Acquisition des données et Online

Christina Agapopoulou (IJCLab/CNRS)

École thématique d’instrumentation de l’IN2P3: “Du détecteur à la mesure”
Roscoff, 23-28 Novembre 2025

Plan

- Mercredi:
  - Introduction et concepts de base
  - Acquisition des données (Front-end et Back-end)
  - Trigger Hardware
  - Readout et ferme des calculs
  - Distribution d’horologe
  - Control et monitoring

- Aujourd'hui:
  - Un paradigm different: sans trigger hardware
  - Architectures heterogenes
  - Analyse en temps réel
  - Tendances nouvelles et futures: IA et al.

Pourquoi vivre sans *hardware* trigger?

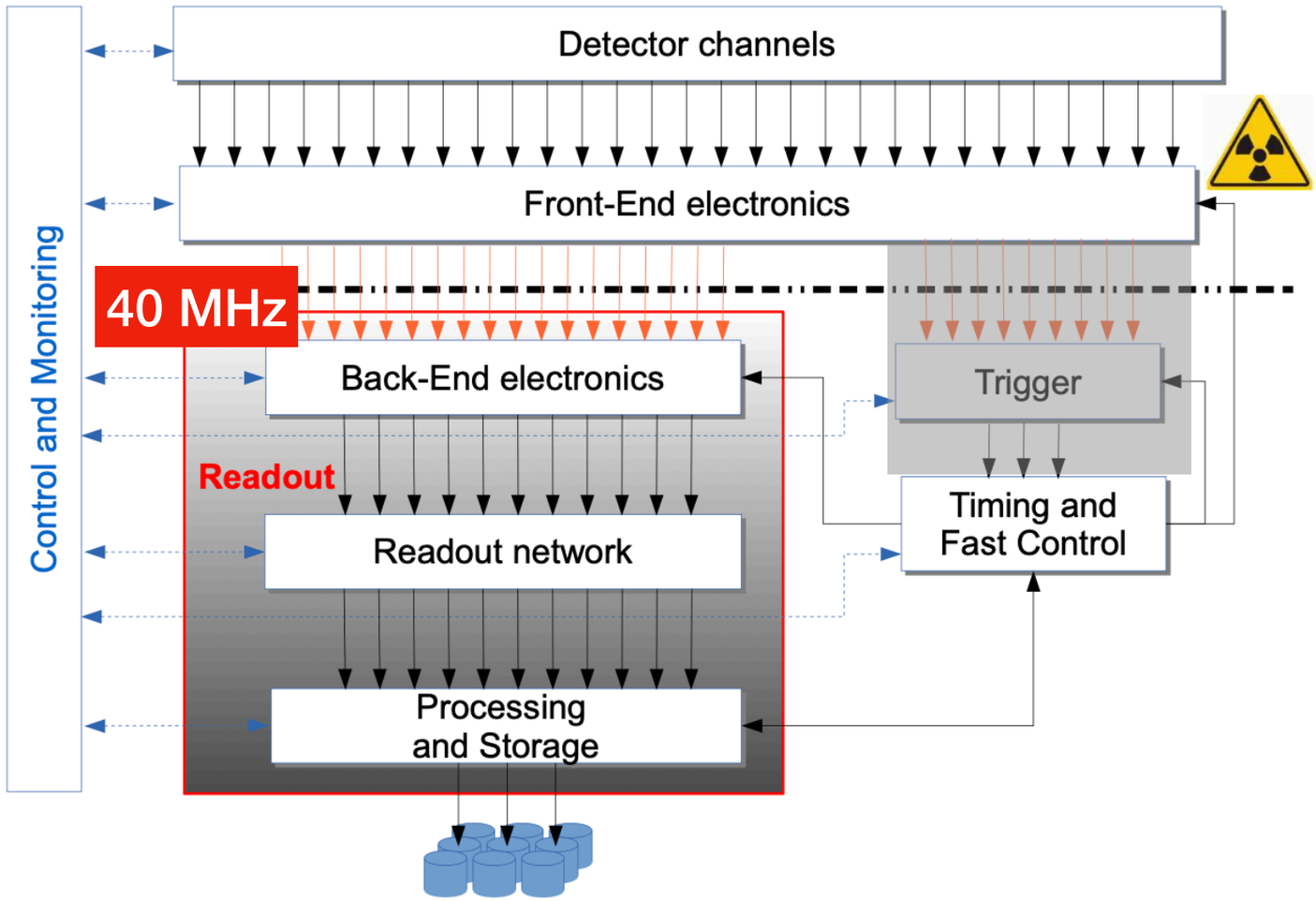
Peut-être il n'y a pas besoin!

- Des experiences qui cherchent des interactions rares, de particules inconnues avec un bruit du fond très bas

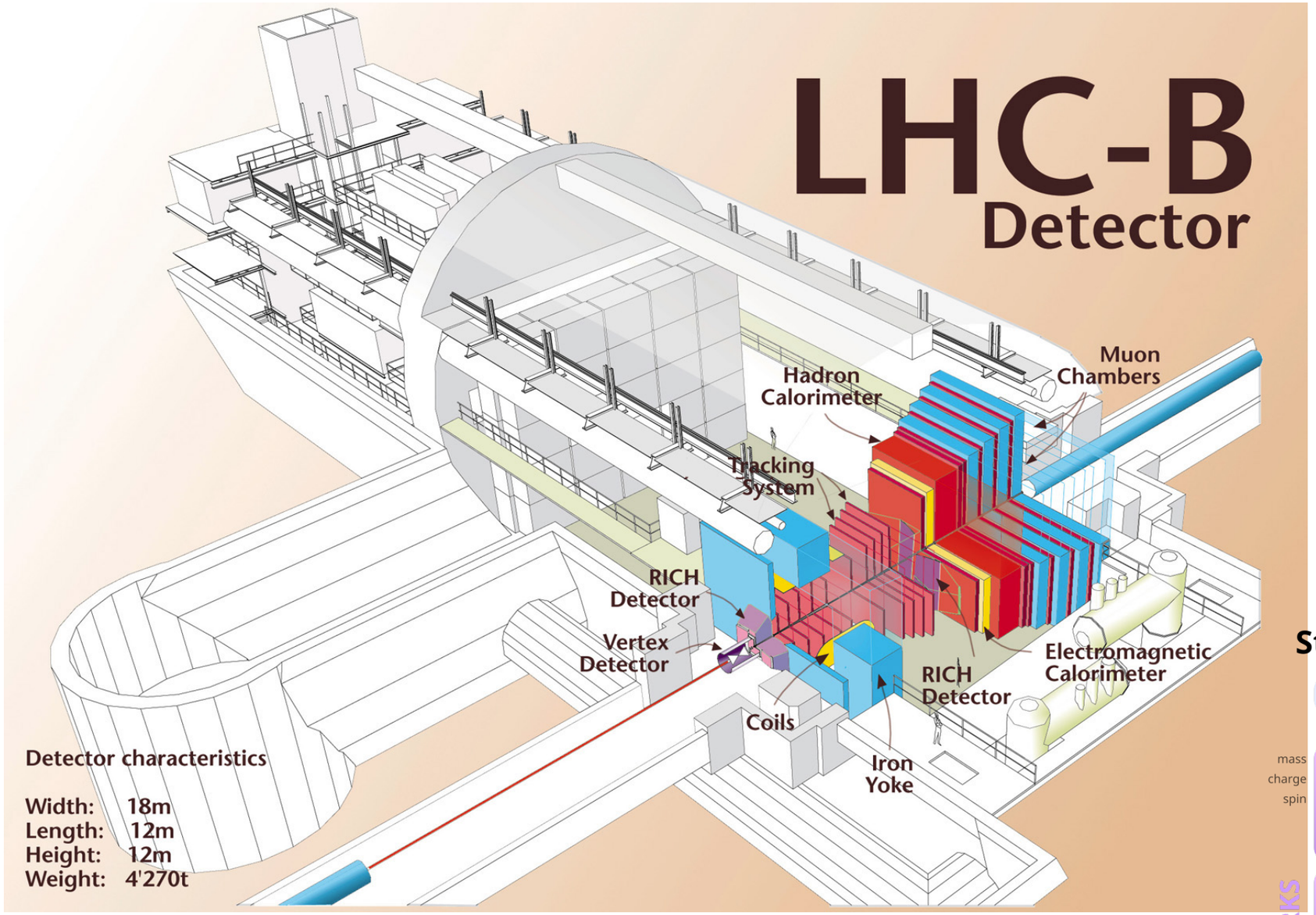
Ou peut-être le trigger hardware n'est pas assez performant → l**e cas qu'on va étudier aujourd'hui**

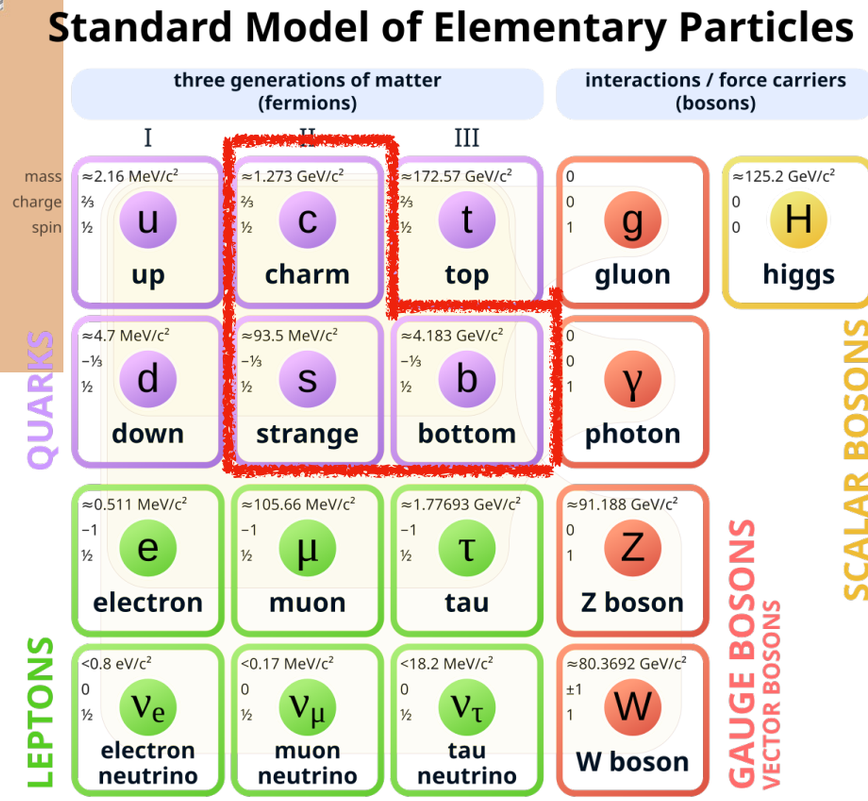
C'est un cas extreme: enlever le trigger hardware dans une experience LHC

- Ca veut dire qu'on doit faire le read-out de tout le détecteur à la fréquence des collisions - 40 MHz



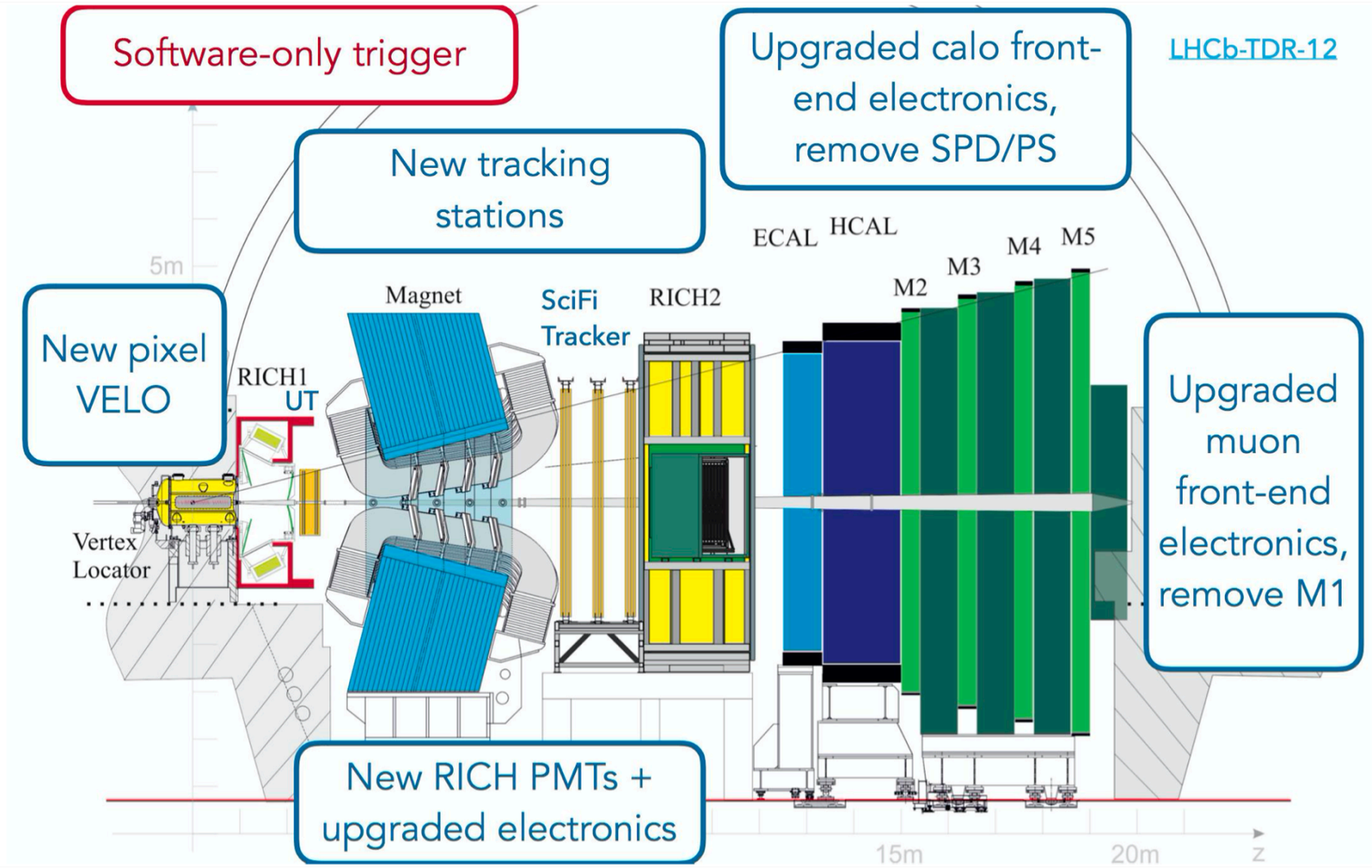
Le détecteur LHCb a CERN





Une experience en couverture de la region de l’avant - spécialisée a l’etude des quarks b et c

La mise a jour de LHCb pour Run 3

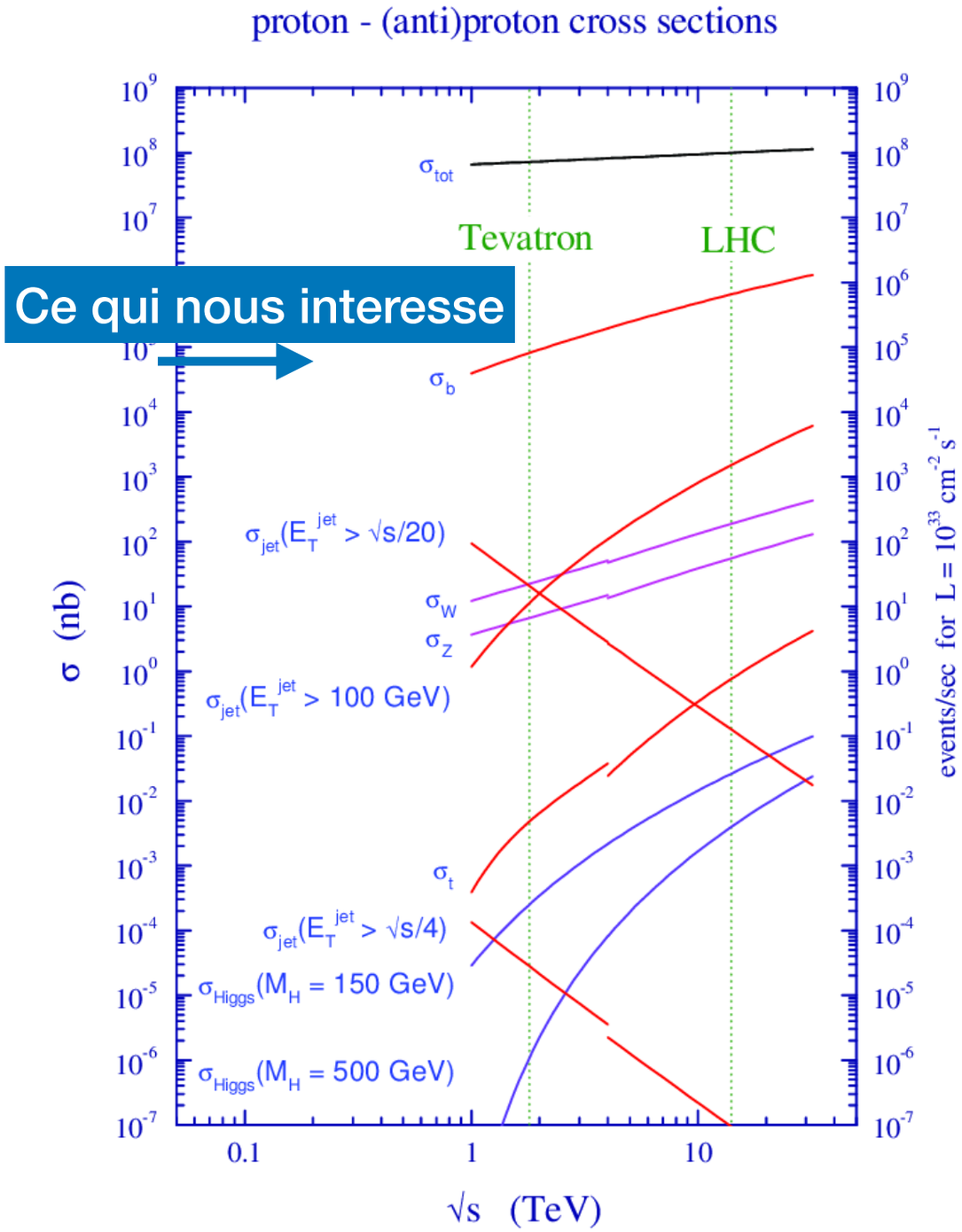


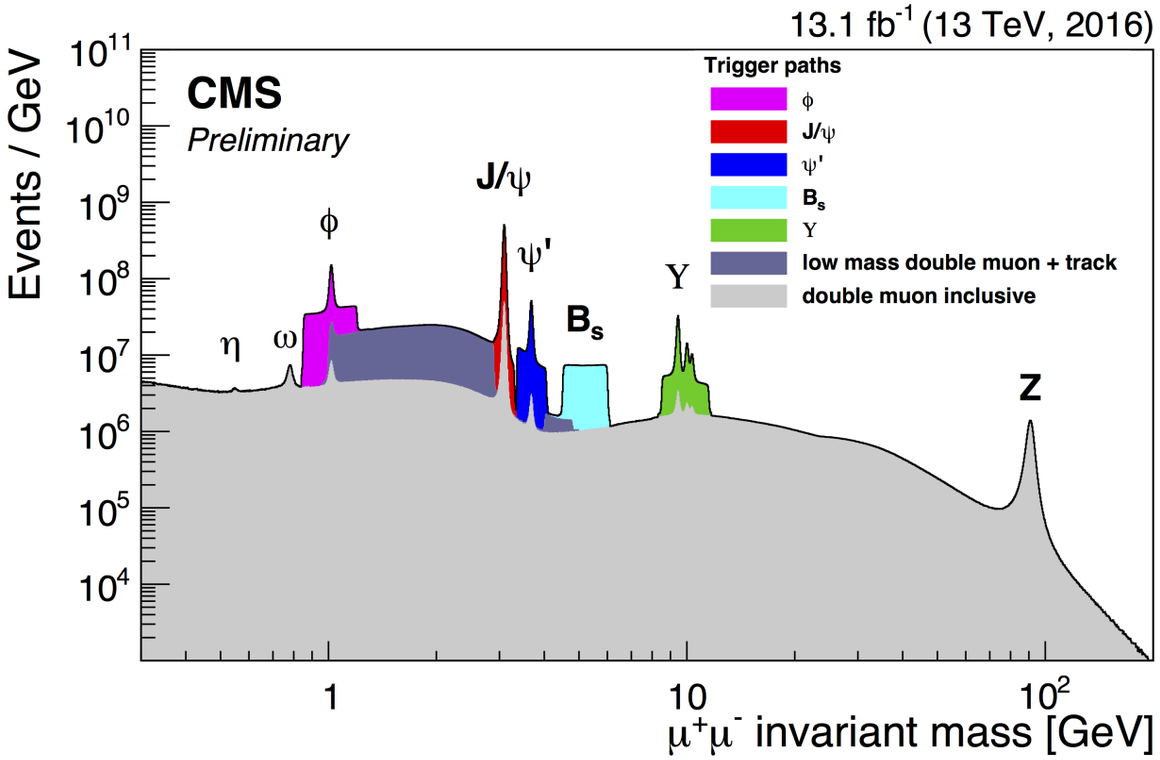
Entre Run 2 et Run 3 LHCb a été mis en jour

- Luminosité instantanée augmentée par un facteur 5
- Nouveau système de trajectographie, mis a jour des électroniques de tous détecteurs
- LHCb a enlevé son système de trigger hardware → système purement software

Pourquoi?

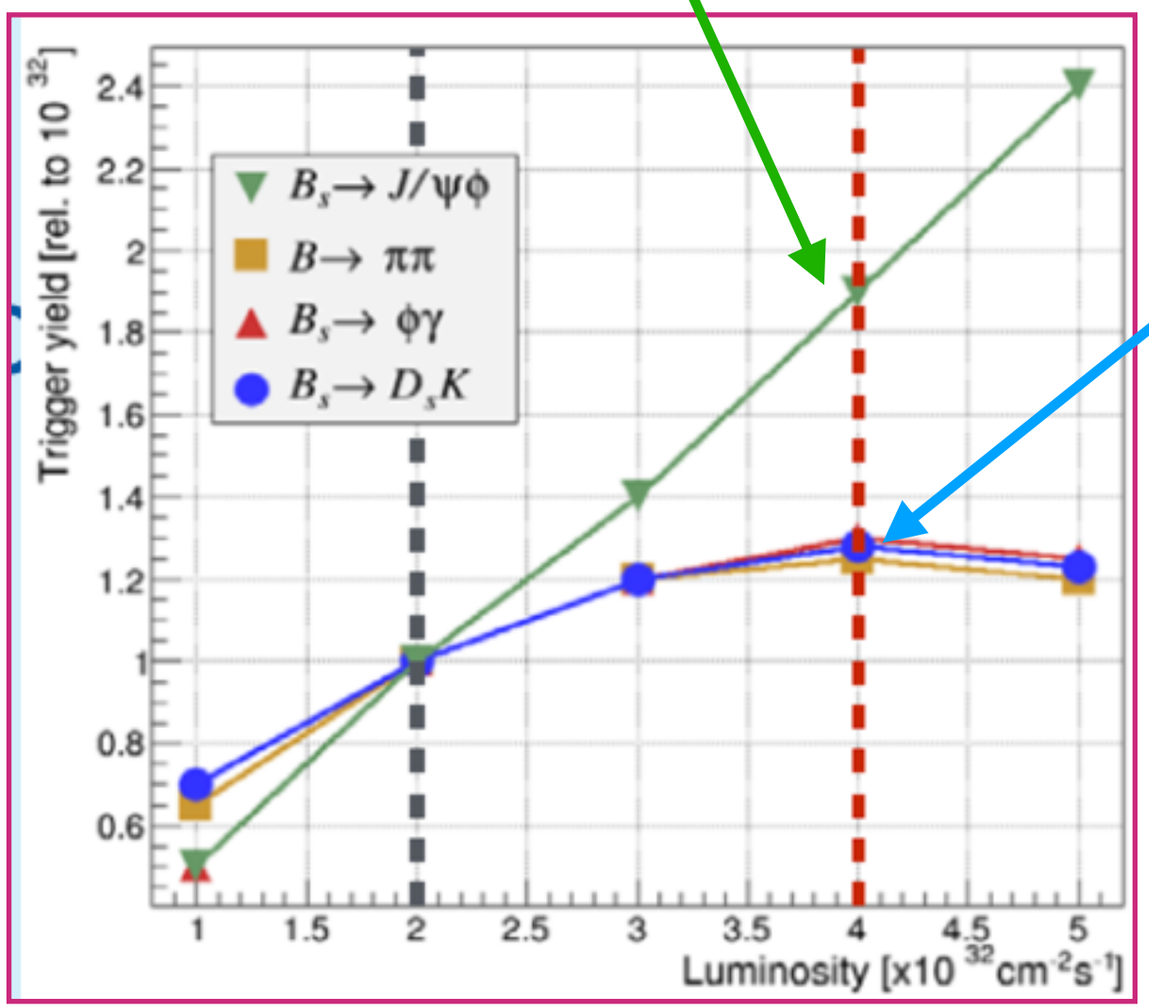
- Production de quarks b assez élevée dans LHC (en comparant avec le Higgs)
- m(b) ~ 4 GeV/c^2 (~0.01 le m(Higgs))
  - Signatures aux energies beaucoup plus faibles que la physique du Higgs
  - Souvent signatures qui ne contiennent pas un muon/objet calo a haute énergie!





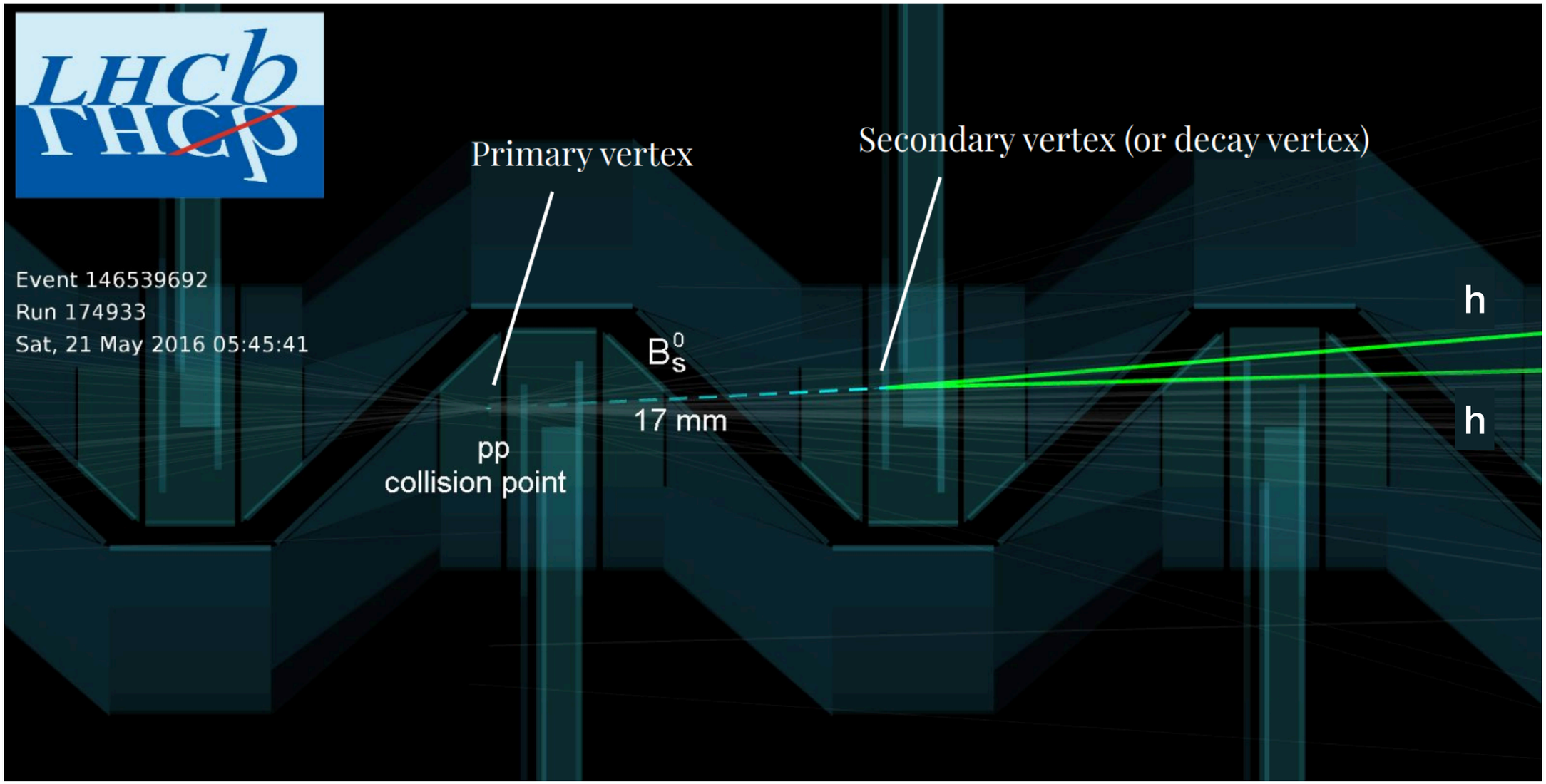
Si on gardait le trigger hardware

Signatures avec muons toujours ok



Saturation du trigger hardware pour les désintégrations B sans muons

Le meilleur façon de sélectionner une désintégration B

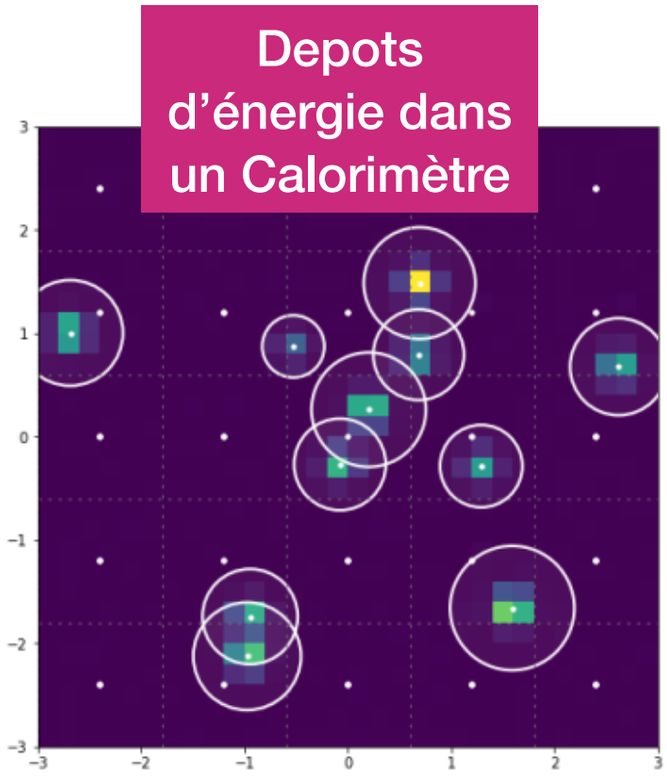


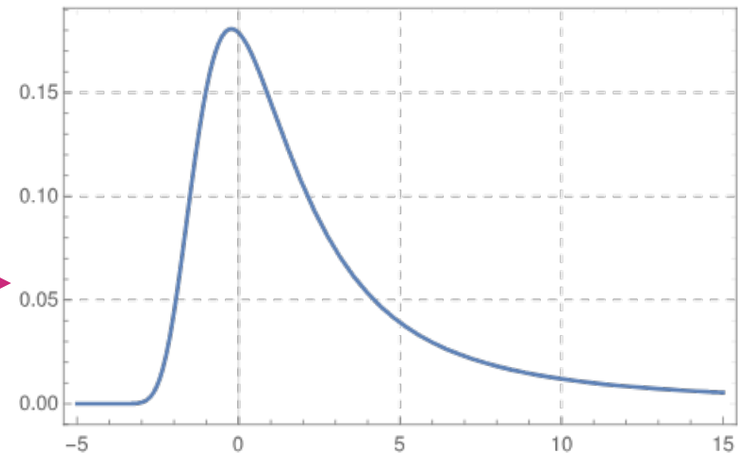
Un système trigger qui exploite l'information de tout l'évènement

Et en particulier, la reconstruction des traces et vertices

Où: trigger software après l'Event Builder

Le nouveau schema





Chemin DAQ:

- Quelle était l'énergie déposée dans ma voie?
- Digitisation doit être precise, pas possible de récupérer l'info analogique!
- Exigence de transmission: ~~la fréquence du trigger~~ la fréquence de collisions (40 MHz à LHC)

DAQ

Trigger

Chemin Trigger:

- Avait-il du signal interessant dans ma voie?
- Mesure doit être très vite!
- Exigence de transmission: la fréquence de collisions (40 MHz à LHC)

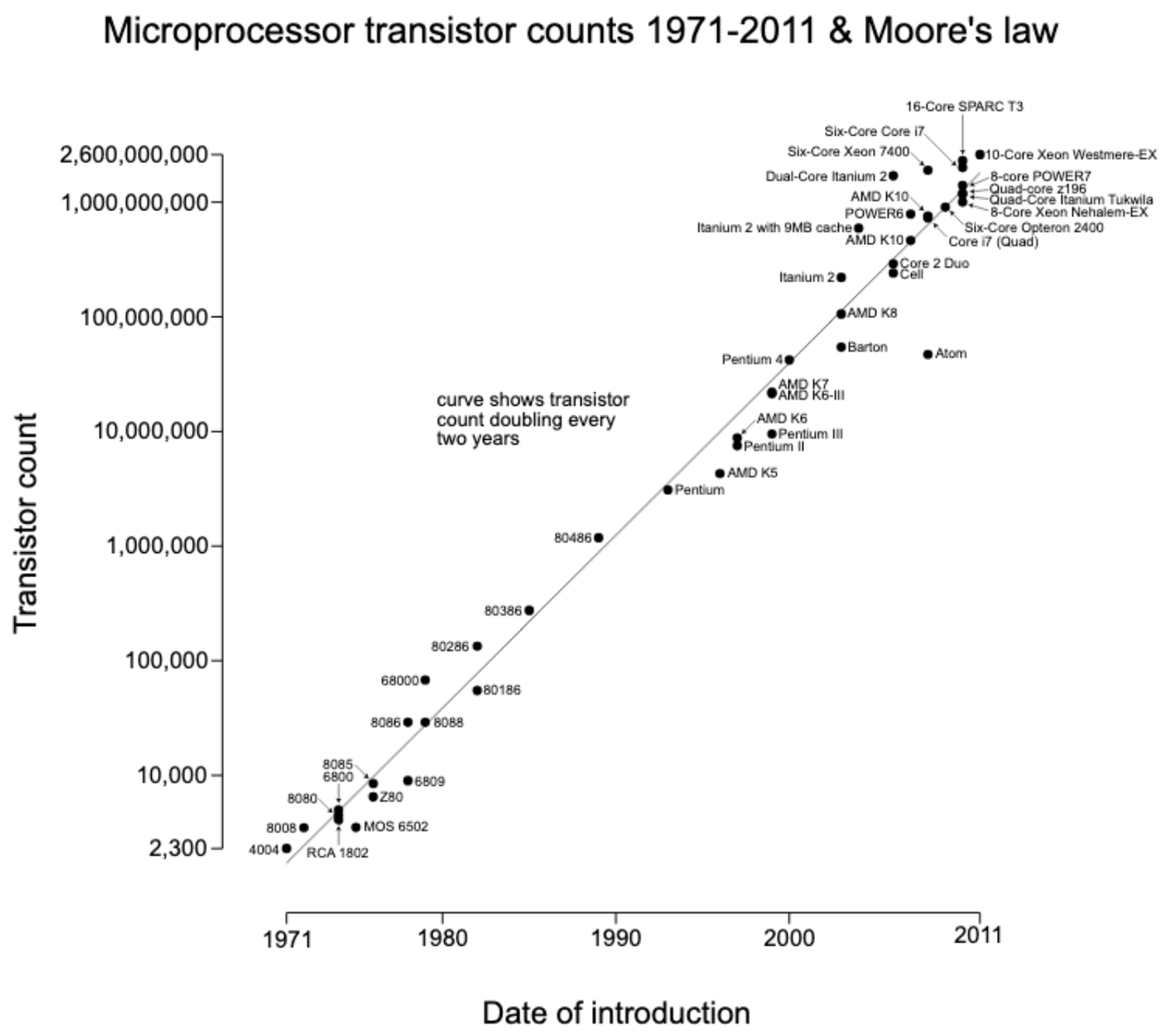
Quelques ingredients necessaries

Pouvoir transporter les données:

- Vitesse des protocols des liens optiques a LHC augmente (ex. GBT -> lpGBT)
- La geometrie de LHCb est particulierement favorable: un détecteur pas hermétique, plus d'espace pour sortir les cables
- Vitesse d'ethernet a un trend très bon les dernières années

Quelques ingredients necessaries

Et les traiter… on pensait que c'était ok

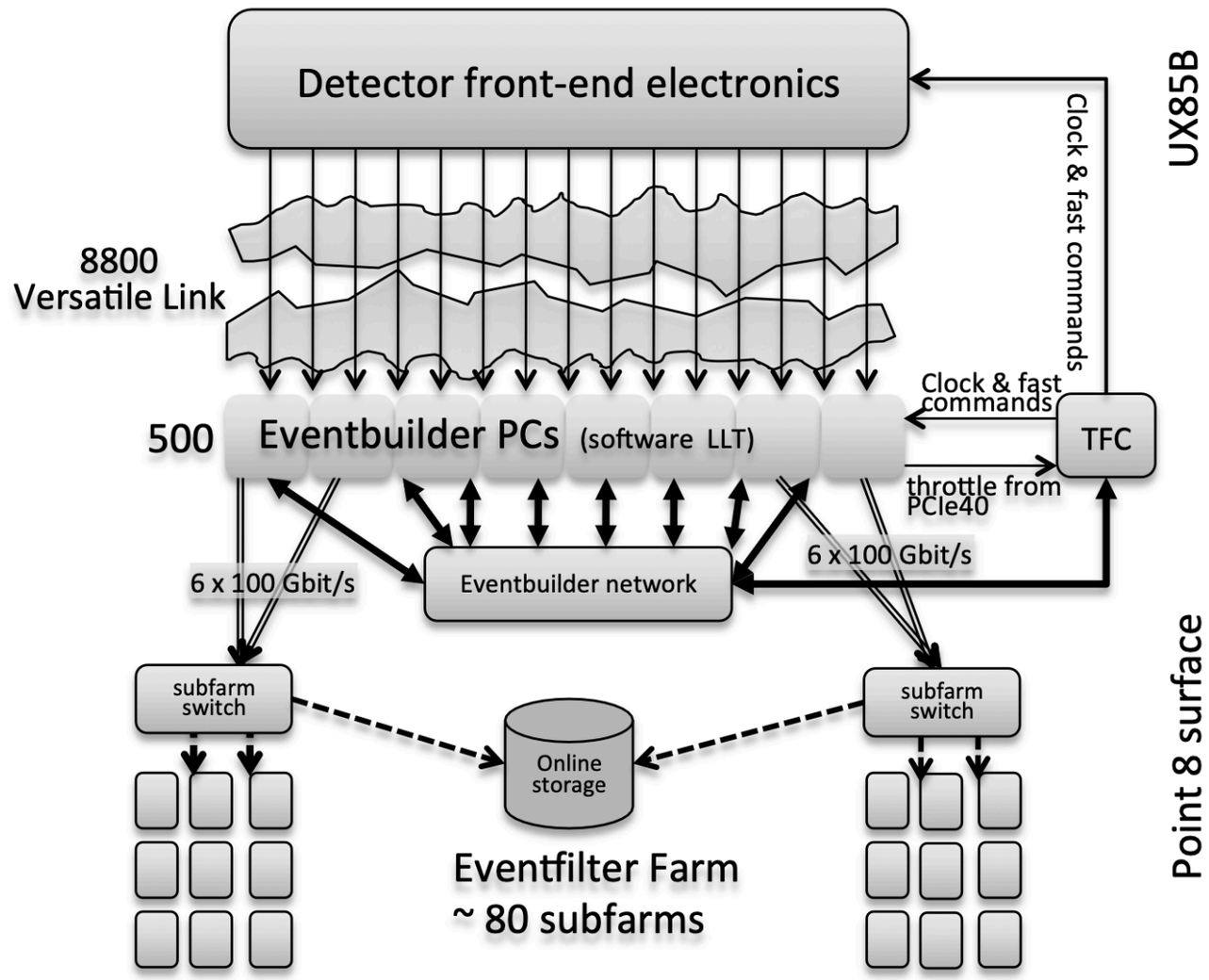


Nouveau schema de readout

Prioritisation des operations Zero-suppression / compression, pour diminuer le n. des liens

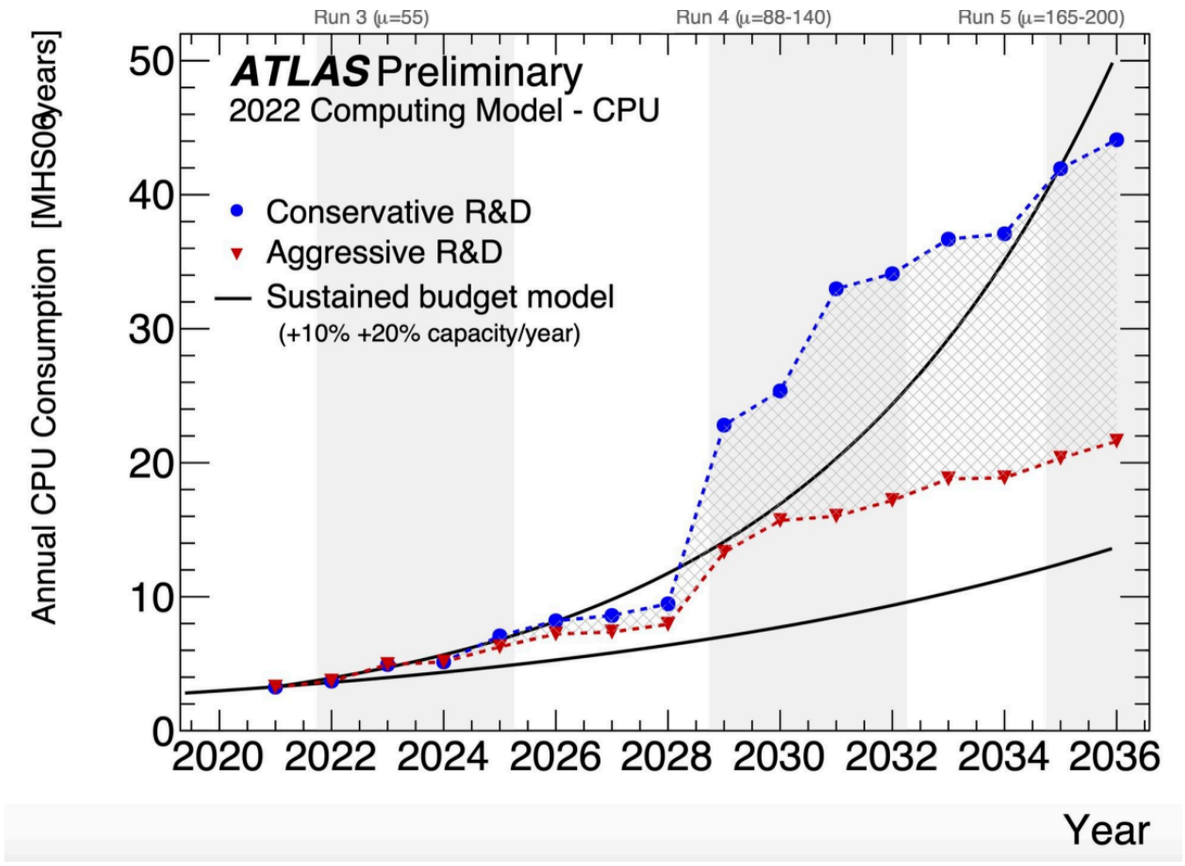
Event Builder dans la surface (liens de 300 m) - design en style data center:

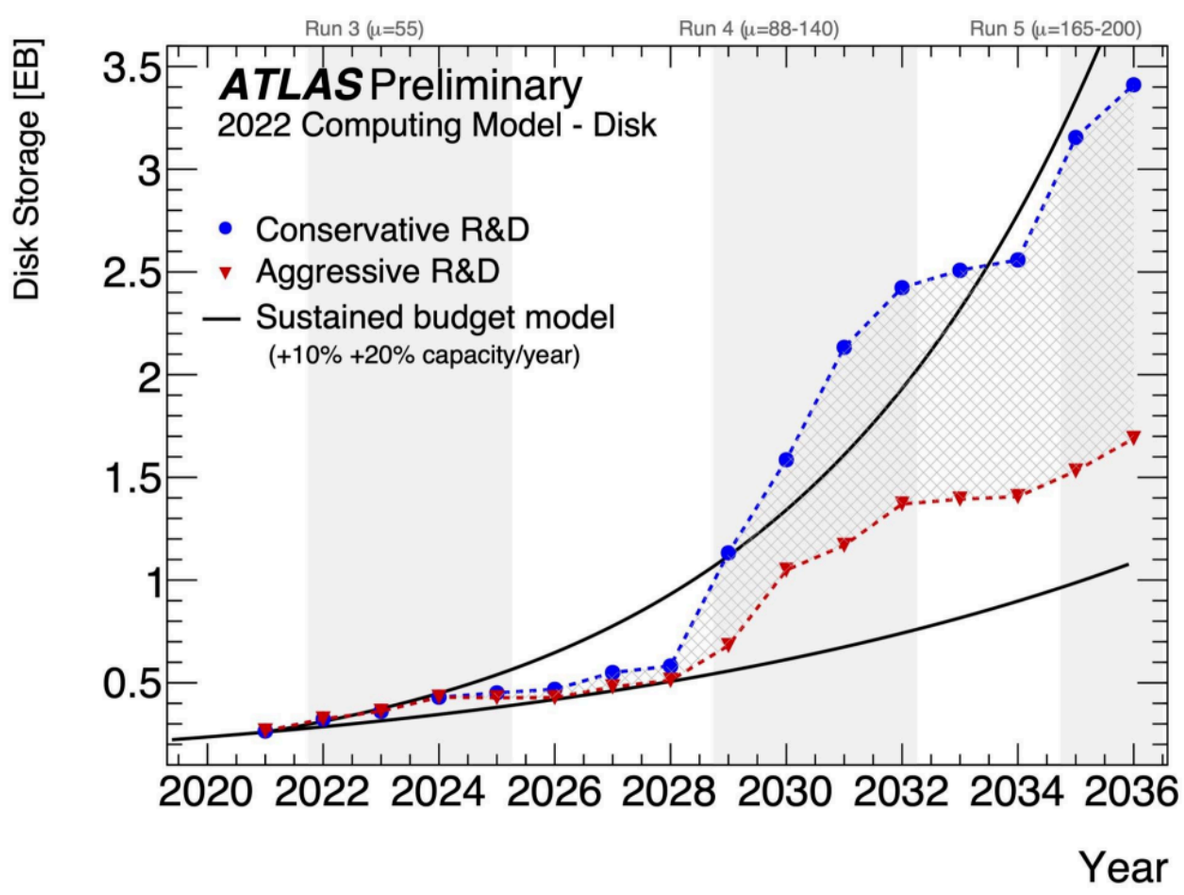
- Back-end intégré dans ce système
- Utilisation de COTs (Custom Off the Shelf), cooling standard
- Grande quantité de memoire à faible cout → **possible de realiser l'event building dans les serveurs**
  - Système est **throughput-bound,** not latency-bound



La loi du Moore est mort

Et les traiter… on pensait que c'était ok, mais finalement non

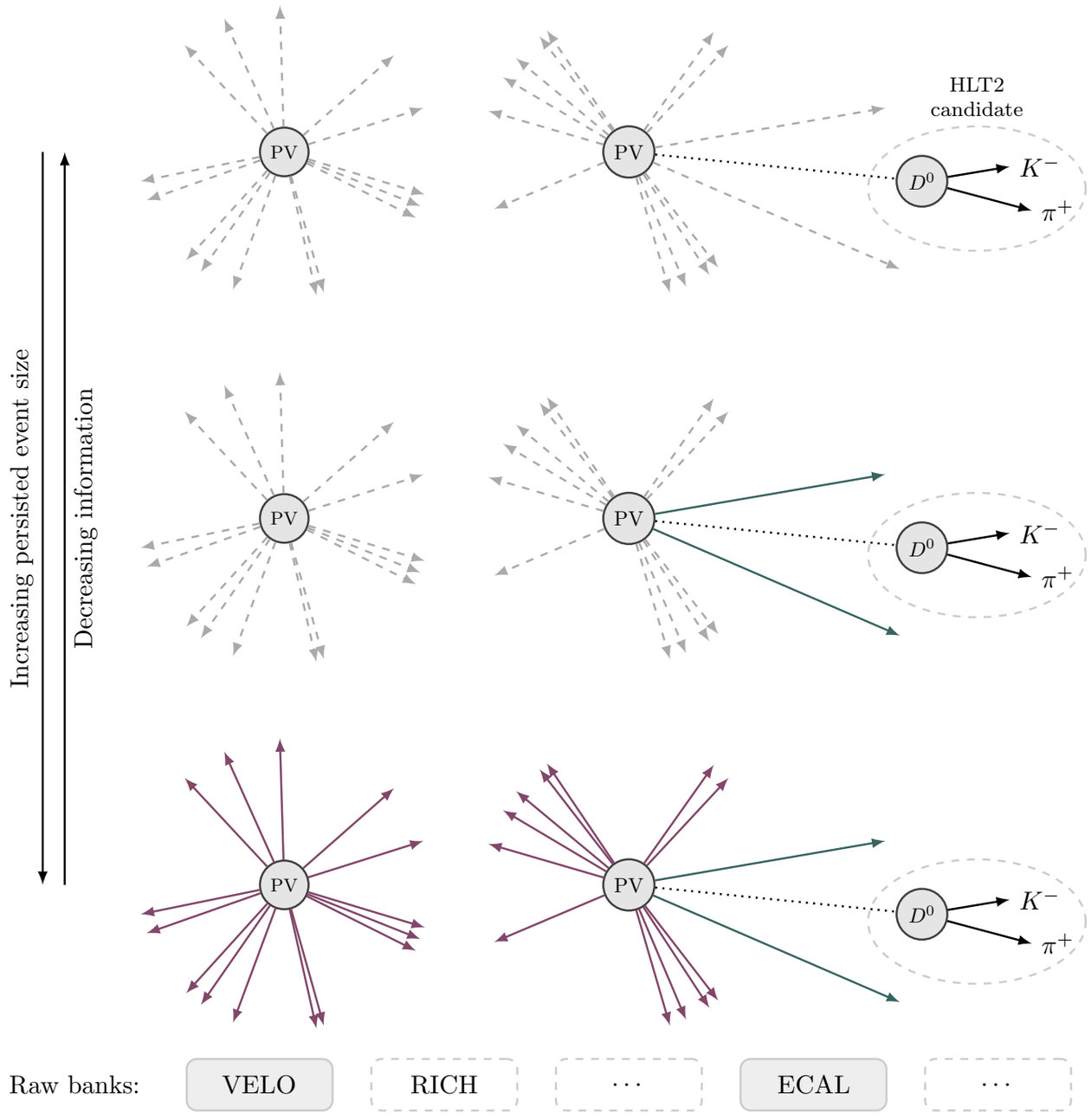




Les modèles des calculs actuels basées sur les CPUs et les disques dures commencent a sortir hors du budget sans un R&D "agressive"

Le concept d'Analyse en Temps Reel (RTA)

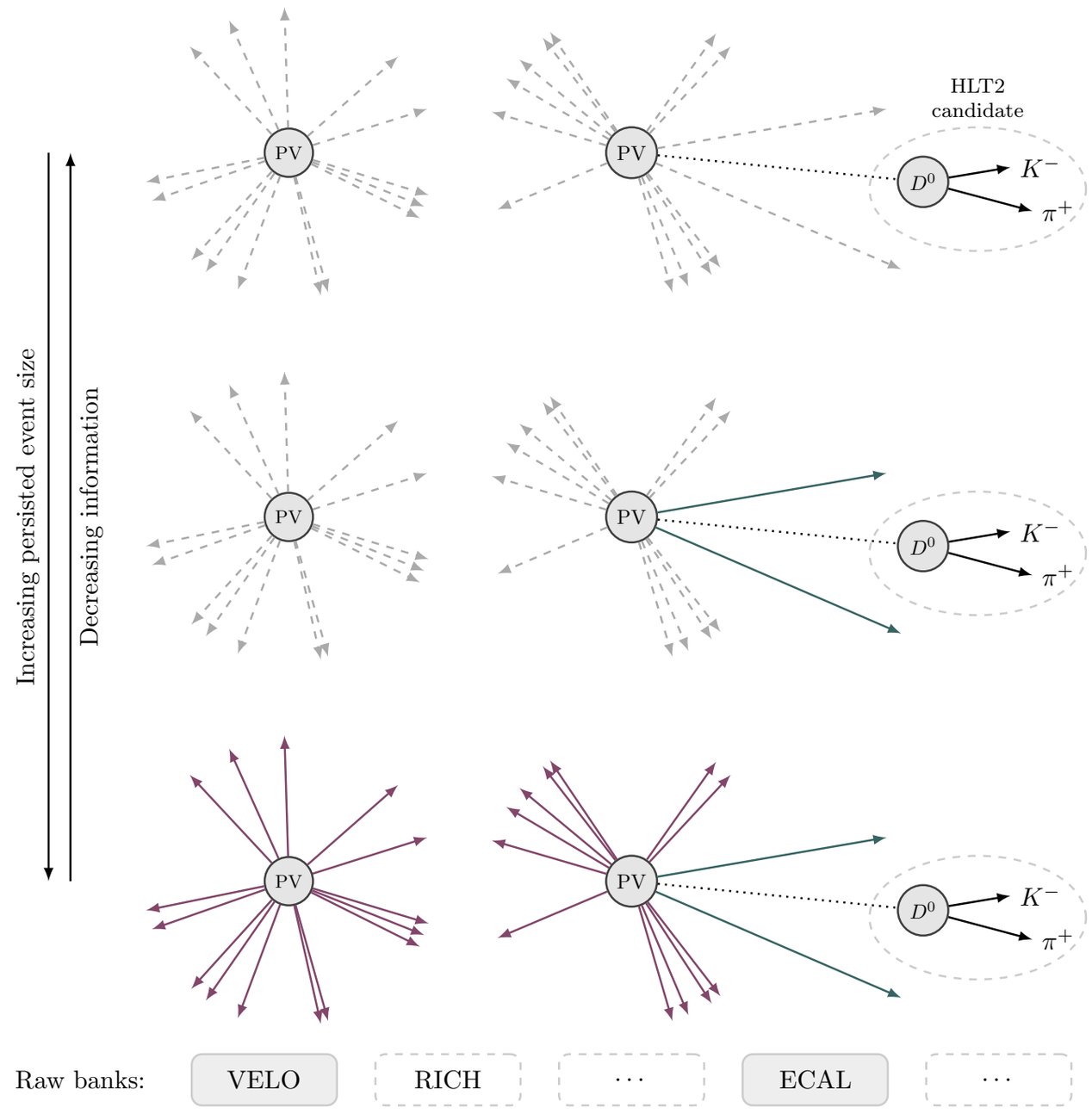
- Tendance des upgrades LHC: augmenter les collisions simultanées dans un Bunch Crossing (pile-up)
  - Plus de collisions, plus de chance qu'il y a une signature interessante
  - ATLAS/CMS passent de 40 a 200, LHCb de 5 a 40
- Un trigger "classique" choisit les Bunch Crossings qui contiennent la collision interessante et sauvegarde toute l'information brute
  - évènement = bunch crossing
- Un système avec un scaling pas optimale pour le futur:
  - Augmentation de luminosité = plus de événements
  - Augmentation de complexité par évènement



Le concept d'Analyse en Temps Reel (RTA)

Et si on changeait le façon dont on définit l'évènement ?

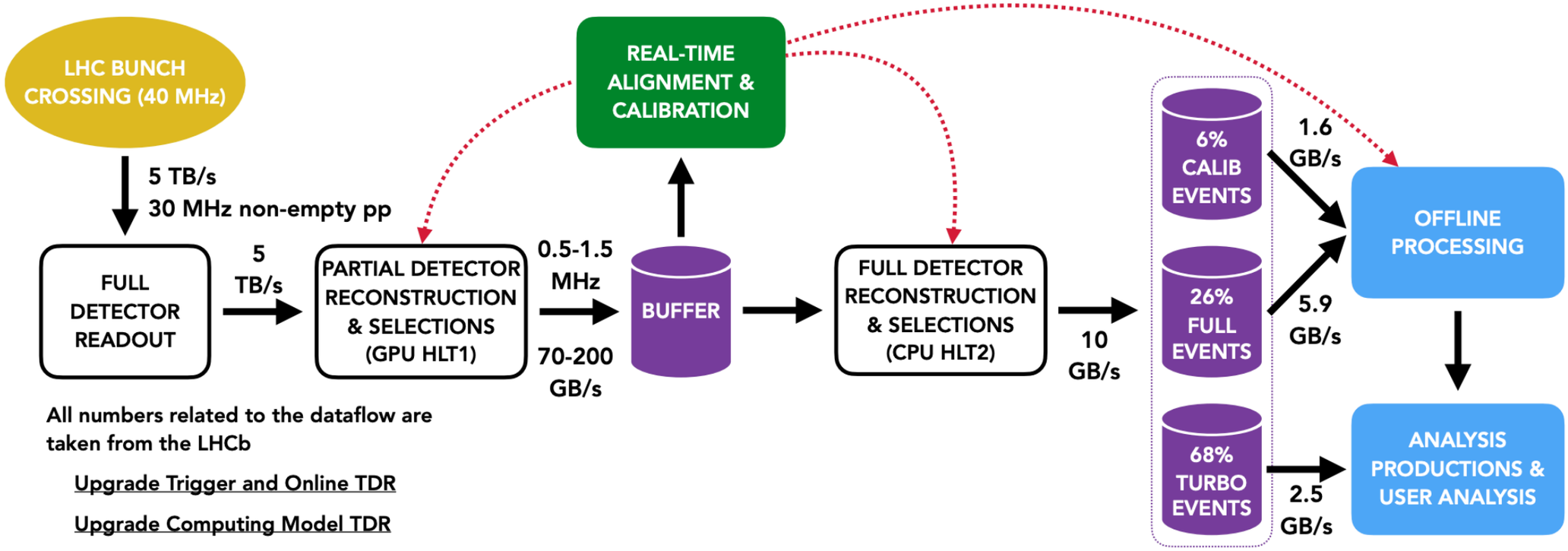
- Concept de "persistence selective":
  - Sauvegarder juste l'information qui nous intéresse vraiment
  - Sauvegarder l'information reconstruite et pas brute
- Nécessite une performance de reconstruction du trigger égale a la reconstruction "offline" (i.e. come si on avait tout le temps de le faire)
  - Alignement et calibration doivent être faites en temps réel



- Reduction de la taille de "l'évènement" par un facteur 10 (LHCb) - on peut stocker plus de physique!
- Analyse des données peut être faite directement a la sortie du trigger
- Plus possible de re-faire la reconstruction de l'évènement en cas d'erreur!

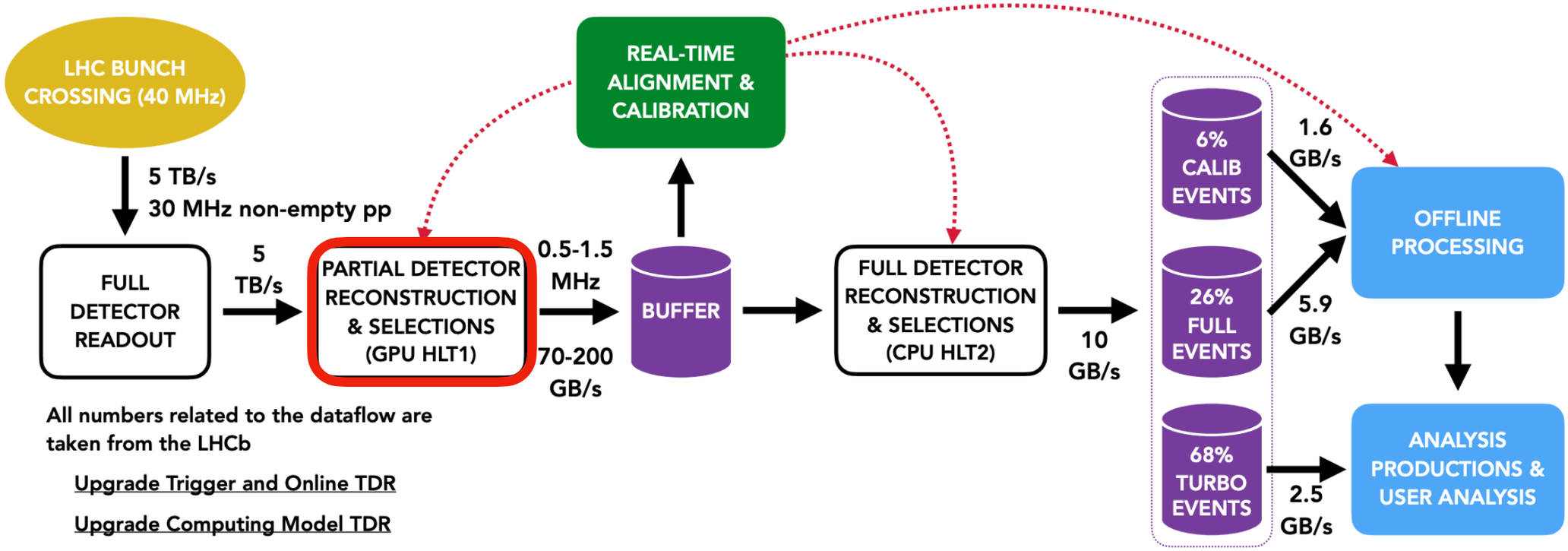
L’implementation LHCb du concept RTA

- **HLT1:** premier passage en **GPU**, reconstruction de l’évènement avec precision réduite, selections inclusives - reduction de rate de 5 TBps → 100 GBps
- **Buffer** temporaire de 32 PB pour faire **l’alignement et calibration du détecteur en temps reel**
- **HLT2**: deuxième passage, reconstruction de l’évènement avec precision “offline” en CPU, selections inclusives et exclusives - reduction de rate de 100 → 10 GBps



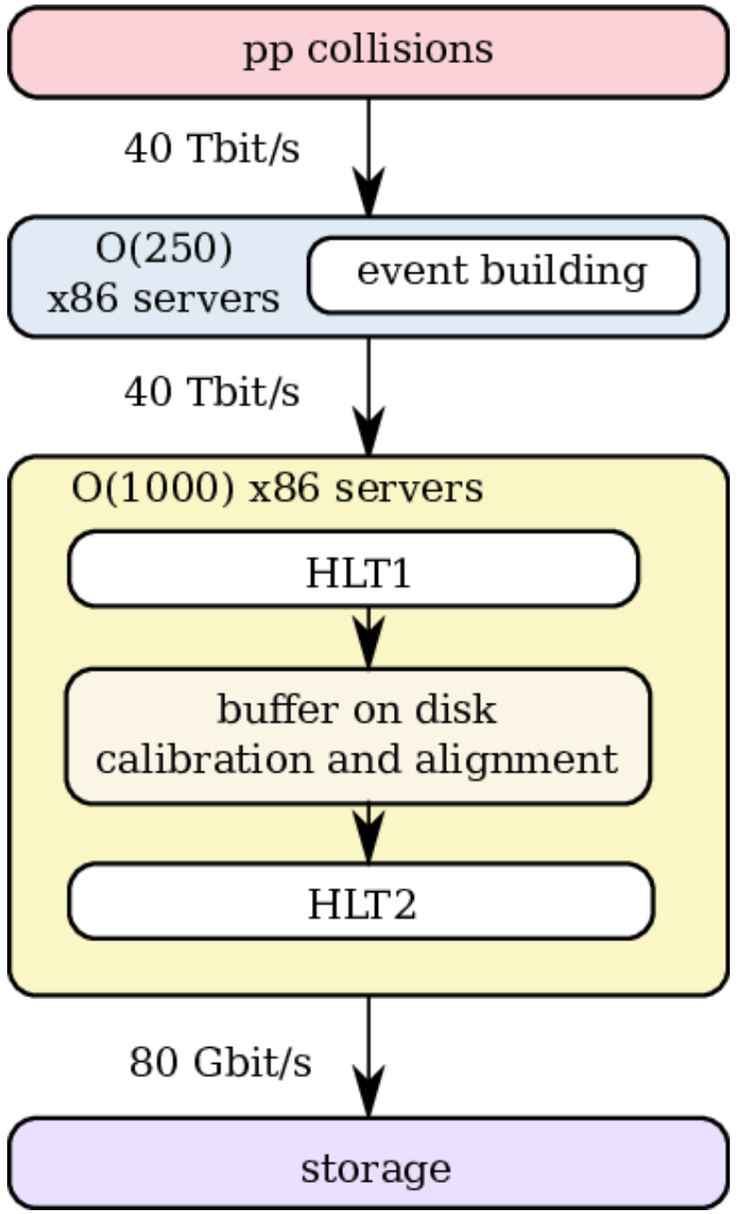
L’implementation LHCb du concept RTA

- **HLT1:** premier passage en GPU, reconstruction de l’évènement avec precision réduite, selections inclusives - reduction de rate de 5 TBps → 100 GBps
- **Buffer** temporaire de 32 PB pour faire **l’alignement et calibration du détecteur en temps reel**
- **HLT2**: deuxième passage, reconstruction de l’évènement avec precision “offline”, selections inclusives et exclusives - reduction de rate de 100 → 10 GBps
- 3 façons de sauvegarder les données finales
  - Turbo (68%): sauvegarder seulement la désintégration interessante
  - Full (26%): sauvegarder l’info reconstruite complete
  - Calib (6%): sauvegarder l’info brute et reconstruite complete



Architecture d’un trigger software CPU

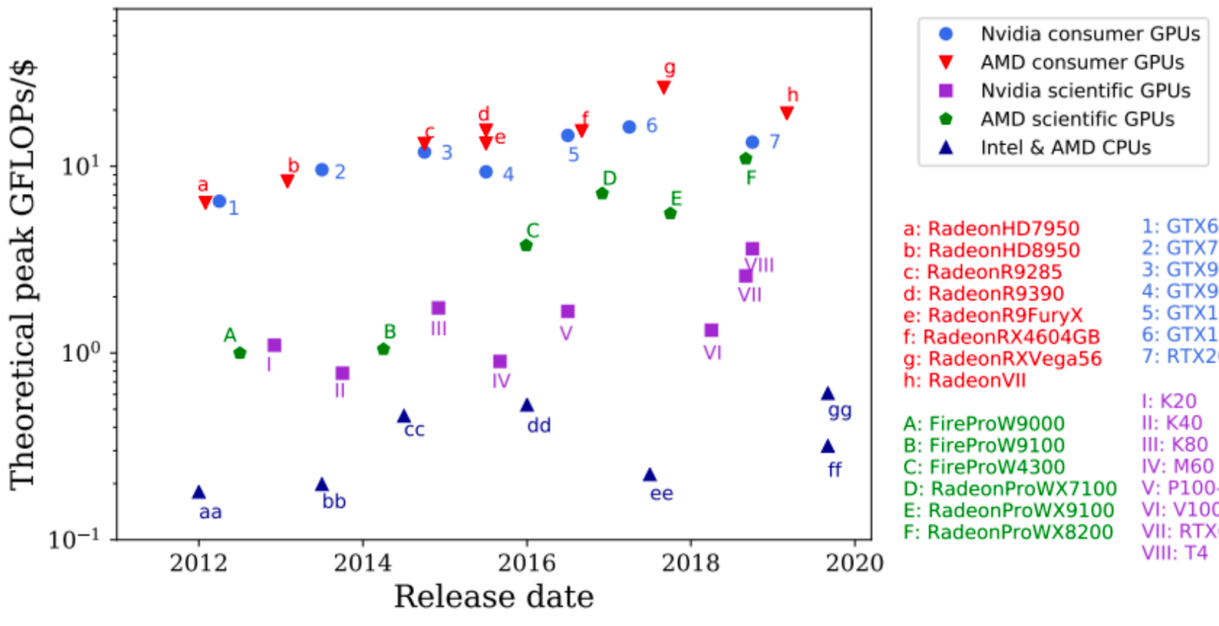
- Readout at 40 MHz
- Event builder
- Réseau entre l’event builder et la ferme de calculs
- Ferme de calculs: ~ 4000 CPUs
- 2 etapes de trigger software (HLT)

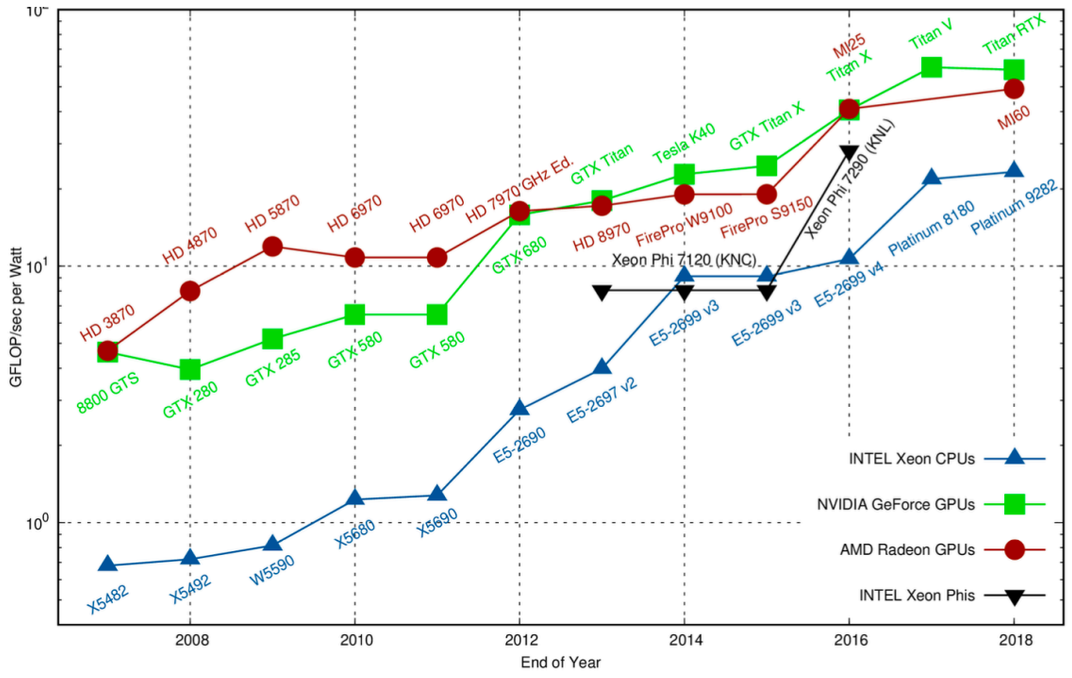


Une alternative aux CPUs

- **G**raphics **P**rocessing **C**ards : GPUs
- Originellement conçues pour les opérations de mise en image (ex jeux videos)
- Mais depuis:
  - Utilisation pour l'entrainement des modèles IA
  - Le farming de cryptomonais
  - Les fermes de calculs a haute performance
  - … **Et les calculs scientifiques!**

Pourquoi?





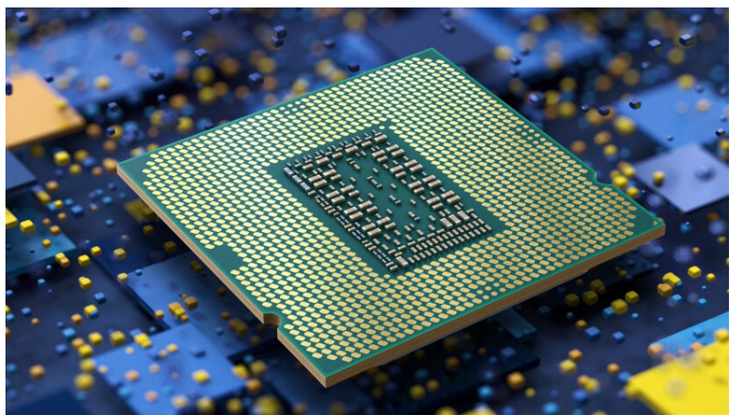
https://arxiv.org/pdf/2003.11491.pdf

GPU vs CPU

CPU: minimisation de latence

- Grande memoire de haute vitesse
- Hautes frequencies
- Executions speculatives

➡Excellente performance pour operations en serie



GPU: oublier la latence

- Petite memoire pas aussi vite
- Bases frequencies
- Executions fixes
- Milliers des operations d’executions en parallel (threads)

➡Excellente performance pour operations en parallel

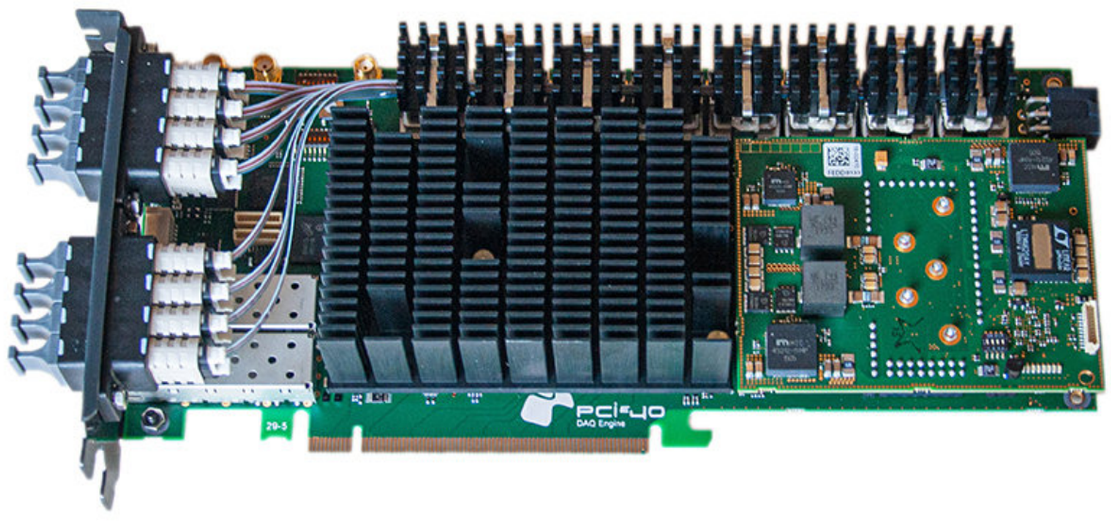
GPU vs FPGA

GPU

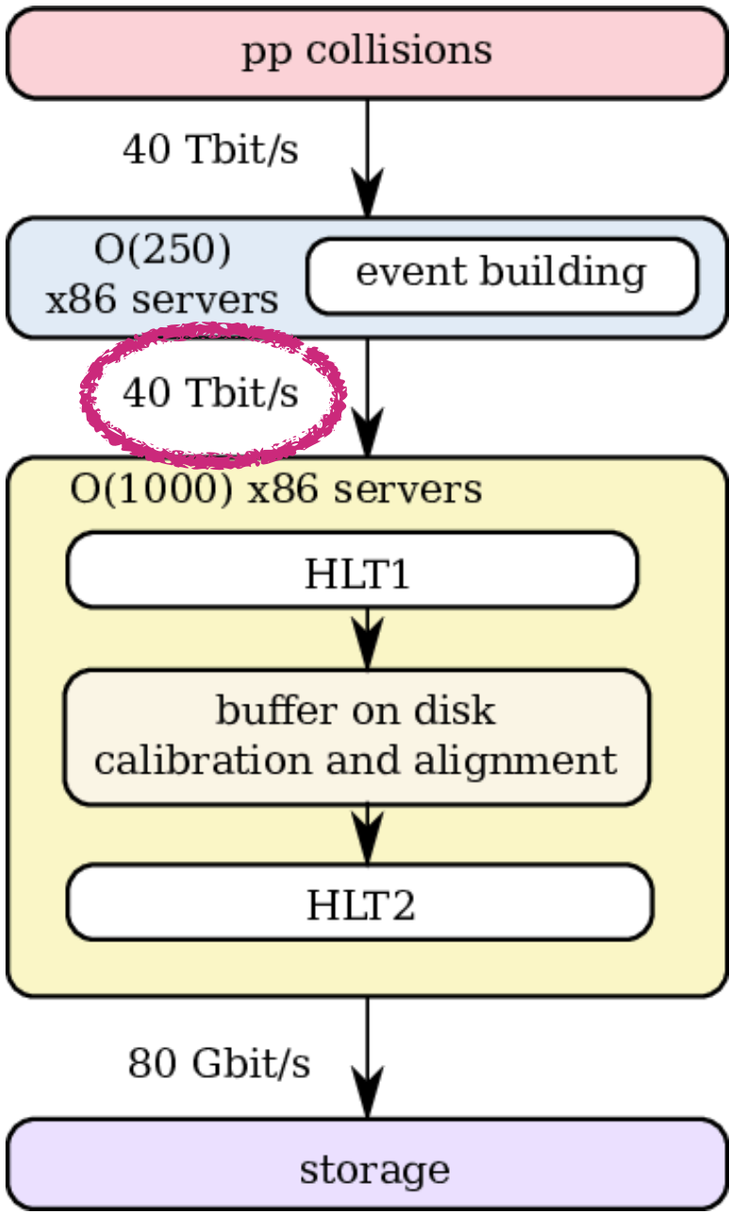
- Latence plus grande
- Bande passante limité par connection PCIe / NVLink
- Très bonne performance de precision flotante
- Compatibilité entre différents versions facile
- Moins de cout (achat, mais aussi d'ingenerie et dev. Soft)

FPGA

- Petite latence
- Haute Bande passante & facilité de connection
- Moyenne performance de precision flotante
- Compatibilité entre différents versions difficile
- Cout plus élève



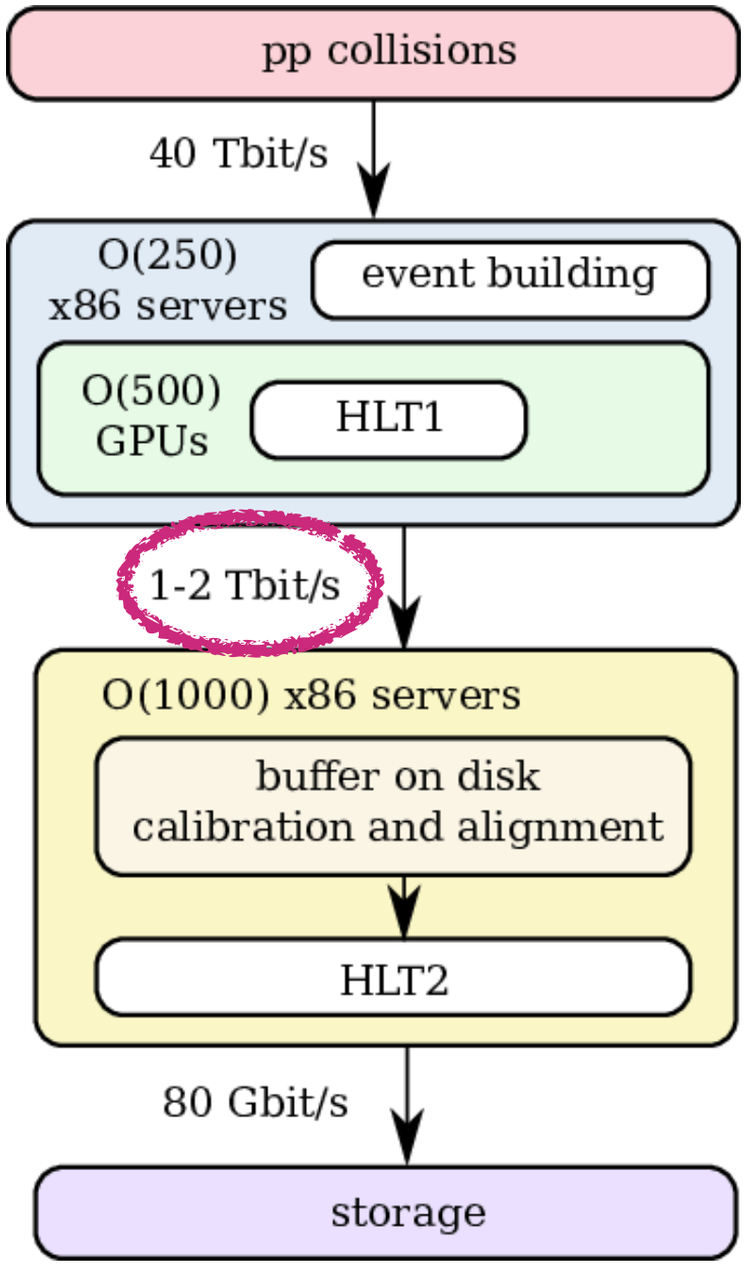
Une choix a faire



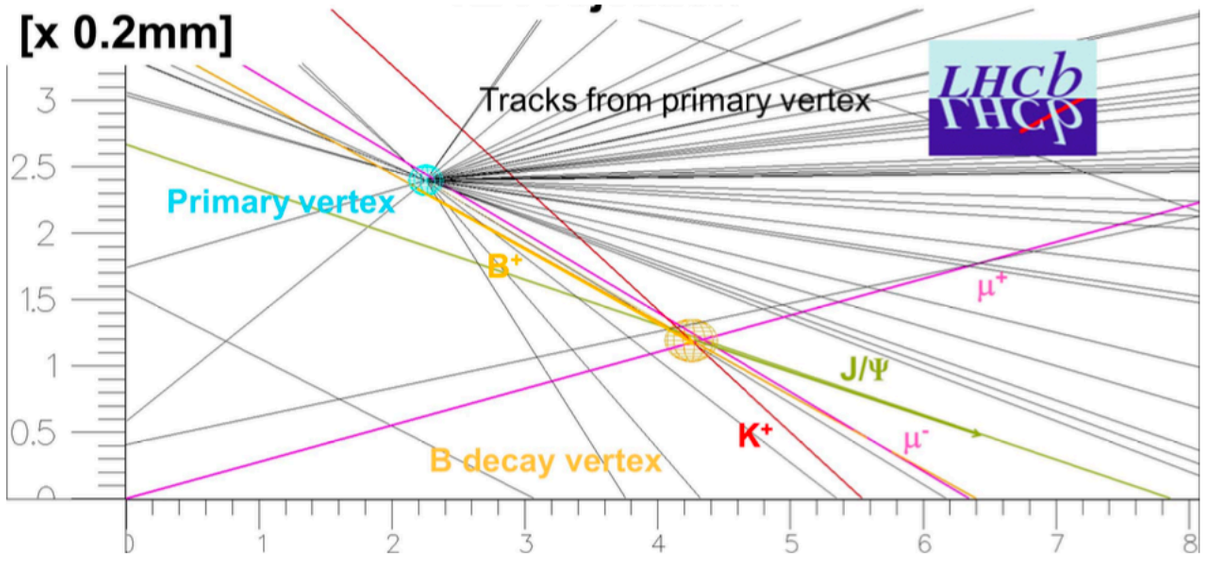
Deux options:

1. Envoyer les 40 Tb/s dans une ferme de calculs CPU → besoin de 2eme réseau

2. Mettre des GPUs directement dans les serveurs EB → réduire les données dans le EB avant de les envoyer



La reconstruction d'un évènement



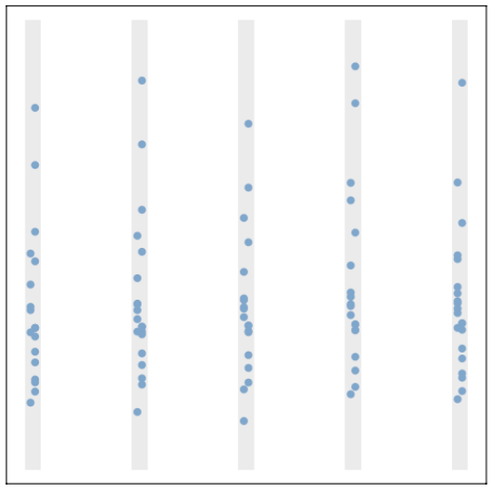
Chaque 25 ns, il y a une collision p-p

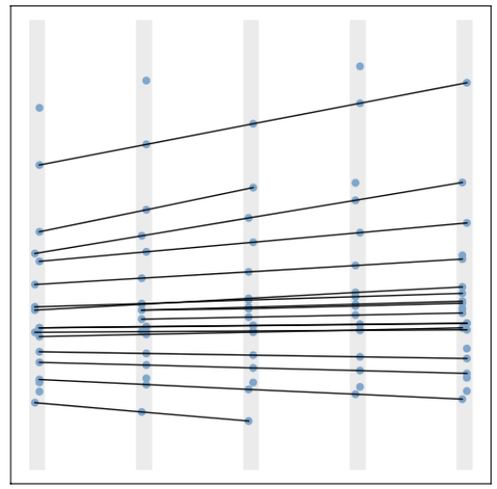
Chaque collision (évènement) est un process indépendante

Événements peuvent être traités en parallel!

Dans chaque évènement on doit:

- Transformer les données brutes dans une collection de hits pour chaque détecteur (x,y,z, énergie etc)
  - **On peut paralleliser les operations pour chaque détecteur, ou meme pour chaque FE**
- Reconstruire les objets de physique (traces, vertices, particules)
  - Operations de pattern recognition (quel hit appartient dans quel particule?)
  - **Paralleliser entre hits, objets reconstruites et finalement particules**





Parallelisation est possible dans chaque étape de la chaine!

L’architecture

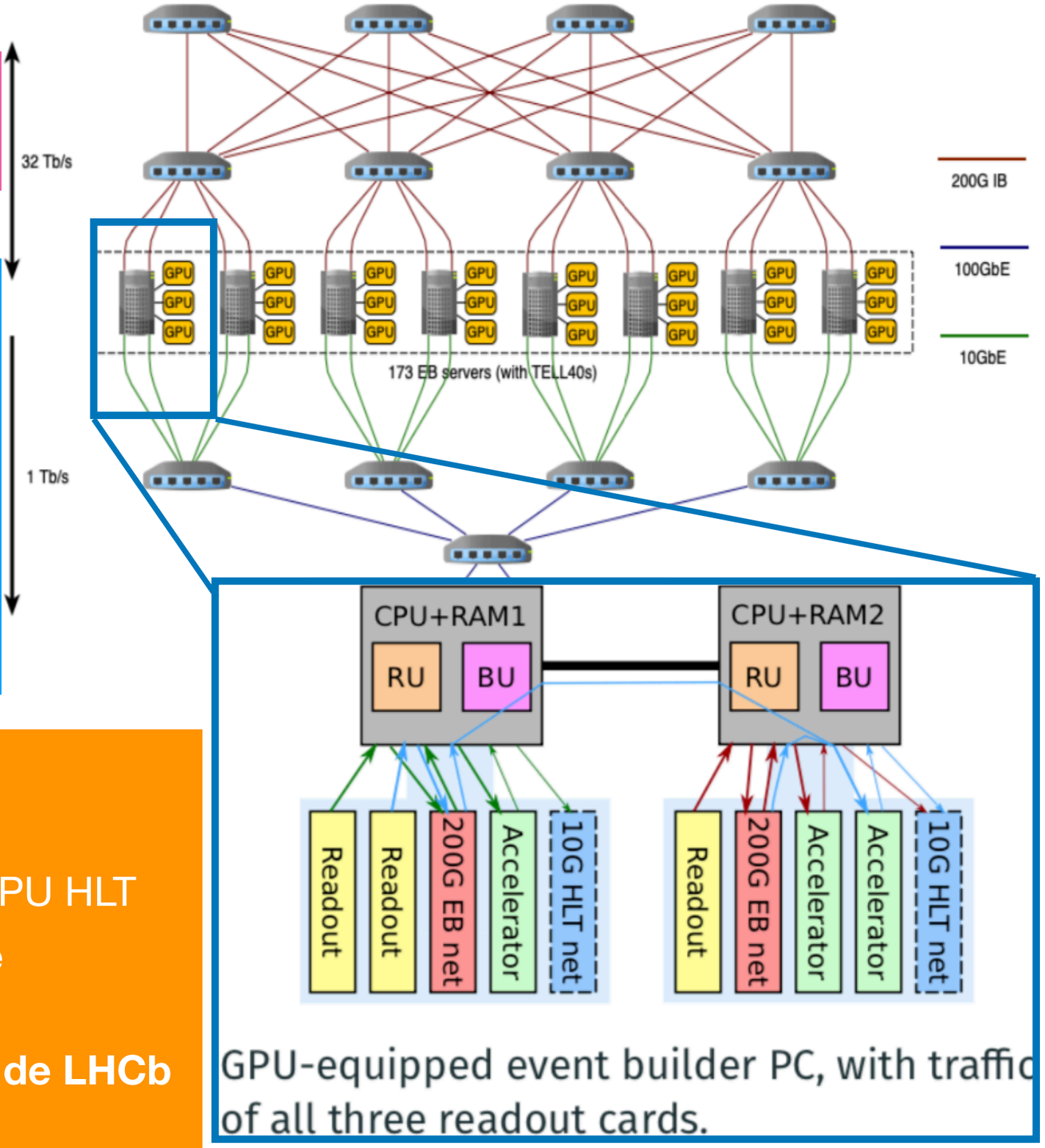
- Ferme d’EB avec 173 serveurs

- Chaque serveur a 3 portes PCIe libres
  - Peuvent être utilisés par des GPUs
  - Refroidissement & alim. suffisant
  - Si les GPUs ont pas besoin de communiquer entre eux, elles fonctionnent comme une carte réseau (data in , data out)

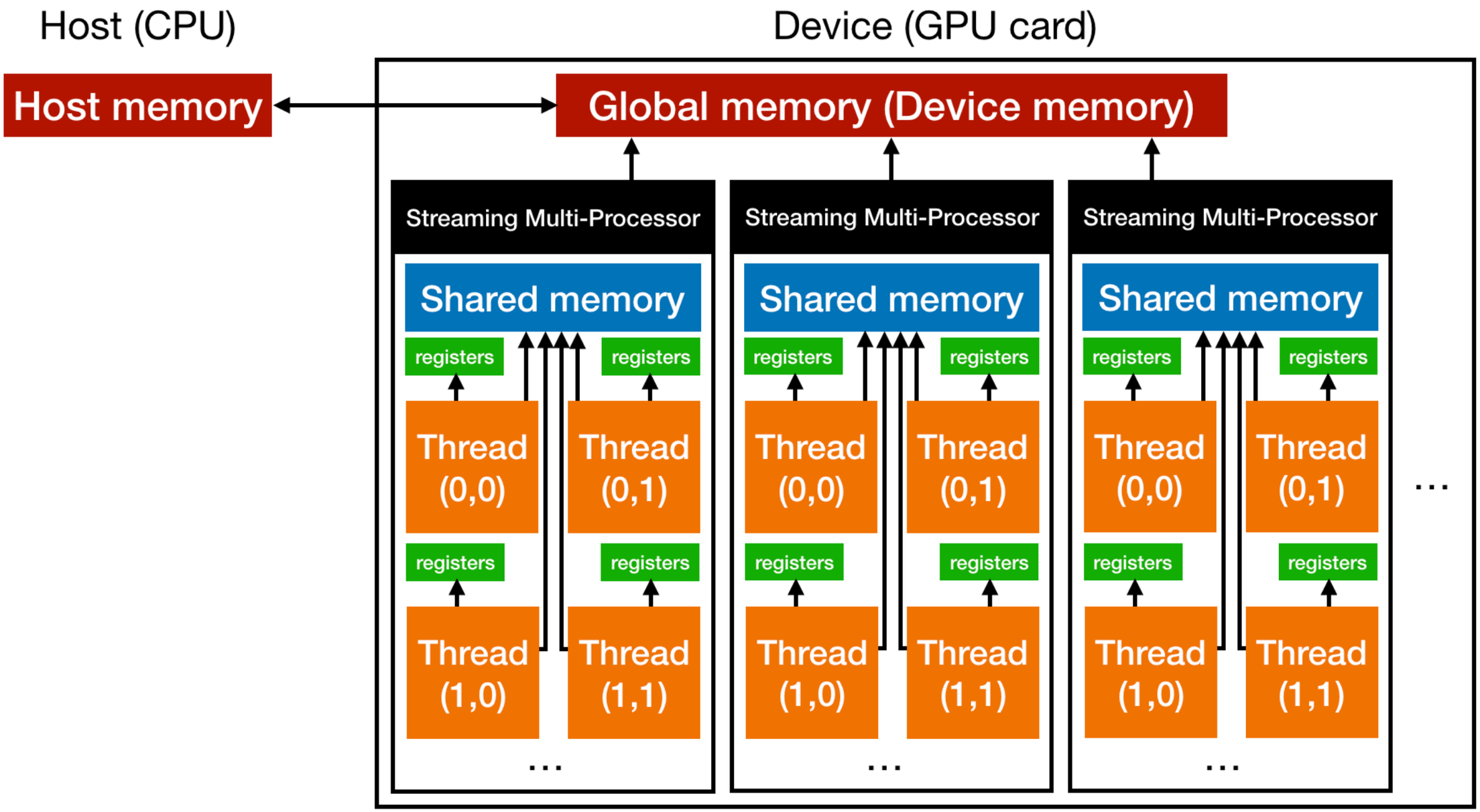
- Mapping des GPUs très bien adapté
- Taches de reconstruction parallelisables
- Reduction de taille de réseau entre EB & CPU HLT
- Moins de cout et une solution qui peut être redimensioné dans le futur

➡**A était choisi et implementé pour Run 3 de LHCb**

➡**500 Nvidia RTX A5000 GPUs**

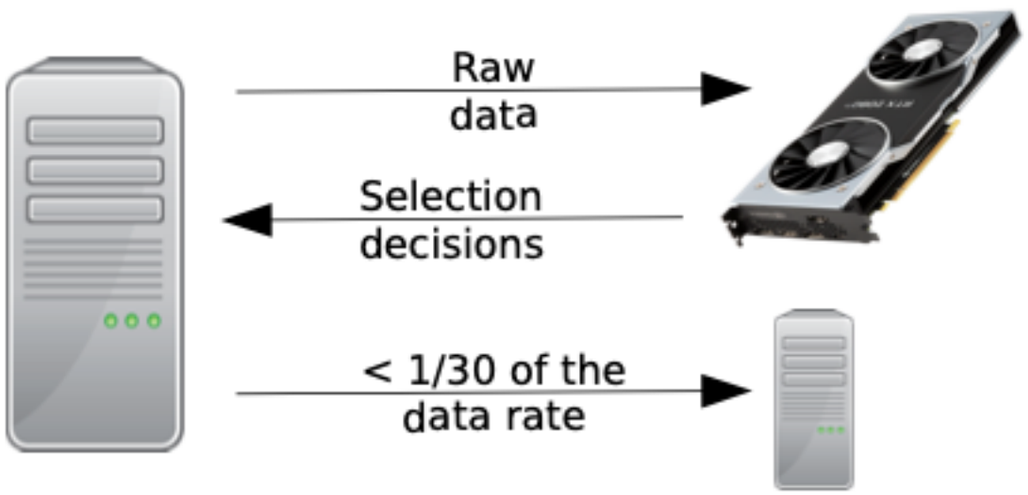


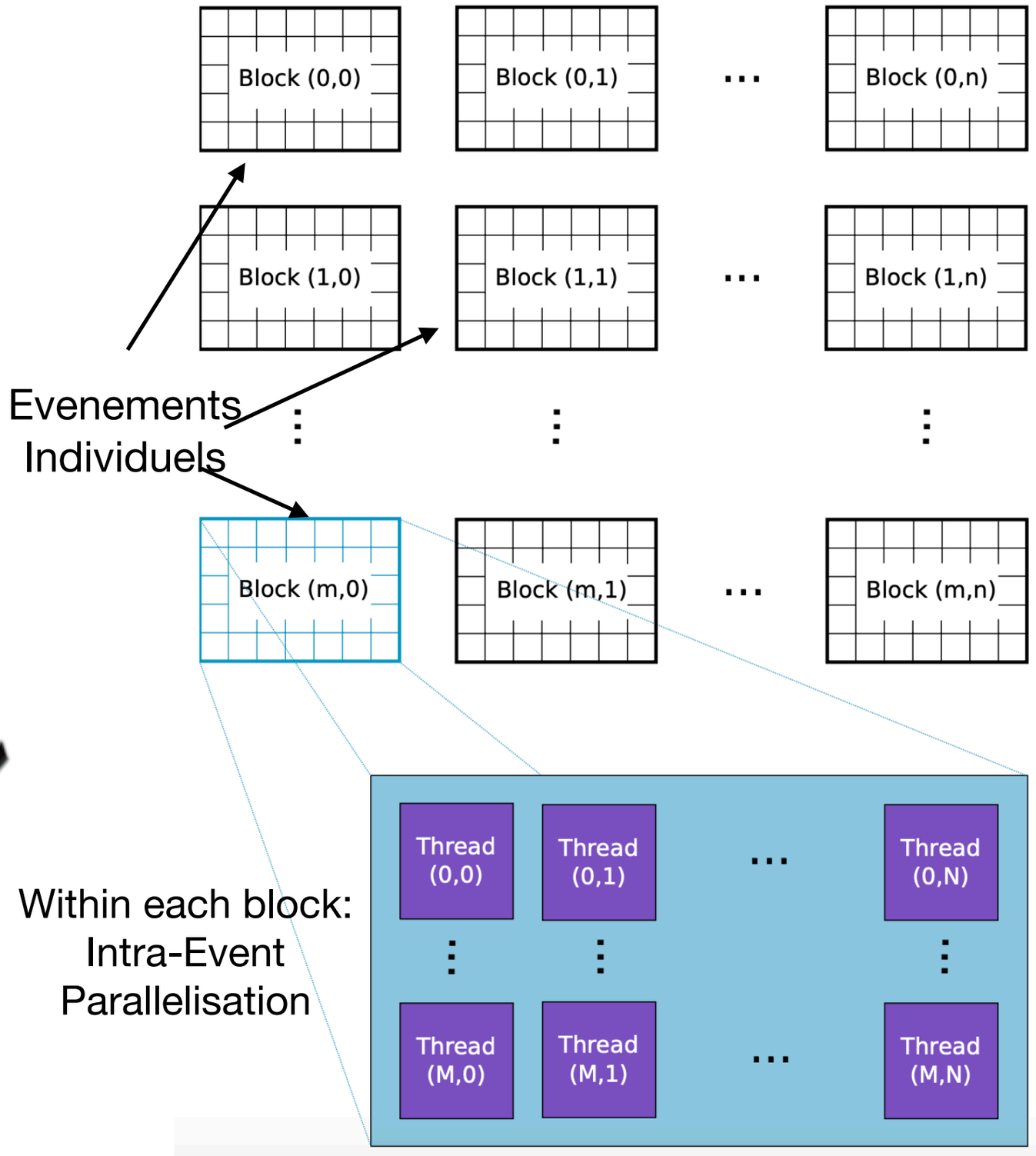
Une GPU


Host (CPU)
Device (GPU card)
Host memory
Global memory (Device memory)
Streaming Multi-Processor
Shared memory
registers
Thread (0,0)
Thread (0,1)
Thread (1,0)
Thread (1,1)
...

Schema de parallelisation

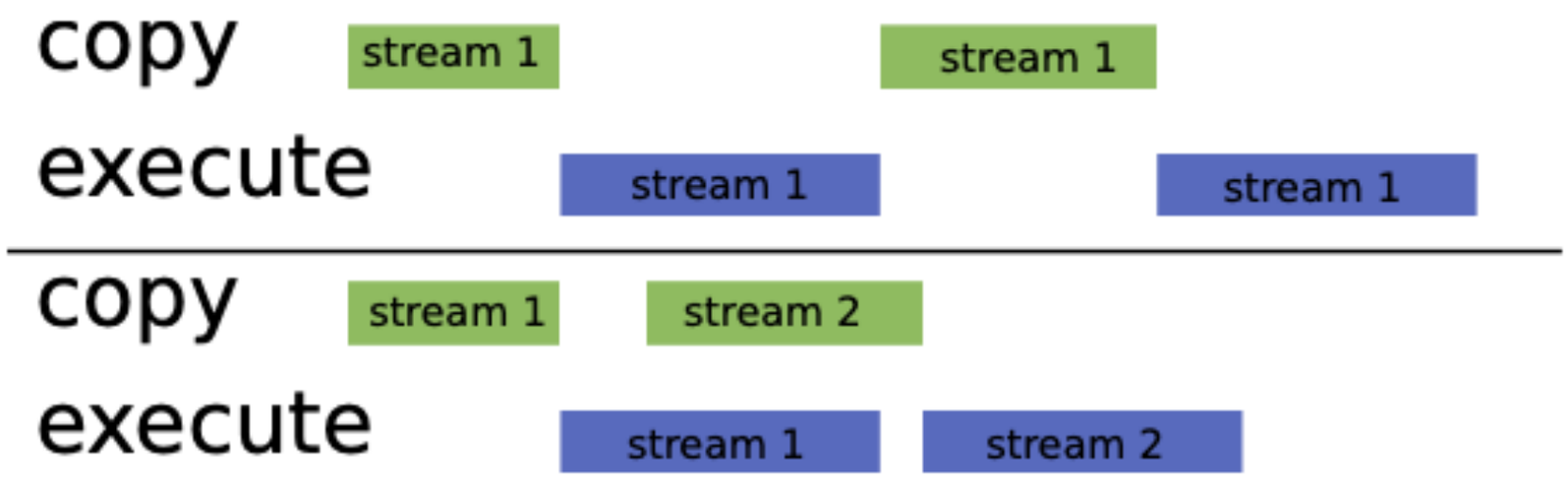
- Structure de grille, threads organisées en blocks
- Chaque block traite événements différents
- Chaque thread d'un block reconstruit des différentes parties de l'évènement en parallel
- Les communications de la GPU avec le monde extérieur sont chers, alors ils sont minimisés: les données brutes entrent et les selections finales sortent





Utilisation de batching et streaming

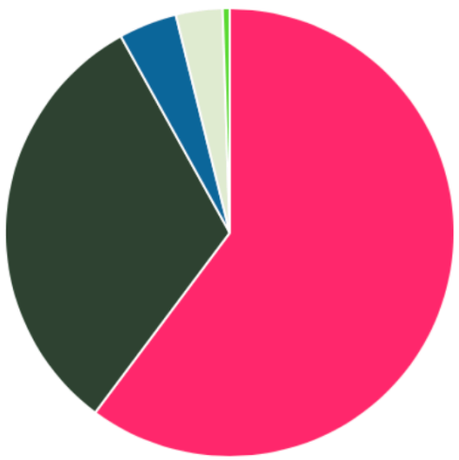
- Nous voulons minimiser les communications entre l'hôte et le GPU.
- Et notre unité de données, l'événement, est petite — de l'ordre de ~100 kB.
- Nous transférons les événements vers le GPU par lots : typiquement 1000 événements par lot, mais c'est configurable
- Nous utilisons des *CUDA streams* lors de la planification des lots d'événements à traiter
- Chaque *stream* traite la même séquence sur un lot différent
- Les constantes de configuration, etc., sont copiées vers le GPU pour chaque *stream*.



Allen: un framework de software trigger

- Un projet publique: [gitlab repo]

- Support CPU, Nvidia GPU (CUDA, CUDACLANG), AMD GPUs (HIP)
- Code GPU écrit en CUDA mais plutôt “caché” dans le backend
- Compatibilité avec les autres architectures (HIP, CPU) via macros



Documentation

```
#include "SAXPY_example.cuh"

INSTANTIATE_ALGORITHM(saxpy::saxpy_t)

void saxpy::saxpy_t::set_arguments_size(
  ArgumentReferences<Parameters> arguments,
  const RuntimeOptions&,
  const Constants&) const
{
  set_size<dev_saxpy_output_t>(arguments, first<host_number_of_events_t>(arguments));
}

void saxpy::saxpy_t::operator()(
  const ArgumentReferences<Parameters>& arguments,
  const RuntimeOptions&,
  const Constants&,
  const Allen::Context& context) const
{
  global_function(saxpy)(dim3(1), m_block_dim, context)(arguments, m_saxpy_factor);
}

/**
 * @brief SAXPY example algorithm
 * @detail Calculates for every event y = a*x + x, where x is the number of velo tracks in one event
 */
__global__ void saxpy::saxpy(saxpy::Parameters parameters, const float saxpy_scale_factor)
{
  const auto number_of_events = parameters.dev_number_of_events[0];
  for (unsigned event_number = threadIdx.x; event_number < number_of_events; event_number += blockDim.x) {
    Velo::Consolidated::ConstTracks velo_tracks {
      parameters.dev_atomics_velo, parameters.dev_velo_track_hit_number, event_number, number_of_events};
    const unsigned number_of_tracks_event = velo_tracks.number_of_tracks(event_number);

    parameters.dev_saxpy_output[event_number] = saxpy_scale_factor * number_of_tracks_event + number_of_tracks_event;
  }
}
```

Code qui resemble a du C++! Mais avec quelques particularités:

- Pas des objets de taille variable (vectors etc)
- Necessite une gestion de parallelisation et des operations de syncrhonisation
- Et une bonne gestion de memoire

Les algorithmes de HLT1

Reconstruction

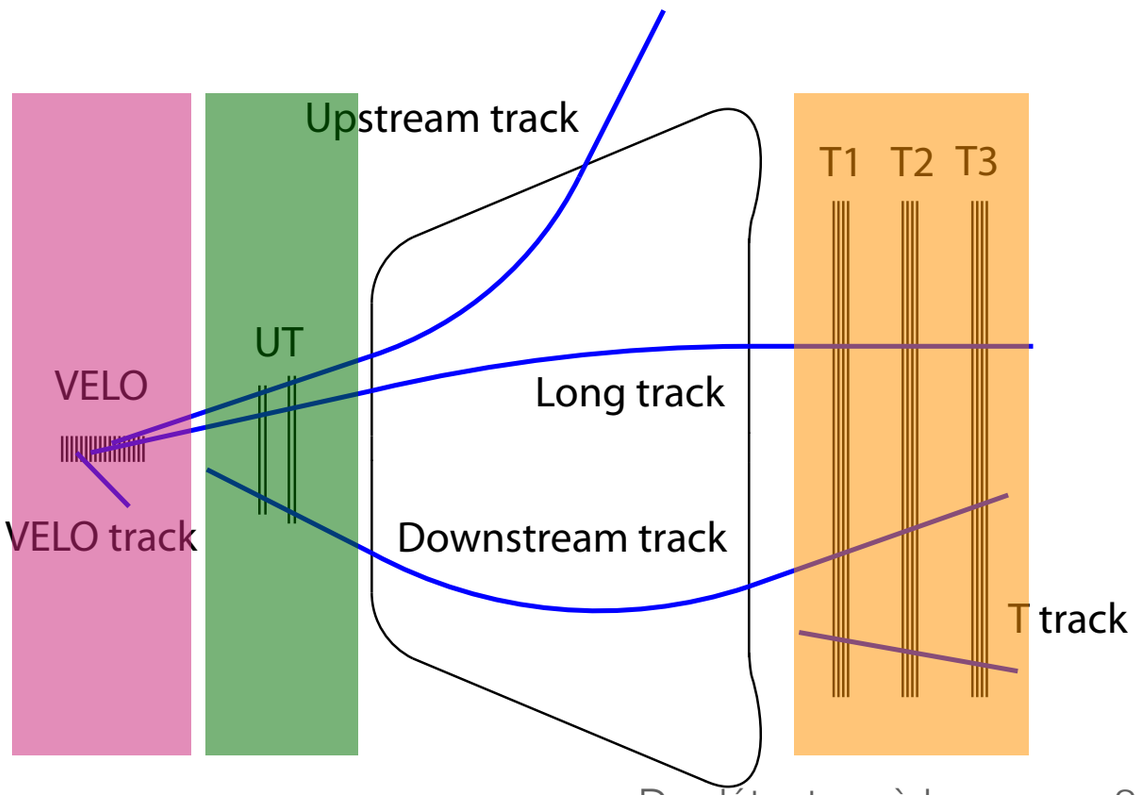
Tracking a la coeur de la reconstruction HLT1

3 détecteurs:

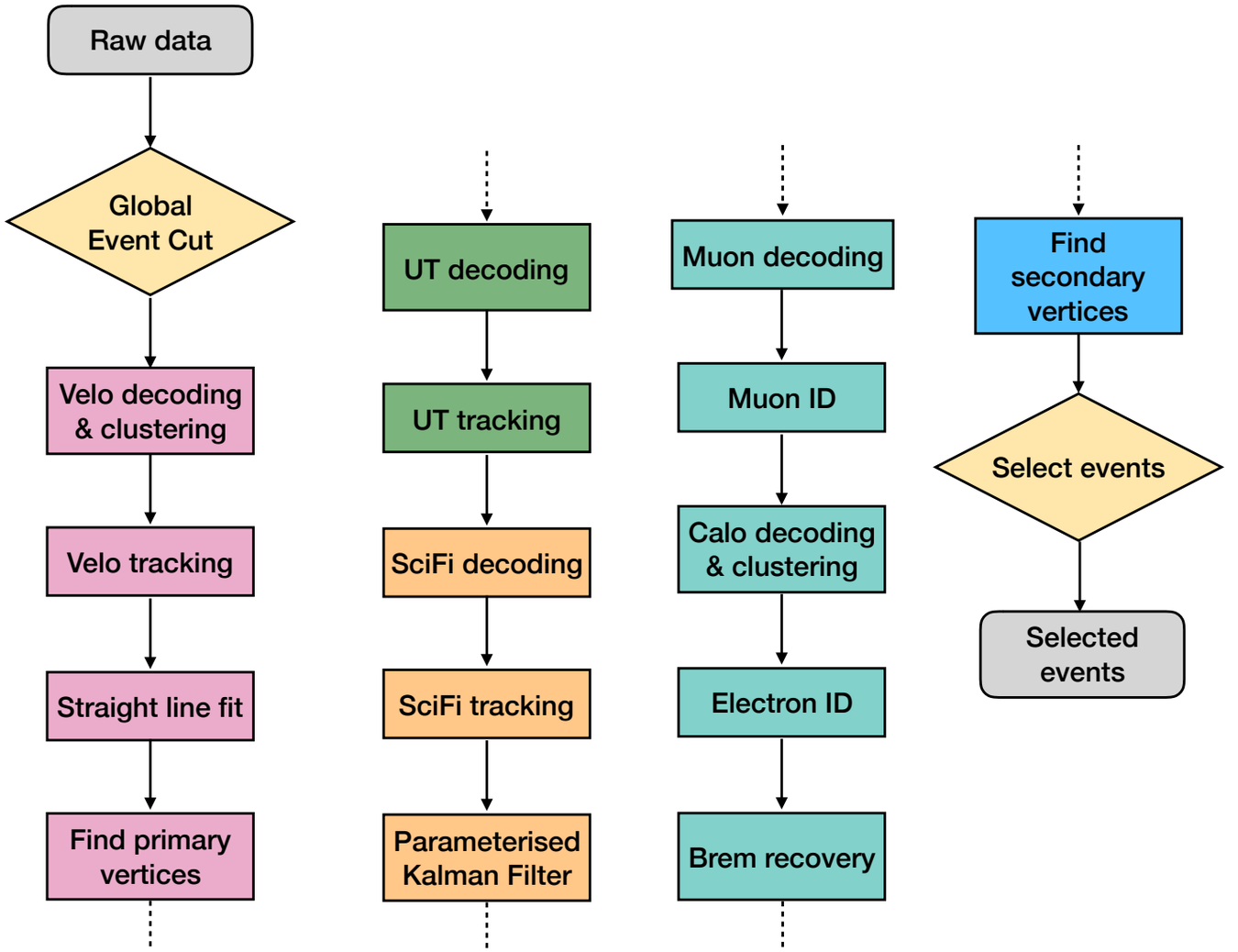
- VELO: pixel, sans champs magnétique, precise tracking et vertexing
- UT: strip, un peu de B, important pour la rejection des ghost tracks et la resolution
- SciFi: après l'aimant, mesure de l'impulsion

+

PID from muon stations + **Calo**



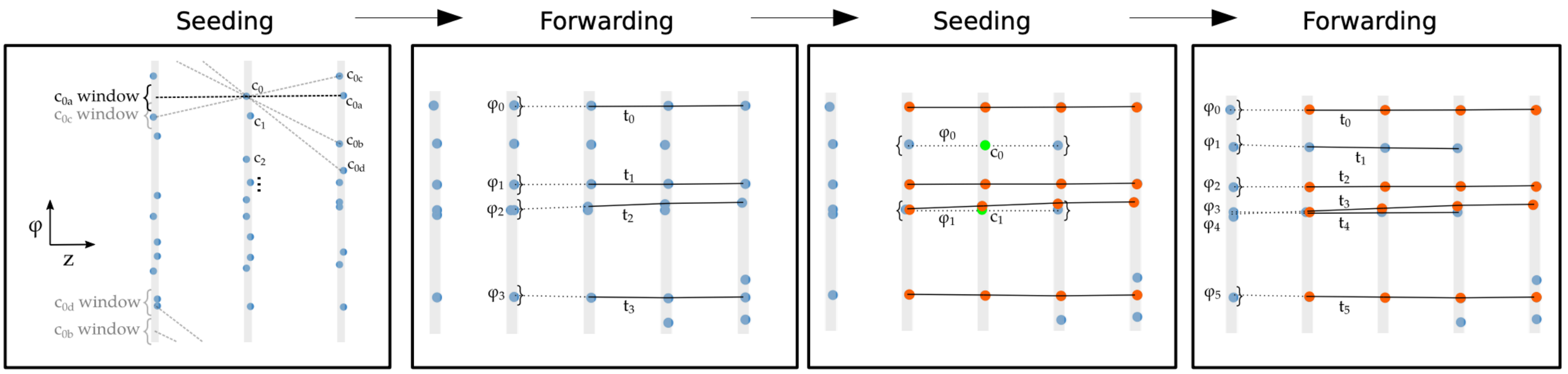
Lignes de sélection → **Selection des évènements**



Un example d'un algorithme GPU

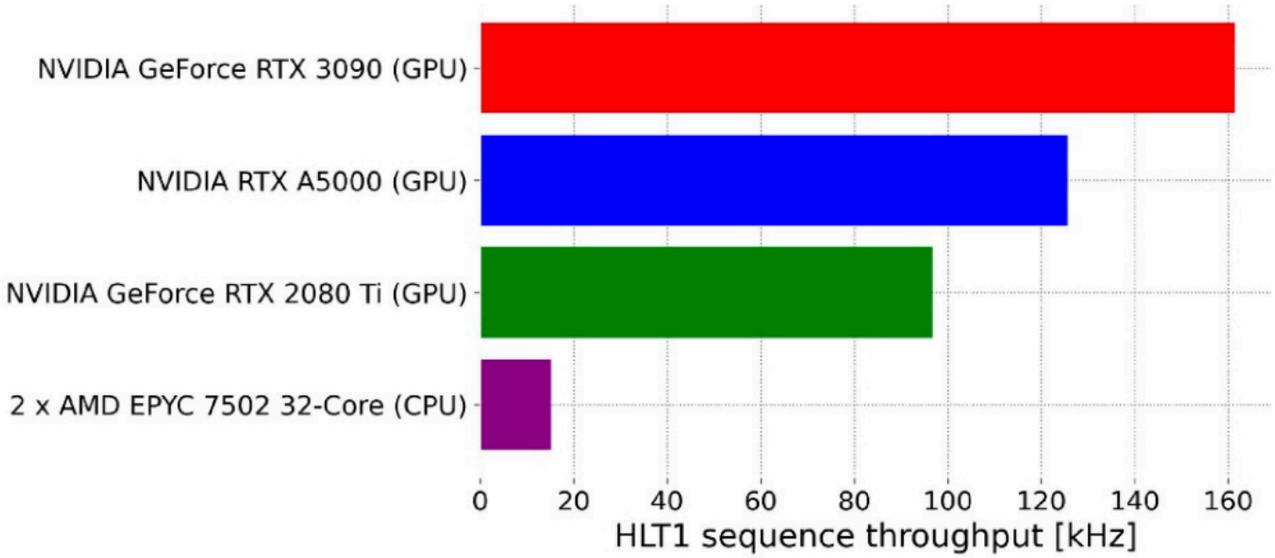
Velo tracking: *Journal of Computational Science, vol. 54, 2021*

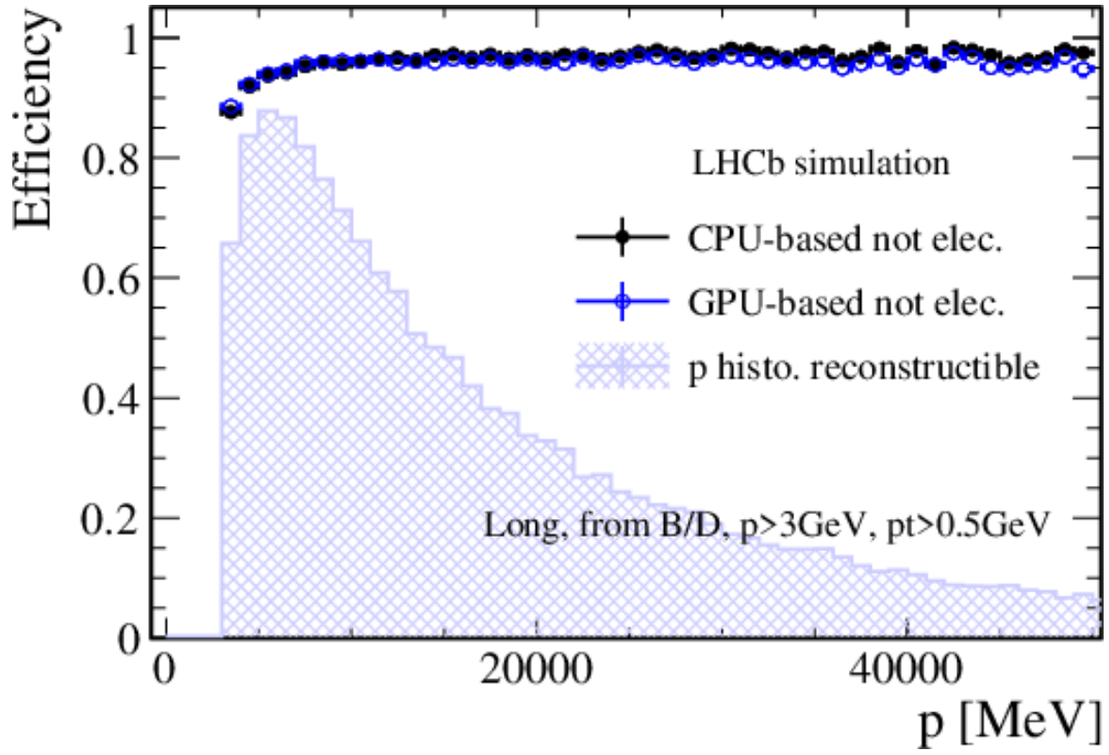
- 26 silicon pixel modules avec $\sigma_{x,y} \sim 5\ \mu m$
- Algorithme de reconstruction parralelisé: *Search by Triplet*



- Construction des "triplets" (3 hits consecutives) → parallelisation
- Choix des meilleurs selon l'alignement en phi
- Sorting des hits selon phi → access a memoire optimisée
- Extension des triplets dans les couches suivantes → parallelisation

Performance

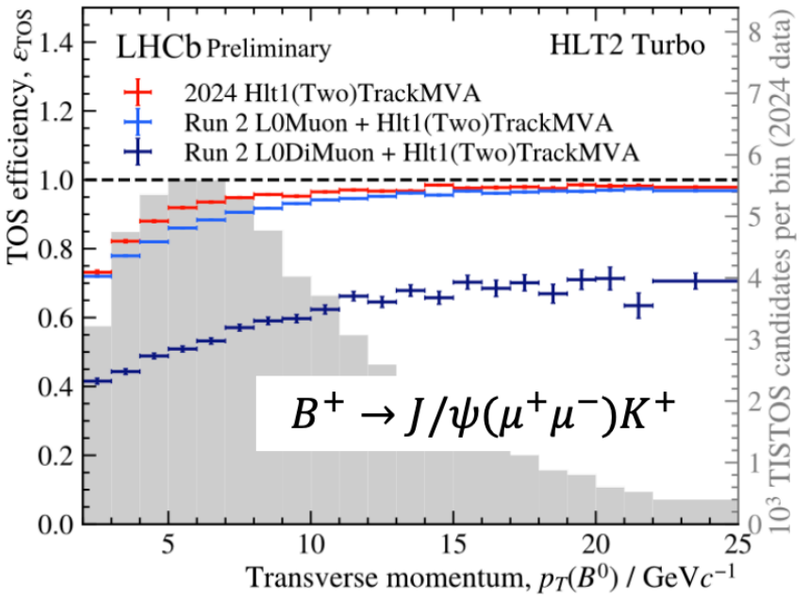




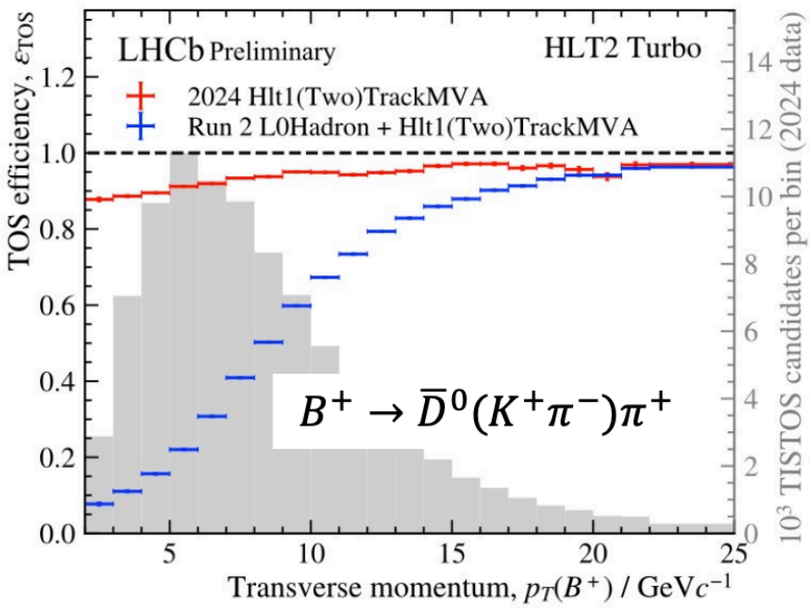
- 40 MHz avec O(500) GPUs
- Pouvait accueillir plus d’algorithmes que prévu
- Efficacité du tracking comparable entre GPU et CPU!

A la fin

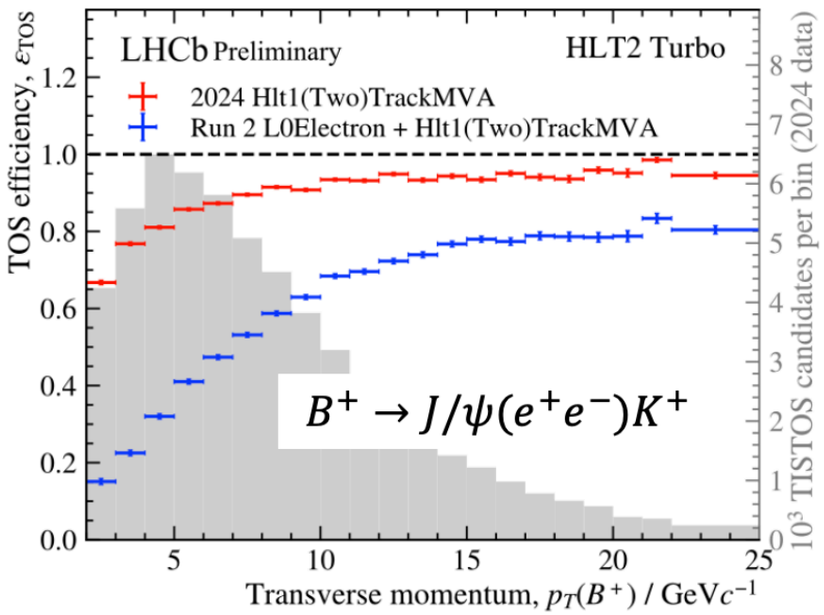
Muonic



Hadronic

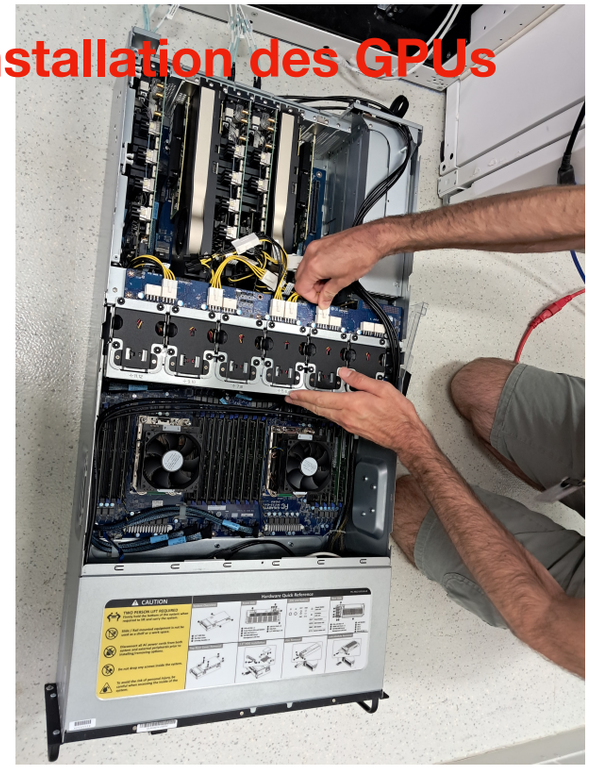


Electronic



Le concept du trigger de LHCb avec un système purement software a été validé avec les données de Run 3!

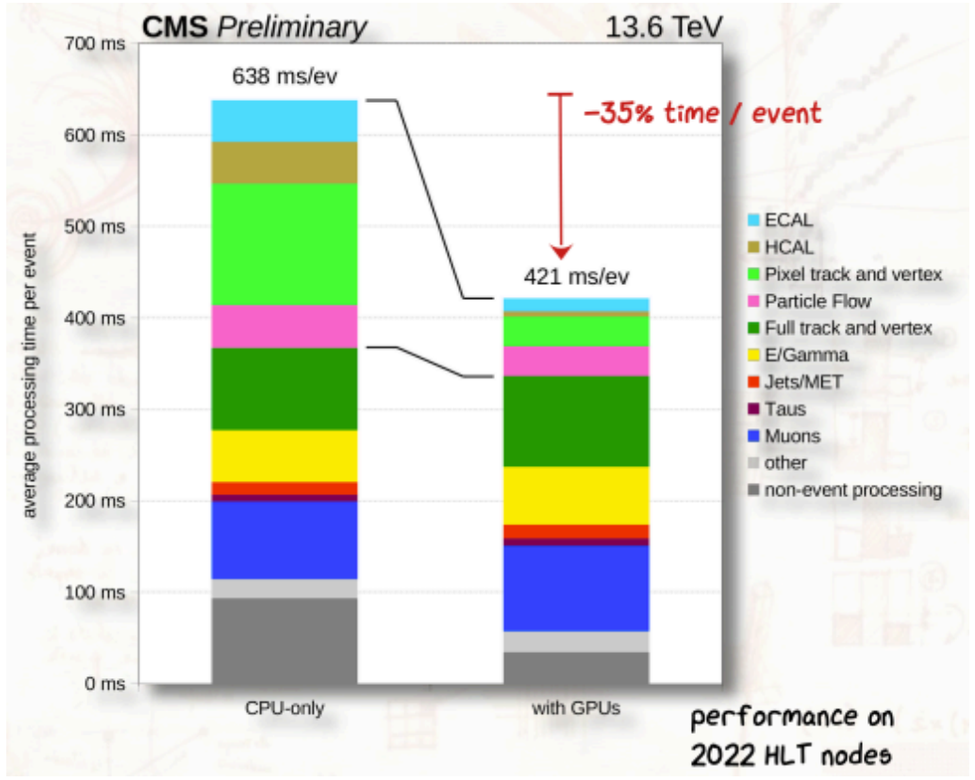
Dans la vraie vie

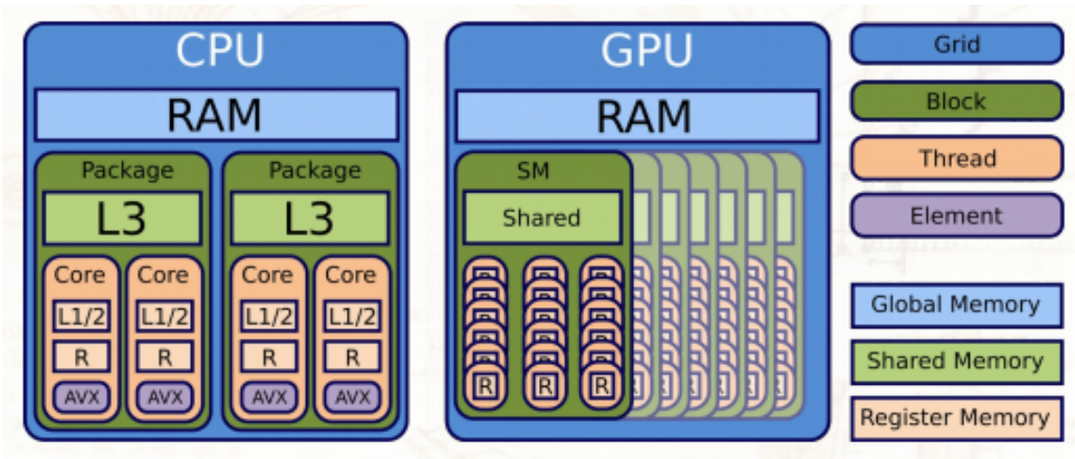


Autres exemples des systèmes avec GPUs

CMS:

- En 2022, CMS a construit une ferme mixte CPU/ GPU pour son étape HLT → **50% augmentation de vitesse en comparant avec la solution purement CPU**
- Ferme très hétérogène, avec plusieurs types des CPUs et GPU → Il nous faut un framework qui peut s'adapter
- Développement alpaka , une C++17 librairie d'abstraction pour les developments hétérogènes
- Projet open source: https://github.com/alpaka-group/alpaka/

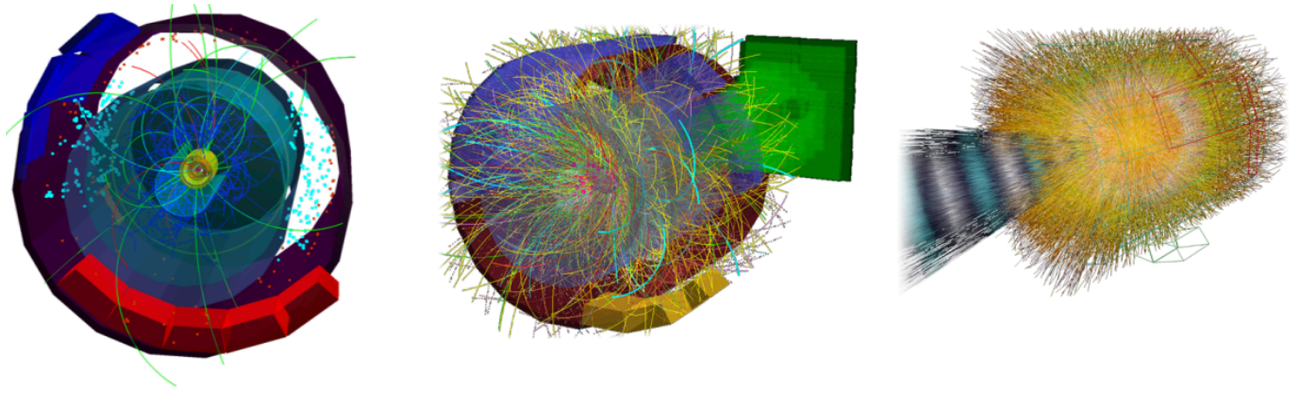




Autres exemples des systèmes avec GPUs

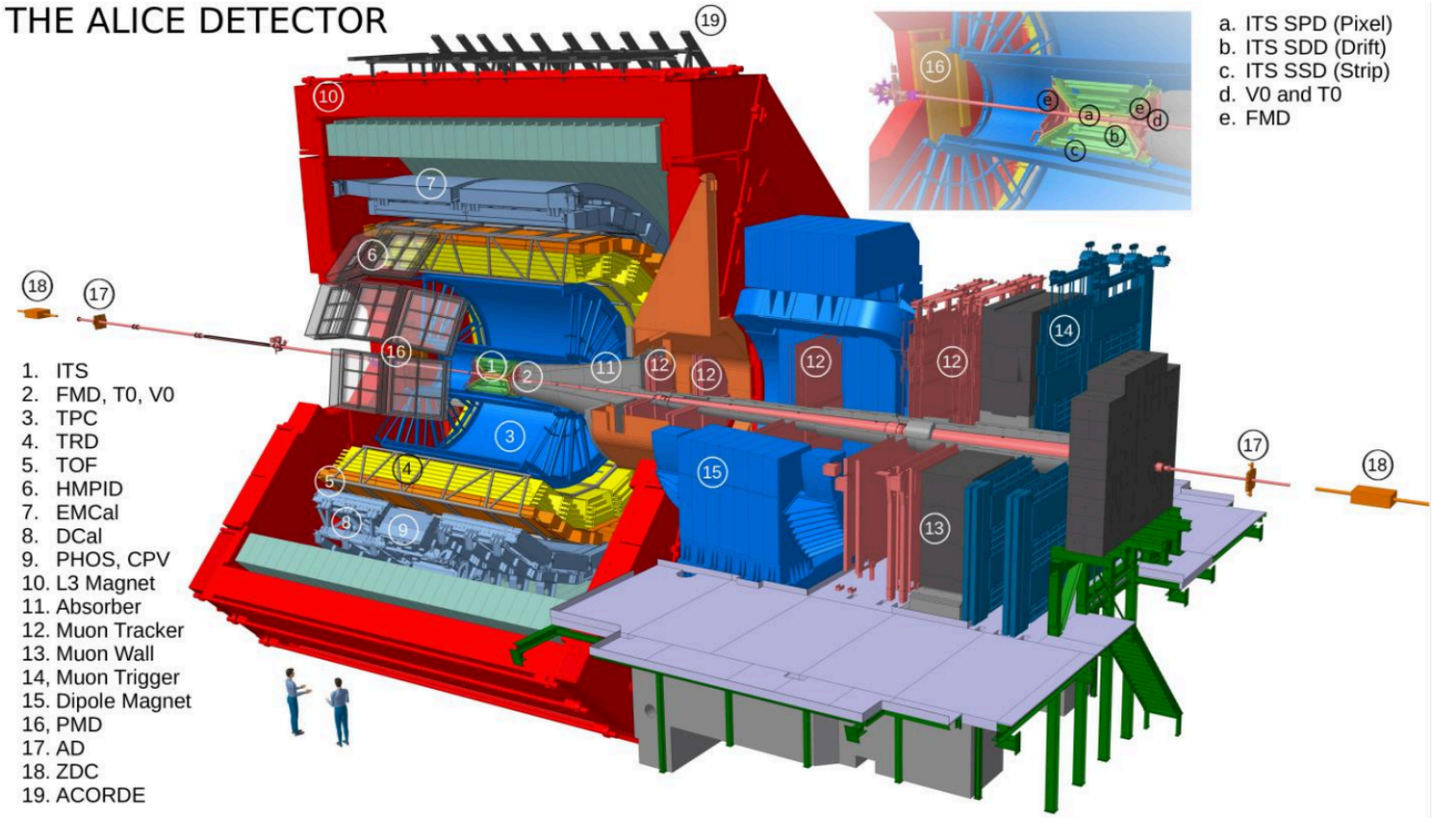
ALICE: experience dédiée a l'etude des interactions PbPb

- PbPb vs p-p : fréquence des interactions beaucoup moins élevée (~x0.001%)
- Mais avec des évènements beaucoup plus grandes (x1000)



p-p *p-Pb* *Pb-Pb*

- Le but pour Run 3: un grande sample de minimum bias
  - Pas de trigger (ni hardware ni software)
  - Read-out continu
  - Compression des données en ligne pour réduire le taux - reduction d'un facteur 20
  - Avec une ferme mixte CPU/GPU



FPGAs comme accélérateurs

FPGAs font deja partie des chaines DAQ

Peut-on les imaginer comme accélérateurs (i.e. intégrés dans les fermes de calculs)?

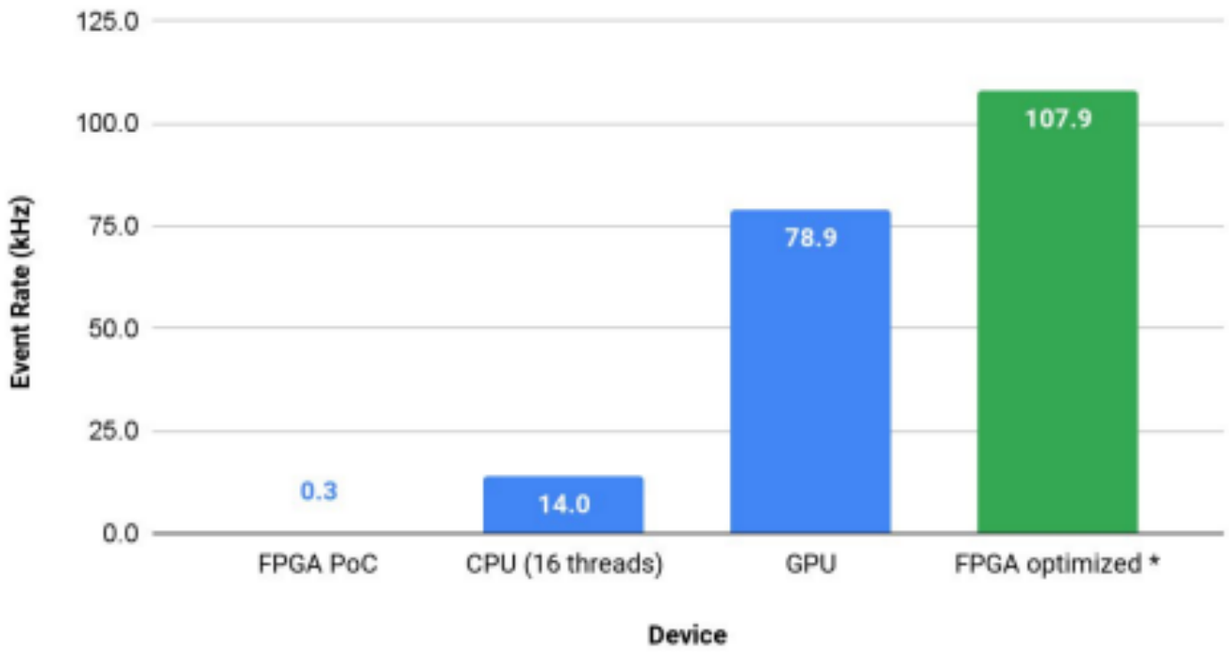
Certaines tâches se prêtent mieux aux FPGA en raison de leur capacité à gérer des opérations au niveau du bit et des opérations combinatoires.

Cependant : il faut des langages de description et une expertise en firmware.

Utils pour traduire les codes c++/python to HDL (High Level Synthesis). Exemples:

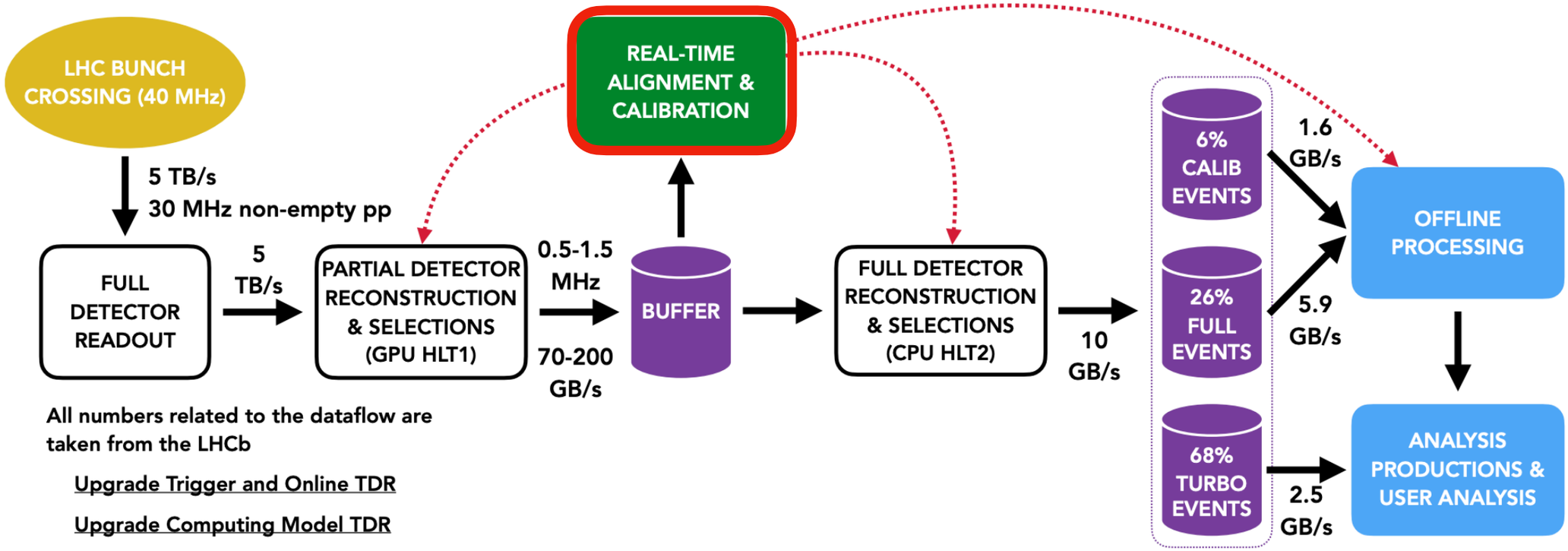
- Intel oneAPI
- hls4ml

Cela nécessite tout de même de repenser les algorithmes et de les optimiser en fonction des spécificités des architectures FPGA pour atteindre les mêmes performances.



A. Perro et al, CHEP 2024

L'implementation LHCb du concept RTA



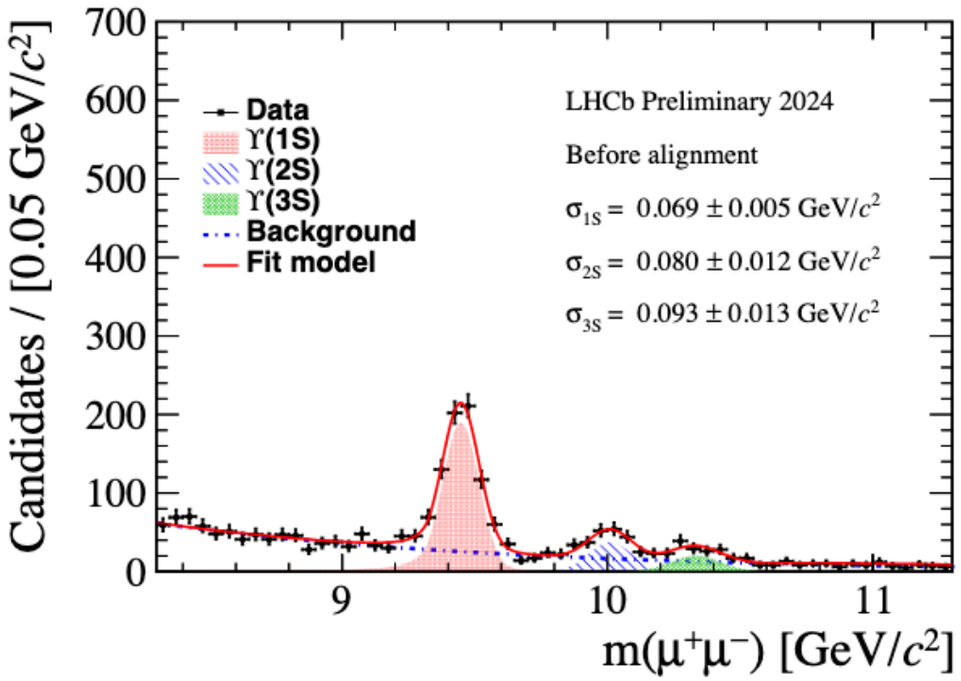
All numbers related to the dataflow are taken from the LHCb

Upgrade Trigger and Online TDR

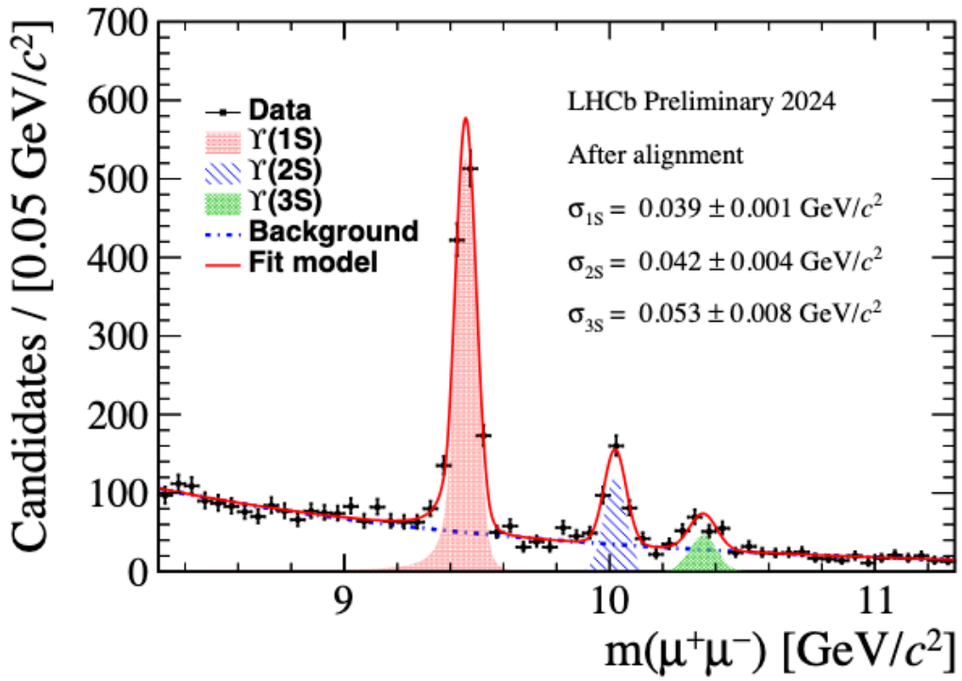
Upgrade Computing Model TDR

L’importance d’alignement et calibration en temps reel

Avant alignement

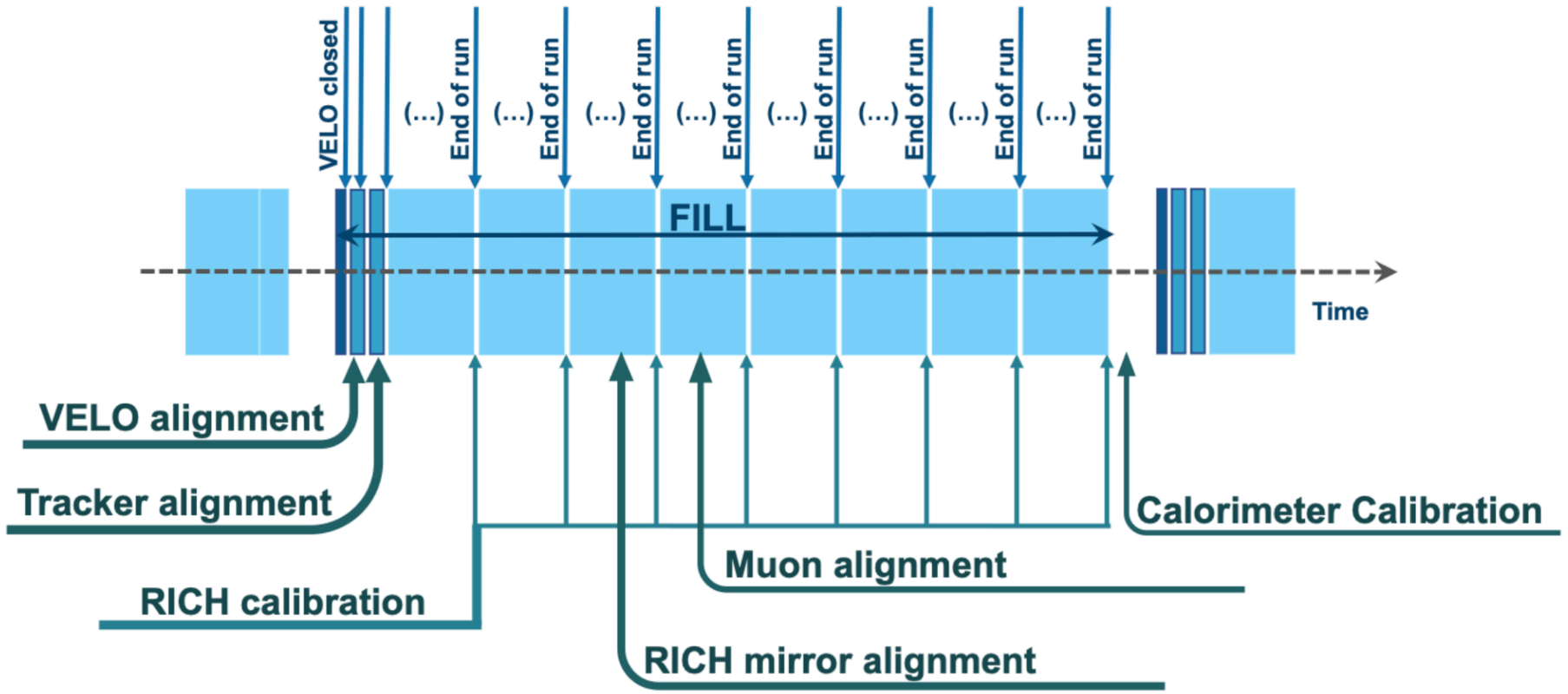


Après alignement

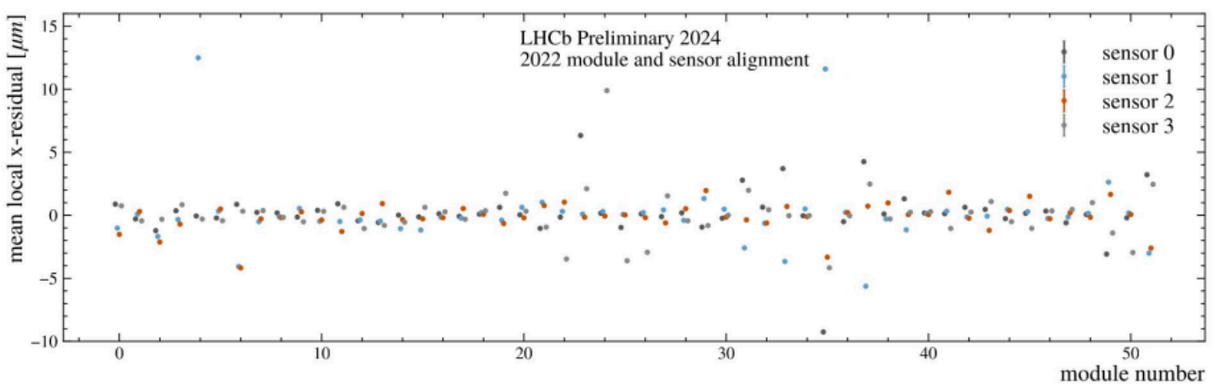


- Performance “offline” ne peut pas être atteinte sans une très bonne alignment & calibration du détecteur
- Si on sauvegarde que l’info reconstruite, il faut aligner en temps reel!
- A LHCb: taches d’alignement les plus rapides prennent < 1 minute pour produire des constantes d’alignement

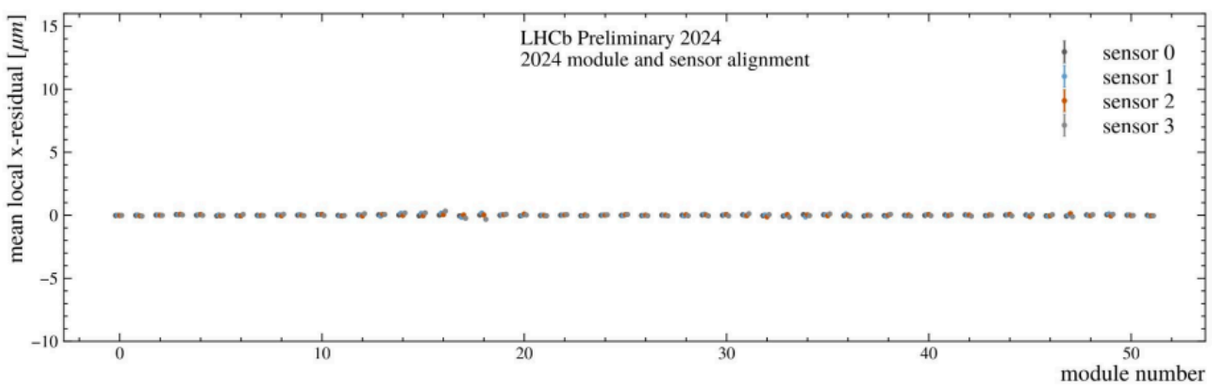
L’importance d’alignement et calibration en temps reel



Before alignment

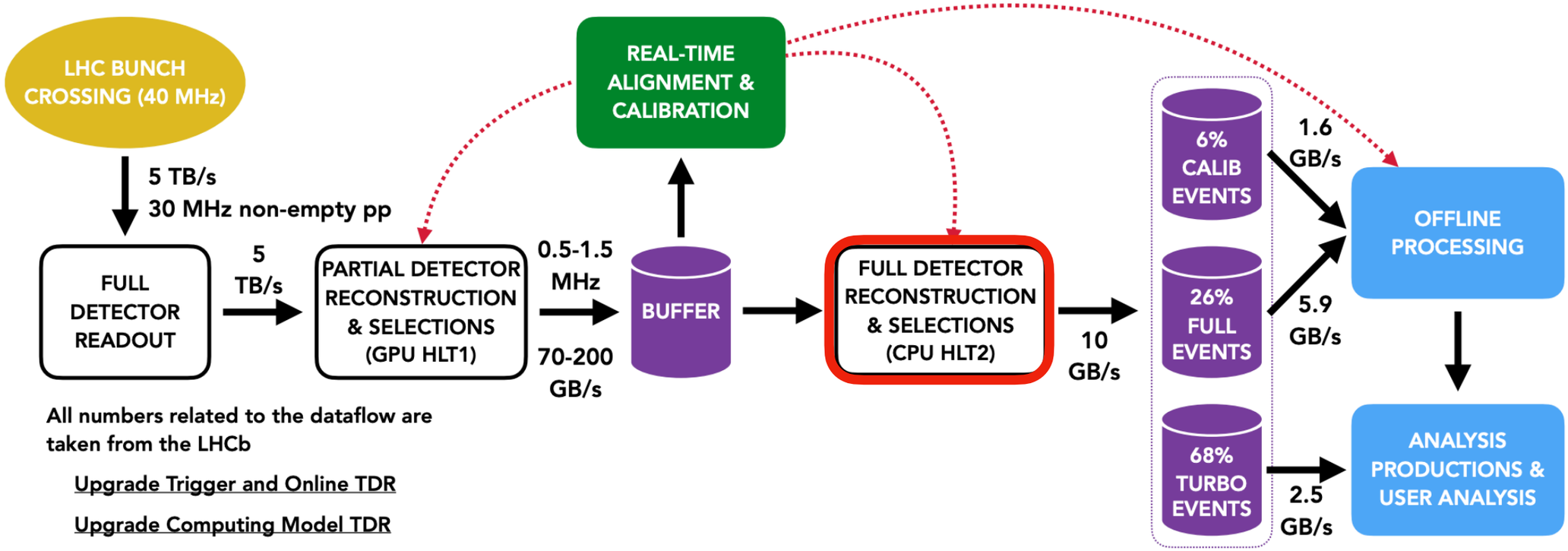


After alignment



[LHCB-FIGURE-2024-009]

L'implementation LHCb du concept RTA



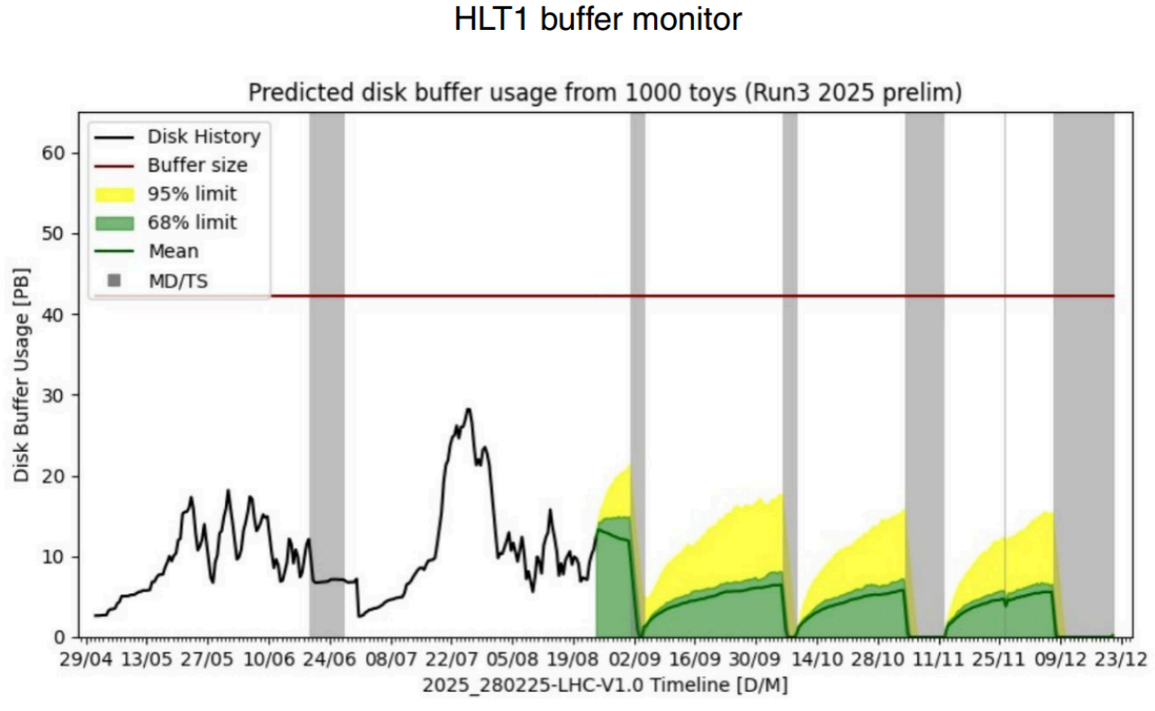
All numbers related to the dataflow are taken from the LHCb

Upgrade Trigger and Online TDR

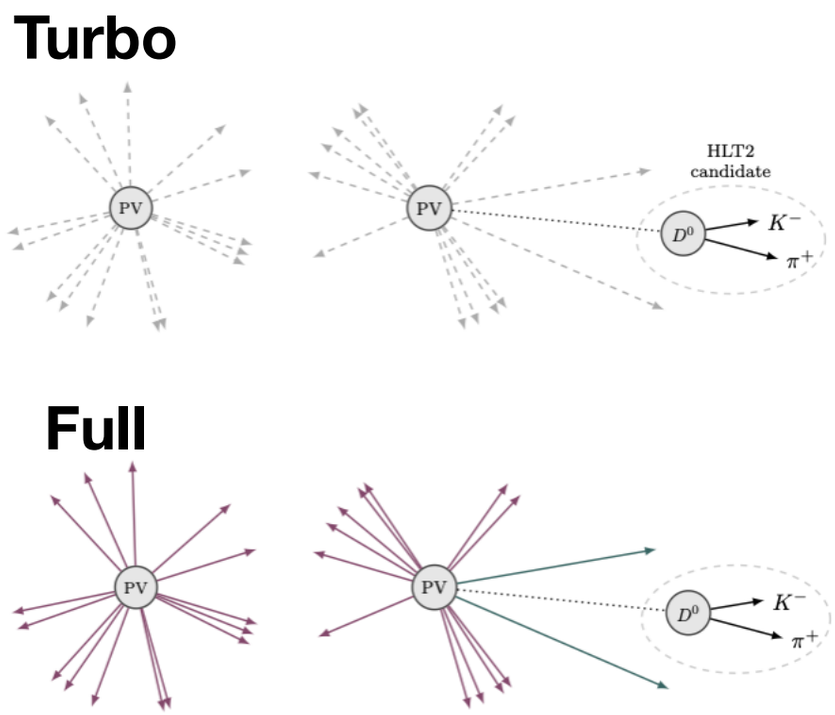
Upgrade Computing Model TDR

L’etape HLT2 de LHCb

- Est un trigger CPU (classique)
- Implementé dans une ferme de ~4000 CPUs
- Est un étape quasi-online: 32 PB de buffer nous permettent d’attendre un peu pour les constants d’alignement et de calibration
- Regle: pour avoir un système qui marche bien, HLT2 throughput doit être ~ 0.5 HLT1 output rate (LHC a une efficacité d’environ 50%)

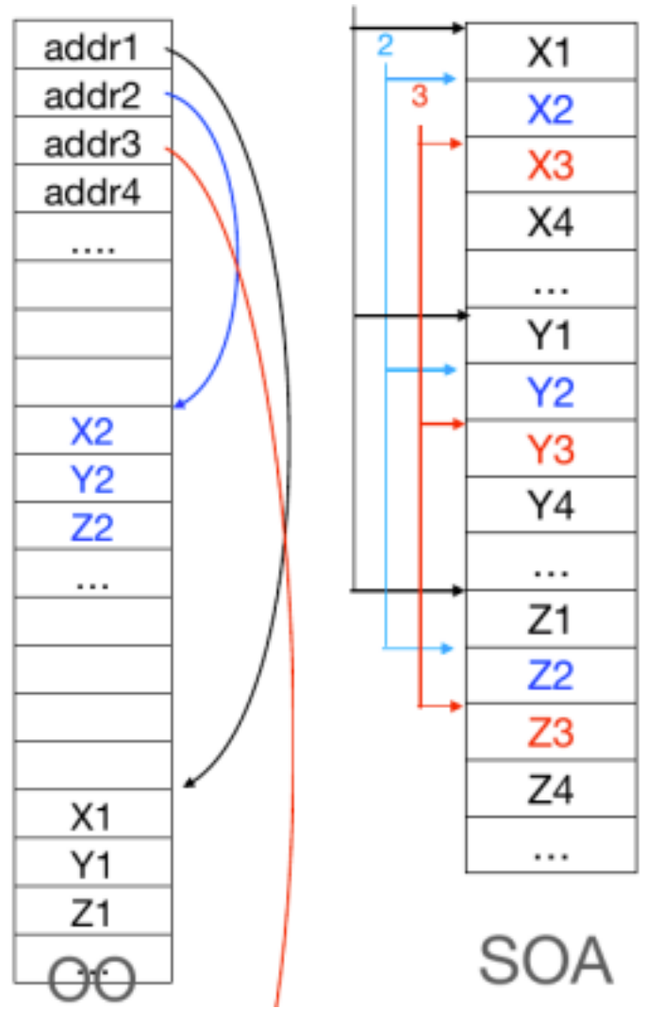


- 3 façons de sauvegarder les données finales
  - Turbo (68%): sauvegarder seulement la désintégration interessante
  - Full (26%): sauvegarder l’info reconstruite complete
  - Calib (6%): sauvegarder l’info brute et reconstruite complete (pour calibrations et etudes supplémentaires offline)

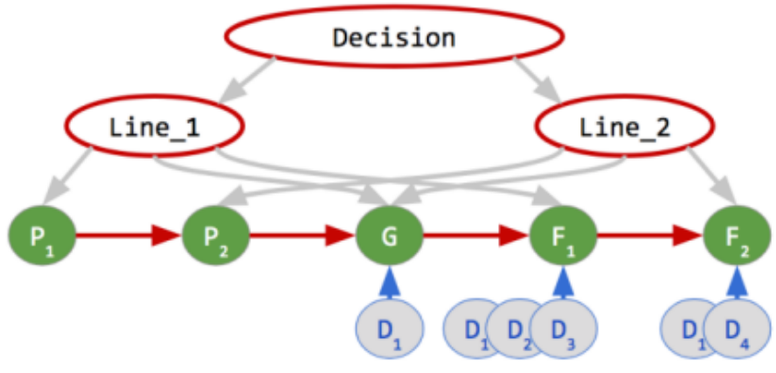


Les algorithms CPUs

- Le deuxième étape du trigger LHCb est toujours un trigger soft “classique” (purement CPU)
- Doit faire une reconstruction complète de l’évènement avec la meilleure qualité possible
- Selections très variées: O(4000) types de selection (lines)
- Quelques strategies d’optimisation des calculs:
  - Vectorisation: chaque thread CPU traite un évènement different
  - Utilisation de layout memoire “Structure of Arrays (SoA)”, pour optimiser les operations vectoriels
  - Precision flottante quand possible

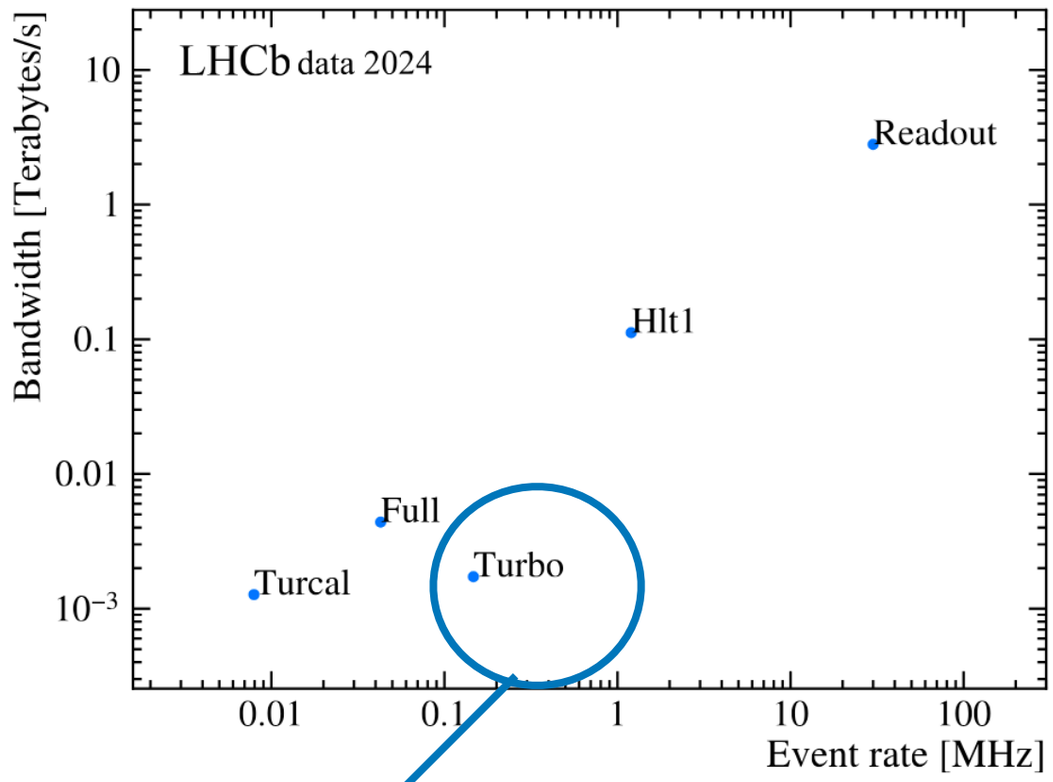


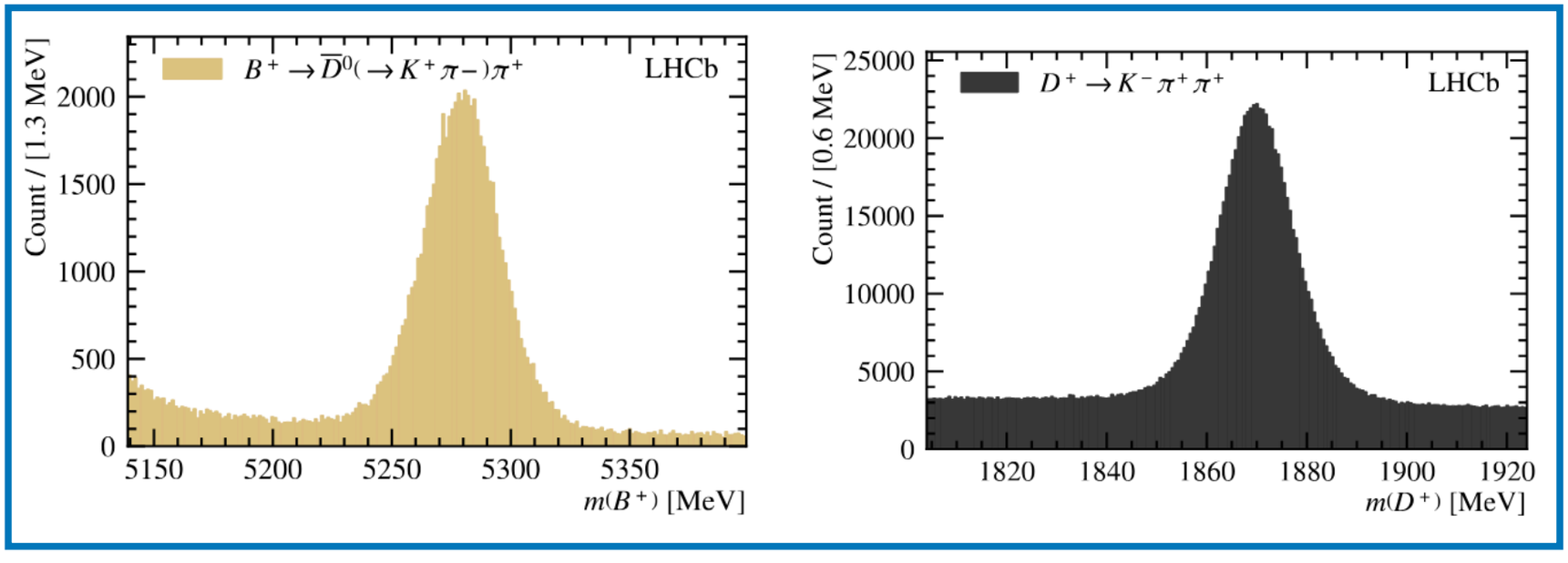
- O(4000) algorithmes - une problématique de dependences et ordre très compliquée:
  - Développement des utils pour automatisation de control et data flow



Directement sortis du four

- Des données de qualité offline qui sortent prêtes a être analysés!





Tendances futures

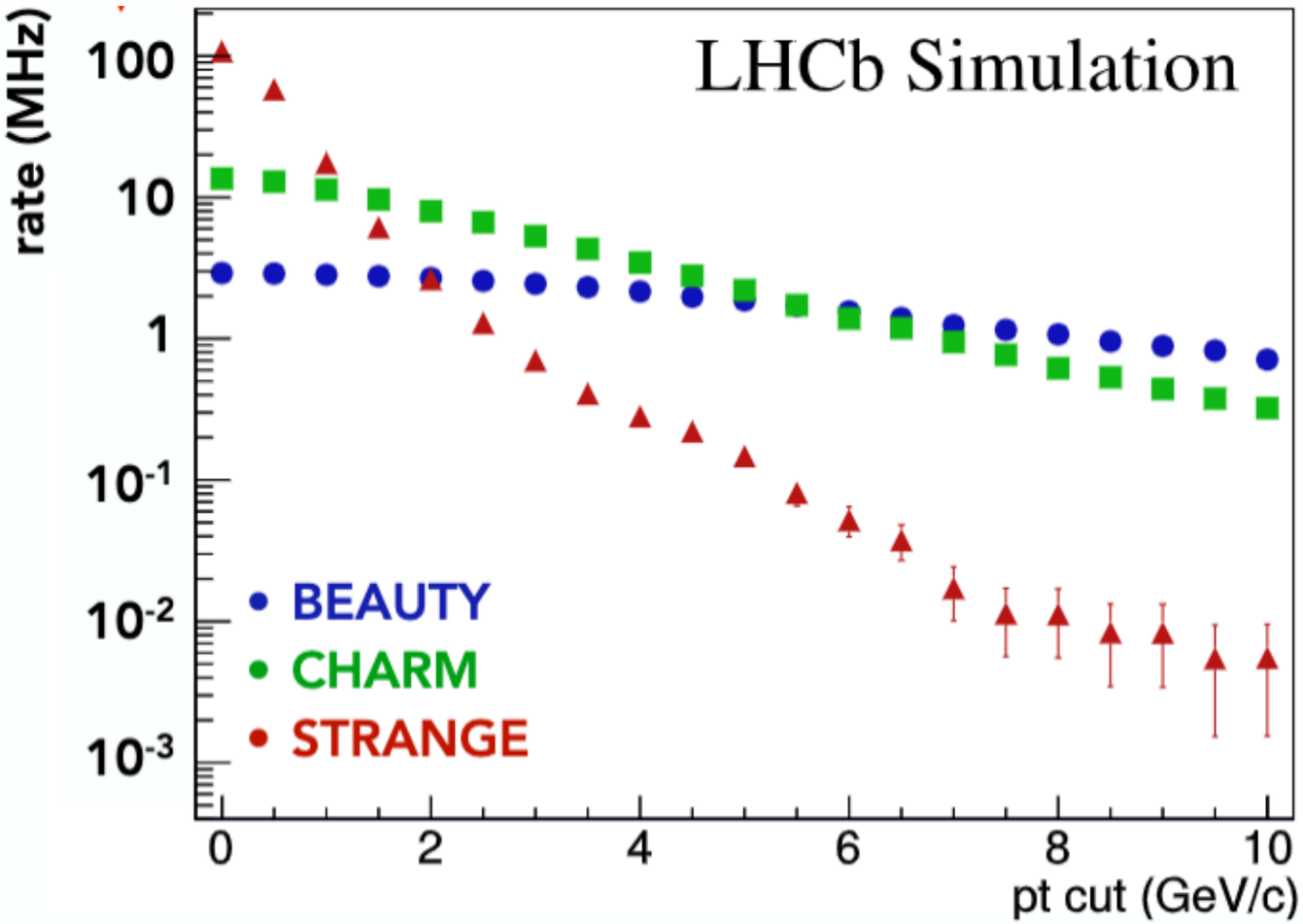
Encore plus des données

Pour HL-LHC, LHCb va augmenter sa luminosité d'encore un facteur ~5

Presque chaque collision contient un vertex displacé… comment on peut faire un trigger?

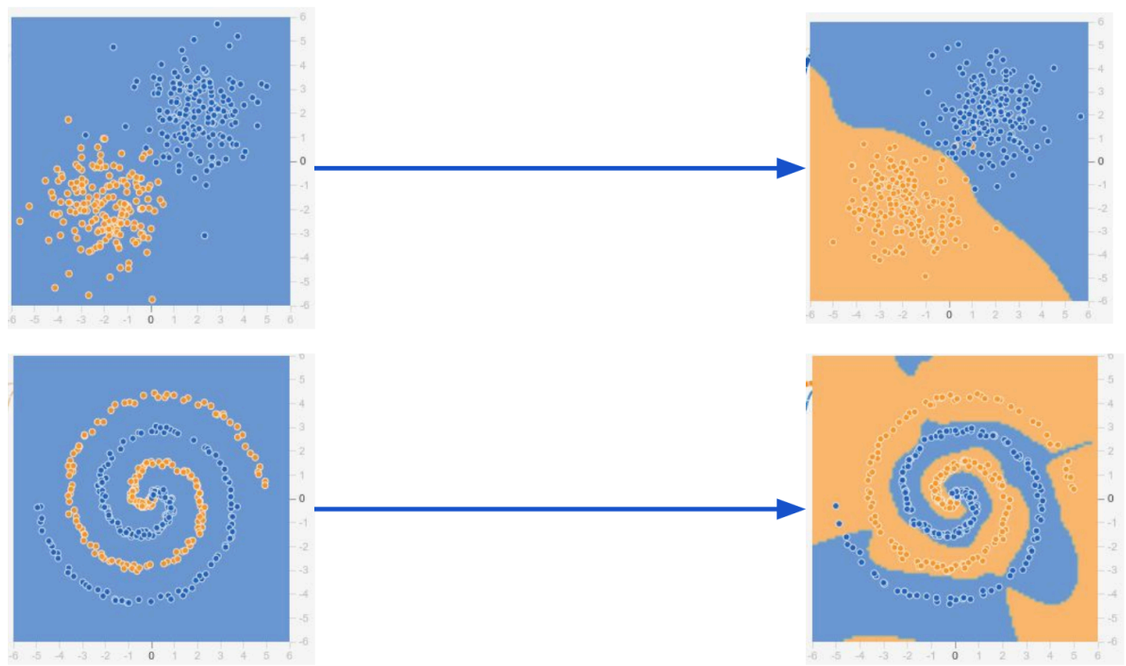
Nous avons pas de réponse pour l'instant… mais il y a des chemins a explorer:

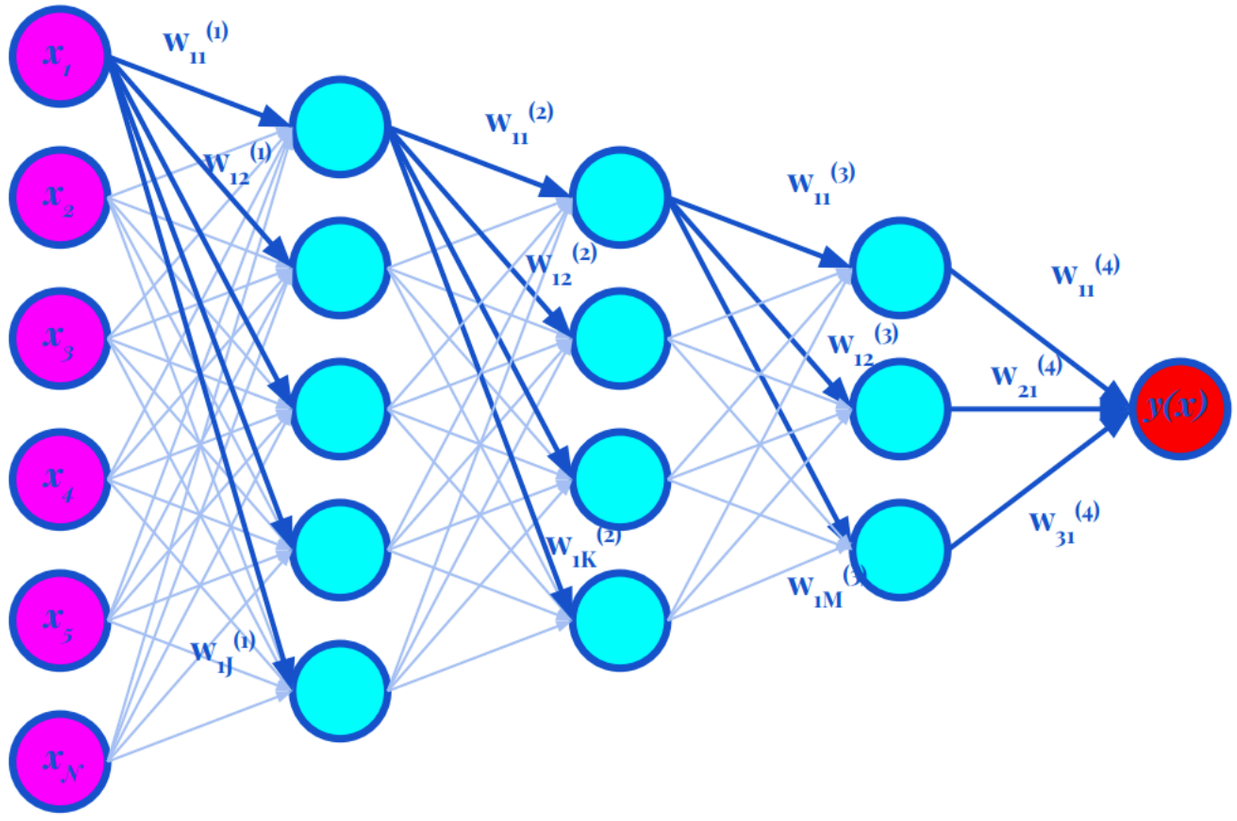
- Utilisation du timing ?
- Faire du pre-processing avant la ferme des calculs (pas a confondre avec un trigger hardware!)?
- Decisions exclusives deja au niveau de HLT1?
- Un melange de tout?



Il faut en parler: IA

- L'utilisation d'IA dans la physique des particules a une histoire assez longue
- **First recorded use I could find of ML in particle physics was 1988!**
- Un example d'IA (ANN):
  - Les données entrent dans l'input (plusieurs variables)
  - Plusieurs couches cachés. Chaque passe les données avec des poids
  - Les poids sont calculés dans la phase d'entrainement du réseau, dans une procedure de minimisation des pertes
  - Pour un bon entrainement, on a besoin d'un bon proxy pour le signal et le bruit du fond

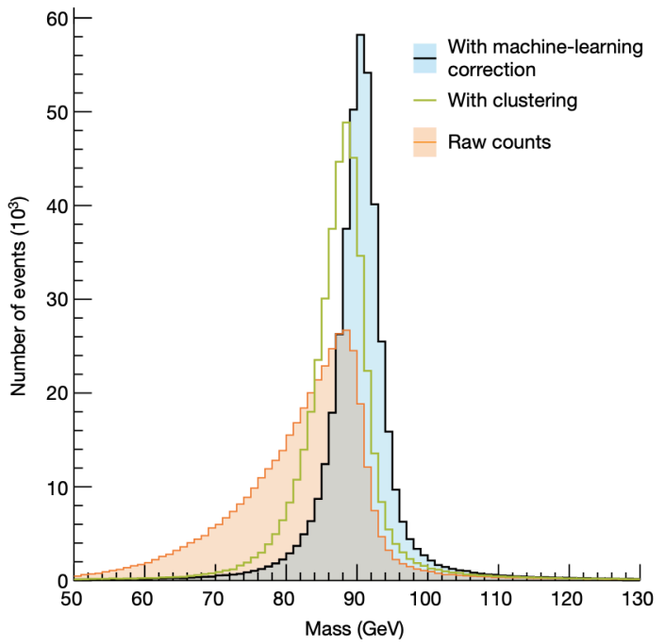


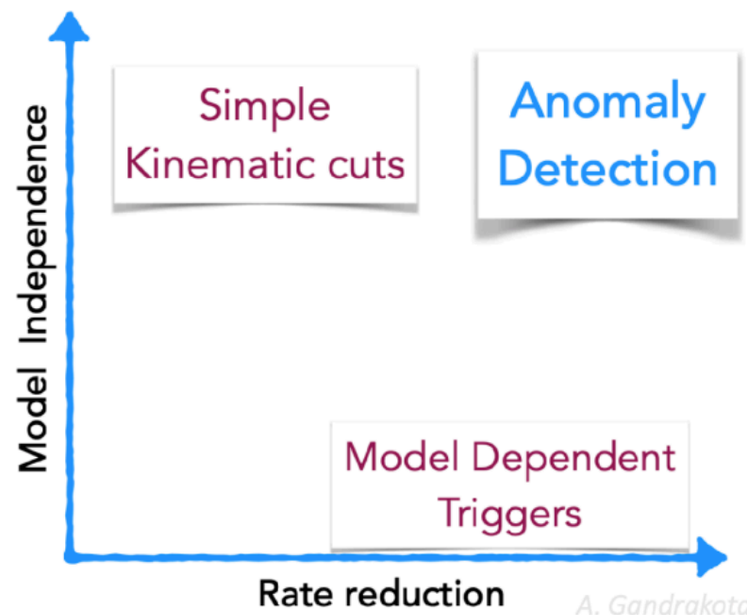


L’IA dans la physique des particules

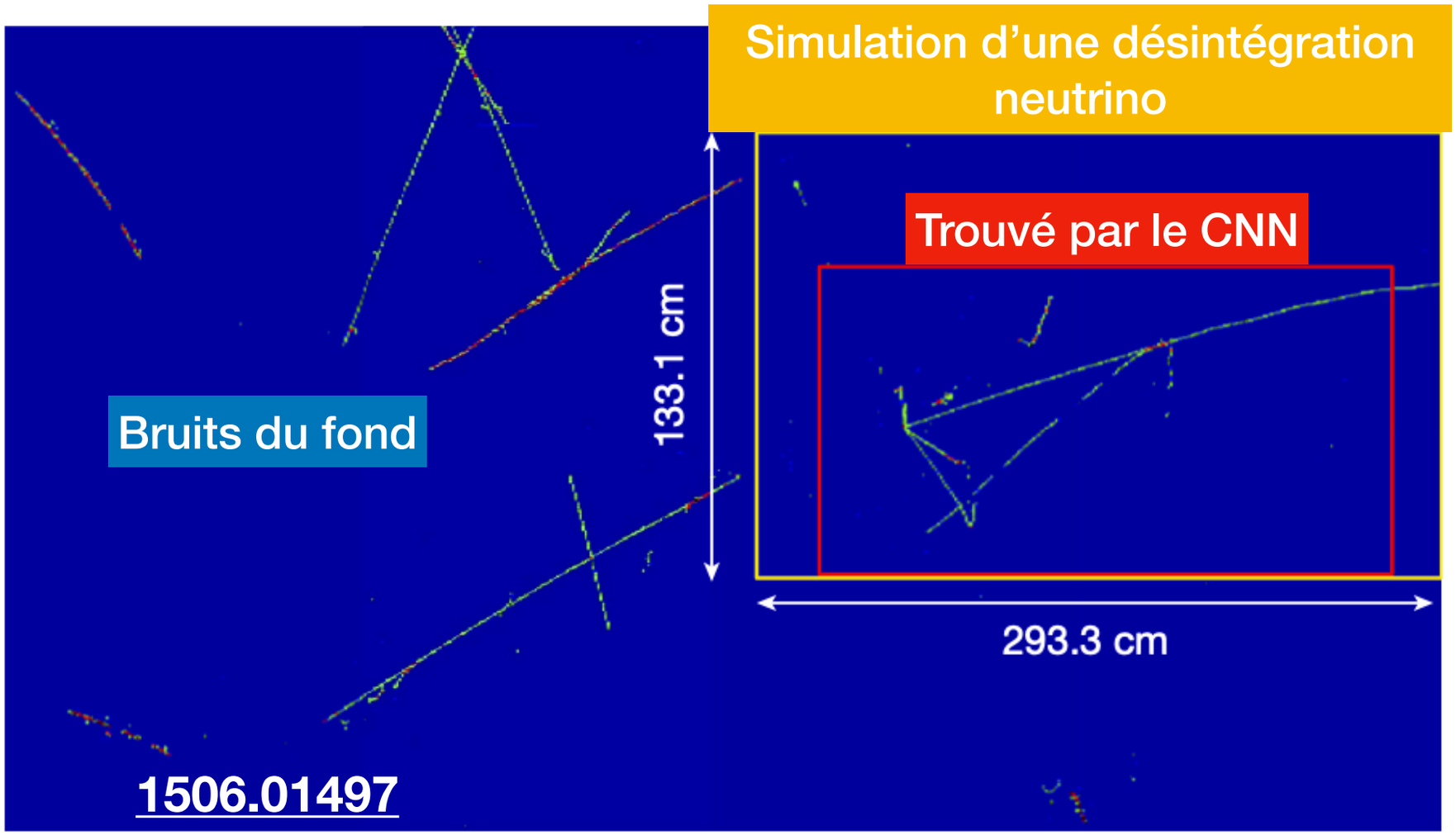
L’utilisation des algorithmes IA en forte hausse a la physique des particules

- Pour les taches “pattern recognition”, selection des données, simulation … !





Mais, la complexité des models modernes IA, est-elle adapté pour le monde online et temps-reel ?



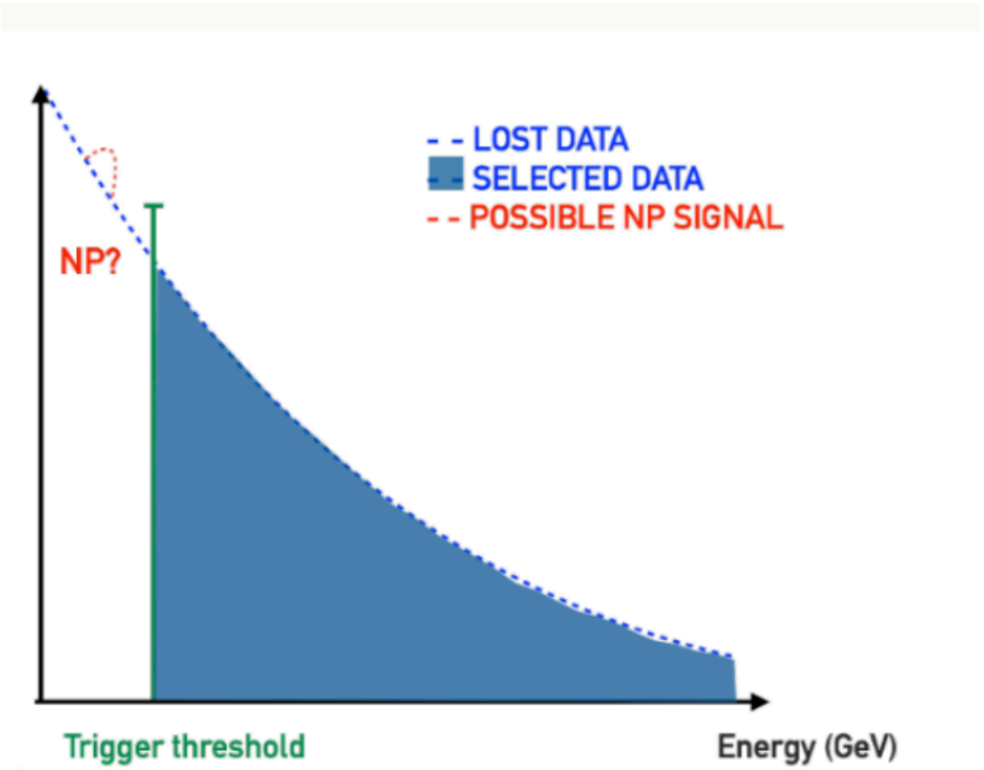
IA comme util d’imagination

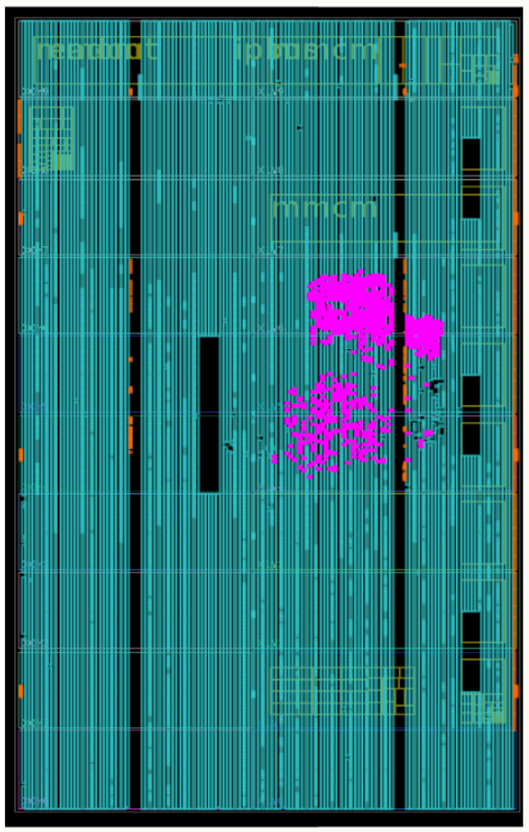
Pour le design d’un bon trigger, il faut savoir ce qu’on cherche…

Mais on peut pas imaginer tous les cas! Et si la nouvelle physique ne resemble pas a ce qu’on imaginait?

IA peut aider: entrainement sans biais (unsupervised learning), pour trouver des anomalies dans les données

Toujours aux physiciens de les interpreter!



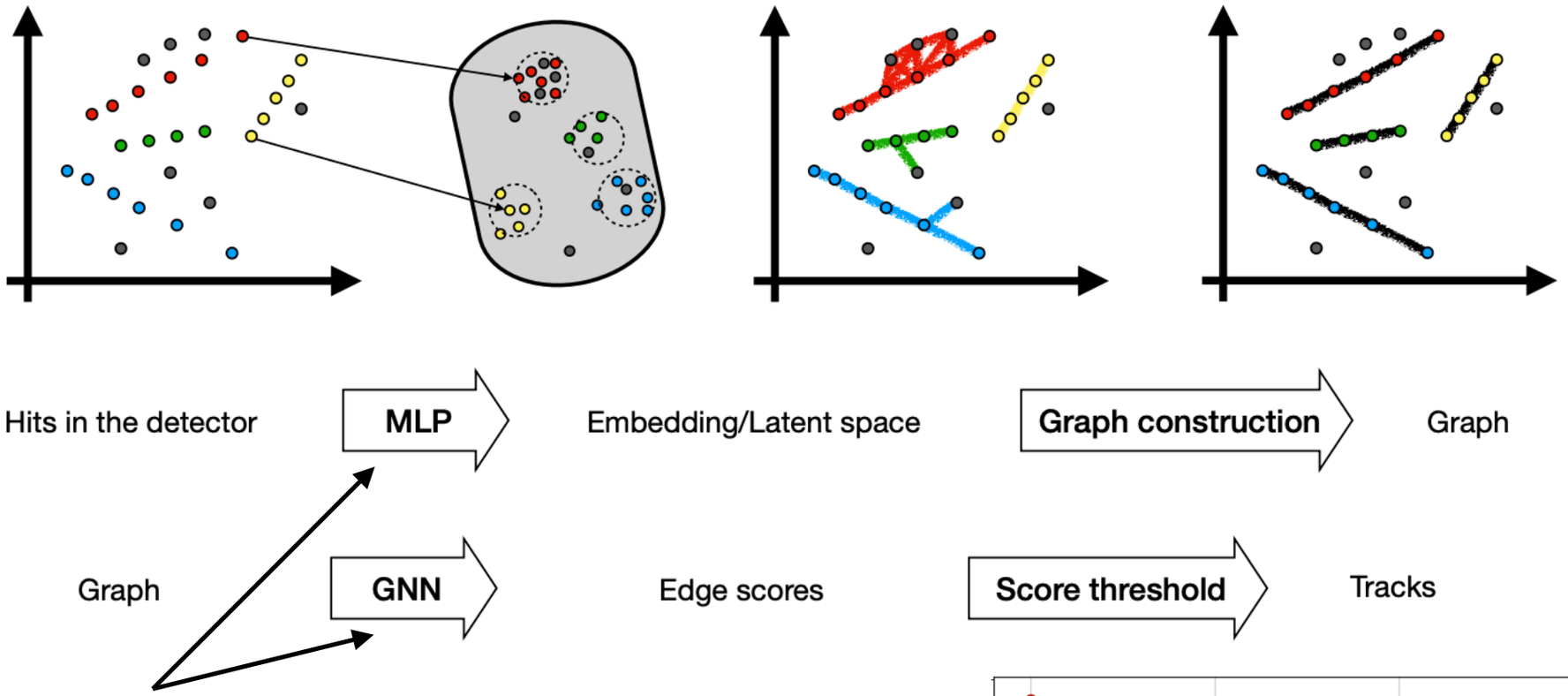


CMS a développé des algorithme IA pour couvrir toutes les bases dans le premier trigger:

- CICADA pour trouver des anomalies dans le calorimètre et AXILOTL pour des anomalies dans toutes les observables du L1
- Développé avec HLS4ML avec optimisations pour base latence (quantisation, optimisation d’espace) - moins de 1% d’utilisation des LUTs et DSPs
- Implémenté pour Run 3 et en train de chercher pour des anomalies!

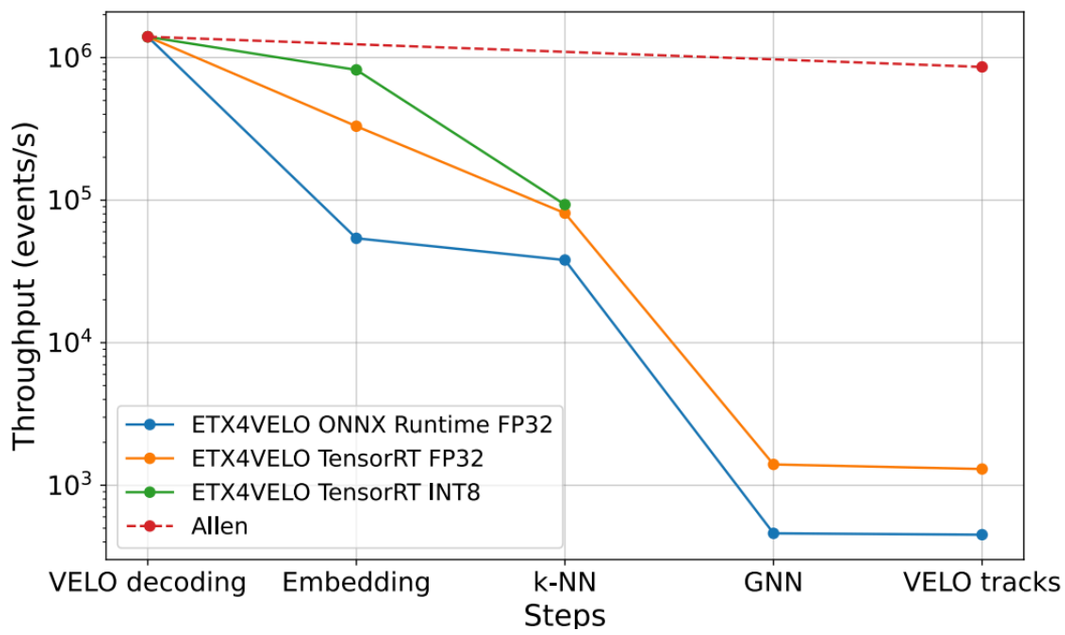
GNN inference dans les GPUs de LHCb

Retournons a la reconstructions des traces a LHCb: problématique bien représentée par des graphes → GNN (Graph Neural Network)



2406.12869

Vitesse depend de la taille du réseau



Implementation dans Allen (GPUs):

- Inference en batches
- Quantisation and optimisation de la pipeline doit être exploré pour atteindre la vitesse demandé par les conditions LHCb

Les defies principaux (deja identifiés) pour FCCee

FCC at a glance:

- Plusieurs étapes d'energie -> besoin d'un système flexible
- Du point de vue DAQ, le plus grand défi sera le pole Z: BC
  - 40 MHz, grand bruit du fond (le rate de physique est ~100 kHz)
  - + détecteurs a haute granularité et besoin de faible taux de matériaux
- 3 scenarios imaginés pour l'instant: triggerless, minimally triggered, hardware trigger

Quelques questions:

- N'importe quel scenario, on attend des ~100 ab^{-1} des données
- Impact des bruits du fond de la machine aux taux des données
- Contraintes de materiel (cables) sur la physique

FCC-ee parameters		Z	W^+W^-	ZH	ttbar
$\sqrt{s}$	GeV	91.2	160	240	350-365
Luminosity / IP	10^{34} cm^{-2} s^{-1}	**140**	20	7.5	1.5
Bunch spacing	ns	25	160	680	5000
"Physics" cross section	pb	35,000	10	0.2	0.5
Total cross section	pb	70,000	30	10	8
Event rate	Hz	**100,000**	6	0.5	0.1
"Pile up" parameter [μ]	**10^{-6}**	2,500	1	1	1

J. Bracinik @ FCC week May '25

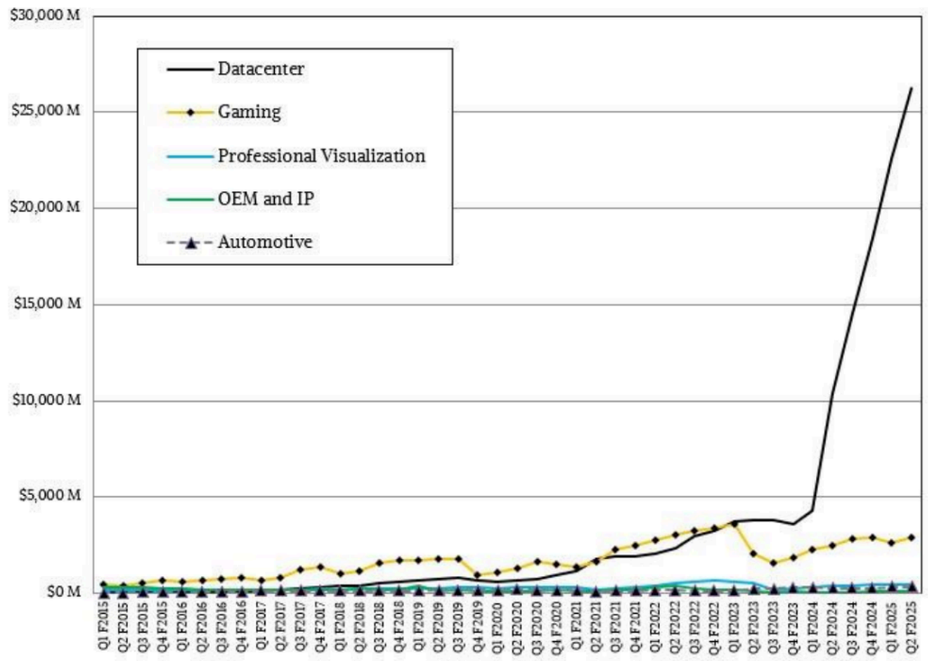
Privilegier les algorithmes du pre-processing et explorer le transfert des données a très haute bande passante?

Un mot sur le marché

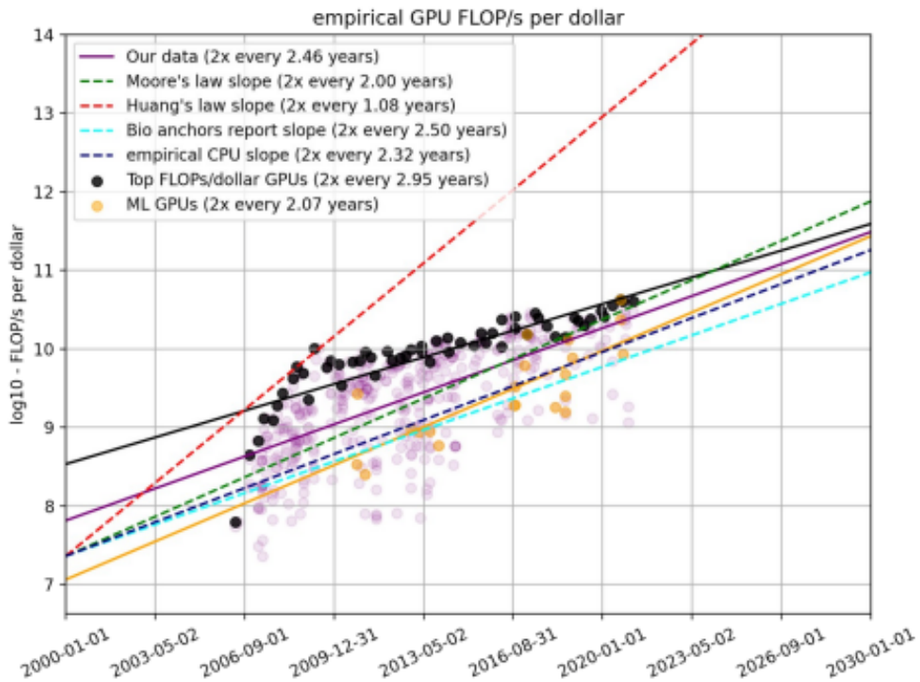
A. Sciama @CHEP24

Peut-on faire de predictions? Très difficile après 1-2 years…

- Les GPUs toujours très promettants mais
  - Il faut faire attention a l'evolution de la **memoire**
  - L'industrie commence a oublier la **precision flotante** (IA n'a pas besoin) - peut-t-on adapter nos algorithmes de reconstruction?
  - Cout d'alimentation et taille des nouvelles GPU en forte hausse - l'integration online qui a été fait par LHCb peut-etre sera pas valable dans 10 ans



Nvidia revenues (Source: The Next Platform)



L'industrie des processeurs évolue trop rapidement en ce moment - il faut éviter a prendre des decisions sur la technologie trop tot, mais plutôt garder les portes ouvertes

Conclusions

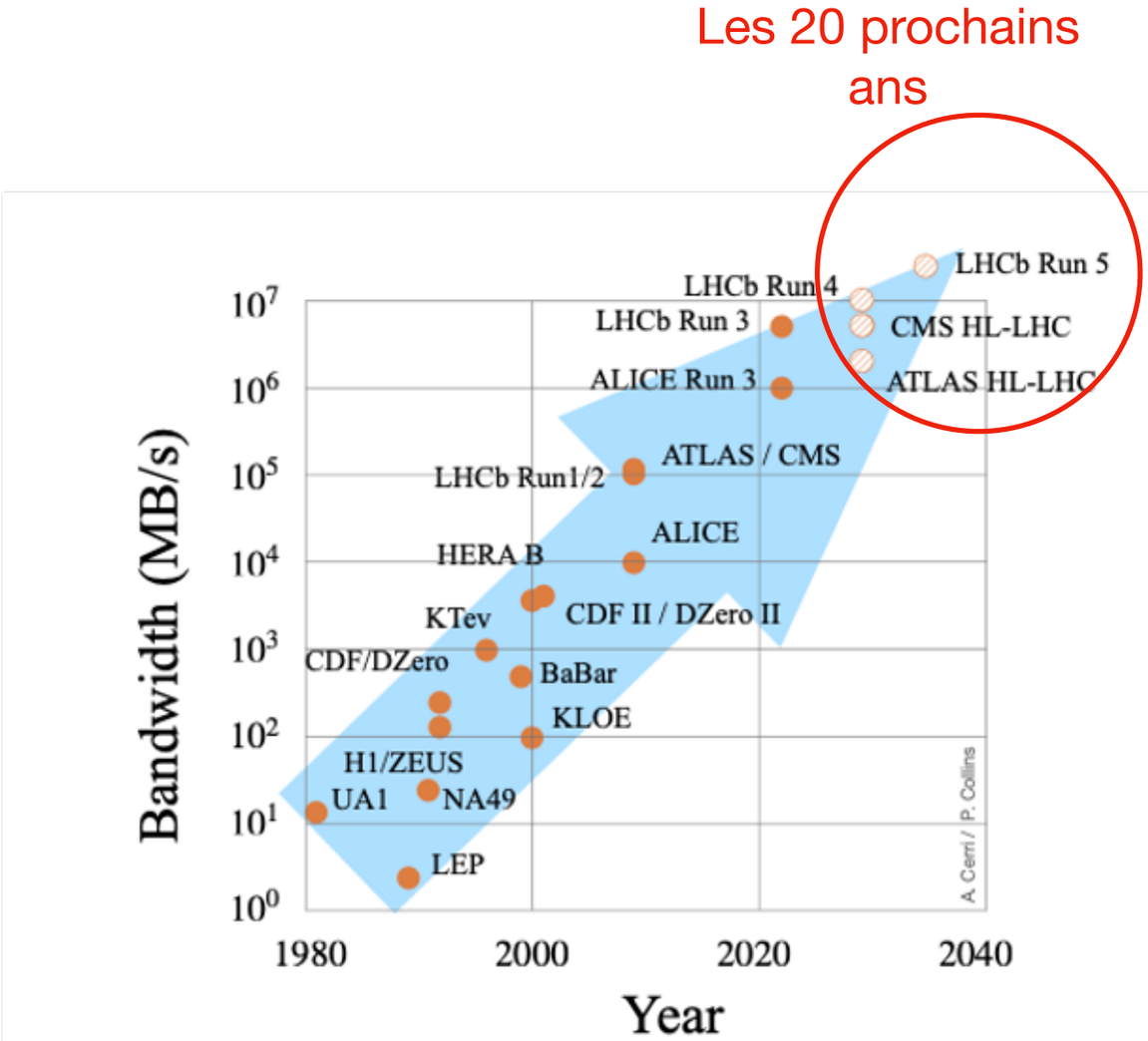
Dans ce cours on a vu:

- Quelques bases sur les systèmes d'acquisition de physique de particules

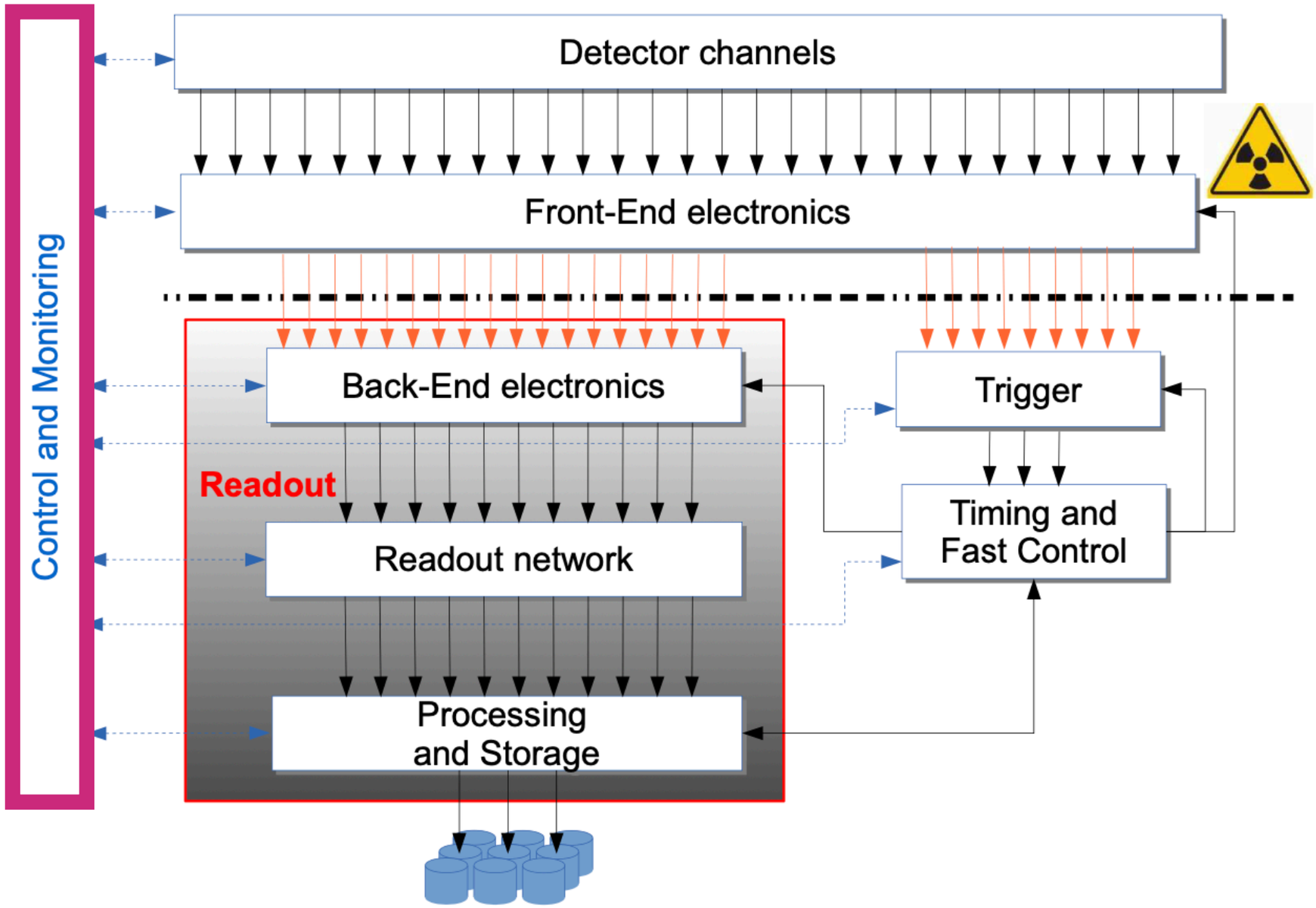
- Les principes des systèmes trigger hardware, qui étaient la base pour toutes les experiences LHC jusqu'a 2018

- Le mouvement vers les systèmes "triggerless" et analyse en temps réel

- Les defies majeurs pour le futur

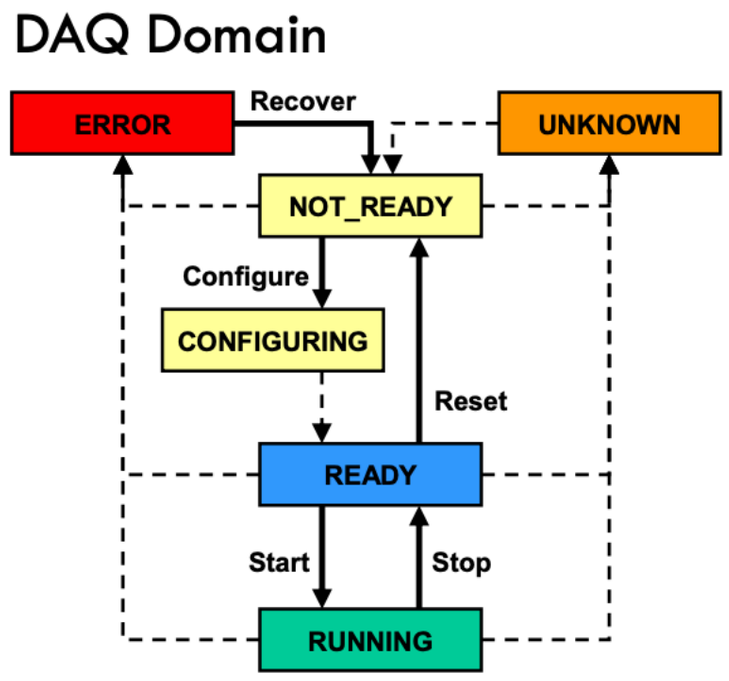


Extra: Contrôle et monitoring

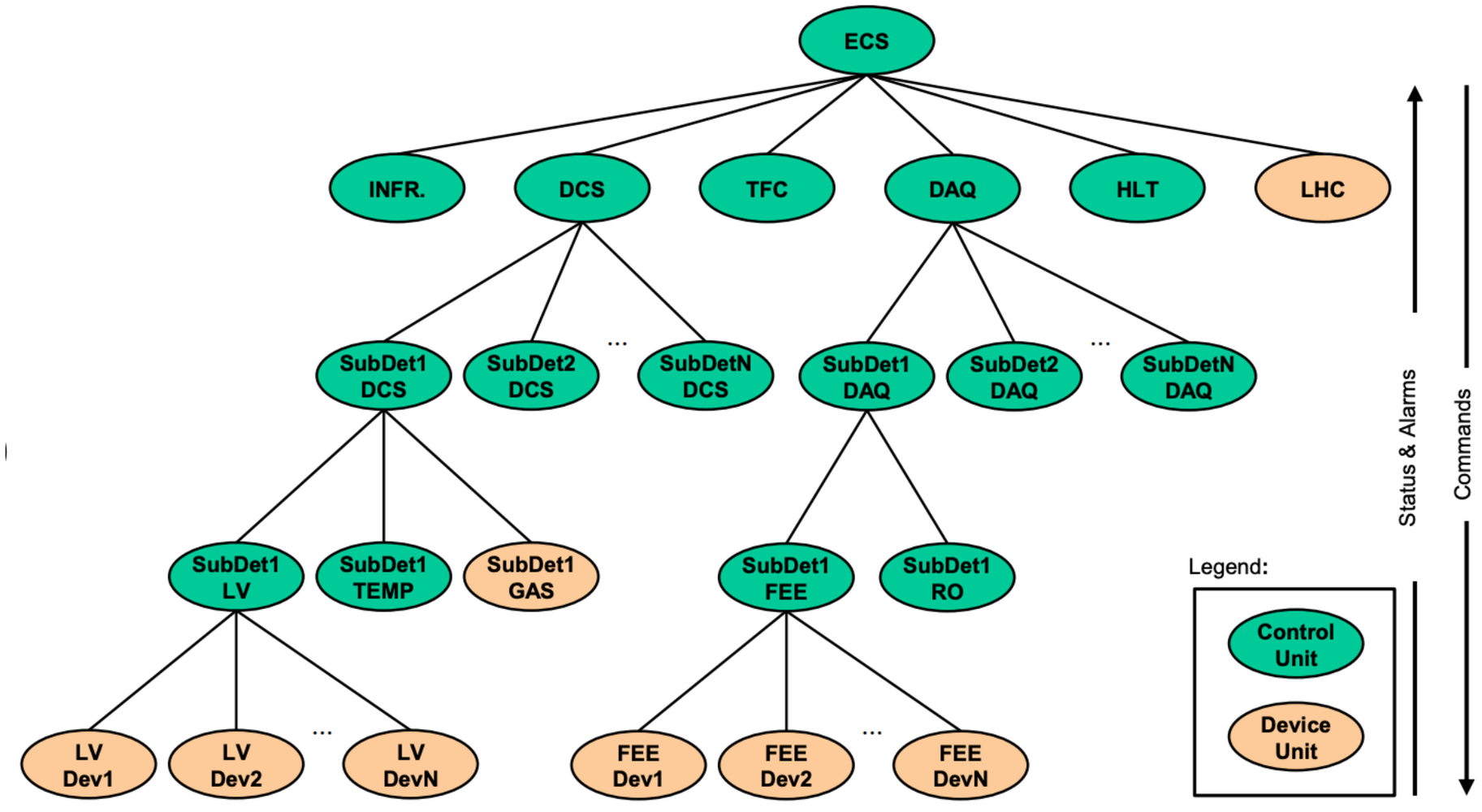


Experiment Control System (ECS)

- Assure la supervision global de l'experience
- Pilotage de l'acquisition des données, de l'EB et le trigger
  - Chargement des paramètres de la prise des données, démarrage, arrêt changement de RUN
  - Detection des erreurs
  - Pilotage des sous-systèmes (slow control, HV, temperature, gaz) et infrastructure (refroidissement ventilation etc)
  - Monitoring
  - Interface avec accélérateur, autres experiences etc

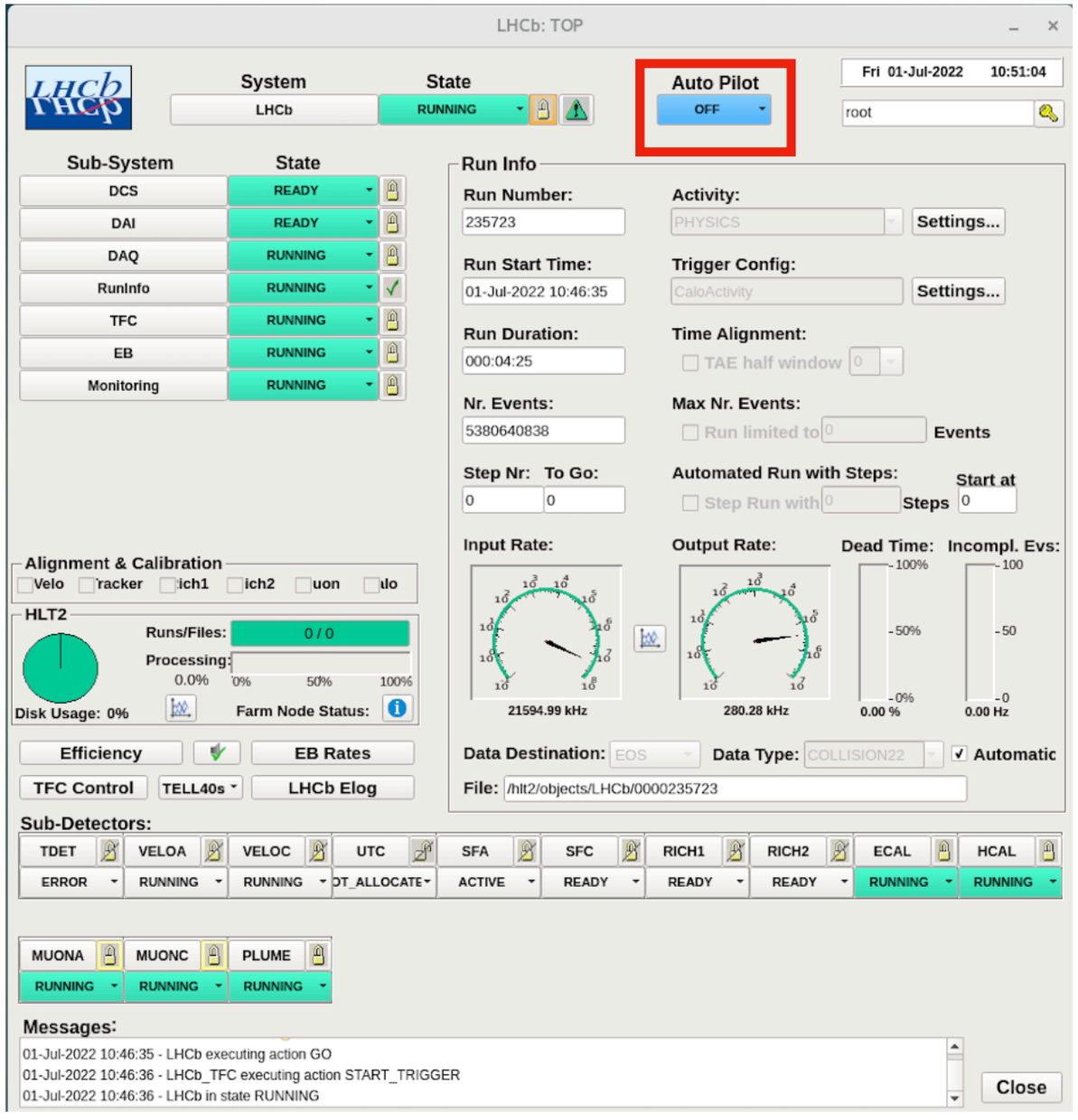


Pilotage des sous-systèmes



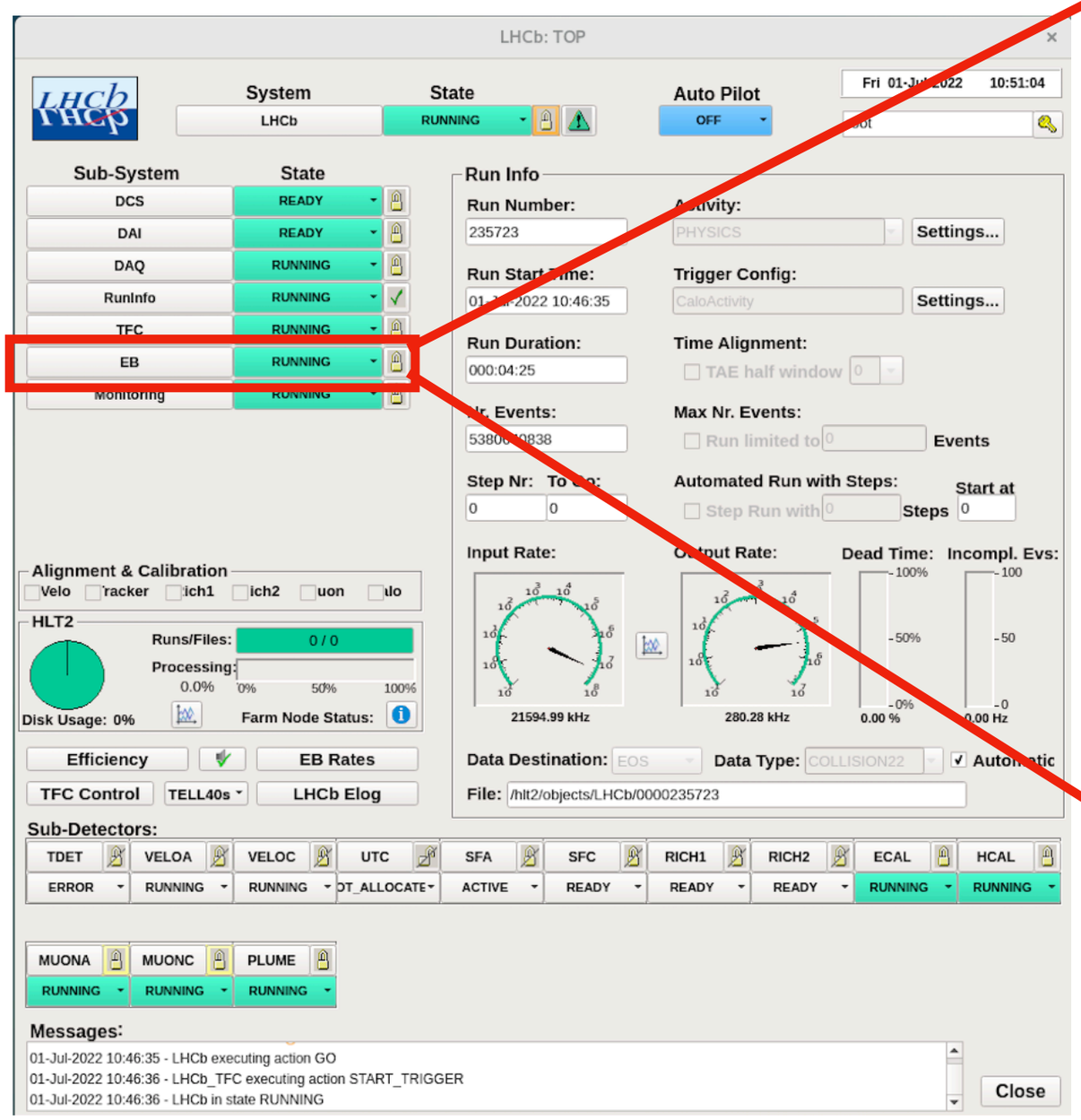
Organisation en arbres hiérarchiques, avec unités physiques (device) et unités de controle (groupent les unités physiques)

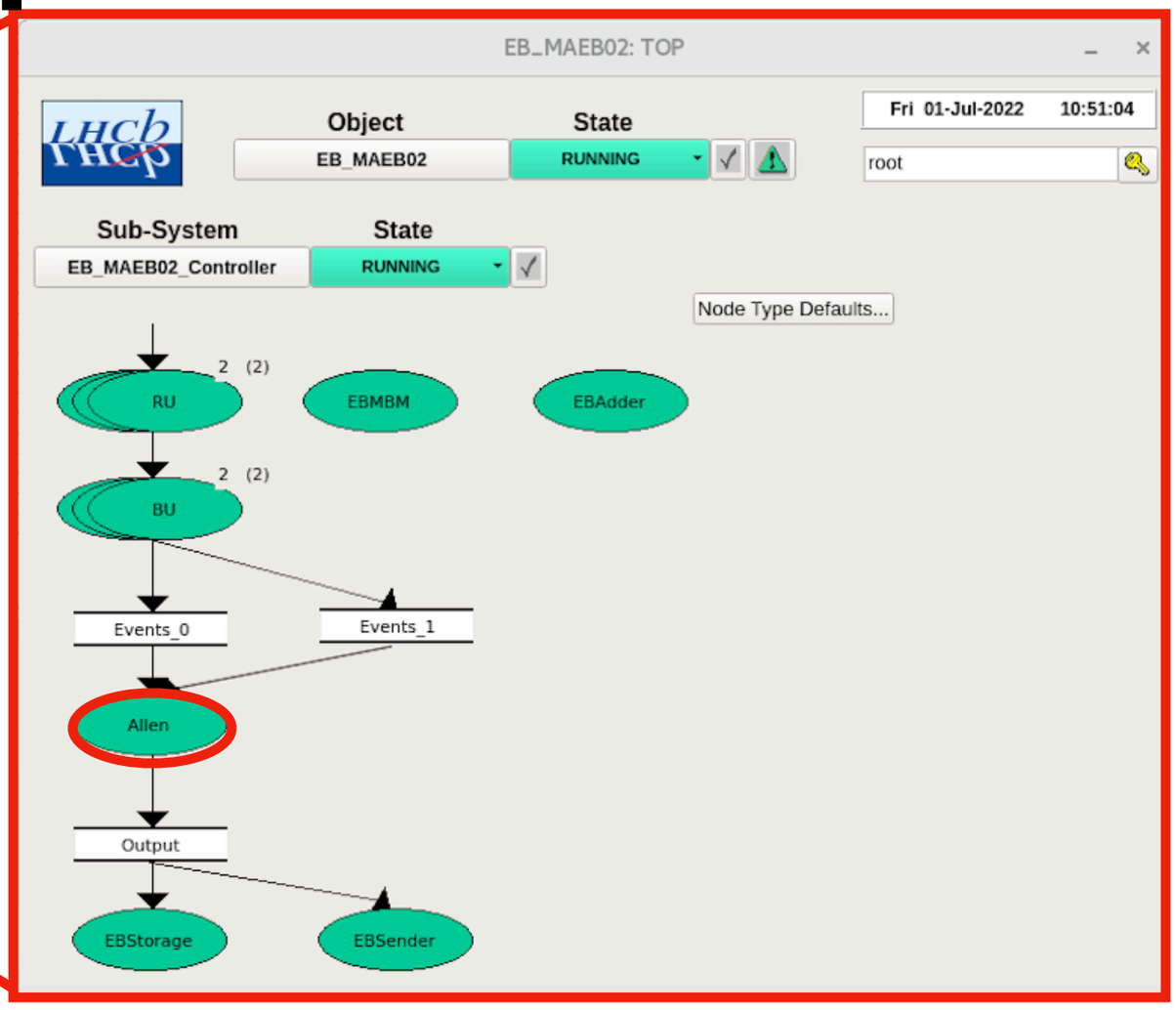
Panneaux de supervision



- Facilité d'utilisation
- Mode automatique de prise des données (auto-pilot) indispensable sur le longue terme

Panneaux de supervision



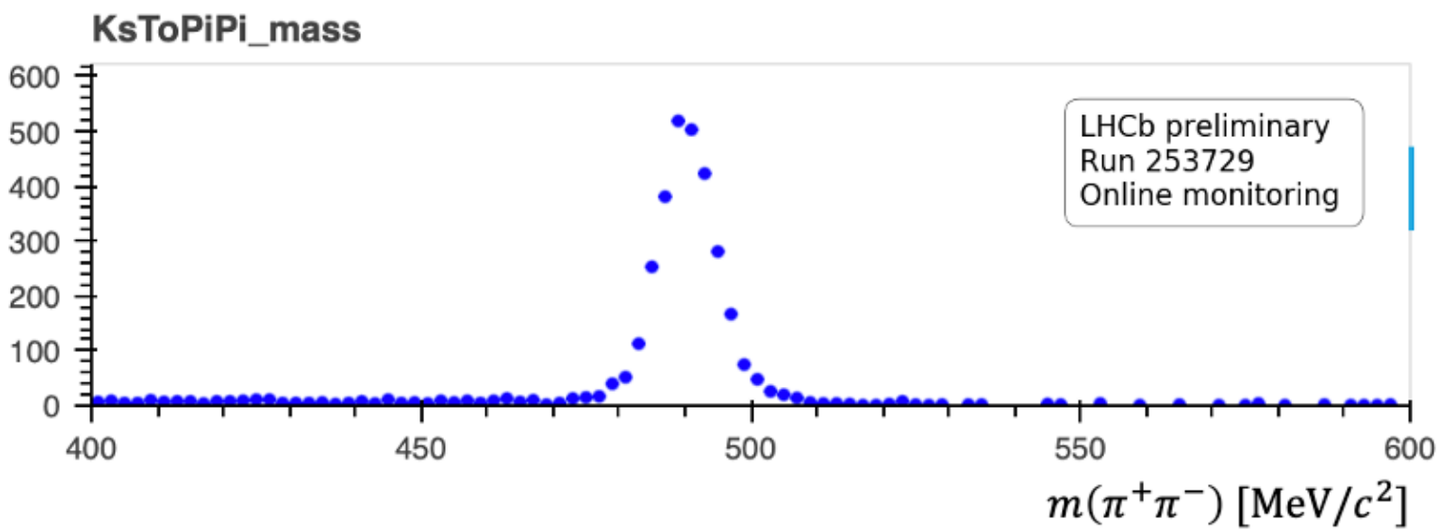


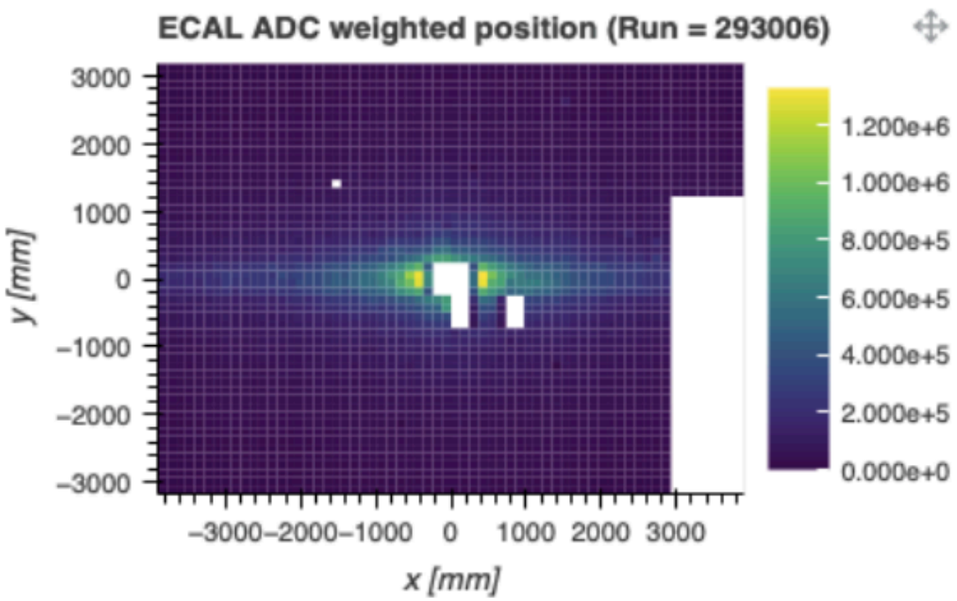
Monitoring

Pourquoi a-t-on besoin?

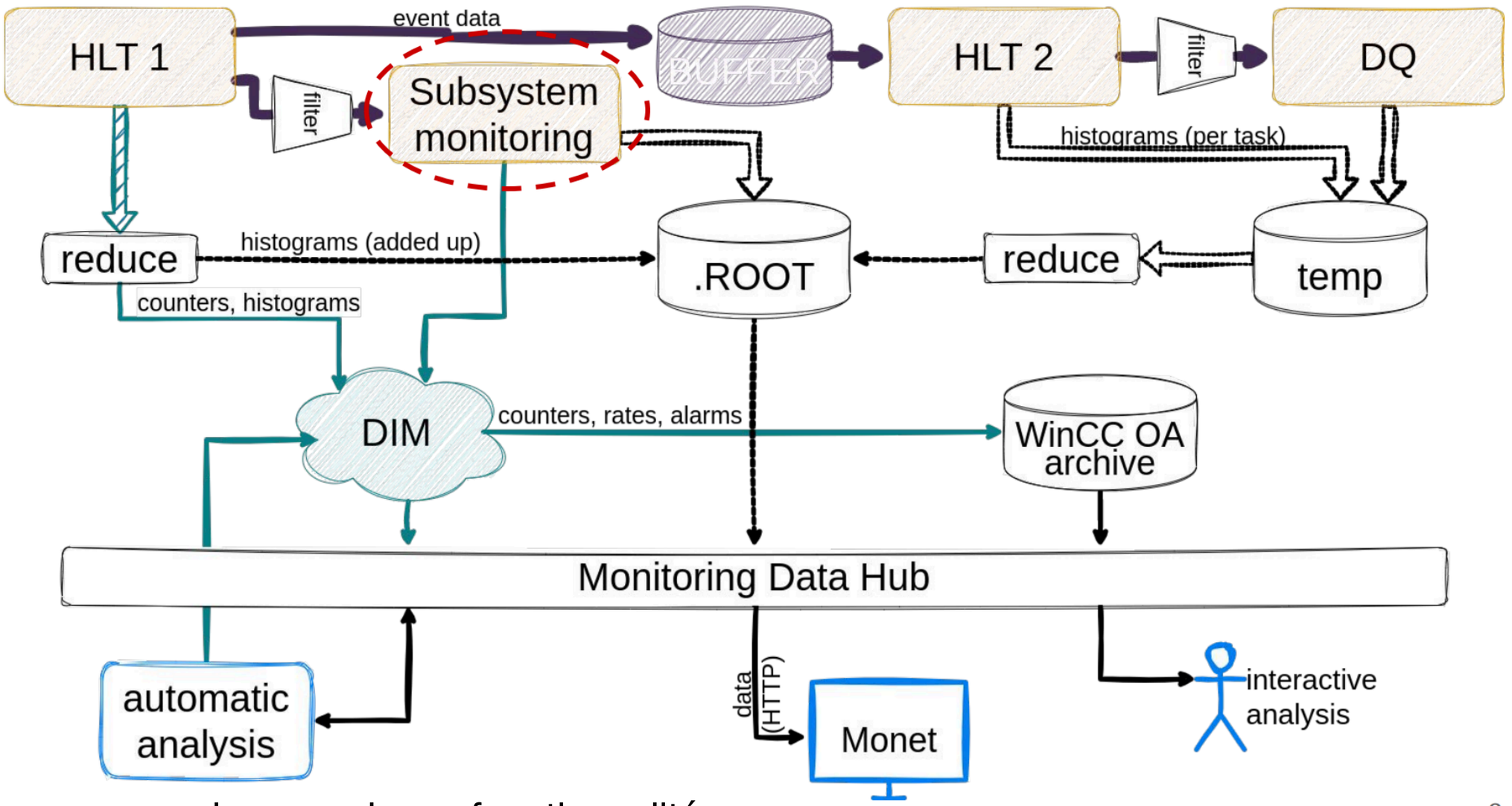
- Detecter les problèmes rapidement: **toutes parties du système sans monitoring sont potentiellement non-fonctionnelles**
- Qualité des données
- Faire du debugging
- Sauvegarder les conditions de prise des données
- Donner du feedback aux autres (LHC, autres experiences etc)

- Source des données:
  - Les données de systèmes de HV, refroidissement etc
  - Les données brutes du détecteur
  - Les données du trigger (hardware et software)
  - Données d'une post-analyse a plus haut niveau





Monitoring: un exemple



Système assez complexe, quelques fonctionnalités:

- Monitoring des sous-détecteurs à 10 kHz → histograms
- Monitoring du premier etape du trigger soft a 40 MHz
- Visualisation par interface web (Monet)
- Possibilité de faire analyse en ligne complémentaire des histograms (automatic analysis): fits, ratios, alarms etc

Questions?