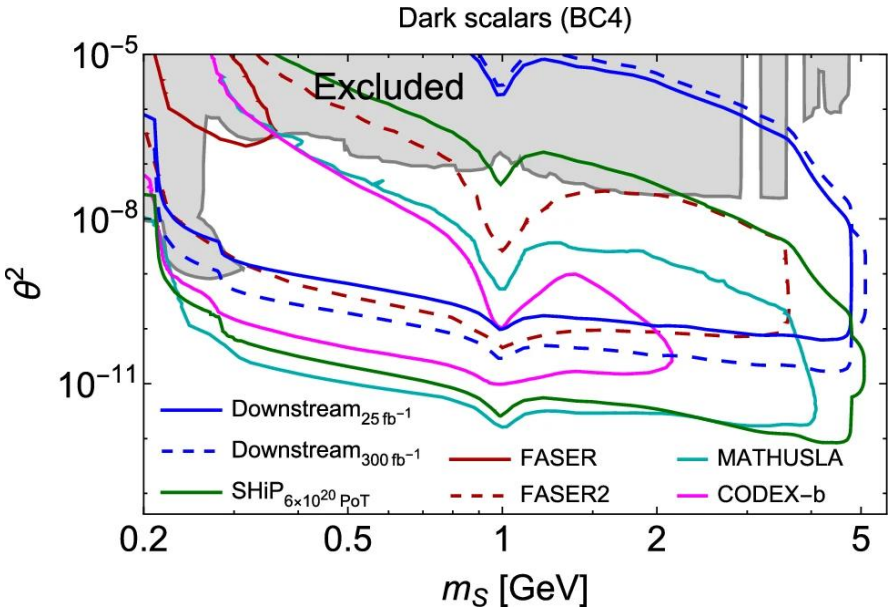


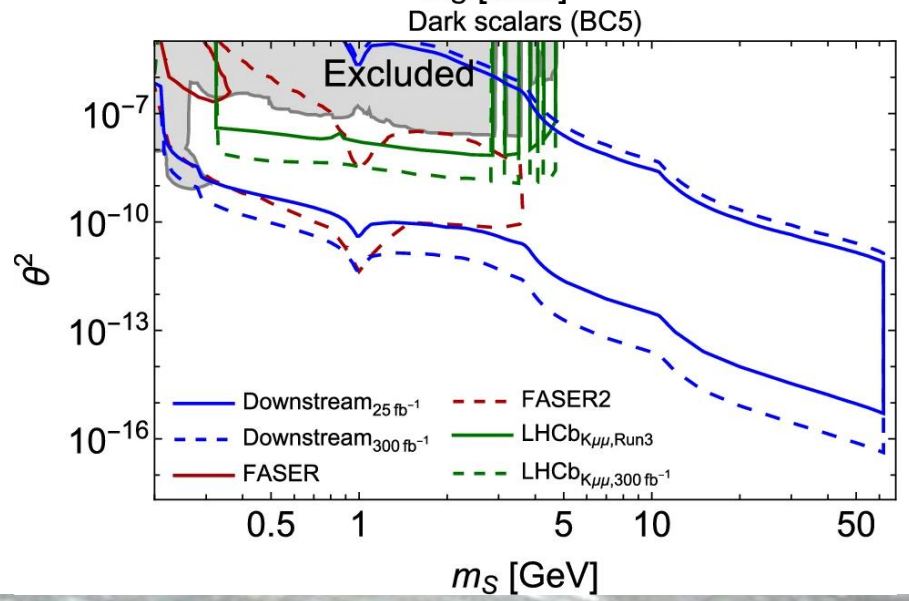
Real-Time Analysis (online): Tracking

10.1140/epjc/s10052-024-12906-3

Very high impact on physics!







GPU programming

- Full Hands-on GPU tutorial with examples - Course repo
- Where to get GPUs ? - https://colab.research.google.com
  - Using GPUs on the cloud for free, based on Jupyter and Python.

GPU session:

Session 1: Introduction: Parallel programming and CUDA C/C++
Exercise notebooks in 01, 02

- Parallel programming
- Hardware architectures and GPGPUs
- Programming languages and libraries

GPU session:

Session 2: <u>Advanced concepts and exercises:</u>
Exercise notebooks in 03

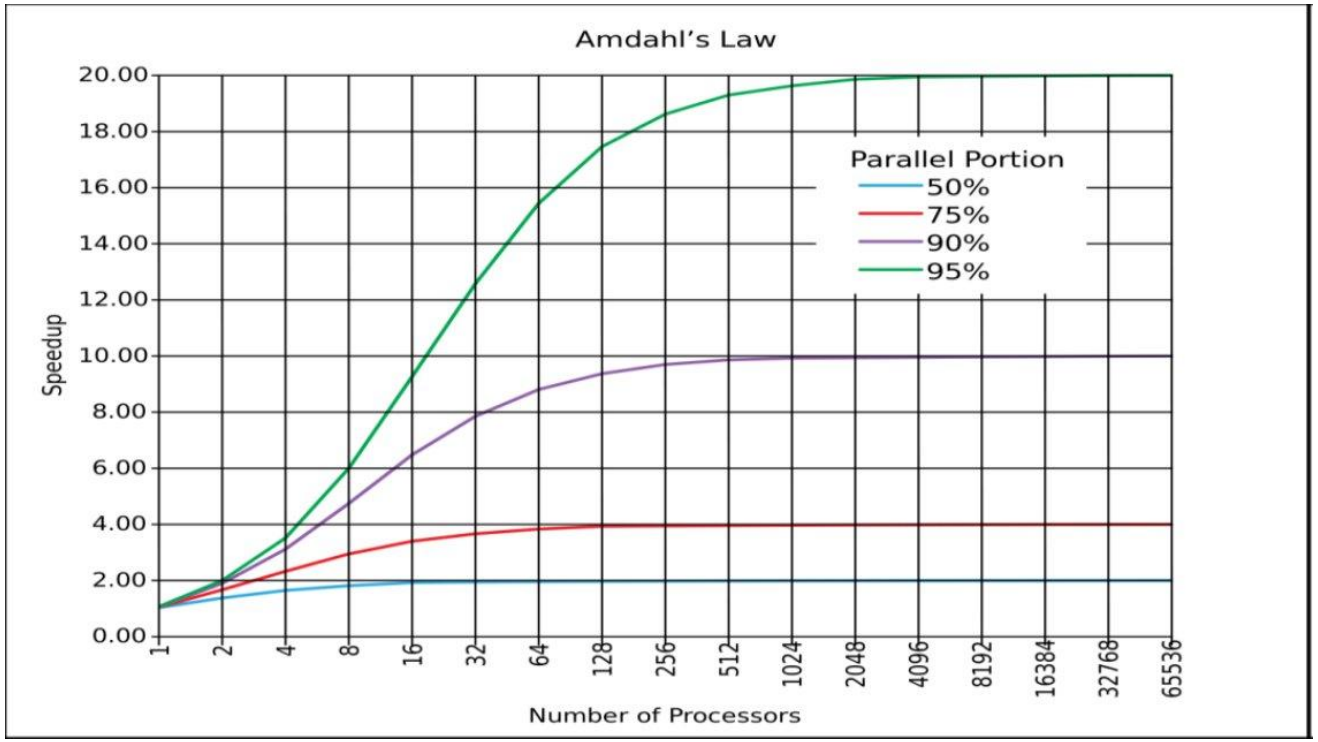
Session 3: <u>Python on GPU Intro and HEP exercises</u>

- Data analysis Python GPU – 04, 10
- CuPy and Numba examples – 04, 10
- Physics Generators - 05
- MC Simulation - 07
- Reconstruction Neural Network - 08

GPU programming: a refresher

Ways to speed up an application:

– Reducing the **complexity** of the algorithm.
– Increasing the speed and capacity of the computing medium: clock **frequency**.
– Searching for tasks within the application that can be performed in **parallel**.

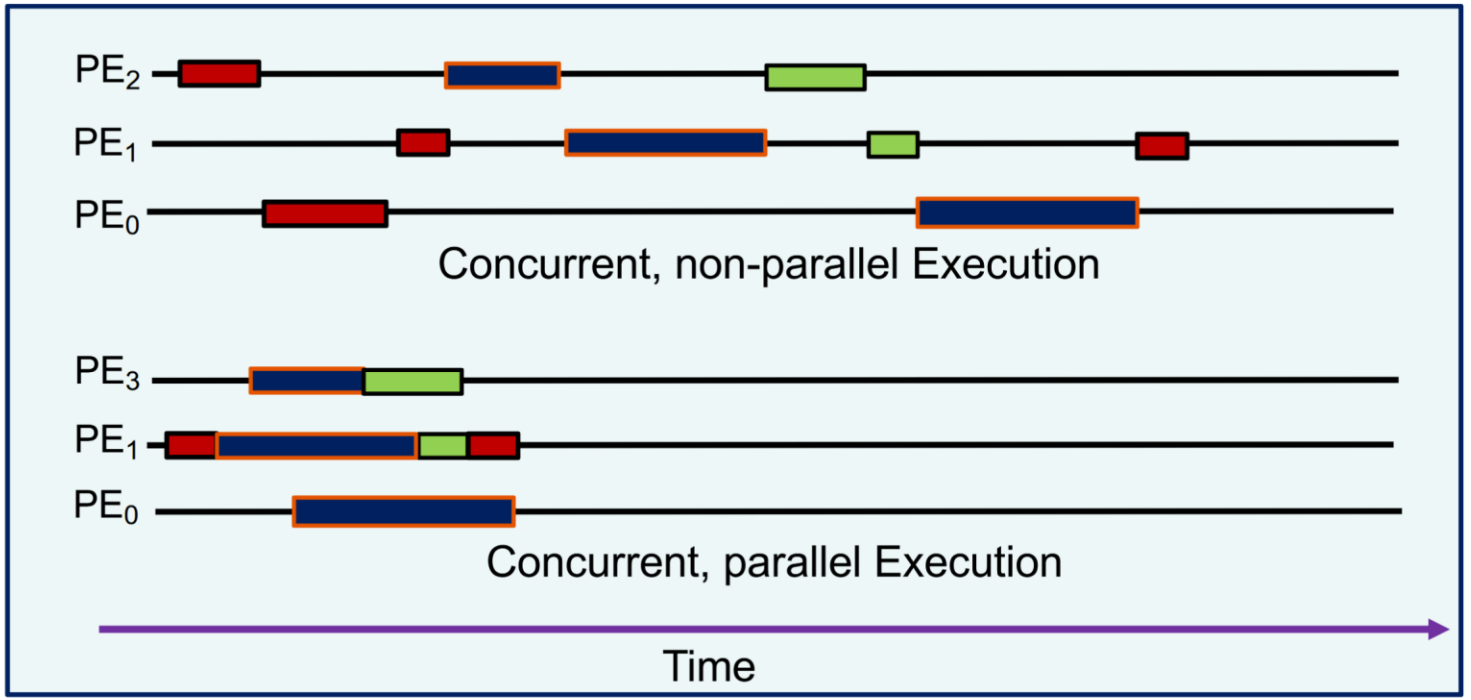


Concurrency vs. Parallelism

- Two important definitions:

  - Concurrency: A condition of a system in which multiple tasks are active and unordered. If scheduled fairly, they can be described as logically making forward progress at the same time.

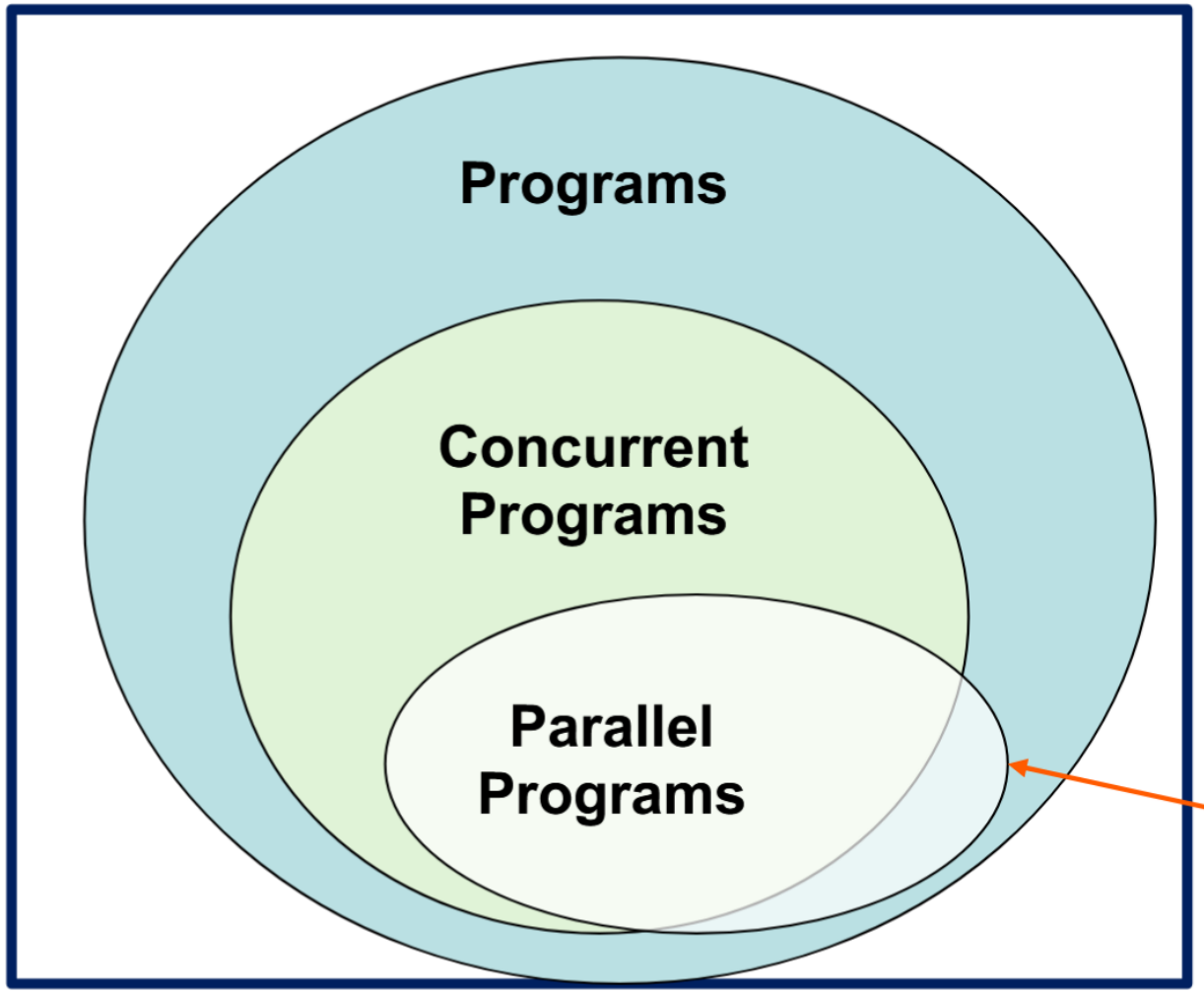
  - Parallelism: A condition of a system in which multiple tasks are actually making forward progress at the same time



PE = Processing Element

Figure from "An Introduction to Concurrency in Programming Languages" by J. Sottile, Timothy G. Mattson, and Craig E Rasmussen, 2010

Concurrency vs. Parallelism



In most cases, parallel programs exploit concurrency in a problem to run tasks on multiple processing elements

We use Parallelism to:

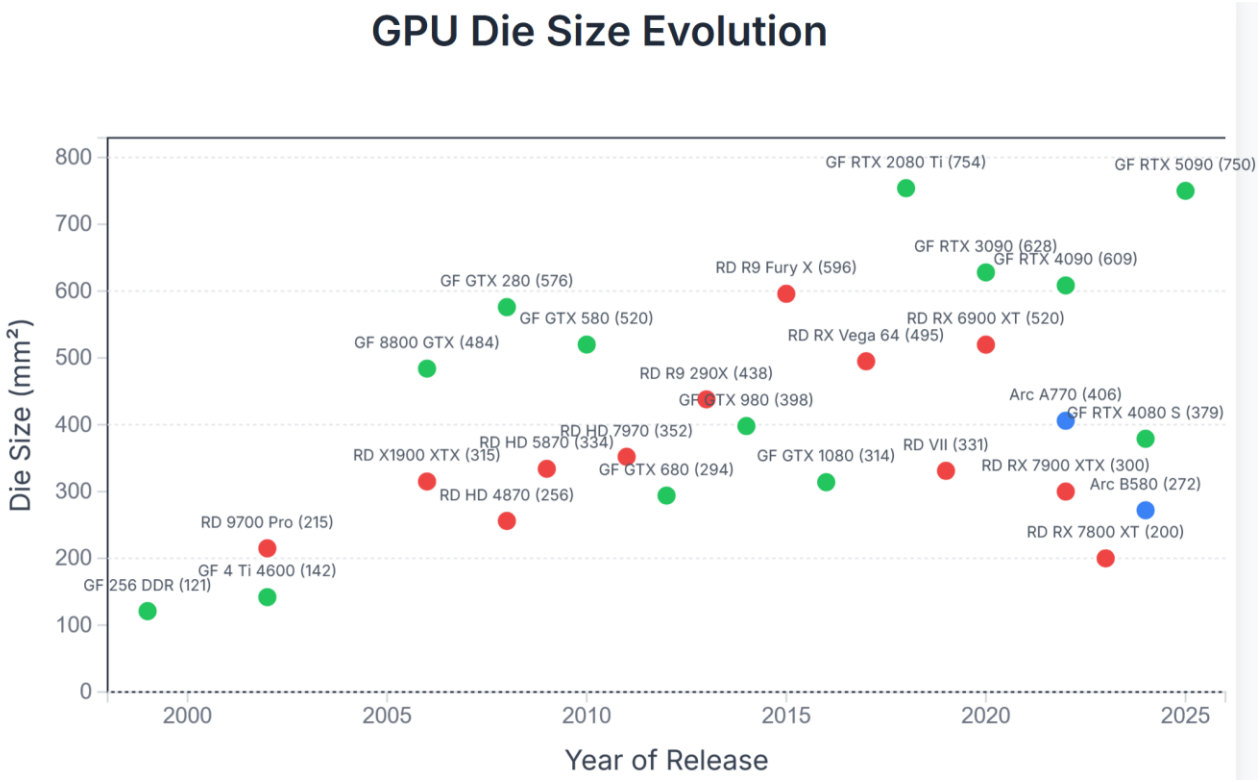
- Do more work in less time
- Work with larger problems

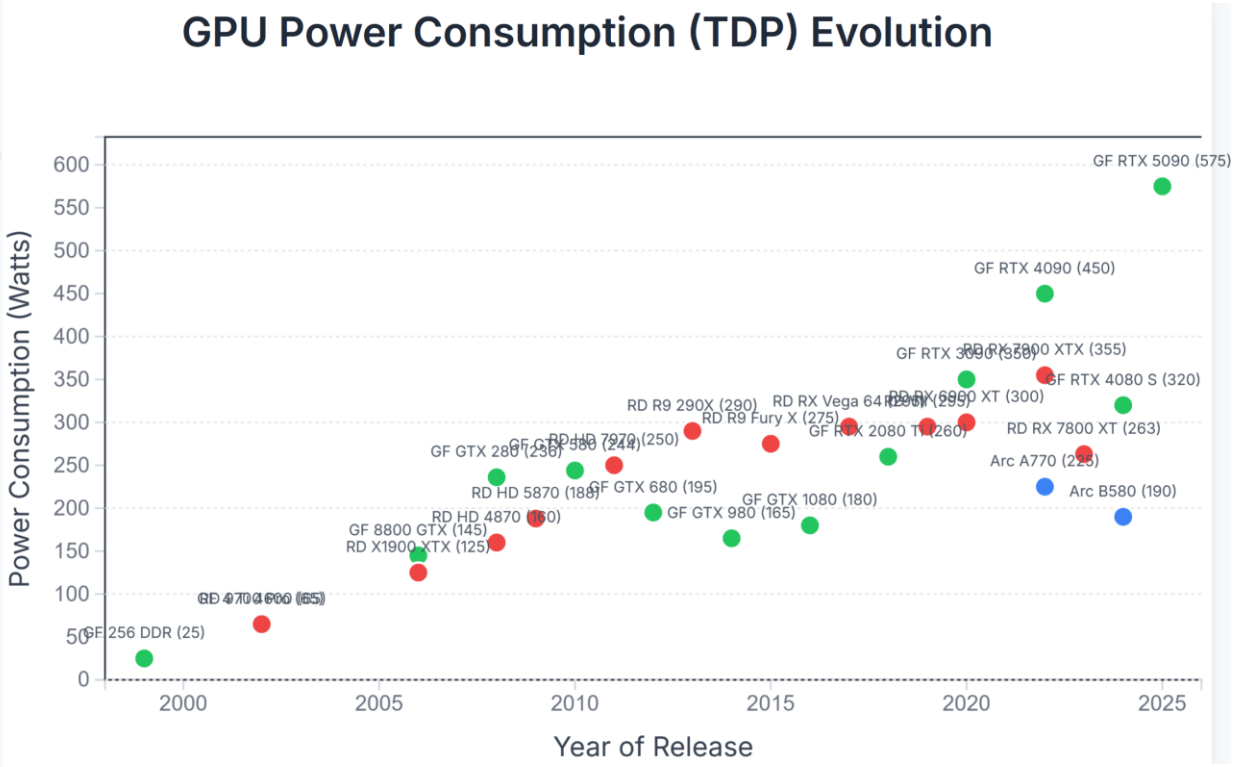
If tasks execute in “lock step” they are not concurrent, but they are still parallel. Example … a SIMD unit.

Figure from “An Introduction to Concurrency in Programming Languages” by J. Sottile, Timothy G. Mattson, and Craig E Rasmussen, 2010

GPU programming: a refresher

Advancements in platforms and architectures.





~ 20 Watts
~15 mW/cm³
~ 86 billion neurons and
~100–1,000 trillion synapses

GPT3
~175 billion parameters (synapses)
~But only ~96 layers × thousands of units per layer ⇒ maybe a few million neurons, tops.

GPGPU languages: Concepts are same, terminologies are different

Language	Cuda	OpenAcc	HIP
Definition	`kernel_name(...)`	`kernel_name(...)`	`kernel_name(...)`
Kernel Launch	`kernel_name<<<grid, block>>>(...)`	`clEnqueueNDRangeKernel(...)`	`hipLaunchKernelGGL(kernel_name, ...)`
Thread Indexing	`blockIdx.x`, `threadIdx.x`	`get_global_id(0)`, `get_local_id(0)`	`hipBlockIdx_x`, `hipThreadIdx_x`
Block Size	`blockDim.x`	`get_local_size(0)`	`hipBlockDim_x`
Grid Size	`gridDim.x`	`get_global_size(0) / get_local_size(0)`	`hipGridDim_x`
Shared Memory	`__shared__`	`__local`	`__shared__`
Memory Fence	`__syncthreads()`	`barrier(CLK_LOCAL_MEM_FENCE)`	`__syncthreads()`
Atomic Functions	`atomicAdd(...)`, `atomicSub(...)`, etc.	`atomic_add(...)`, `atomic_sub(...)`, etc.	`atomicAdd(...)`, `atomicSub(...)`, etc.

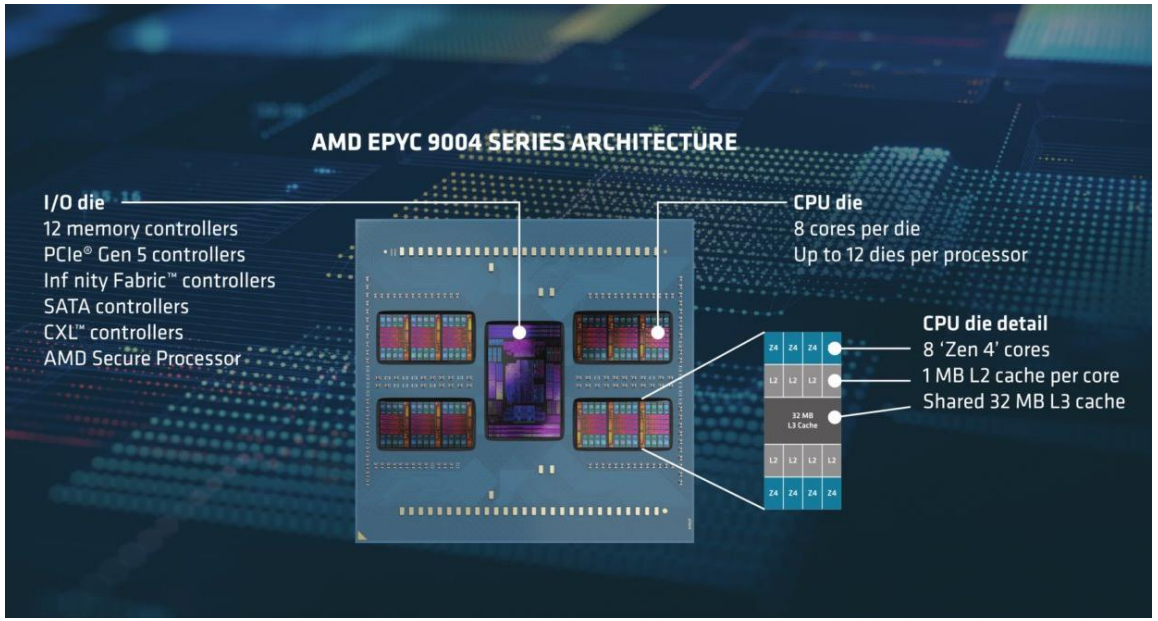
Performance portability layer

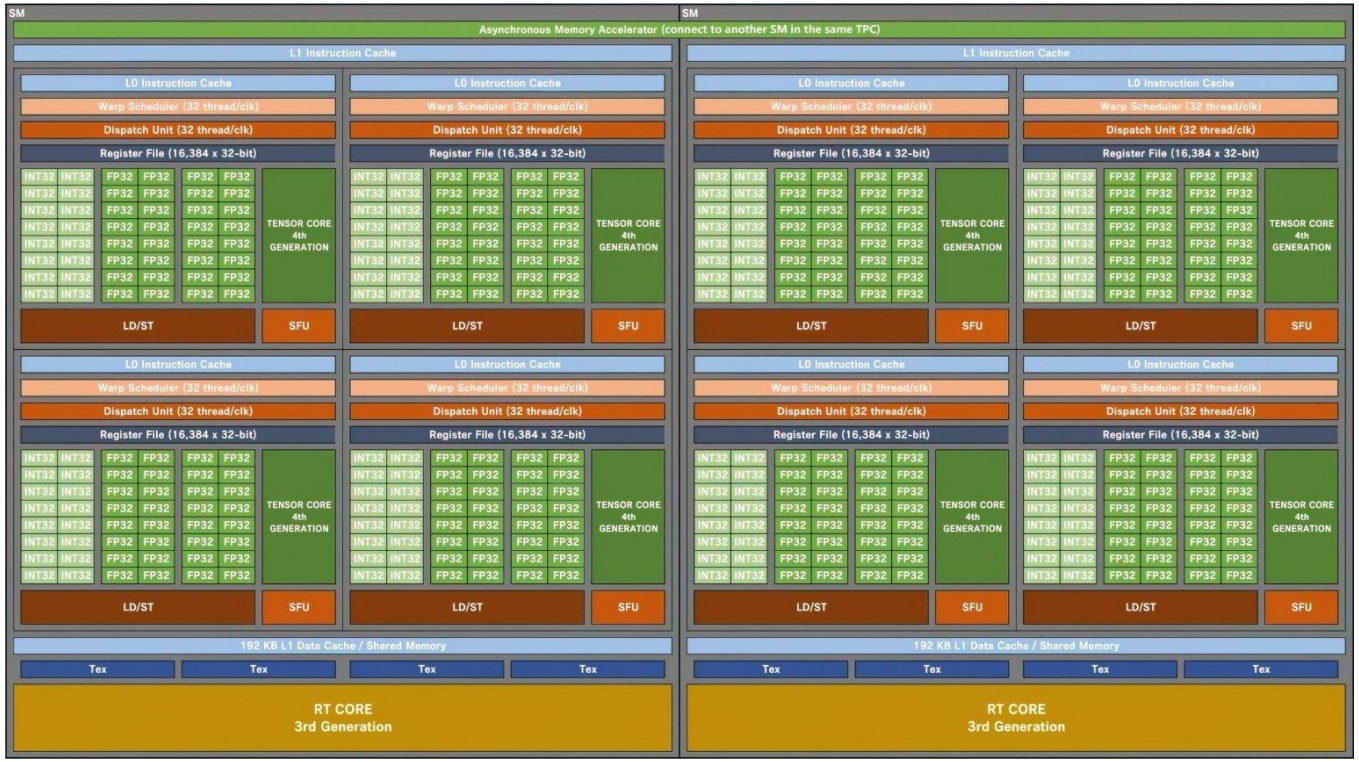
- Single source compilation: backend selection header-only library
- Other examples are ALPAKA, CUPLA, OneAPI etc etc….

```
// Host / device compiler identification
#if defined(TARGET_DEVICE_CPU) || (defined(TARGET_DEVICE_CUDA) && defined(
    __CUDACC__)) ||
  (defined(TARGET_DEVICE_HIP) && (defined(__HCC__) || defined(__HIP__)))
------
// Dispatch to the right backend
#if defined(TARGET_DEVICE_CPU)
#include "CPUBackend.h"
#elif defined(TARGET_DEVICE_HIP)
#include "HIPBackend.h"
#elif defined(TARGET_DEVICE_CUDA)
#include "CUDABackend.h"
#endif
-----
```

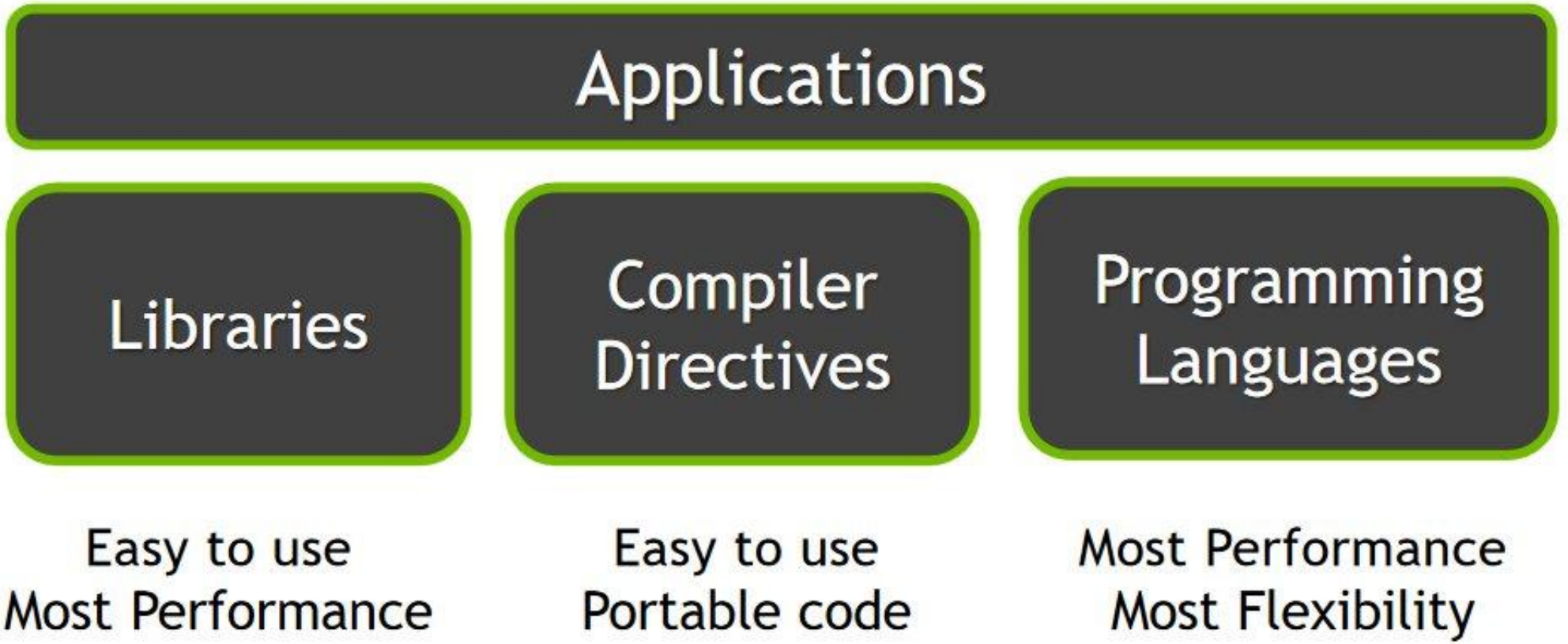
```
#if defined(DEVICE_COMPILER)
namespace Allen {
  namespace device {
    // Dispatcher targets
    namespace target {
      struct Default {
      };
      struct CPU {
      };
      struct HIP {
      };
      struct CUDA {
      };
    } // namespace target

```

GPU programming: a refresher

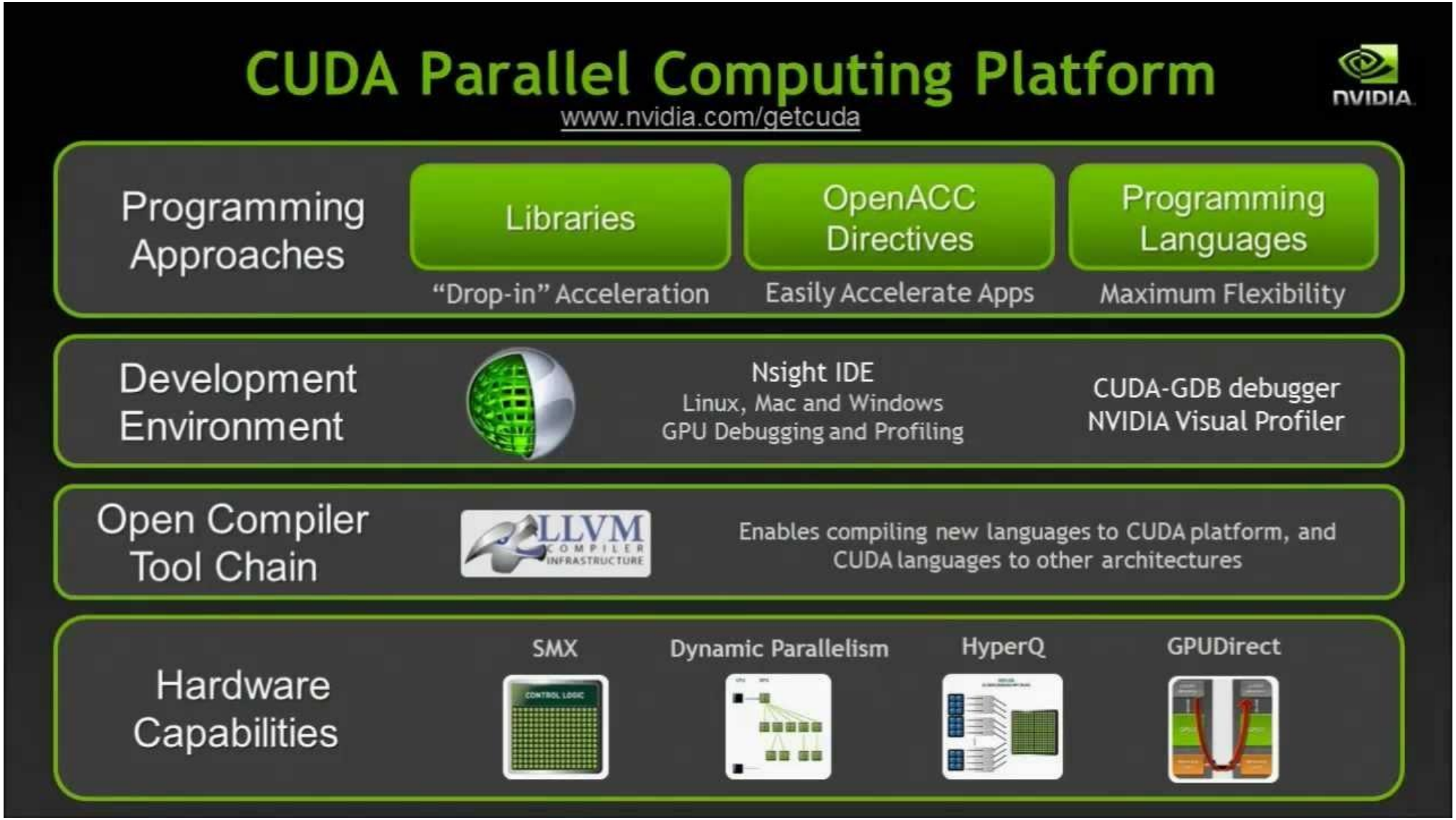




GPU programming: a refresher



GPU programming: a refresher



GPU programming: a refresher

GPU Accelerated Libraries

Vector Addition in Thrust

```
thrust::device_vector<float> deviceInput1(inputLength);
thrust::device_vector<float> deviceInput2(inputLength);
thrust::device_vector<float> deviceOutput(inputLength);

thrust::copy(hostInput1, hostInput1 + inputLength,
    deviceInput1.begin());
thrust::copy(hostInput2, hostInput2 + inputLength,
    deviceInput2.begin());

thrust::transform(deviceInput1.begin(), deviceInput1.end(),
    deviceInput2.begin(), deviceOutput.begin(),
    thrust::plus<float>());
```

GPU programming: a refresher

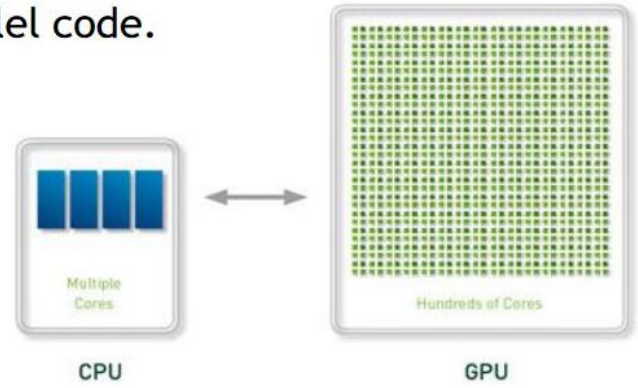
Compiler Directives: Easy, Portable Acceleration

- **Ease of use:** Compiler takes care of details of parallelism management and data movement
- **Portable:** The code is generic, not specific to any type of hardware and can be deployed into multiple languages
- **Uncertain:** Performance of code can vary across compiler versions

```c
#include <stdio.h>

int main() {
    int n = 100000;
    float a[n], b[n], c[n];

    // Initialize arrays
    #pragma acc parallel loop
    for (int i = 0; i < n; i++) {
        a[i] = i;
        b[i] = 2 * i;
    }

    // Perform vector addition
    #pragma acc parallel loop
    for (int i = 0; i < n; i++) {
        c[i] = a[i] + b[i];
    }

    // Print the result
    for (int i = 0; i < n; i++) {
        printf("%f + %f = %f\n", a[i], b[i], c[i]);
    }

    return 0;
}
```

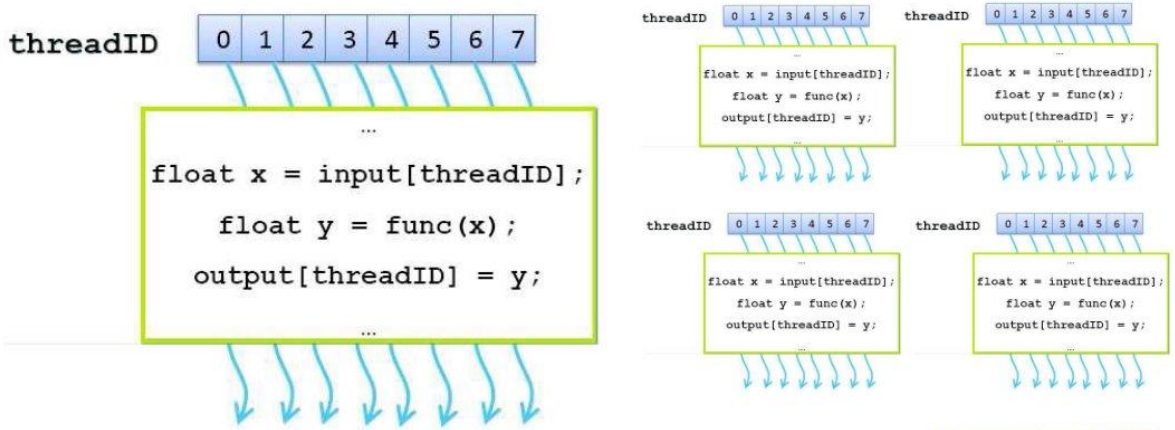
GPU programming: a refresher

- **CUDA** allows:
  - **To scale** the parallelism to thousands of **lightweigth** threads executed in hundreds of processors.
  - **To abstract** the GPU to programmers.
  - To compose an ***heterogenous system*** (CPU + GPU):
    - CPU for sequential code and control.
    - GPU for parallel code.



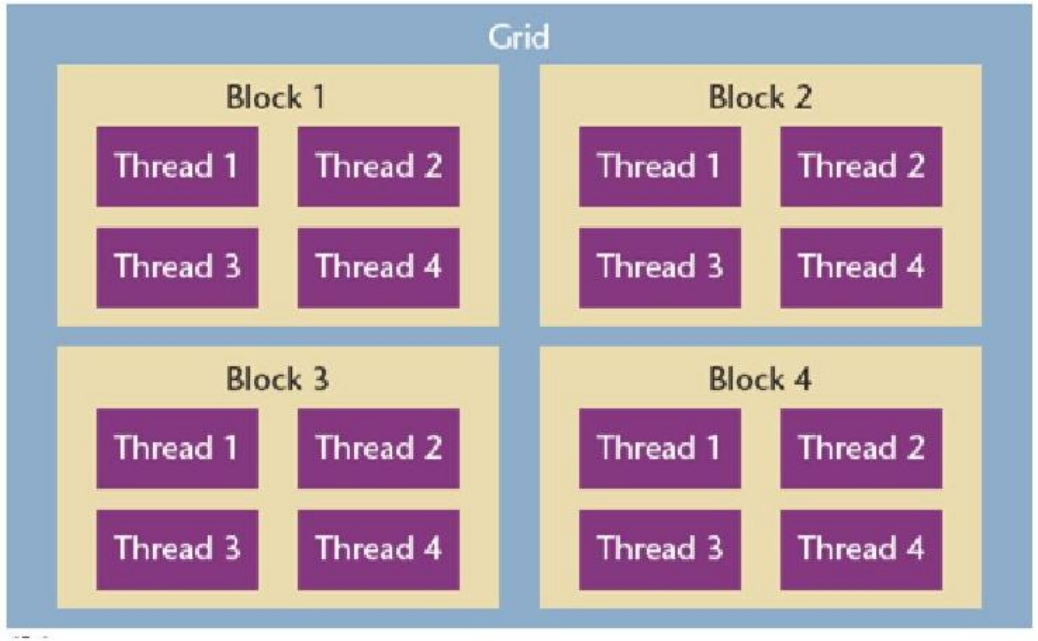
Introduction to CUDA

- The execution of threads is **SPMD** (**S**ingle **P**rogram **M**ultiple **D**ata): they run the same code (kernel) over different data (previously copied to the GPU).

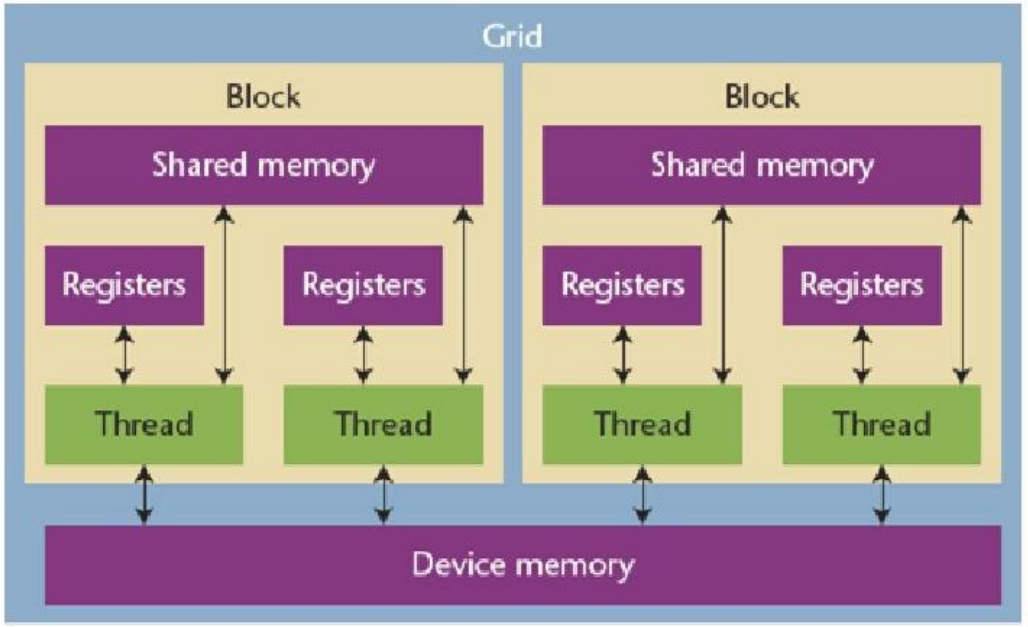


GPU programming: a refresher

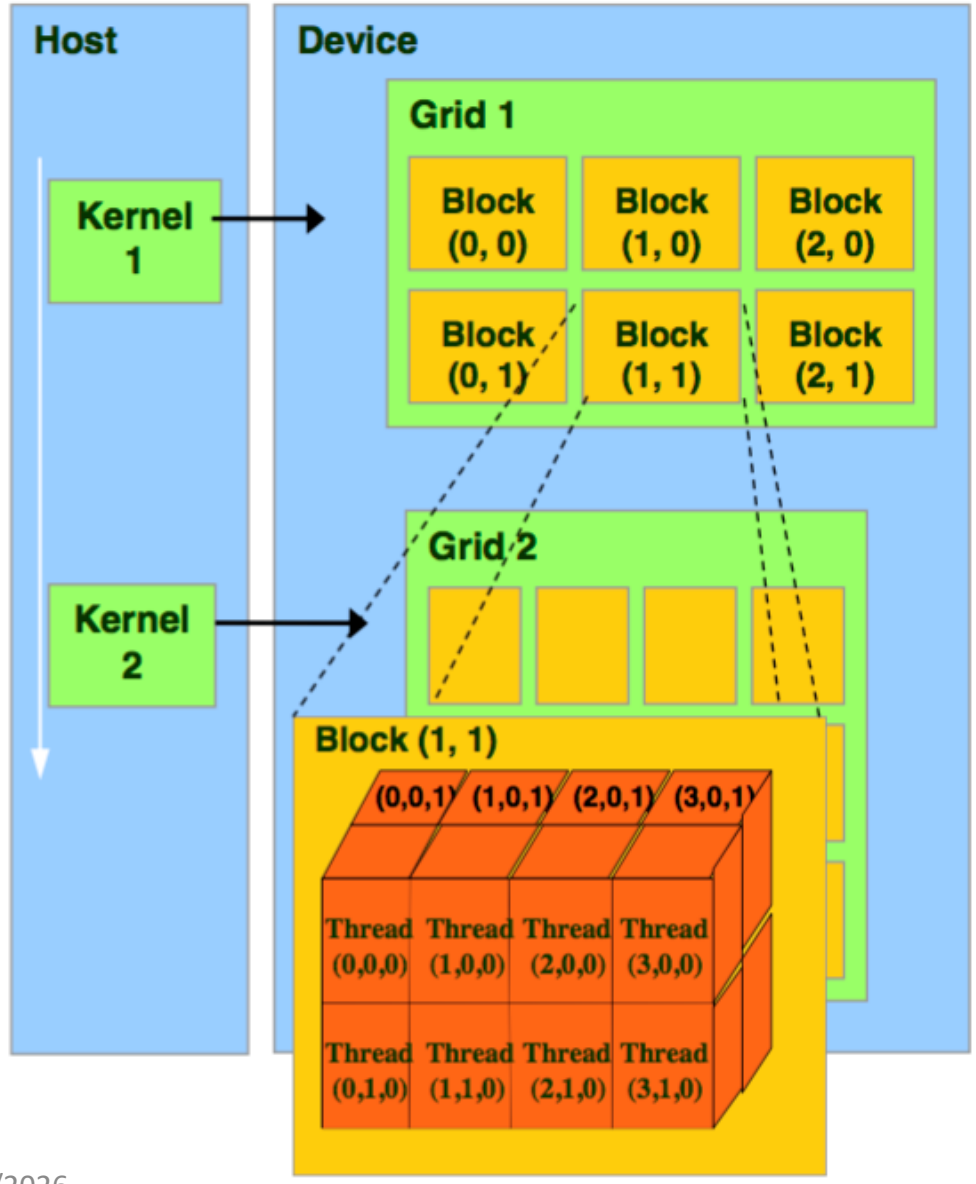
- **Hierarchy of threads** in the model.

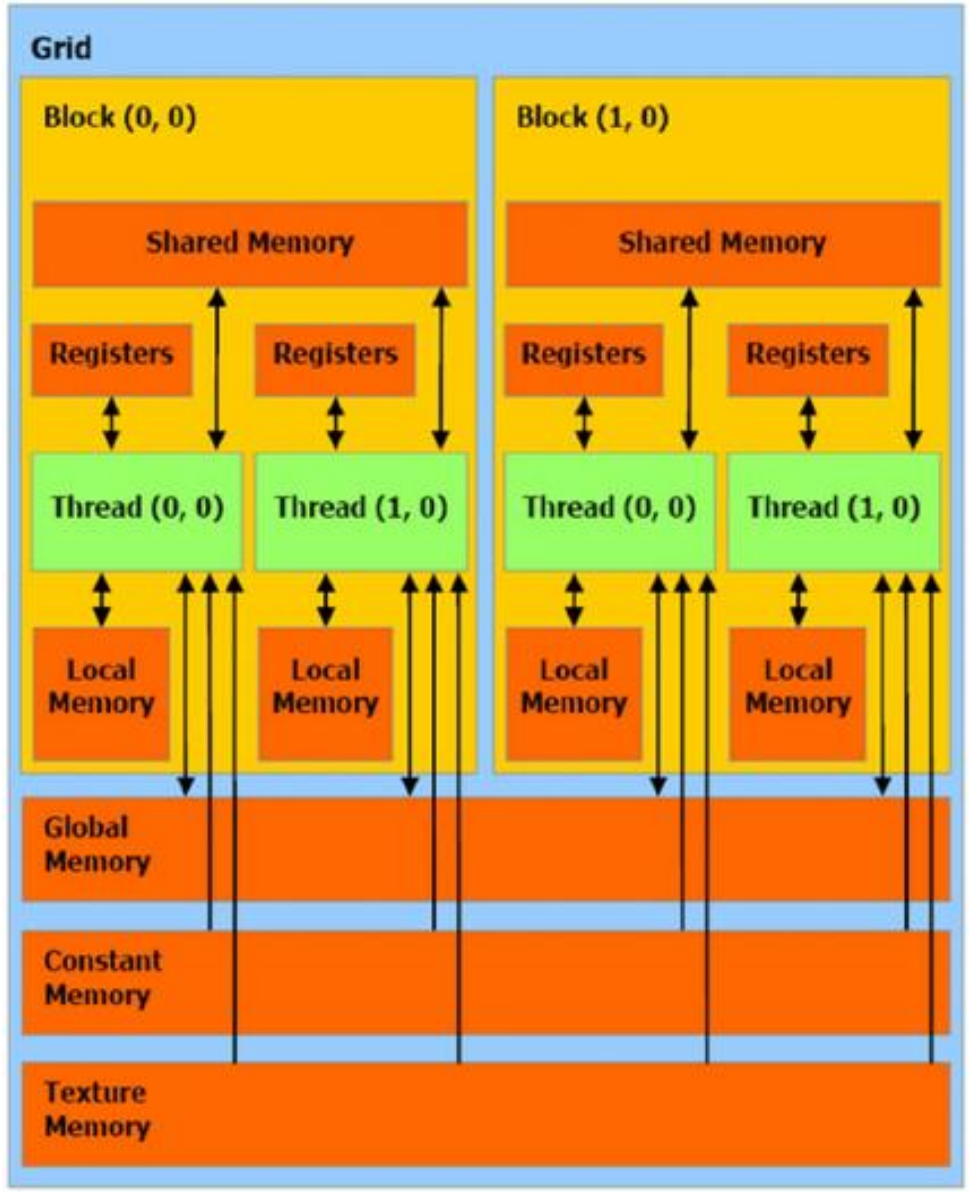


- **Hierarchy** of **memory** in the model:

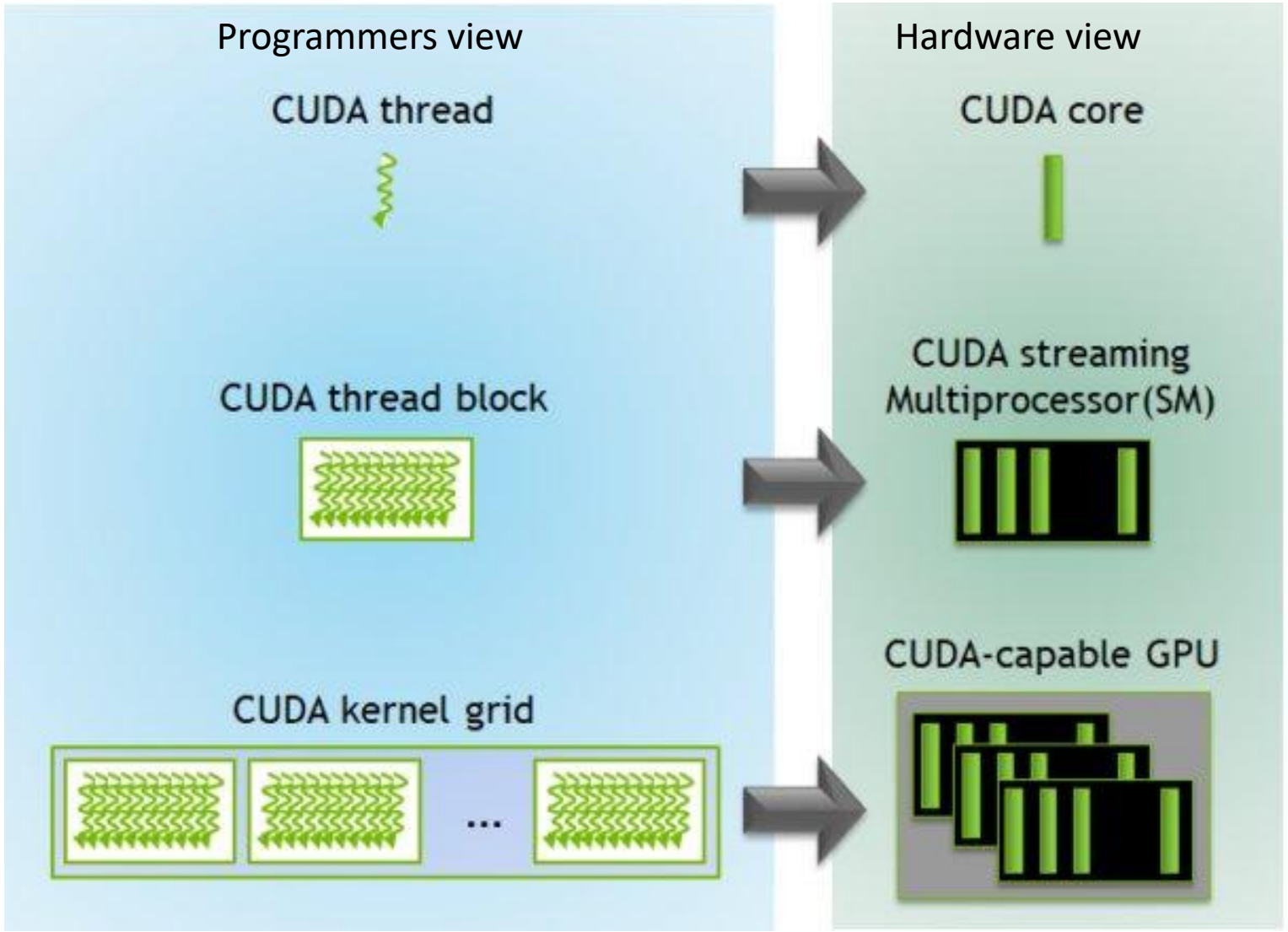


GPU programming: a refresher

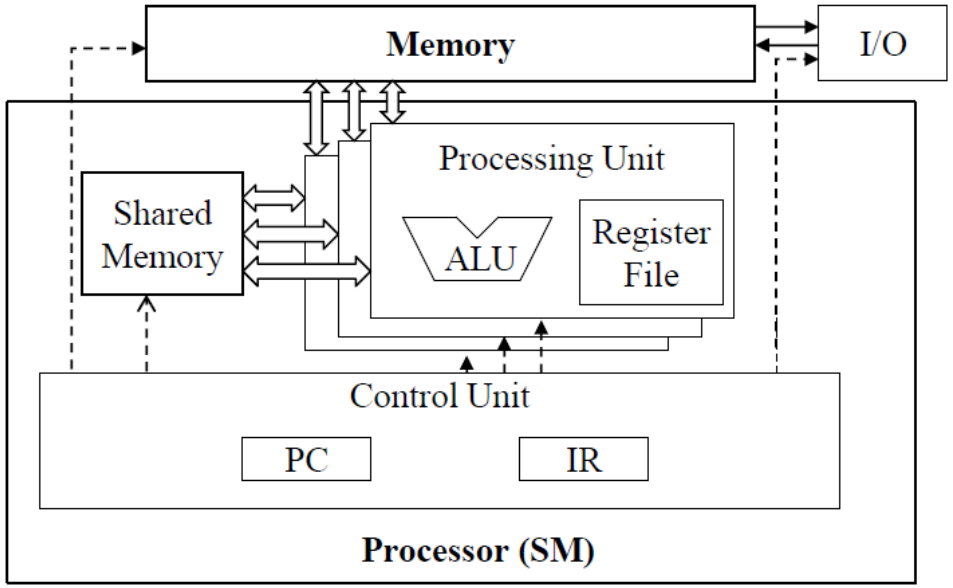




GPU programming: a refresher



Architecture: GH100 Full GPU with 144 SMs

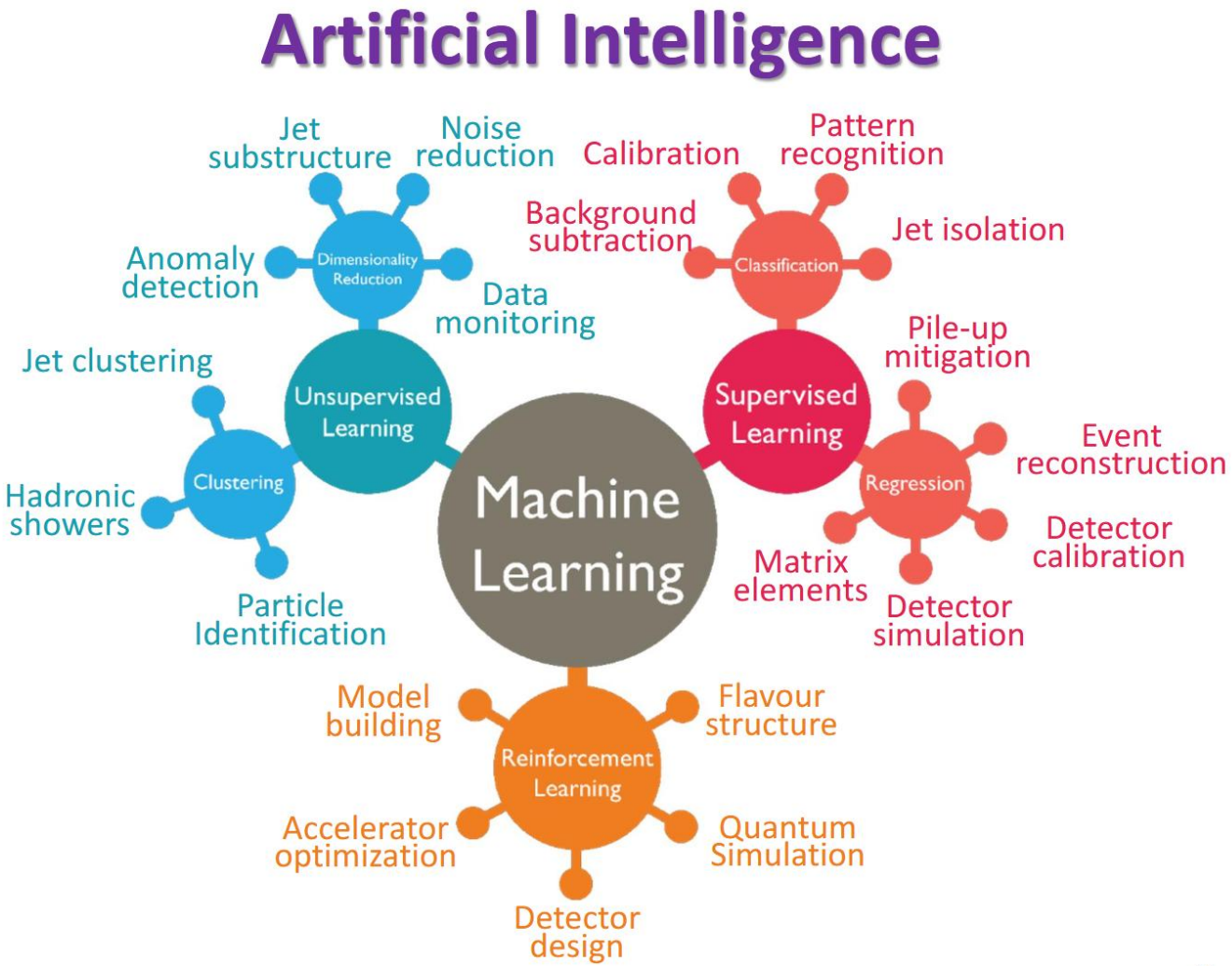


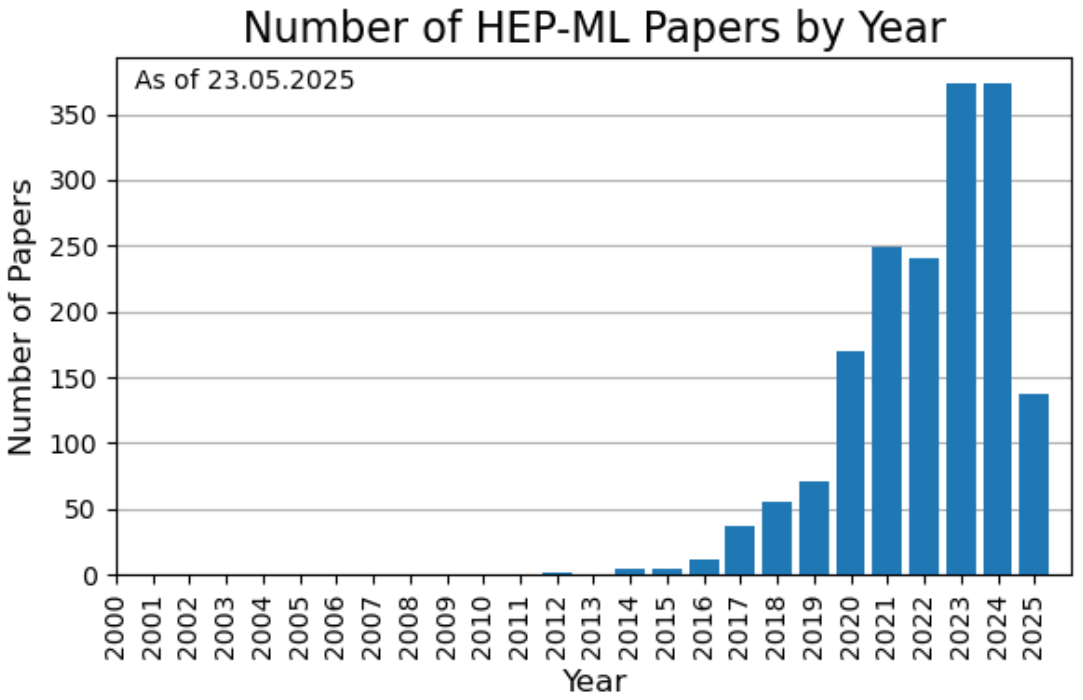
Single Streaming Multiprocessor (SM)

7. Future ?

Generative AI

https://iml-wg.github.io/HEPML-LivingReview/

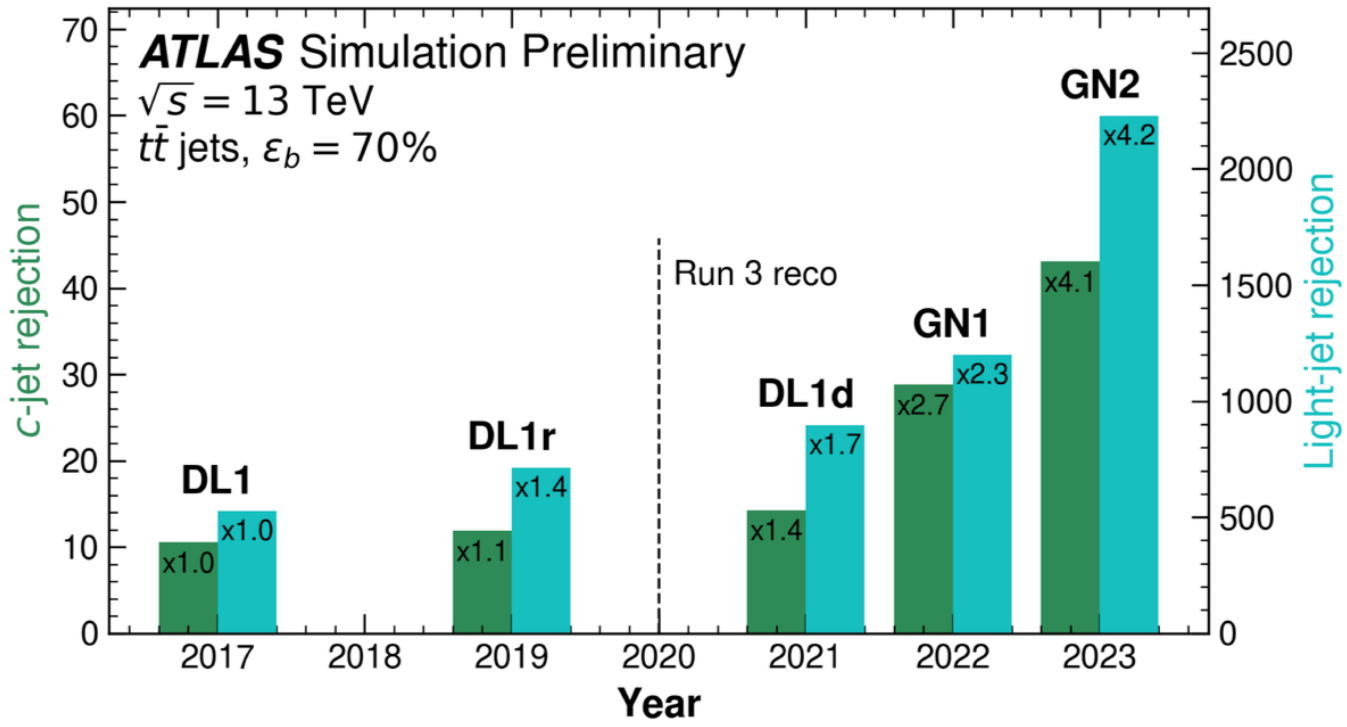




AI and physics performance

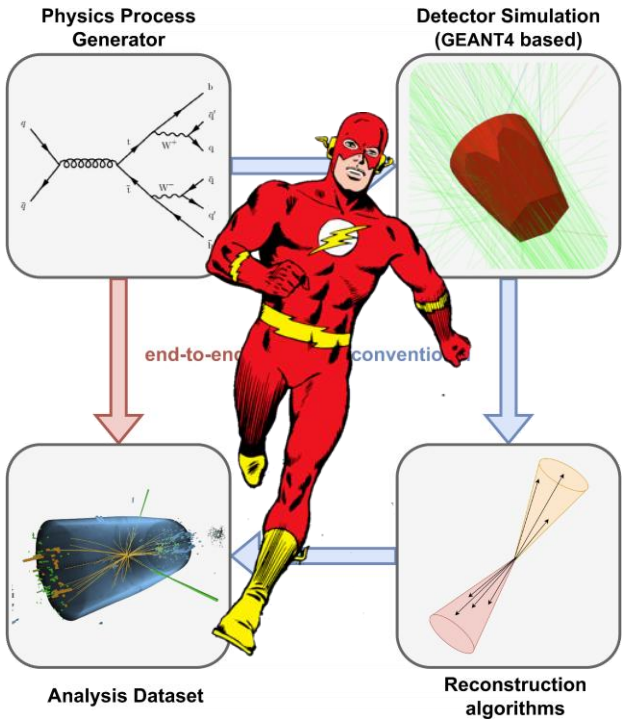
- Machine Learning (ML) / Deep Learning (DL) models in production in HEP since years
- The models are more and more optimised for better physics performance
- Not necessarily impactful in terms of distributed computing
- Generally, no need for special hardware and/or facilities
- No obvious speedup or improvement in computing performance

Flavour Tagging (ATL-PHYS-PROC-2024-081)



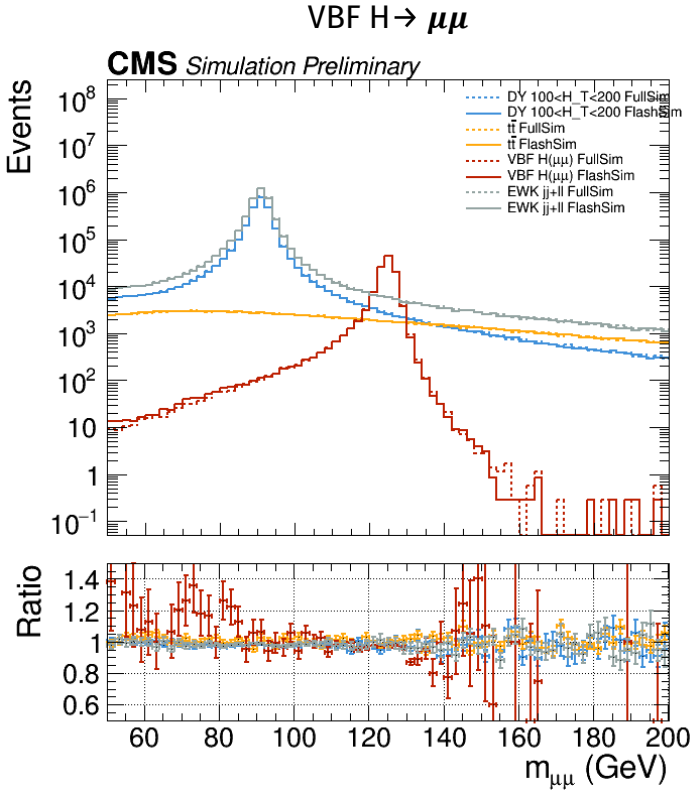
AI and computing models

From ML/AI there is potential for groundbreaking Computing Models evolution

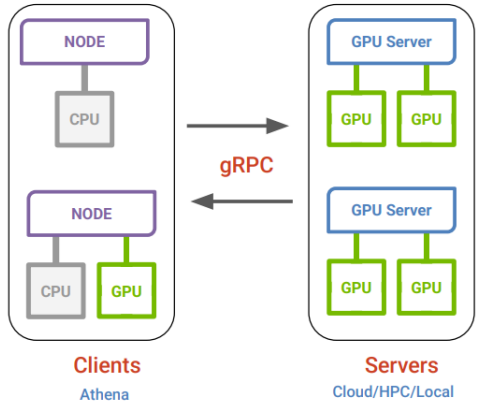


- CMS FlashSim: end-to-end simulation with generative AI
- From Generation to nanoAOD
- Orders of magnitude faster than Full Simulation - precise number irrelevant
- Goal: accuracy “good enough” for analysis

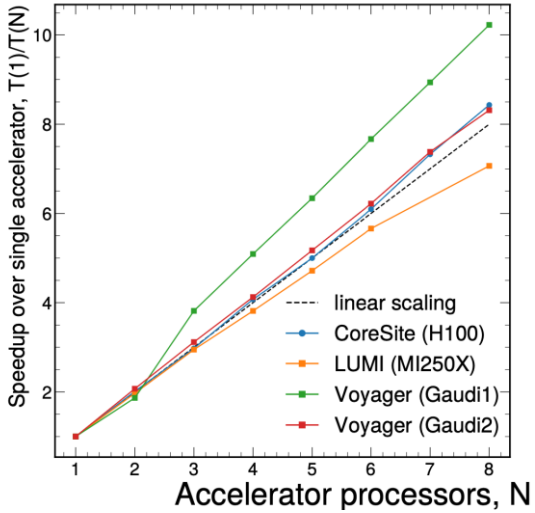
- => Simulation could become “on-the-fly”



AI and computing facilities



TODAY:
inference as a service



TODAY:
training on HPC centers

From the WLCG Heterogeneous Arch. **WORKSHOP**

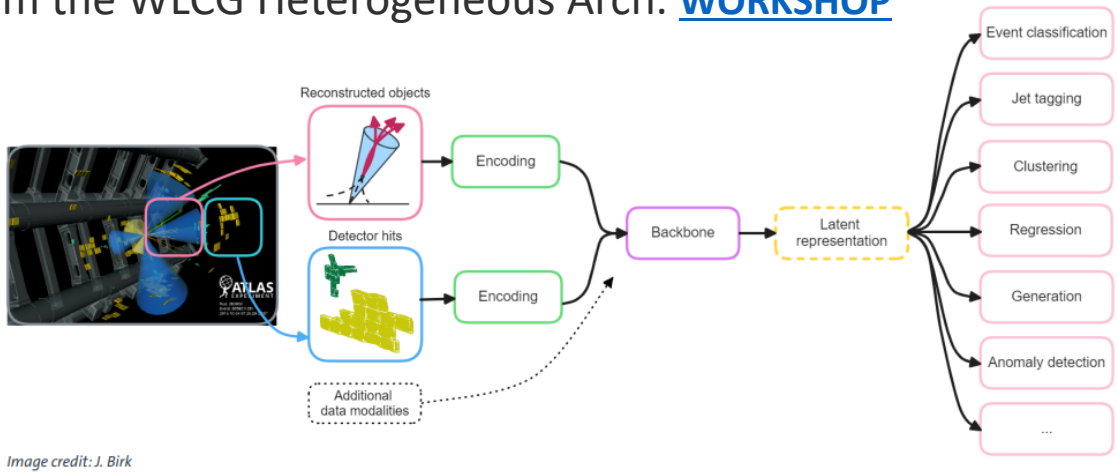


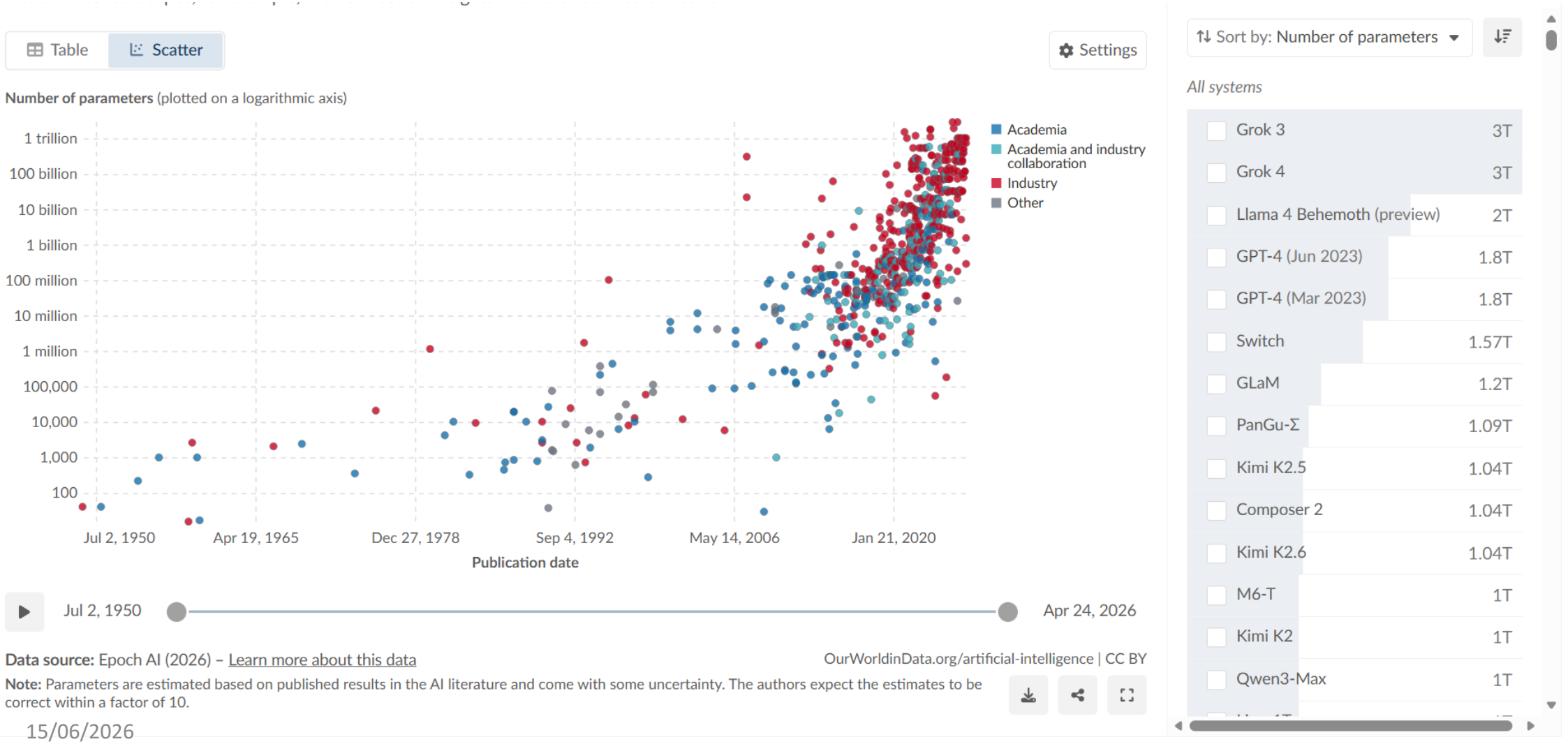
Image credit: J. Birk

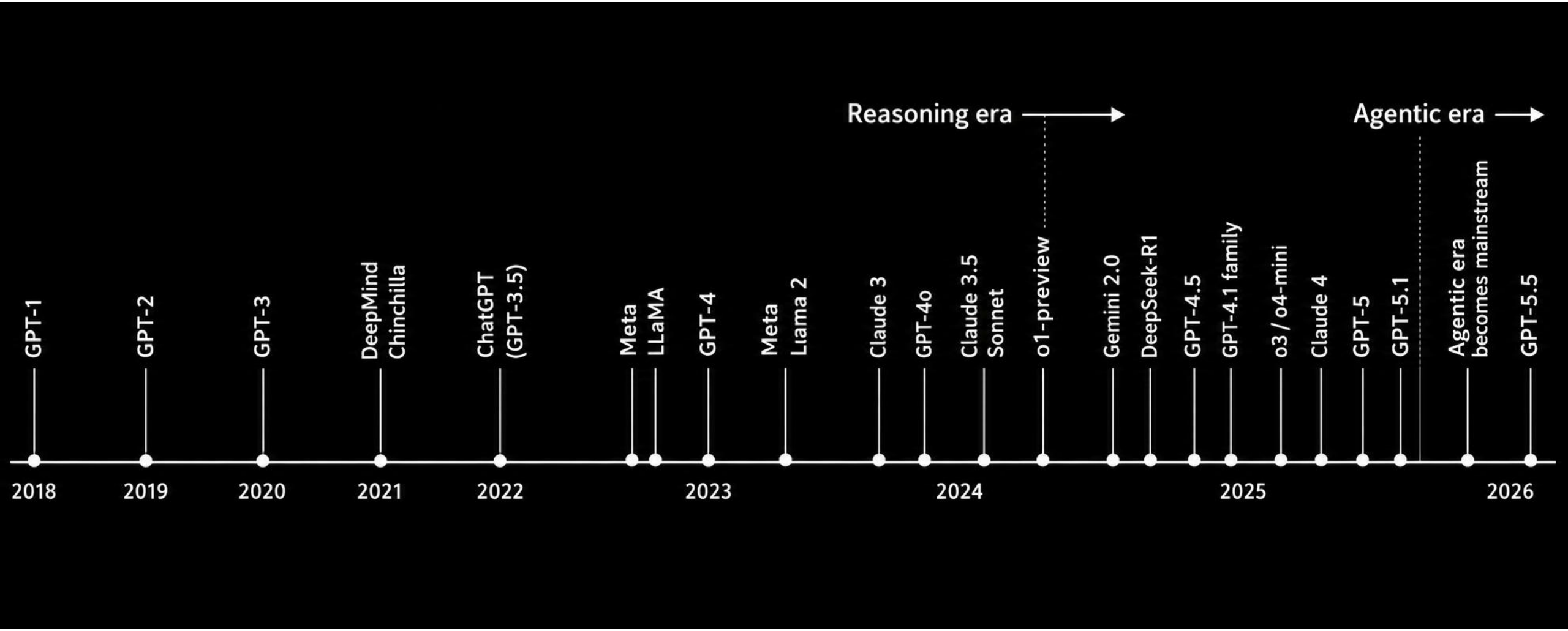
TOMORROW: Foundation Models => AI Factories

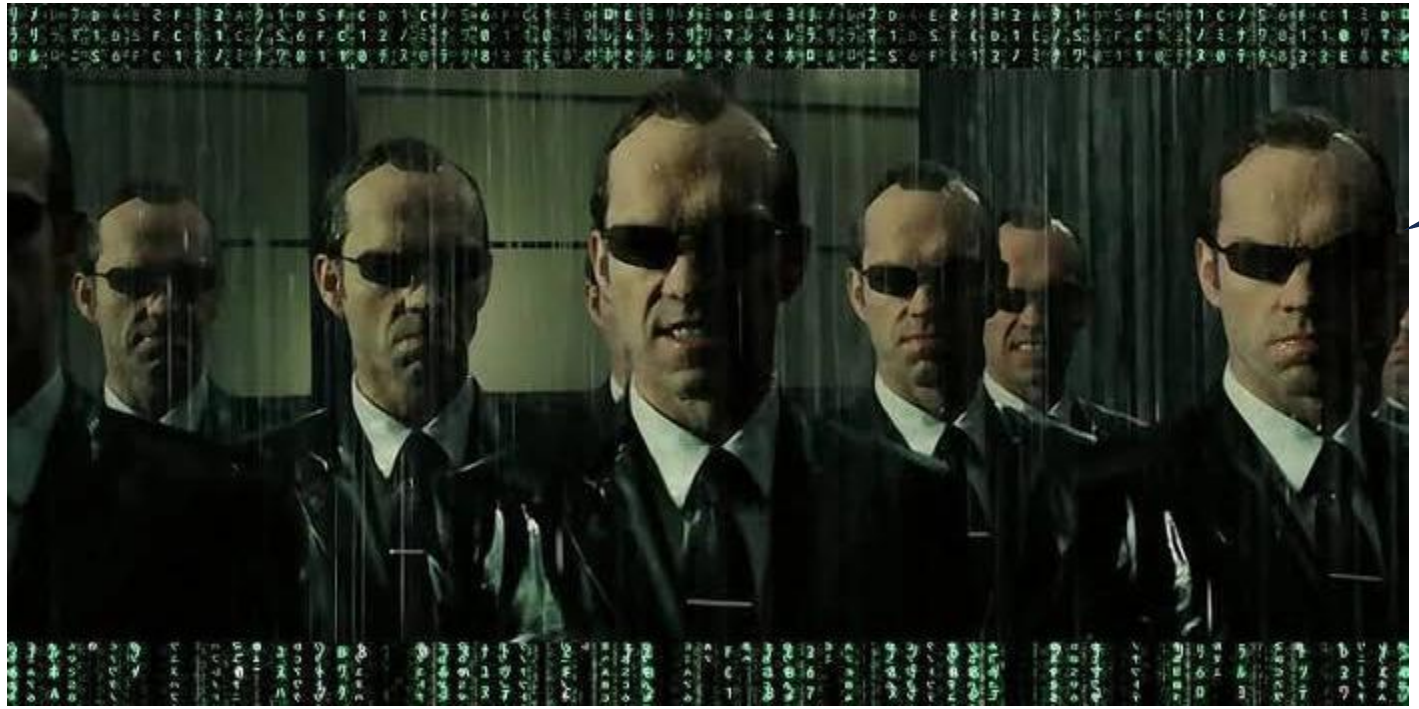
- Learn optimized data representation
- Adapt to multiple tasks
- Trained on large scale, multi-modal data sets

Interactive Parameter Size Map

How Large is Large?




Reasoning era
Agentic era
GPT-1
GPT-2
GPT-3
DeepMind Chinchilla
ChatGPT (GPT-3.5)
Meta LLaMA
GPT-4
Meta Llama 2
Claude 3
GPT-4o
Claude 3.5 Sonnet
o1-preview
Gemini 2.0
DeepSeek-R1
GPT-4.5
GPT-4.1 family
o3 / o4-mini
Claude 4
GPT-5
GPT-5.1
Agentic era becomes mainstream
GPT-5.5
2018
2019
2020
2021
2022
2023
2024
2025
2026

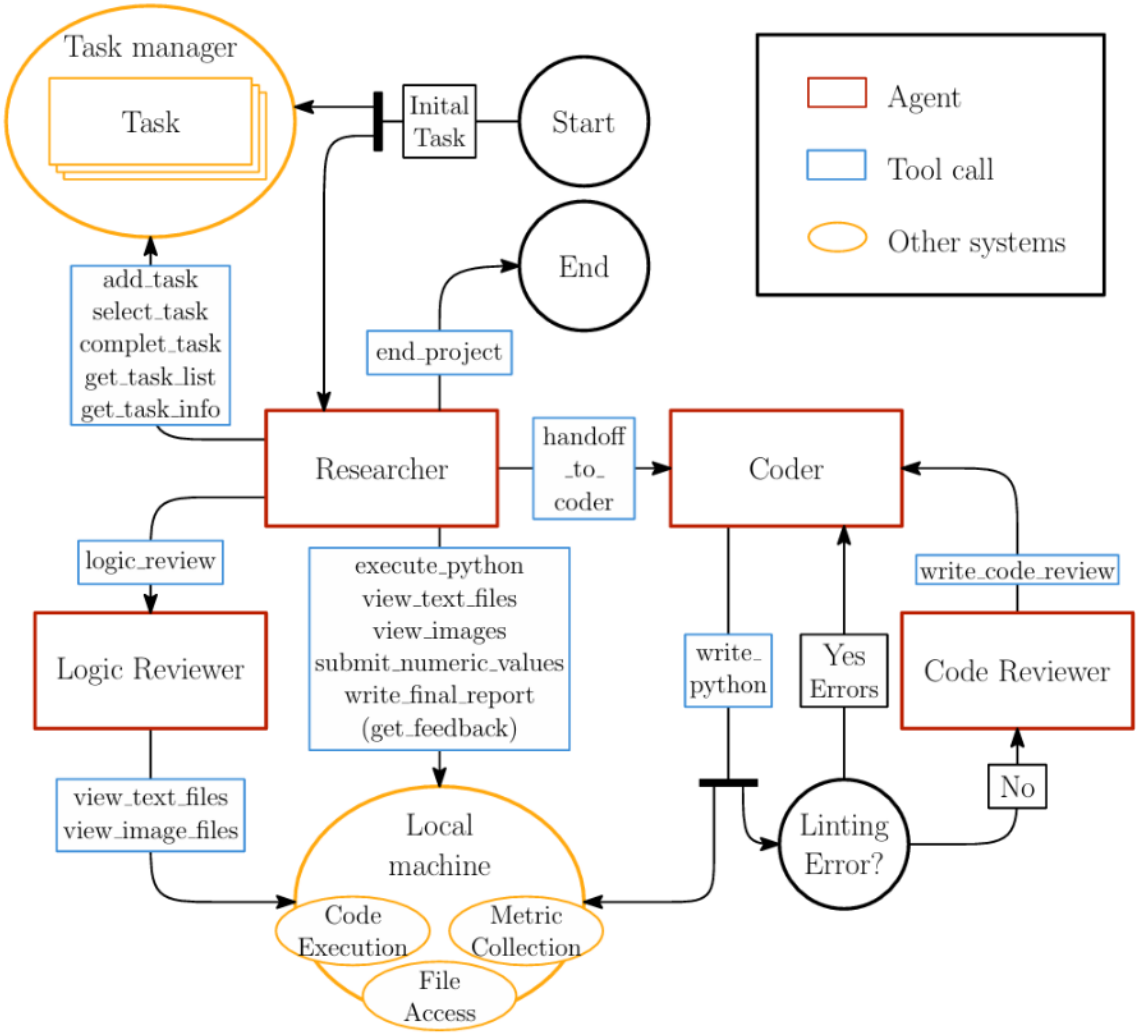


Wait, did you say “Agent”

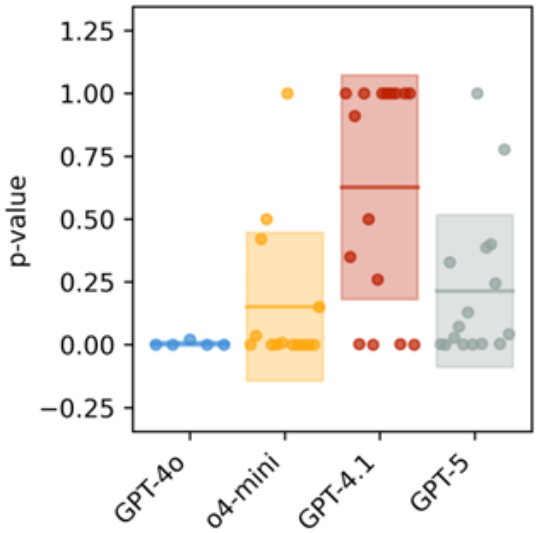
Agents of Discovery

This was released in August 2025

- Note they already have a sophisticated feed back loop to execute the tasks.
- All written as a handwritten agentic harness
  - No open-source harness was up to the task yet



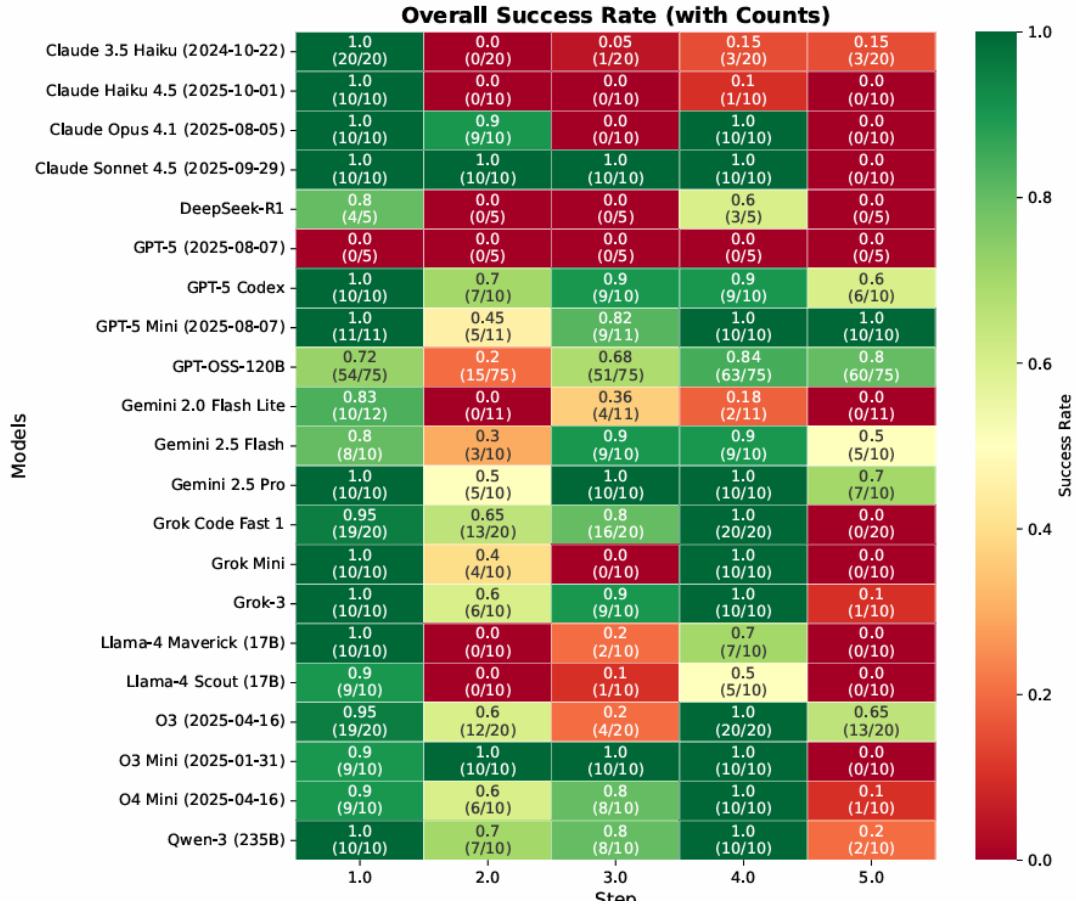
"To investigate the capabilities of current-generation commercial LLMs, we consider the task of anomaly detection via the publicly available and highly-studied LHC Olympics dataset."



Automating High Energy Physics Data Analysis with LLM-Powered Agents

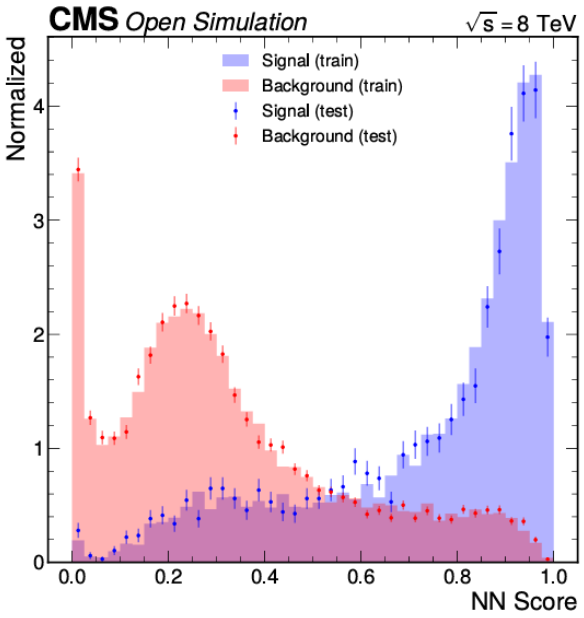
We introduce a framework in which LLM-based agents are integrated into a Snakemake-managed workflow. Agent interventions are bounded to well-defined tasks such as code generation, event selection, and validation while the underlying directed acyclic graph (DAG) maintains determinism and provenance.

ATLAS Open Data Analysis: cross-section measurement of the Higgs boson decaying to two photons. Collision data and simulation samples from the 2020 ATLAS Open Data release are used, corresponding to 10 fb^{-1}.



1. ROOT File Inspection
2. Ntuple Conversion
3. Preprocessing
4. S-B Separation
5. Categorization

First to do multiple analyses



Each analysis contains a full log and evaluation in the paper

It took 4-10 hours for each analysis start to finish

This work did not focus on correctness

We do not guarantee the physics validity of any of these results, but instead, show them as examples of what is already possible with a single prompt.

Analysis	Initial prompt (abridged)	Data	Time	Ref.
$H \to \tau\tau$	Measure the Higgs boson signal strength in the $\mu\tau_h$ final state at $\sqrt{s} = 8$ TeV	CMS	6h 30m	A/Github
Z lineshape + α_s	Measure Z lineshape parameters (M_Z, Γ_Z, σ^0_{had}), N_ν from invisible width, and extract $\alpha_s(M_Z)$ from R_ℓ	ALEPH	4h 58m	B/Github
Lund plane	Measure the primary Lund jet plane density in hadronic Z decays	ALEPH	7h 14m	C/Github
R_b, R_c, and A^b_{FB}	Measure R_b, R_c, and A^b_{FB} using signed impact parameter b-tagging and jet charge	ALEPH	4h 06m	D/Github
Energy-energy correlators	Measure the two-point EEC, its asymmetry (AEEC), and mean charged multiplicity in hadronic Z decays	ALEPH	3h 19m	E/Github
N_ν from Γ_{inv}	Measure the number of light neutrino generations from the Z invisible width	DELPHI	5h 15m	F/ Github
Lund jet plane	Measure the primary Lund jet plane density in hadronic Z decays	DELPHI	6h 59m	G/Github
Energy-energy correlators	Measure the two-point energy-energy correlator in hadronic Z decays	DELPHI	5h 23m	H/Github

Generative AI

https://iml-wg.github.io/HEPML-LivingReview/

Formal Theory and ML

- Theory and physics for ML
- ML for theory

Experimental results.

This section is incomplete as there are many results that directly and indirec techniques. We will try to highlight experimental results that use deep learnir

- Performance studies
- Searches and measurements where ML reconstruction is a core component
- Final analysis discriminate for searches
- Measurements using deep learning directly (not through object reconstruction)

Classification

- Parameterized classifiers
- Representations
- Targets
- Learning strategies
- Fast inference / deployment

Generative models / density estimation

- GANs
- (Variational) Autoencoders
- (Continuous) Normalizing flows
- Diffusion Models
- Transformer Models
- Physics-inspired
- Mixture Models
- Phase space generation
- Gaussian processes
- Other/hybrid

Simulation-based (`likelihood-f

- Parameter estimation
- Unfolding
- Domain adaptation
- BSM
- Differentiable Simulation

Uncertainty Quantification

- Interpretability
- Estimation
- Mitigation
- Uncertainty- and inference-aware learning

Regression

- Pileup
- Calibration
- Recasting
- Matrix elements
- Parameter estimation
- Parton Distribution Functions (and related)
- Lattice Gauge Theory
- Function Approximation
- Symbolic Regression
- Monitoring

Generative AI

GitHub Copilot (… Gemini code assist, Q Developer, JetBrains AI, Cody)

- **Vibe coding** - you express the **what** and let AI handle most of the **how**.
- Reviews your code
- Produces documentation
- Creates issues
- Creates PRs for existing issues
- Assisted code review

AI Agents:

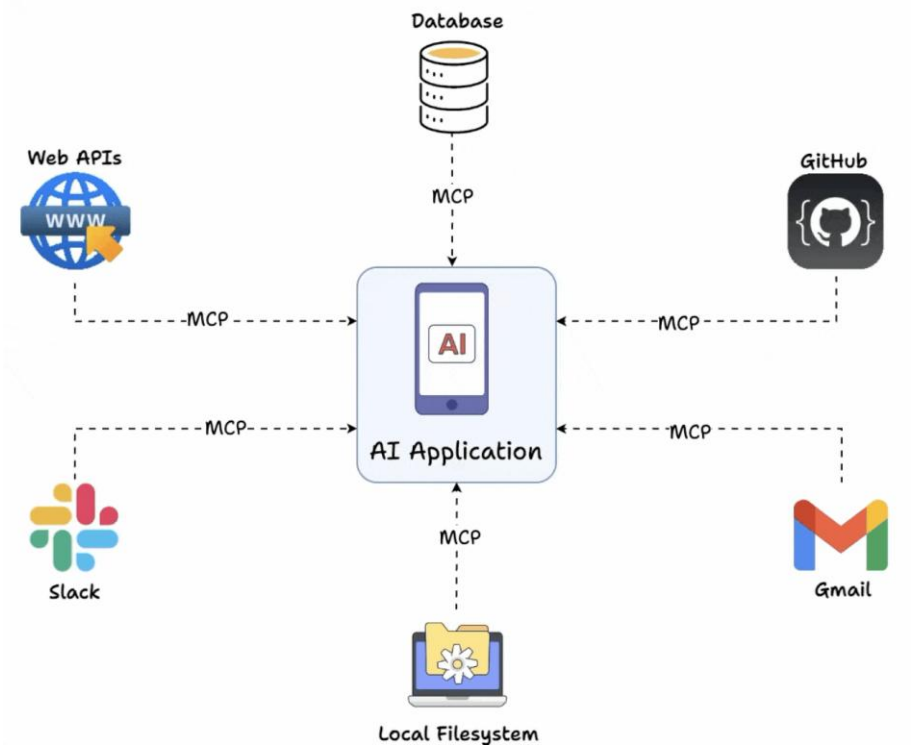
- Reinforcement Learning fine-tuned LLMs
- Can access web - no cutoff dates
- Create and execute code - it is actually nicer language for a lot of tasks

- OpenAI, DeepSearch, WebBrowse, Operator, Perplexity
- LangChain (+LangGraph), Microsoft AutoGen, CrewAI, Manus, Google ADK, WebArena, OSWorld, CodeAct
- AI using the whole computer; Manus and OpenManus

Generative AI: Model Context Protocol

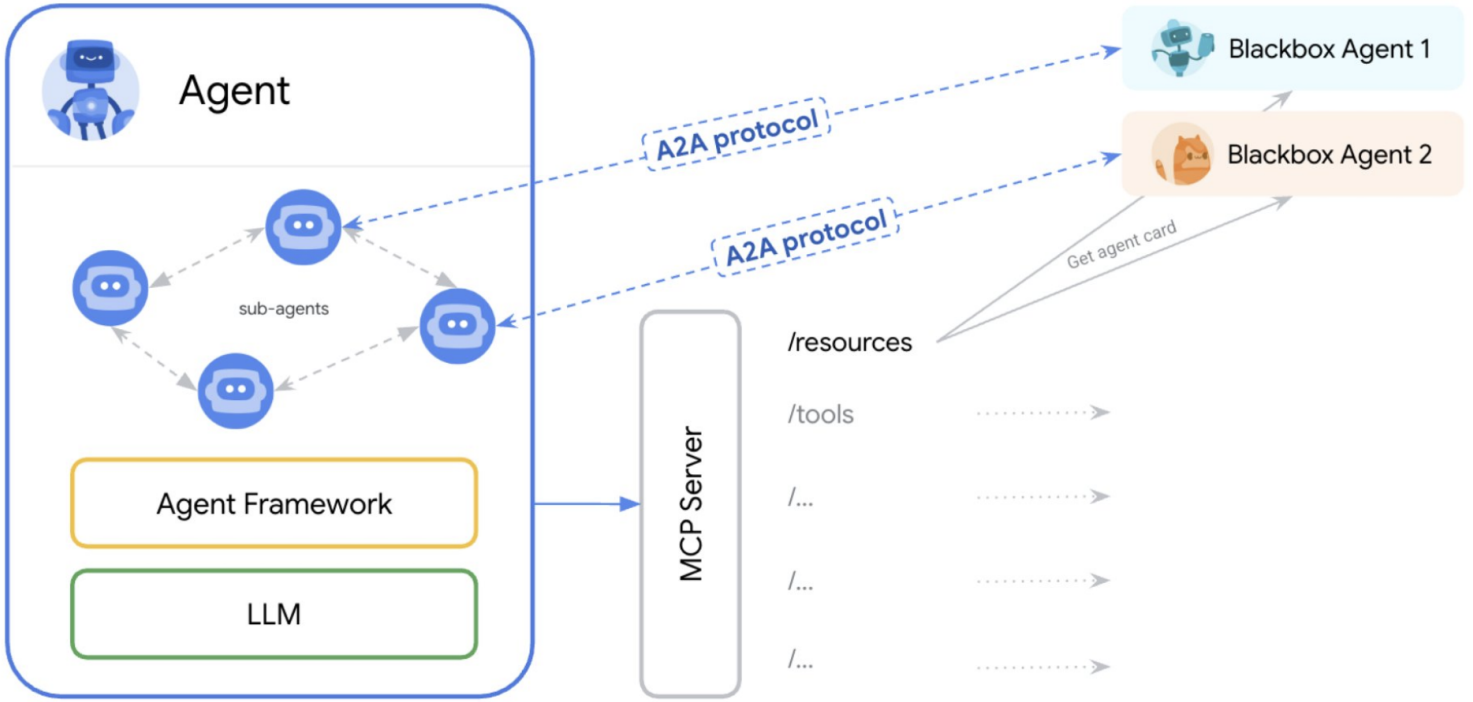
MCP is an open standard that connects Agents to data sources.

- Widely supported: MCP Servers catalogue
- TypeScript, Python, C#, and other MCP SDKs already exist.
- Stdio, SSE or streamable HTTP transport.



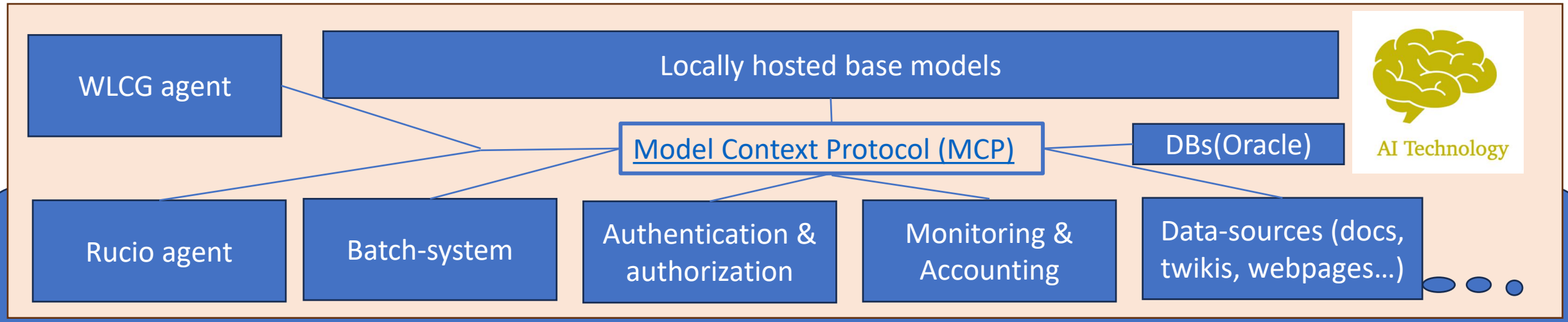
A2A

Agent-to-Agent communication protocol.

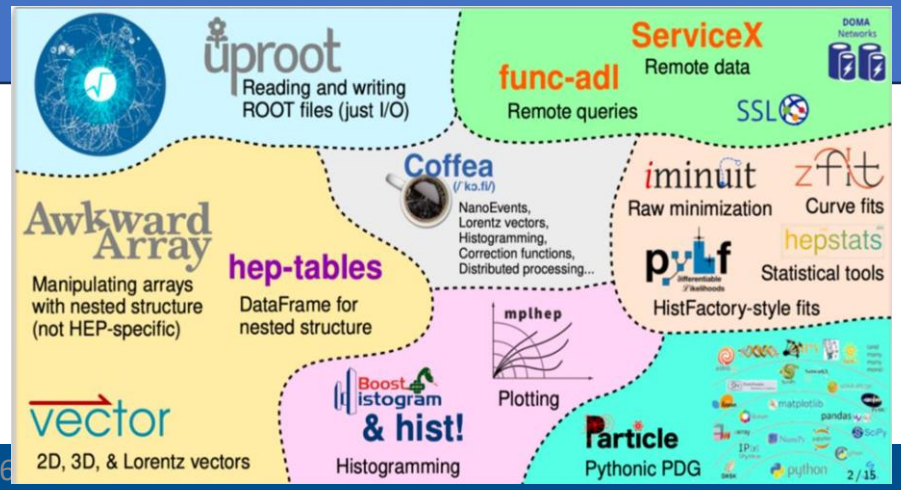


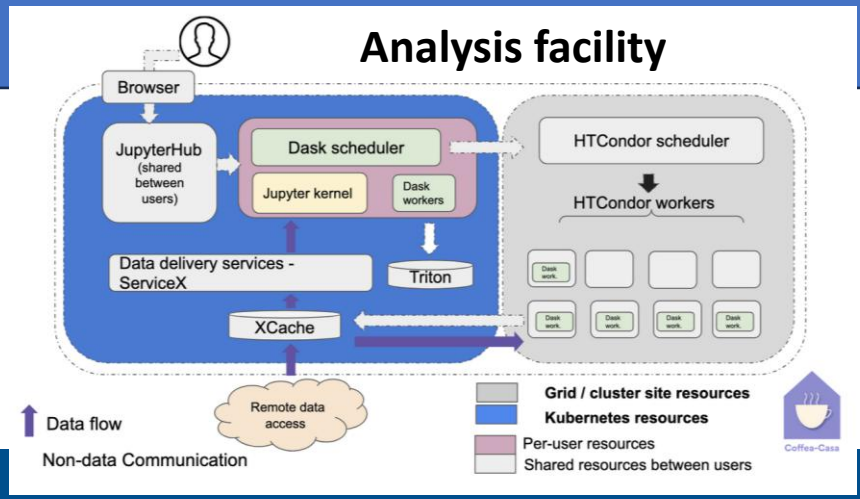
Hay AI, find me HH→ $bb\bar{b}\bar{b}$ from full ATLAS Run2 and Run3 data set along with relevant MC samples with $b\bar{b}$ triggers, use GridPP hosted latest baseline open model with WLCG agent spanning Rucio and gridpp MCPs, for authentication, use my grid token forauthentication, all the code and scritpts you generate, push it to my gitlab HH project and write all the output files to my grid-site-storage, don't worry about background in the first round, just filter out all the events from the data with possible signals

Local Agent Auto



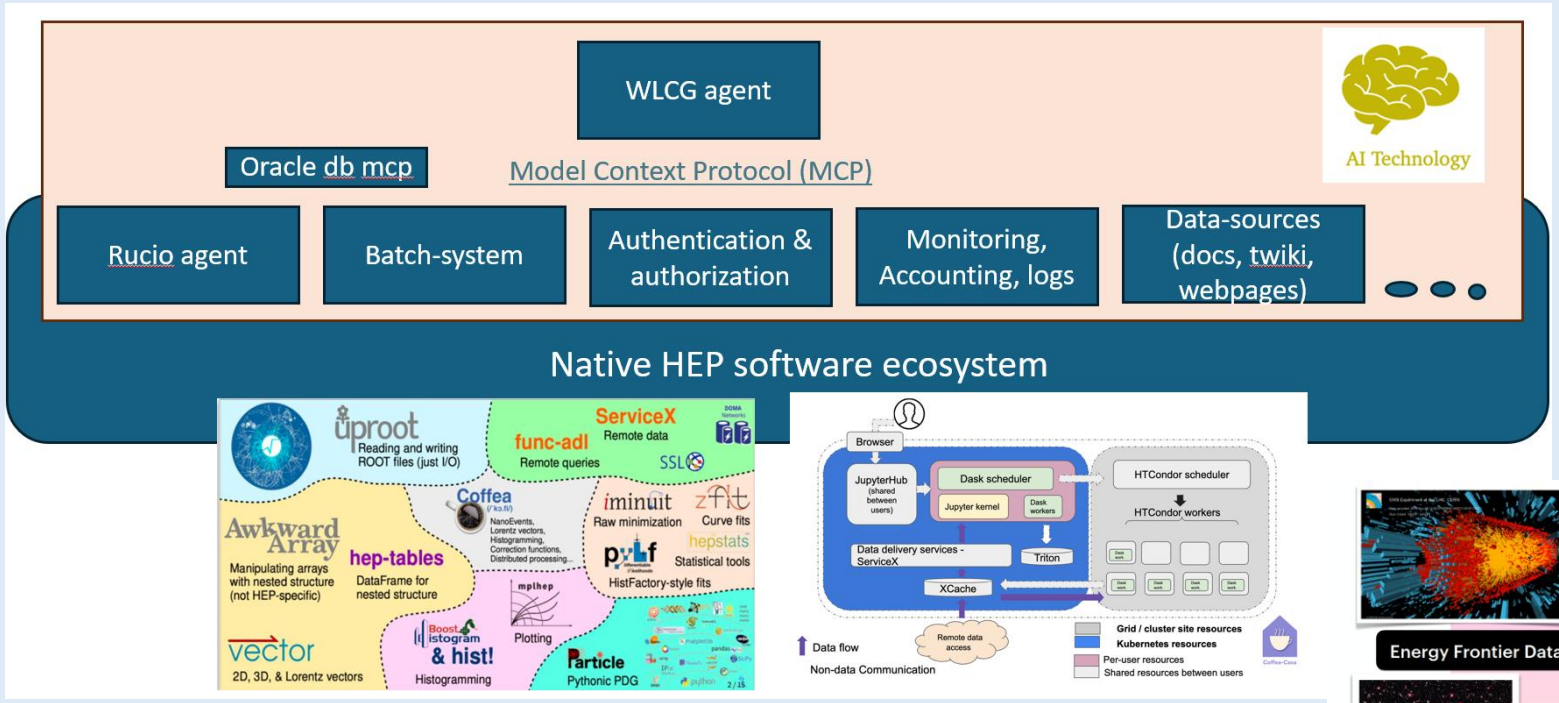
Native HEP software ecosystem

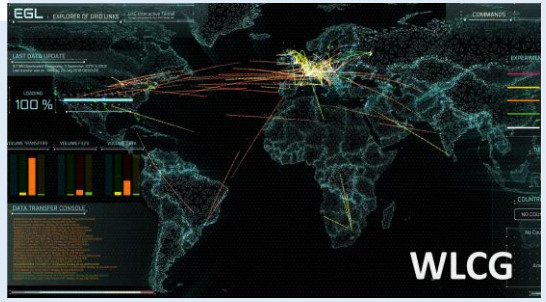


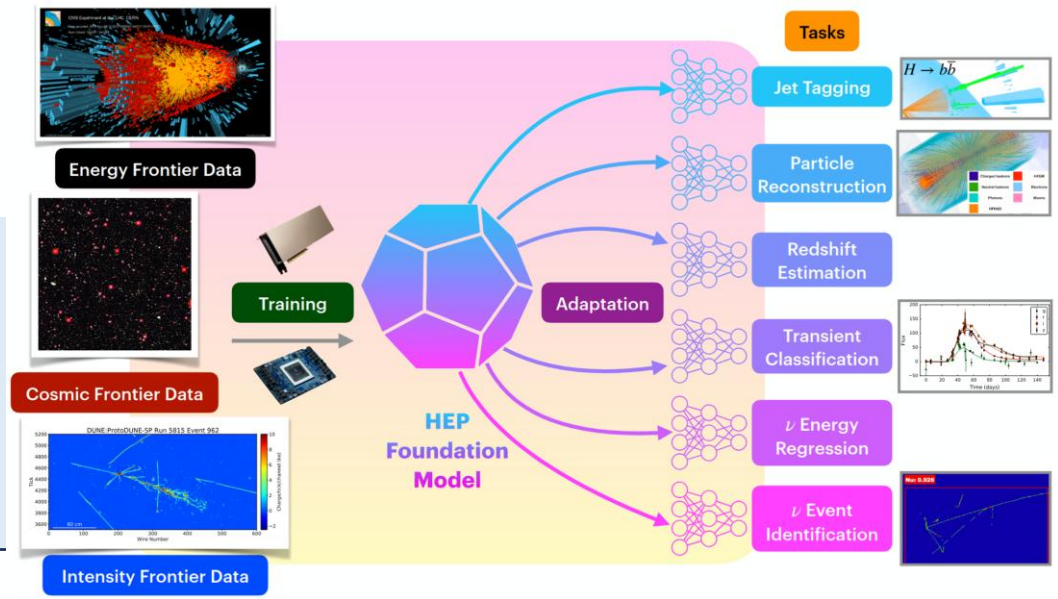


Hay AI, find me HH→ $bb\bar{b}\bar{b}$ from full ATLAS Run2 and Run3 data set along with relevant MC samples with $b\bar{b}$ triggers, use GridPP hosted latest baseline open model with WLCG agent spanning Rucio and gridpp MCPs, for authentication, use my grid token forauthentication, all the code and scritpts you generate, push it to my gitlab HH project and write all the output files to my grid-site-storage, don't worry about background in the first round, just filter out all the events from the data with possible signals

Local Agent Auto







Building an AI-native Research Ecosystem for Experimental Particle Physics: A Community Vision, https://arxiv.org/pdf/2602.17582

High Energy Physics - Experiment

[Submitted on 19 Feb 2026]

Building an AI-native Research Ecosystem for Experimental Particle Physics: A Community Vision

Thea Klaeboe Aarrestad, Alaa Abdelhamid, Haider Abidi, Jahred Adelman, Jennifer Adelman-McCarthy, Shuchin Aeron, Mohamed Aly, Oz Amram, Saeed Ansari Fard, Aram Apyan, John Arrington, Marvin Ascencio-Sosa, Mohammad Atif, Ar Rainer Bartoldus, Amit Bashyal, Aashwin Basnet, Ayse Bat, Lothar A. T. Bauerdick, John Beacom, Chris Bee, Michael E Benelli, Douglas Benjamin, Catrin Bernius, Binod Bhandari, Avinay Bhat, Meghna Bhattacharya, Saptaparna Bhattacha Burak Bilki, Mary Bishai, Kevin Black, Kenneth Bloom, Brian Bockelman, Johan Sebastian Bonilla Castro, Tulika Bose, Gustaaf Brooijmans, Elizabeth Brost, Maria Brigida Brunetti, Quentin Buat, Brendon Bullard, Jackson Burzynski, Paolo Castillo Fernandez, Fabio Catalano, Viviana Cavaliere, Flavio Cavanna, Giuseppe Cerati, Aidan Chambers, Maria Char Sergei Chekanov, Jian-ping Chen, Yi Chen, Zhengyang Chen, J. Taylor Childers, Hector Chinchay, Yuan-Tang Chou, Ta Lopes de Sa, Simon Corrodi, Kyle Cranmer, Matteo Cremonesi, Roy Cruz, Mate Csanad, Mariarosaria D'Alfonso, Carlo De, Patrick de Perio, Klaus Dehmelt, Marco Del Tutto, Carlos Ruben Dell'Aquila, Sarah Demers , Paolo Desiati, Bhesha Markus Diefenthaler, Jeff Dillon, Zelimir Djurcic, Caterina Doglioni, Francois Drielsma, Edmond Dukes, Irene Dutta, Pete Peter Fackeldey, Cristiano Fanelli, Hao Fang, Mattia Fani, Muhammad Farooq, Matthew Feickert, Ian Fisk, Sam Forem Gagnon, Massimiliano Galli, Abhijith Gandrakota, Sudeshna Ganguly, Arianna Garcia Caffaro, Rob Gardner, Rocky Bal Loukas Gouskos, Richard Gran, Heather Gray, Andrei Gritsan, Gaia Grosso, Craig Group, Jiawei Guo, Shubham Gupta Joseph Haley, Eva Halkiadakis, Francis Halzen, Michael Hance, Philip Harris, Harry Hausner, Karsten Heeger, Lukas H Herrmann, David Hertzog, Christian Herwig, Aaron Higuera, Alexander Himmel, Timothy Hobbs, Stefan Hoeche, Tova H Howard, Shih-Chieh Hsu, Fengping Hu, Patrick Huber, Dirk Hufnagel, Daniel Humphreys, Ia Iashvili, Joseph Incandela, Jarvis, Brij Kishor Jashal, Pratik Jawahar, Dulitha Jayakodige, Torri Jeske, Sergo Jindariani, Jay Hyun Jo, Bhishm Shan

Building an AI-native Research Ecosystem for Experimental Particle Physics: A Community Vision

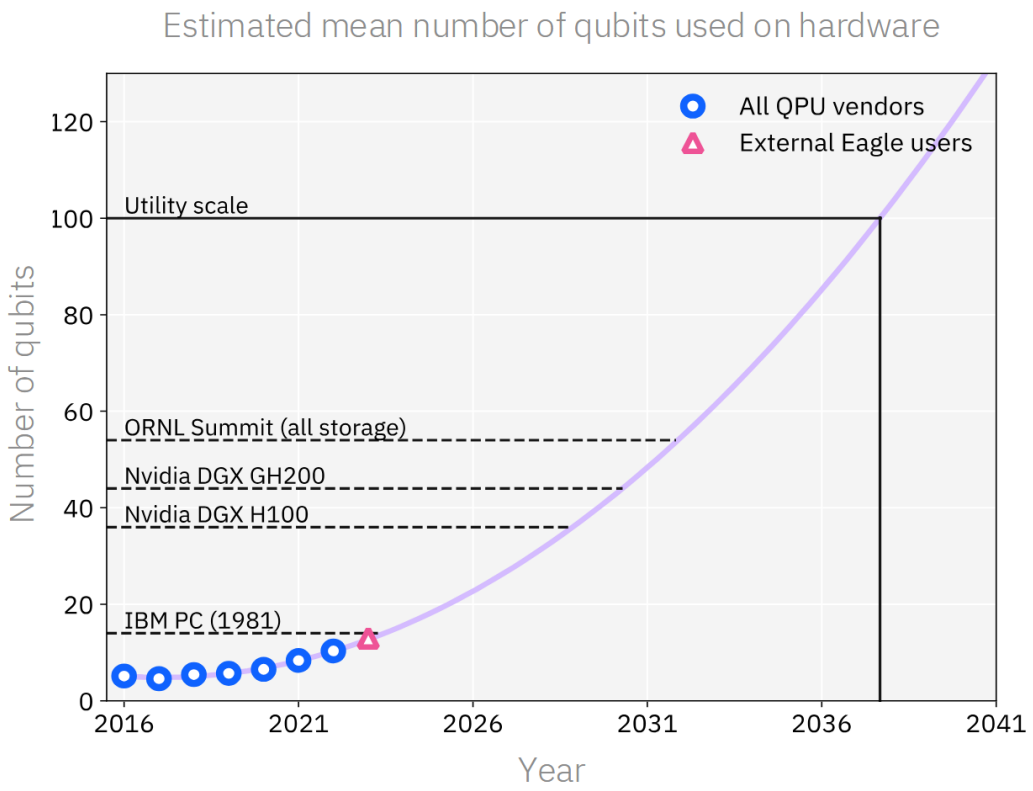
February 20, 2026 - Version 1.00

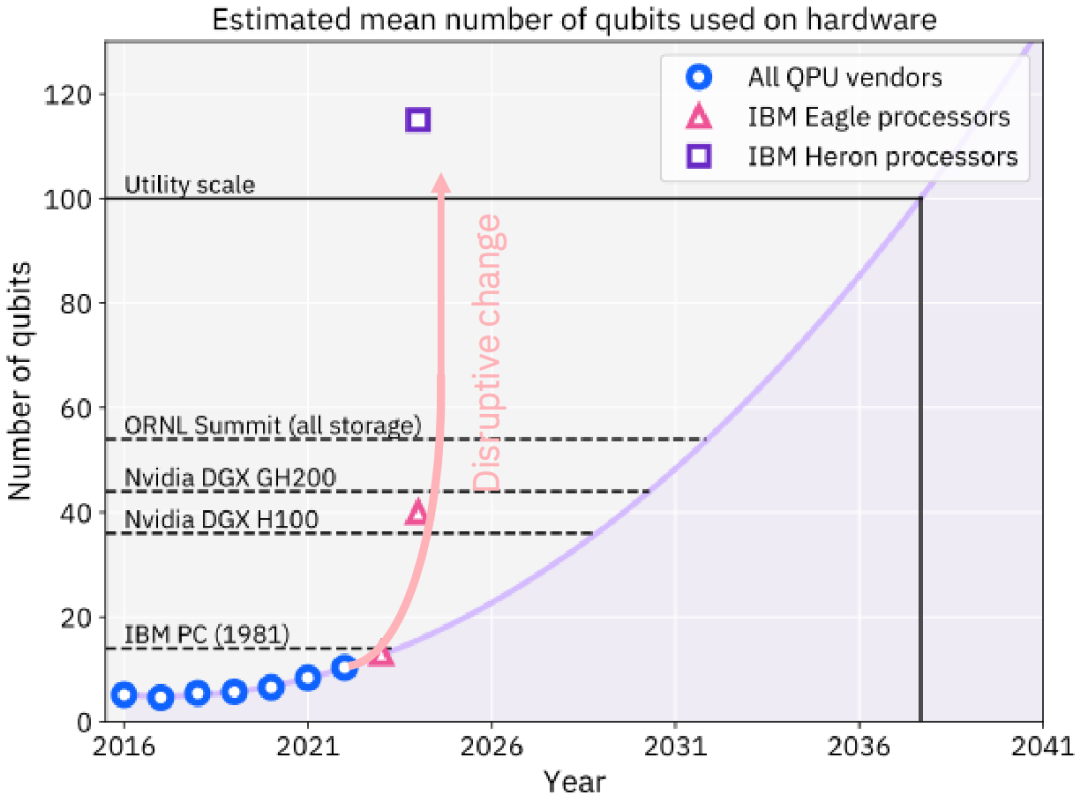
Abstract: Experimental particle physics seeks to understand the universe by probing its fundamental particles and forces and exploring how they govern the large-scale processes that shape cosmic evolution. This whitepaper presents a vision for how Artificial Intelligence (AI) can accelerate discovery in this field. We outline grand challenges that must be addressed to enable transformative breakthroughs and describe how current and planned experimental facilities can implement this vision to advance our understanding of the vast and complex physical world from the smallest to the largest scales. We show how facilities currently under construction, such as the HL-LHC, DUNE and soon EIC, can both benefit from and serve as proving grounds for this vision, while also enabling a longer-term goal for how future experiments— like FCC-ee at CERN, IceCube-Gen2, a Muon Collider in the U.S., and smaller to mid-scale projects—can be fully AI-native. We describe how a truly national-scale collaboration, jointly managed across large funding partners, and involving both DOE laboratories and universities, can make this happen.

Audience: The goal of this whitepaper is to highlight the emerging opportunities and existing gaps for the broader particle physics community, to inform funding agencies in the event new resources become available, and to highlight to policymakers the innovative contributions that experimental particle physics can bring to the field. This document is not intended as a comprehensive proposal for all necessary activities, nor as an exhaustive review of ongoing R&D.

Quantum

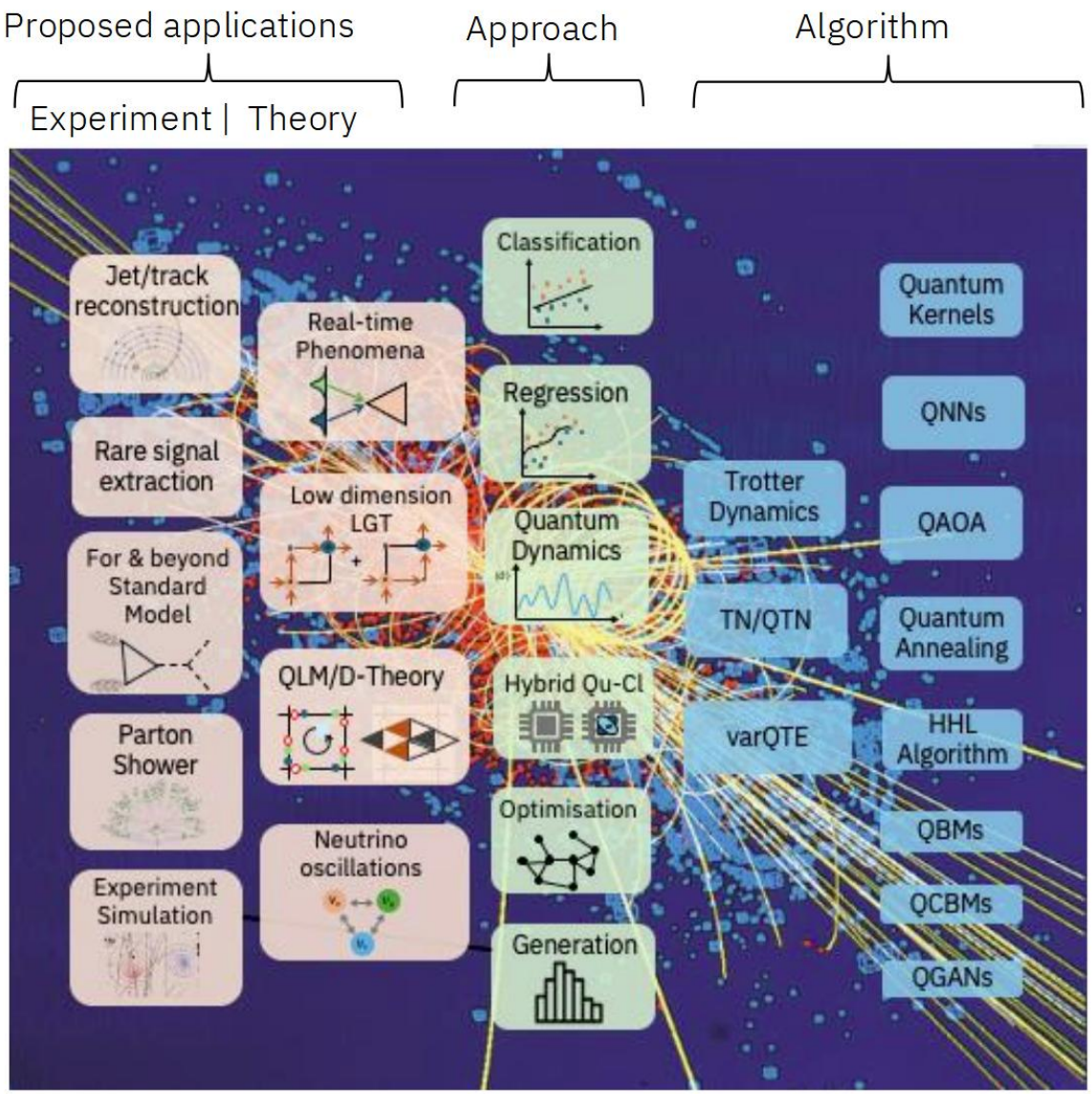
IBM first to put the first quantum computer on the cloud in 2016, quantum computing has largely been in an exploratory phase. Experiments validate the tenets of quantum computation, but do not push the field beyond the reach of classical compute. To move beyond simple experiments to demonstrate the utility of quantum computing in multiple domains,





With quantum systems composed of 100+ qubits, researchers are beginning to explore algorithms and applications at scales beyond brute-force classical computation using IBM Quantum systems

Quantum



The QC4HEP community paper

10.1103/PRXQuantum.5.037001

ROADMAP | OPEN ACCESS

PDF

Quantum Computing for High-Energy Physics: State of the Art and Challenges

Share

Alberto Di Meglio[1,*], Karl Jansen[2,3,†], Ivano Tavernelli[4,‡], Constantia Alexandrou[3,5], Srinivasan Arunachalam[6], Christian W. Bauer[7], Kerstin Borras[8,9], Stefano Carrazza[1,10], Arianna Crippa[2,11] *et al.*

Show more

PRX Quantum **5**, 037001 – **Published 5 August, 2024**

DOI: https://doi.org/10.1103/PRXQuantum.5.037001

Export Citation

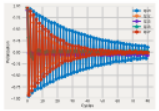
Am score 103 | Citations 76 | Show metrics

Abstract

Quantum computers offer an intriguing path for a paradigmatic change of computing in the natural sciences and beyond, with the potential for achieving a so-called quantum advantage—namely, a significant (in some cases exponential) speedup of numerical simulations. The rapid development of hardware devices with various realizations of qubits enables the execution of small-scale but representative applications on quantum computers. In particular, the high-energy physics community plays a pivotal role in accessing the power of quantum computing, since the field is a driving source for challenging computational problems. This concerns, on the theoretical side, the exploration of models that are very hard or even impossible to address with classical techniques and, on the experimental side, the enormous data challenge of newly emerging experiments, such as the upgrade of the Large Hadron Collider. In this Roadmap paper, led by CERN, DESY, and IBM, we provide the status of high-energy physics quantum computations and give examples of theoretical and experimental target benchmark applications, which can be addressed in the near future. Having in mind hardware with about 100 qubits capable of executing several thousand two-qubit gates, where possible, we also provide resource estimates for the examples given using error-mitigated quantum computing. The ultimate declared goal of this task force is therefore to trigger further research in the high-energy physics community to develop interesting use cases for demonstrations on near-term quantum computers.

Quantum

IBM Quantum CHEP24

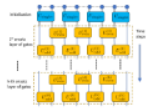


Characterizing quantum processors using discrete time crystals
arXiv:2301.07625
80 qubits / 7900 CX gates

materials

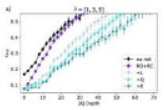
Evidence for the utility of quantum computing before fault tolerance
Nature, 618, 500 (2023)
127 qubits / 2880 CX gates

spin models



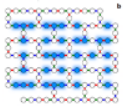
Simulating large-size quantum spin chains on cloud-based superconducting quantum computers
Phys. Rev. Research 5, 013183 (2023)
102 qubits / 3186 CX gates

spin models



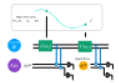
Best practices for quantum error mitigation with digital zero-noise extrapolation
arXiv:2307.05203
104 qubits / 3605 ECR gates

tools



Uncovering Local Integrability in Quantum Many-Body Dynamics
arXiv:2307.07552
124 qubits / 2641 CX gates

materials

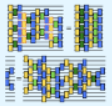


Quantum reservoir computing with repeated Measurements on superconducting devices
arXiv:2310.06706
120 qubits / 49470 gates + meas.

materials

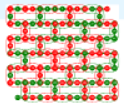
Realizing the Nishimori transition across the error threshold for constant-depth quantum circuits
arXiv:2309.02863
125 qubits / 429 gates + meas.

spin models



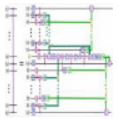
Scalable Circuits for Preparing Ground States on Digital Quantum Computers: The Schwinger Model Vacuum on 100 Qubits
PRX Quantum 5, 020315 (2024)
100 qubits / 788 CX gates

High energy physics



Scaling Whole-Chip QAOA for Higher-Order Ising Spin Glass Models on Heavy-Hex Graphs
arXiv:2312.00997
127 qubits / 420 CX gates

optimization

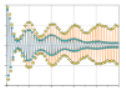


Efficient Long-Range Entanglement using Dynamic Circuits
arXiv:2308.13065
101 qubits / 504 gates + meas

tools

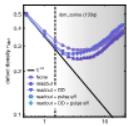
Quantum Simulations of Hadron Dynamics in the Schwinger Model using 112 Qubits
arXiv:2401.08044
112 qubits / 13,858 gates

High energy physics



Unveiling clean two-dimensional discrete time quasicrystals on a digital quantum computer
arXiv:2403.16718
133 qubits / 15,000 CZ gates

materials

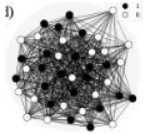


Benchmarking digital quantum simulations and optimization above hundreds of qubits using quantum critical dynamics
arXiv:2404.08053
133 qubits / 1440 CX gates

spin models

Chemistry Beyond Exact Solutions on a Quantum-Centric Supercomputer
arXiv:2405.05068
77 qubits / 3590 CZ gates

chemistry



Towards a universal QAOA protocol: Evidence of quantum advantage in solving combinatorial optimization problems
arXiv:2405.09169
109 qubits / 21,200 gates

optimization

Thank you