Physique des particules : le retour du parallèlisme

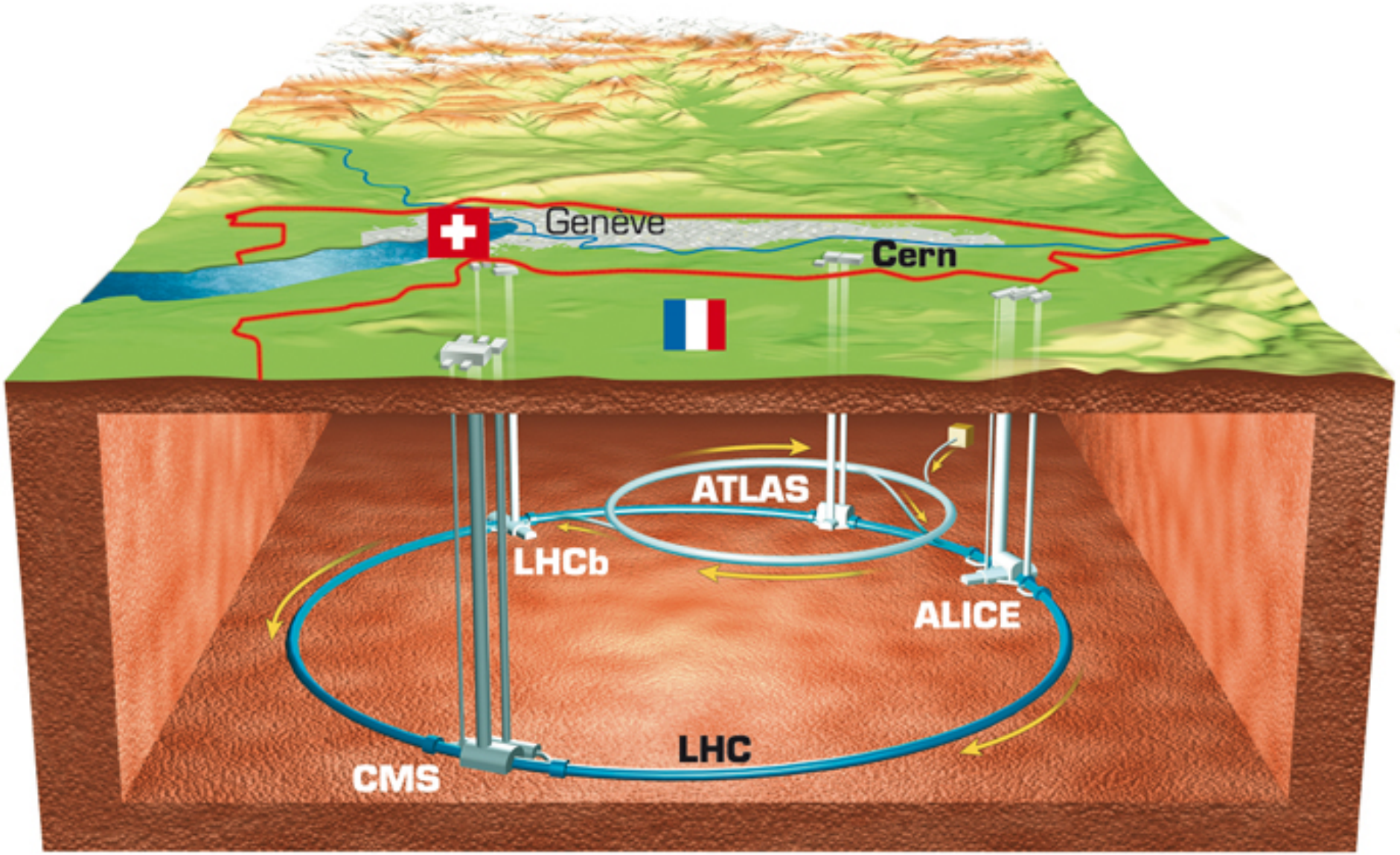
David Rousseau (LAL-Orsay)

Plan

- Contexte
- Changement de paradigme
- Pourquoi c'est difficile

- Remerciements à David Chamont (LLR) pour certains transparents (présentés lors de la journée d'inauguration de la salle vallée de VirtualData https://indico.lal.in2p3.fr/conferenceDisplay.py?confId=2311)
- Remarques préliminaires:
  - Je ne parlerai pratiquement que des expériences au LHC, et surtout d'Atlas (multiplier les chiffres par 2-3 pour le total LHC)
  - Je ne parlerai pratiquement pas des aspects déclenchements, grille de calcul, base de données, réseau, stockage…donc uniquement des logiciels hors-ligne
  - Beaucoup de mes transparents sont en anglais, toutes mes excuses…

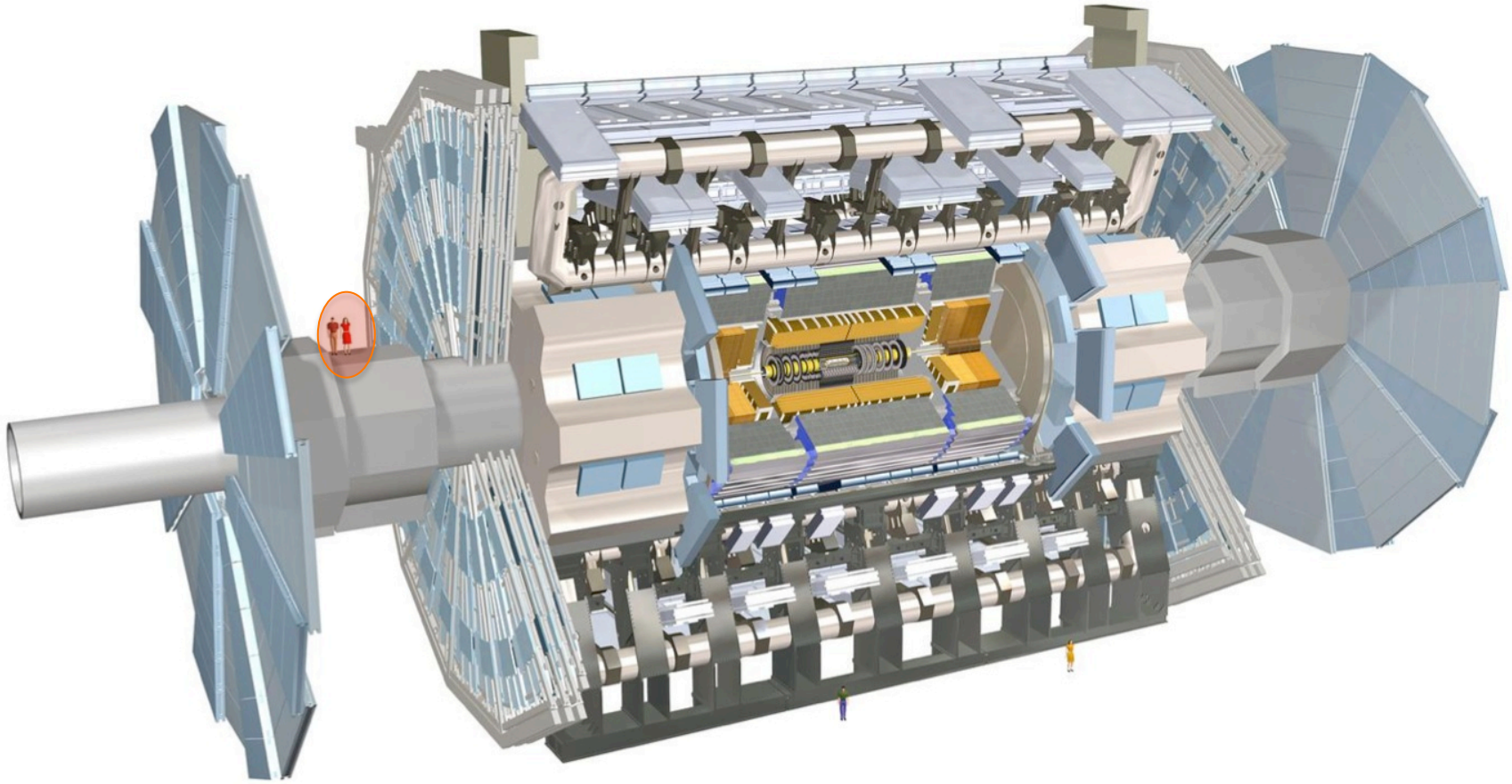
Le LHC



Le détecteur Atlas

Diamètre: 25m
Longueur: 46m
Poids: 7000 tonnes

3000 km de câbles
100 millions de canaux



Collision de protons

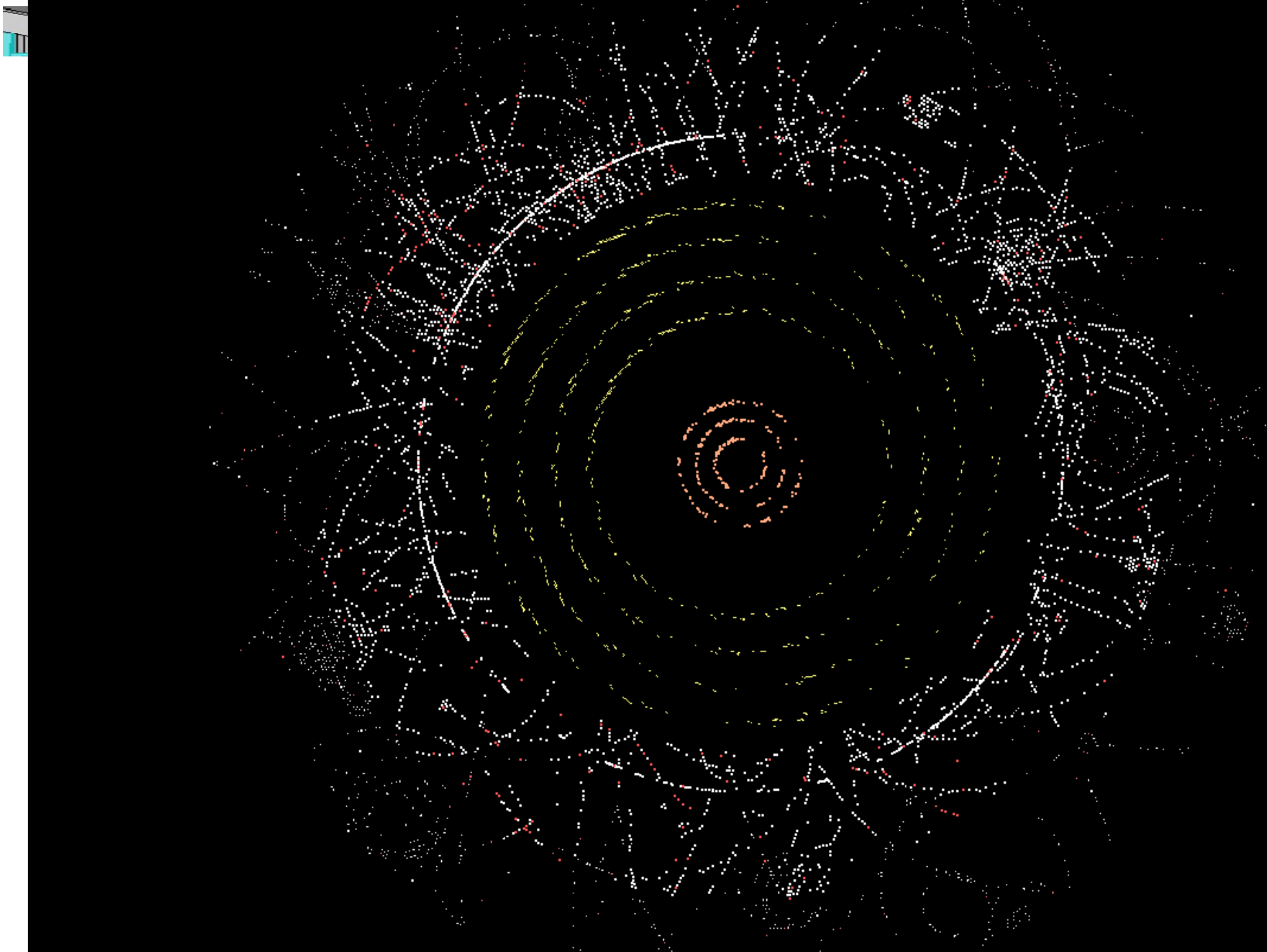
Einstein jeune: $E=mc^2$
c: vitesse de la lumière

Conversion de l'énergie cinétique en masse.

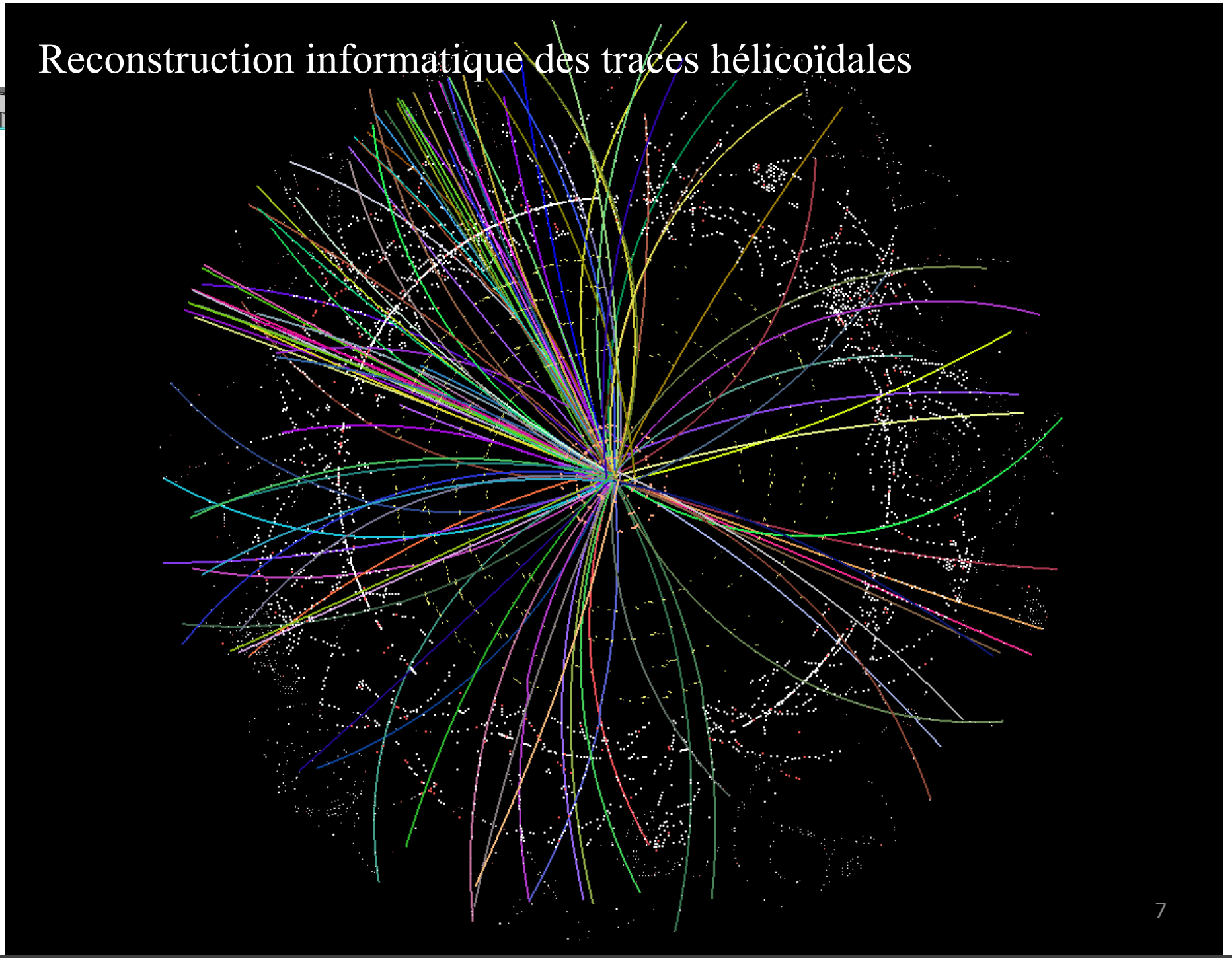
Création de nouvelles particules, d'une centaine de sortes

La plupart se désintègrent immédiatement
⇨Il n'en reste que de ~6 sortes,
qui vont traverser le détecteur.

Détection du passage des particules

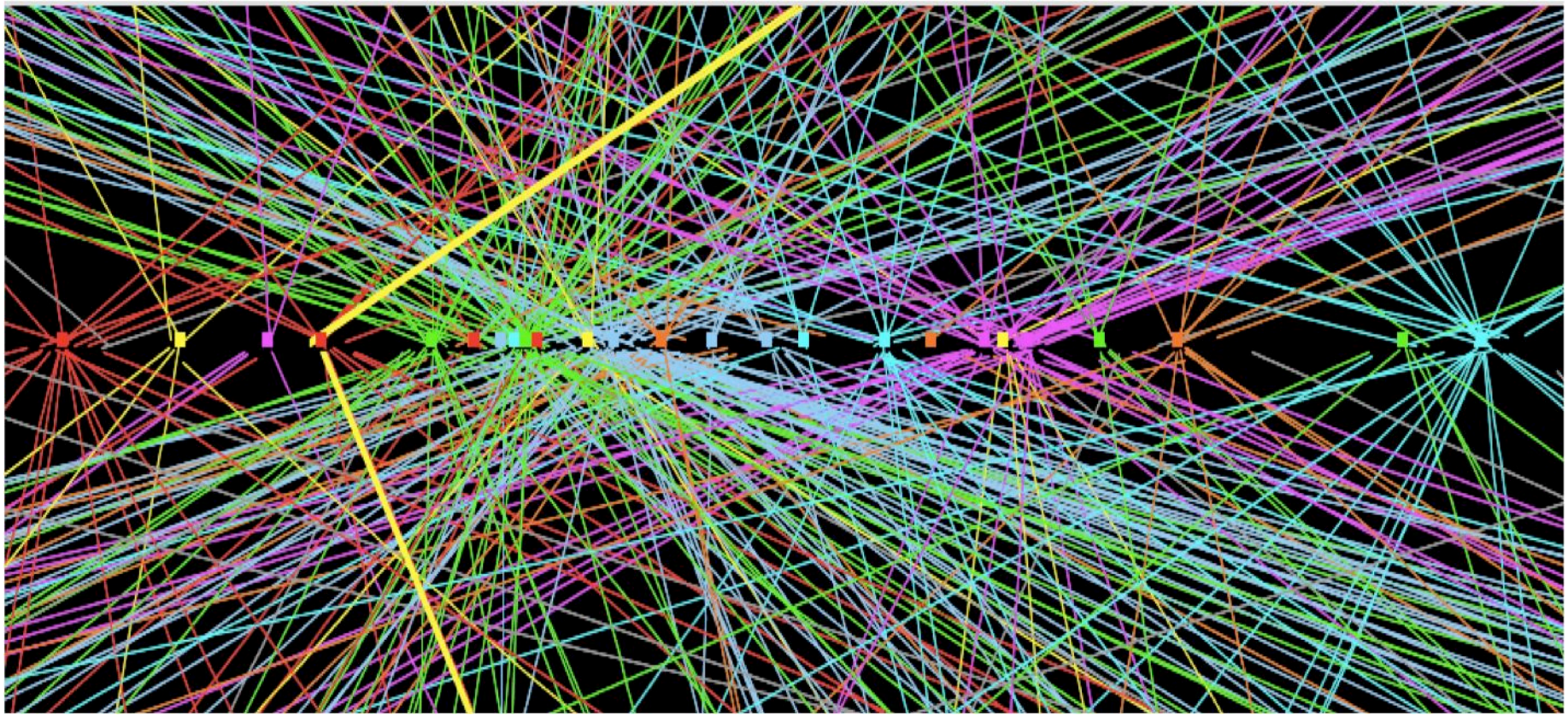


Reconstruction informatique des traces hélicoïdales



The inverse problem….

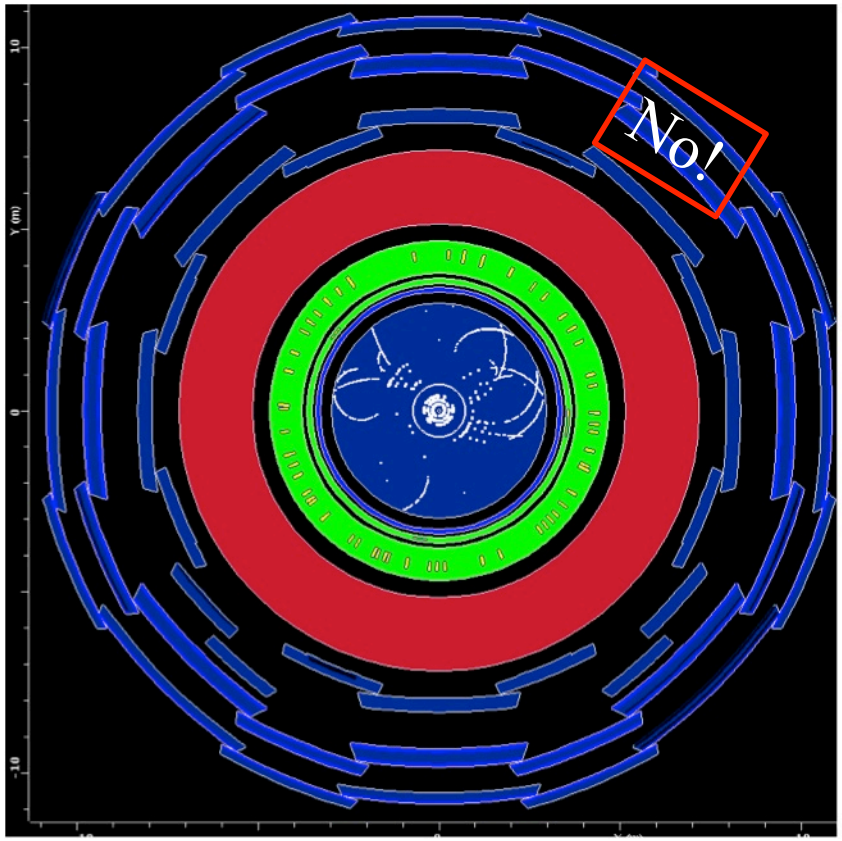
Un événement

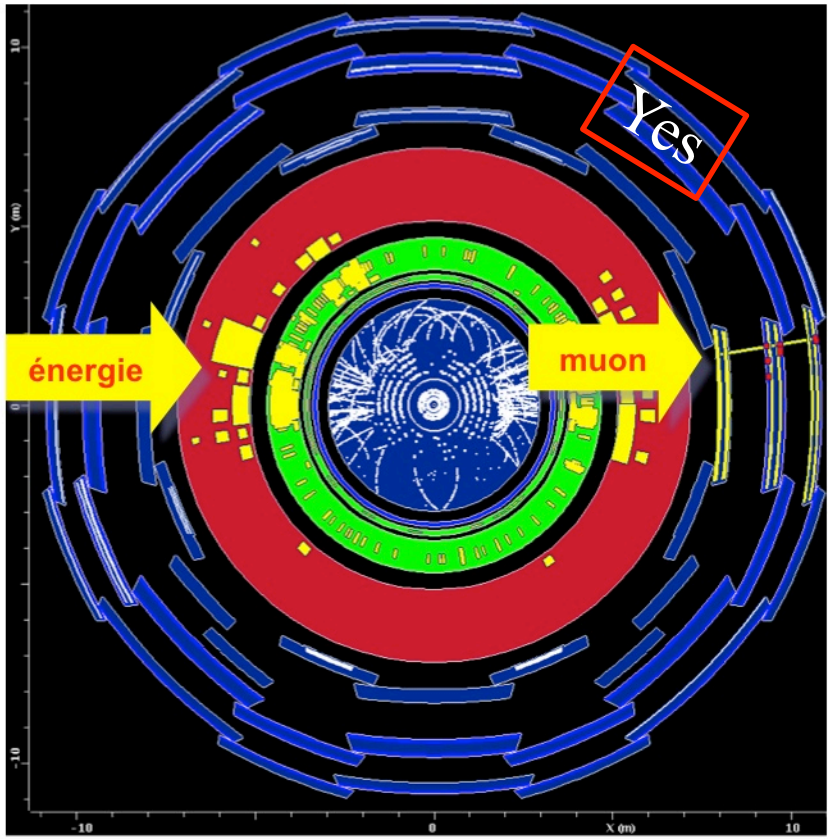


La précision obtenue permet de distinguer les traces venant de la collision intéressante de la 20aine de collision parasites au court de la collision des mêmes paquets de protons

Déclenchement

•20 millions de collision de paquets par seconde
•400 événements sélectionnés (1/50.000) au vol
•⇨échantillonnage en cascade, décision en 1μs-1s sur ~200 *signatures*

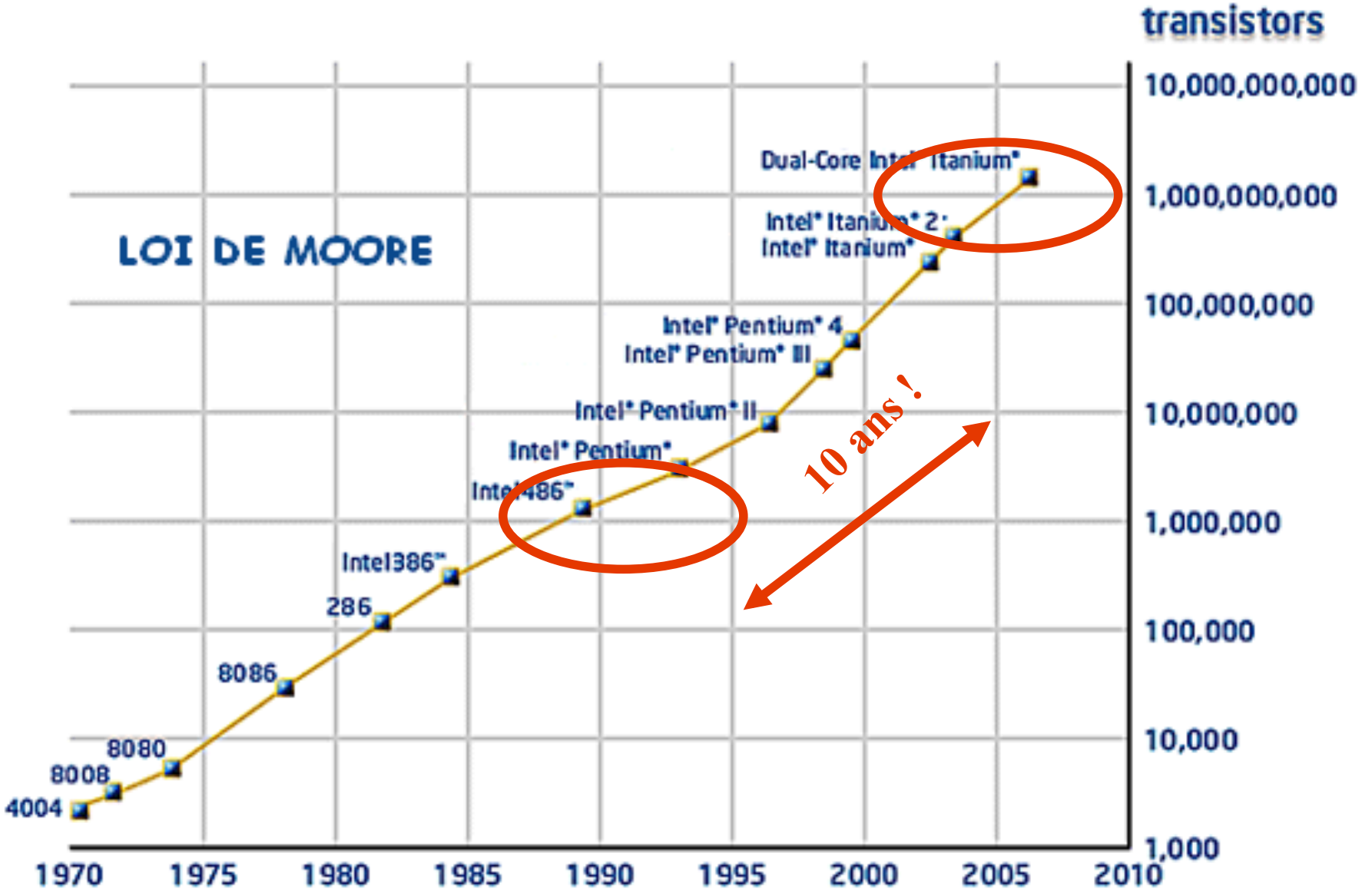




Le traitement des données en chiffres

•quelques PetaOctet de données accumulés chaque année
•données traitées quasi – en ligne par ~6000 coeurs au CERN
•…puis réduites et distribuées dans le monde entier dans les laboratoires,
•et finalement quelques GigaOctets sur les ordinateurs des physiciens
•Parallèlement, 150.000 ordinateurs dans le monde moulinent en permanence pour produire ~1 milliard d'événements simulés par an

Pari technologique

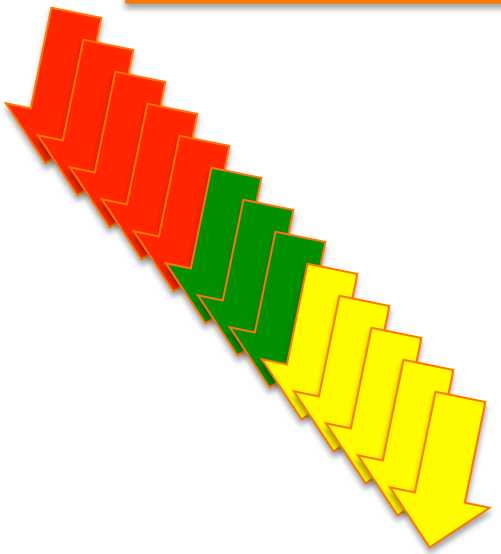


Chaine d'analyse

10^{14} collisions

Tri rapide et grossier

10^{9} événements sur disque



Tri précis

10^{5} événements à 2 gamma

Calcul de la masse
→histogramme

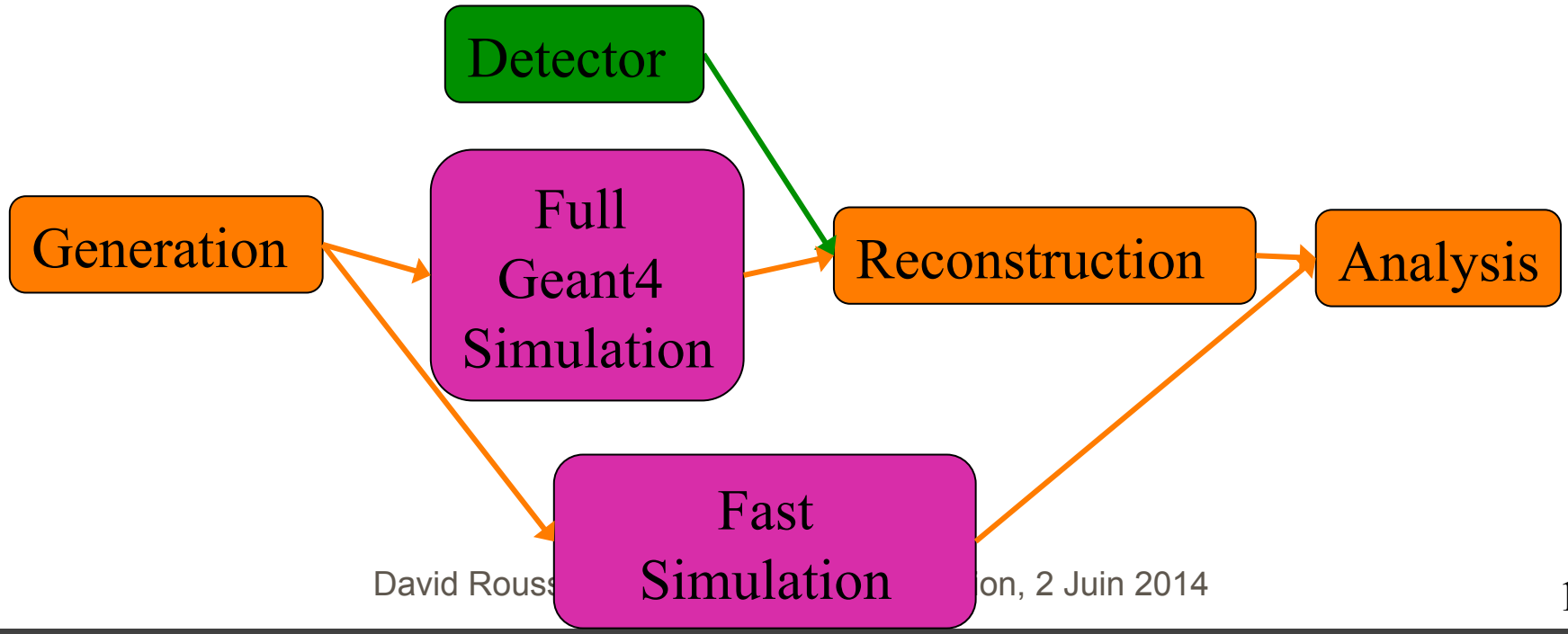
Plusieurs centaines analyses différentes

Processing steps

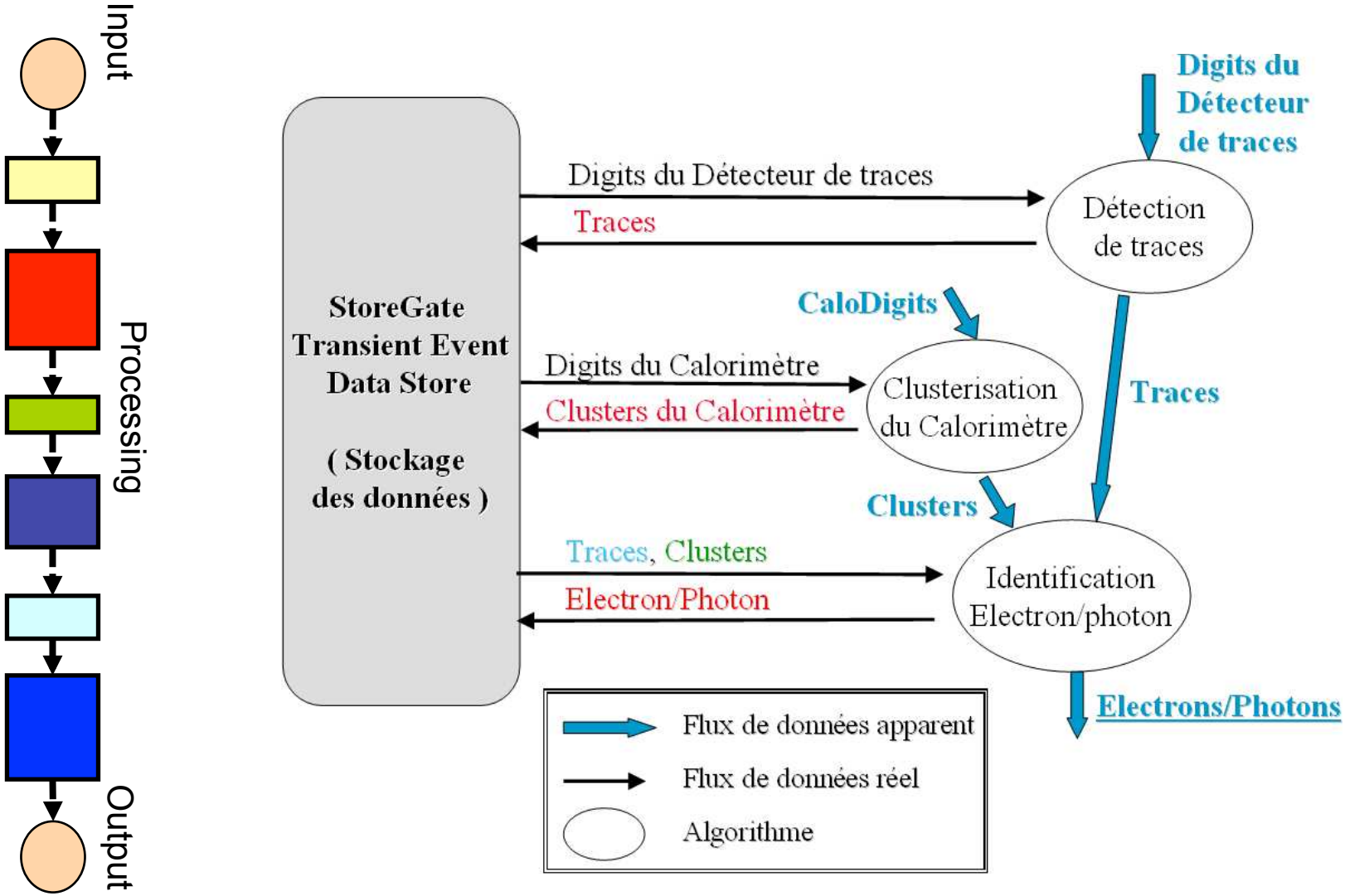
L'événement est l'unité de base de tous nos calculs.
Quelques milliards d'événement à traiter chaque année.
Un traitement prend entre 0.001s (analyse) et 1000s (Geant4)
Chaque événement peut être traité indépendamment des autres.
➔problème scandaleusement parallèle

Atlas sw in a nutshell

- Generators: :Generation of true particle from fondamental physics first principles=>not easy, but no sw challenge
- Full simulation :Tracking of all stable particles in magnetic field through the detector simulating interaction, recording energy deposition. (CPU intensive)
- Reconstruction : for real data as it comes out of the detector, or Monte-Carlo simulation data as above
- Fast simulation : parametric simulation, faster, coarser
- Analysis : Daily work of physicists, running on output of reconstruction to derive analysis specific information (I/O intensive)
- All in same framework (Gaudi/Athena), except last analysis step in Root. All C++.

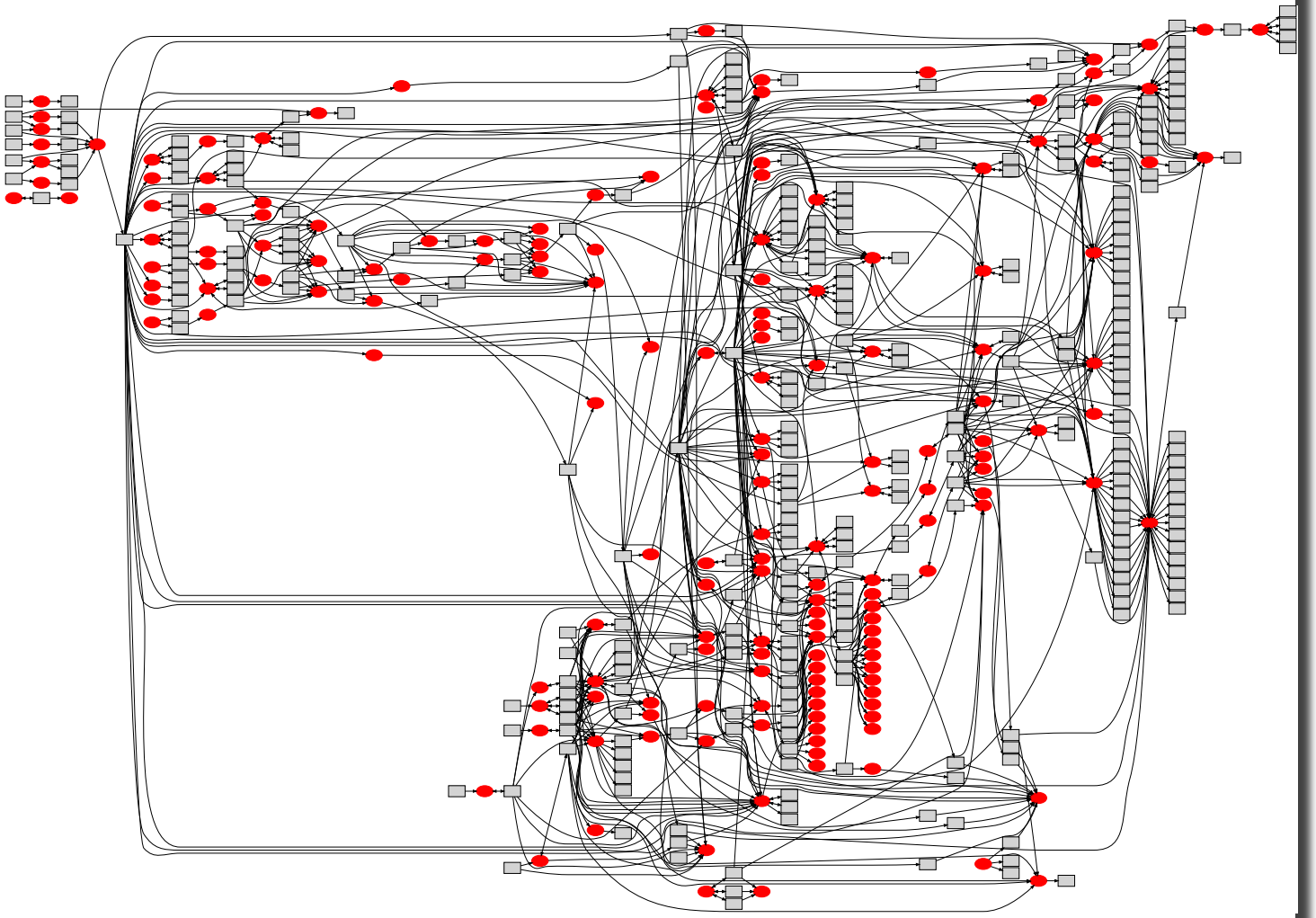


Architecture tableau noir

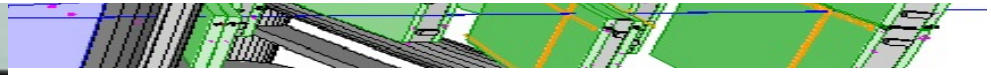


Real life

- Direct Acyclic Graph extracted from real reco job
- Today, algorithms run sequentially



Calcul dans HEP

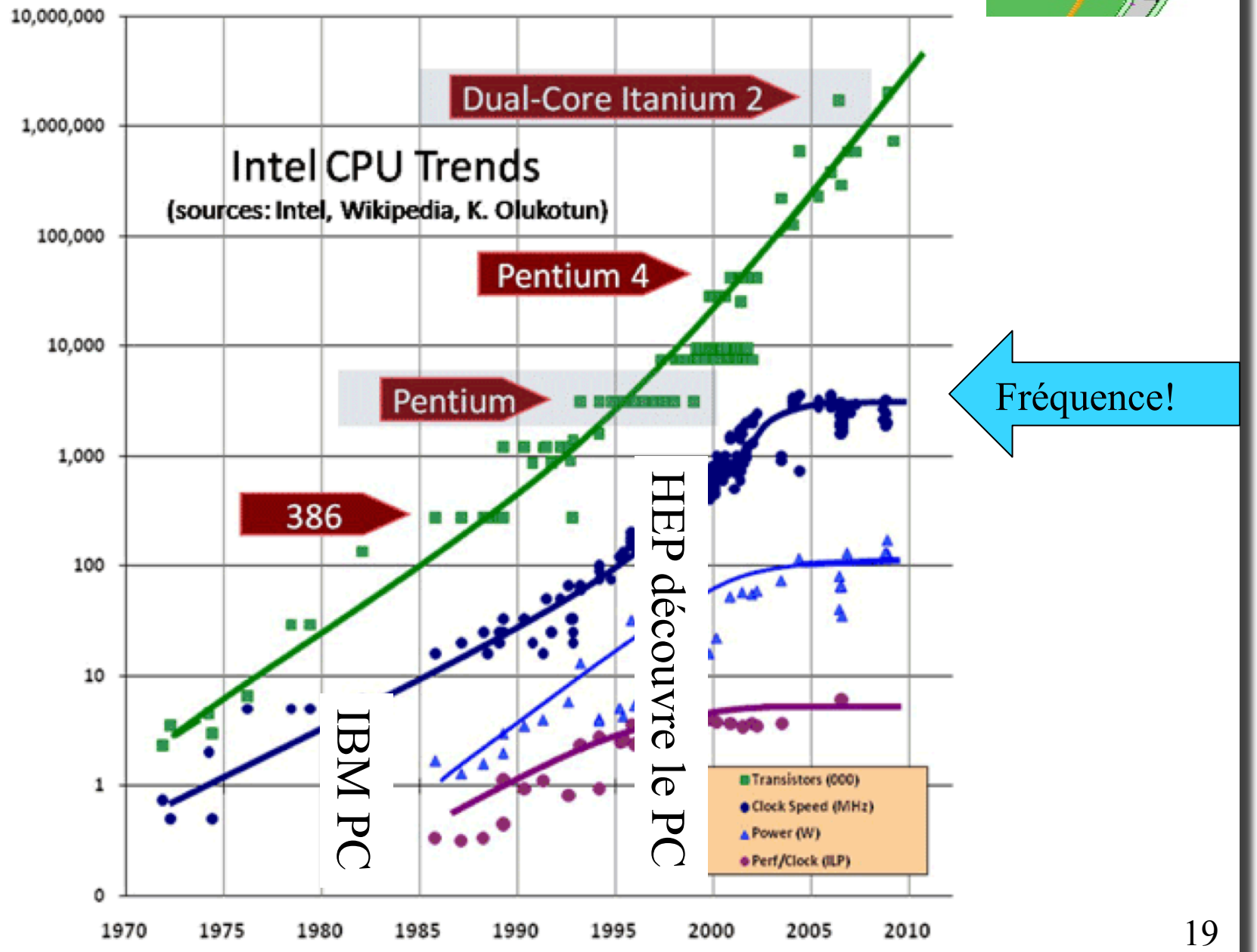


- Depuis les années 50, jusqu'au début des années 90: mainframe IBM, ou (ici au CERN) Cray XMP

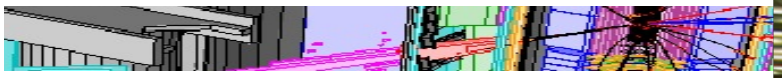
- Pendant les années 90 : ferme de processeurs risc

(ici ferme SHIFT au CERN)

Technologie processeur



Fréquence!



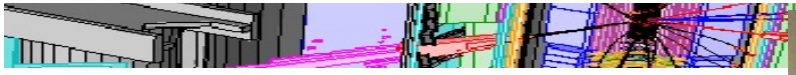
1997@CERN

Ferme PC

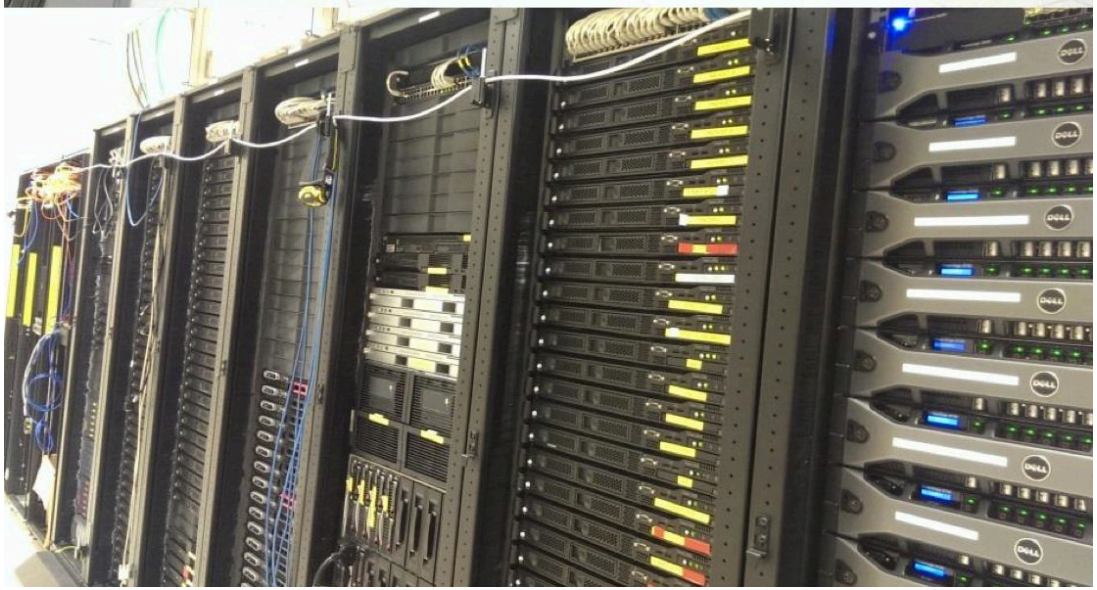
Pentium 200MHz

Un peu plus tard

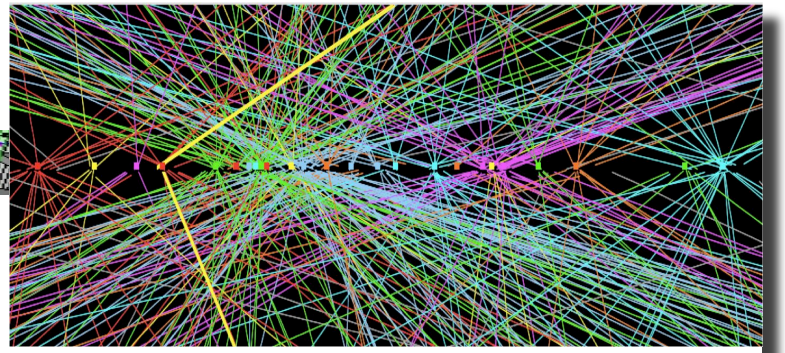
Virtual data



- 2014 @ Orsay
- Salle Vallée
- Toujours des processeurs Intel, mais multicoeurs



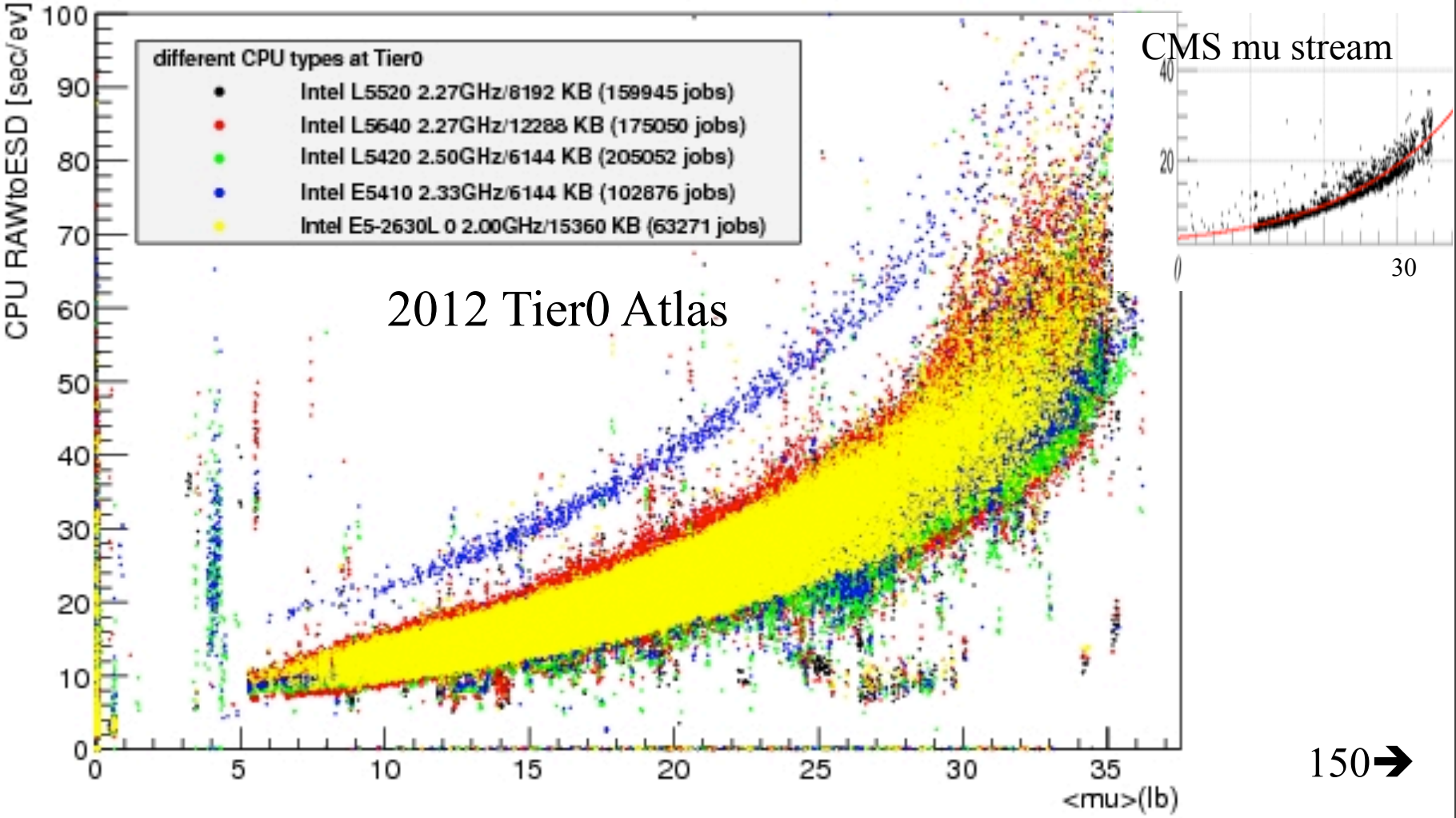
LHC Context



- Run 1 : 2010-2012
- Run 2 : 2014-2018 : énergie x 2, pile-up ~50 au lieu de ~25, N événements x2
- HL-LHC >2025 : pile-up ˜150, N événements x 10
- Flat resources (in euros) and Moore's law give us a factor 10 in CPU power (if and only if we can use the processors as efficiently as today!)
- ➔handling HL-LHC event added complexity, and maintenance/improvement of processor efficiency rely on software improvements. If not, impact on physics.

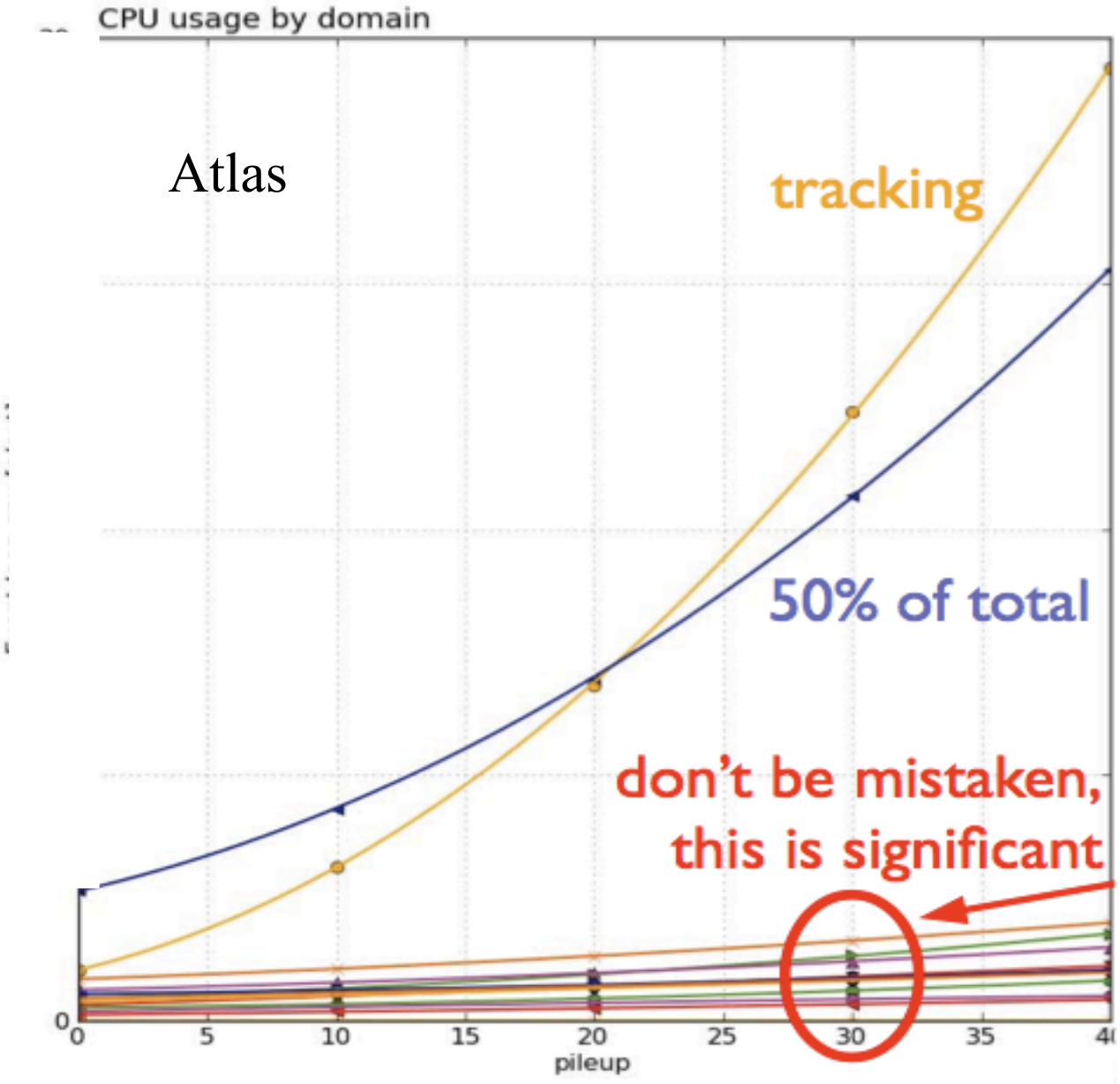
Impact of pileup on reco

Prompt reconstruction CPU time vs pileup



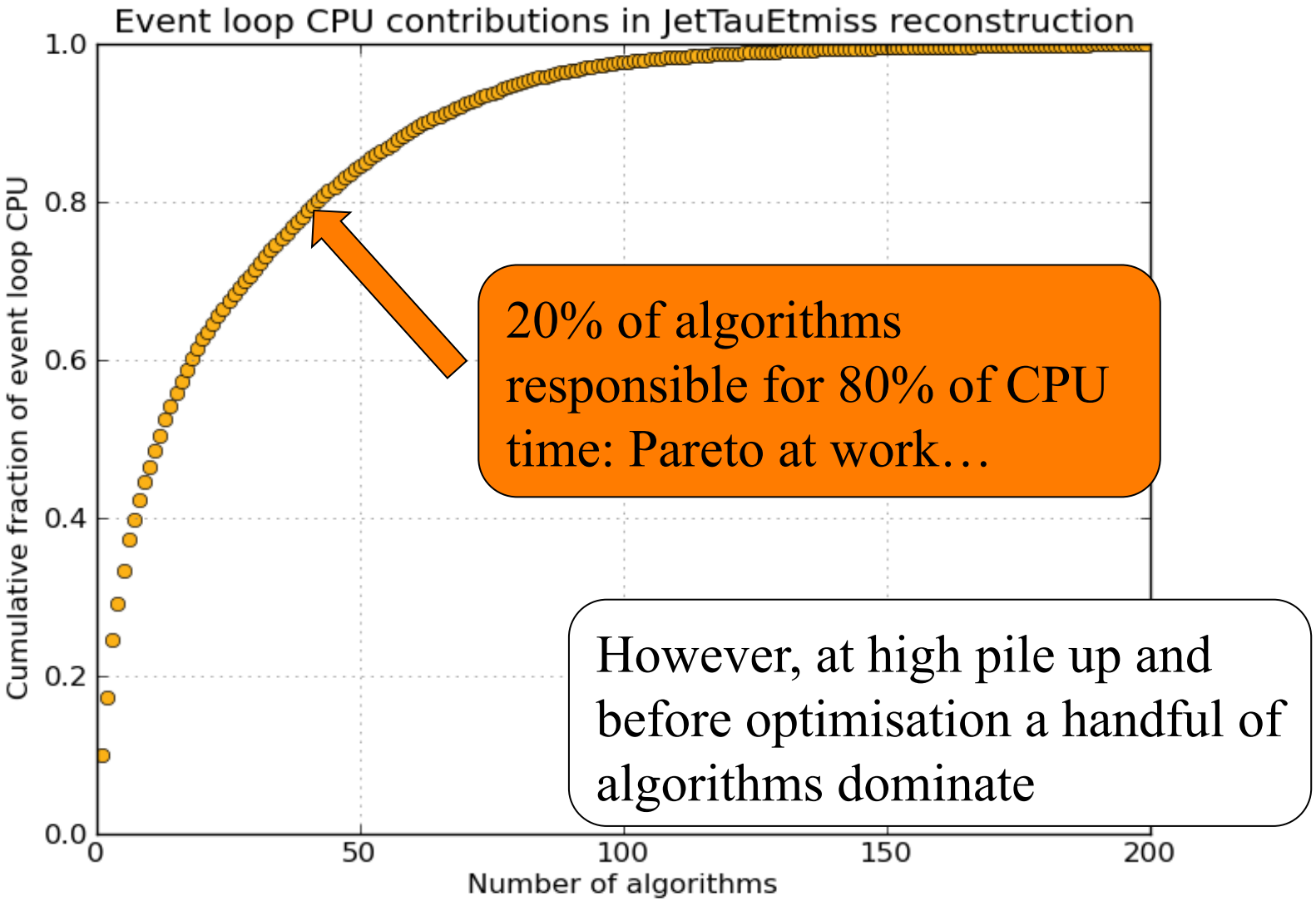
150➔

CPU per domain (reco)



- Tracking dominant in absolute and also non linear behavior
- However, the sum of other domains (calorimeter, jets, muons,…) is similar
- ➔need to improve across the board
- (CMS is slightly more tracking dominated)

CPU per algorithm (reco)

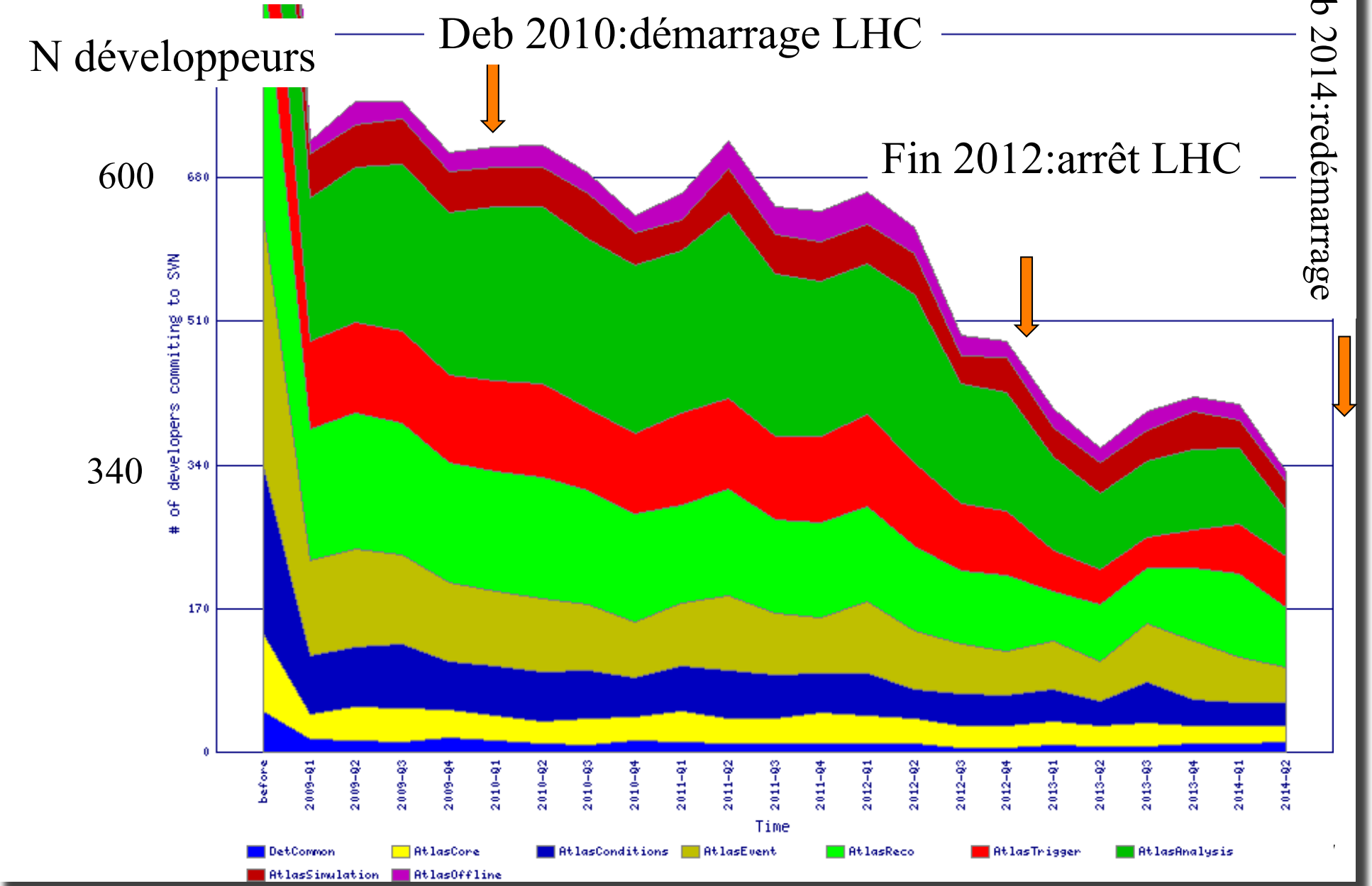


LHC experiments code base

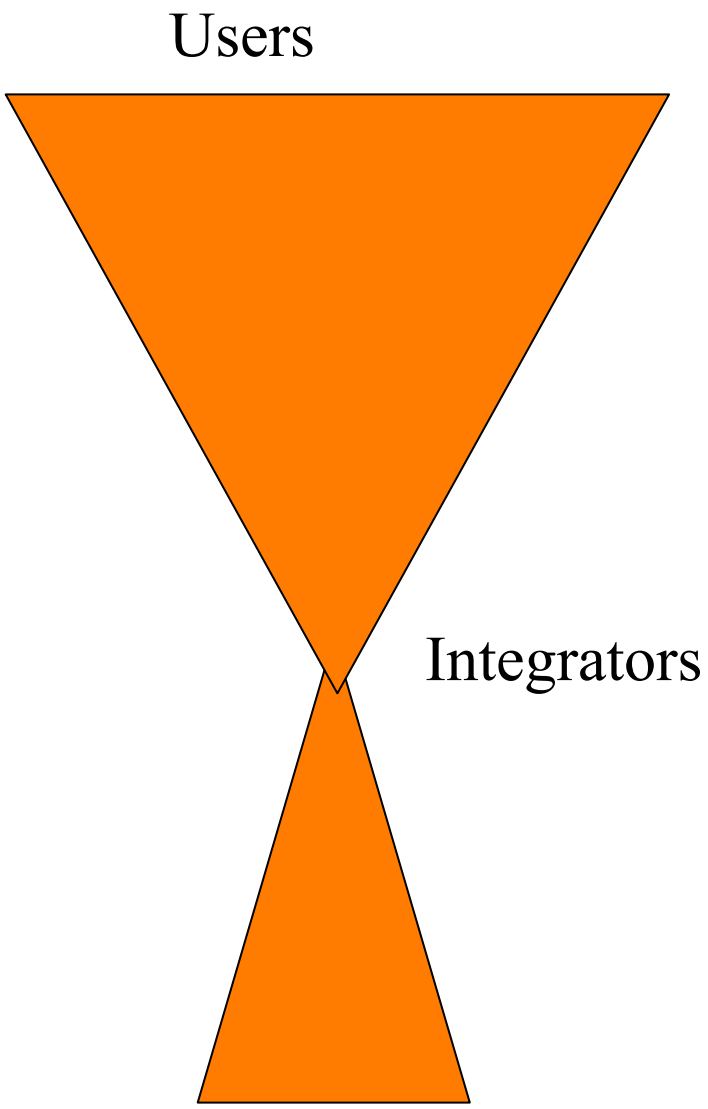
- LHC experiments code base
  - ~5 millions line of code per experiment
  - written by ~ 1000 people per experiment since ~15 years

- Who are they ?
  - Very few software engineers
  - Few physicists with very strong software expertise
  - Many physicists with ad-hoc software experience
- All these people need take part to the new transition

Développement durable ?



Integration challenge

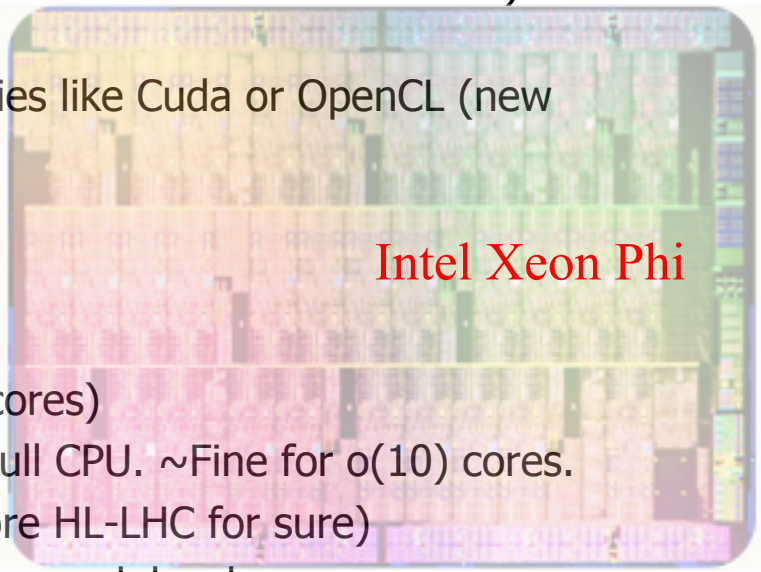


- Large developer community
- Large user community (with large overlap with the developer community)
- Limited number of "integrators"
- ➔elaborate release building infrastructure
  - TagCollector to collect tags
  - Release managers
  - Nightly build system
  - Nightly run time test, fast and slow
  - Shifters to monitor the result of the above, in addition to automatic emails
- In case of a migration (e.g. promotion of best practice), propagating it to all affected packages is often a full time job

Crise majeure ?

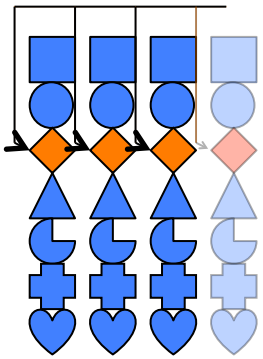
CPU Context

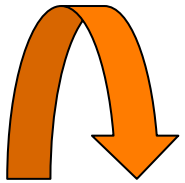
- Remember: no more CPU frequency gain since ~2005
- Two orthogonal avenues (in addition to traditional algorithm improvement):
- Micro-parallelism
  - Modern cores are not used efficiently by HEP software. Many cache misses. SIMD (Single Instruction Multiple Data), ILP (Instruction Level Parallelism) etc… to be used
  - Specialised cores like GPU require use of libraries like Cuda or OpenCL (new languages effectively)
  - Expert task. Focus on hot spots.
  - Immediate benefit on performance
- Macro-parallelism
  - More cores per CPU (and possibly specialised cores)
  - So far largely ignored : treat one core as one full CPU. ~Fine for o(10) cores.
  - Will break down for o(100) cores (when ? before HL-LHC for sure)
  - ➔calling for macro-parallelism, handled at framework level
  - Mitigates impact on sw developers if framework is smart enough
  - However does not help throughput if already enough I/O and memory



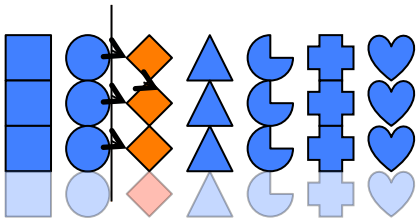
Note on data organisation

Array of objects





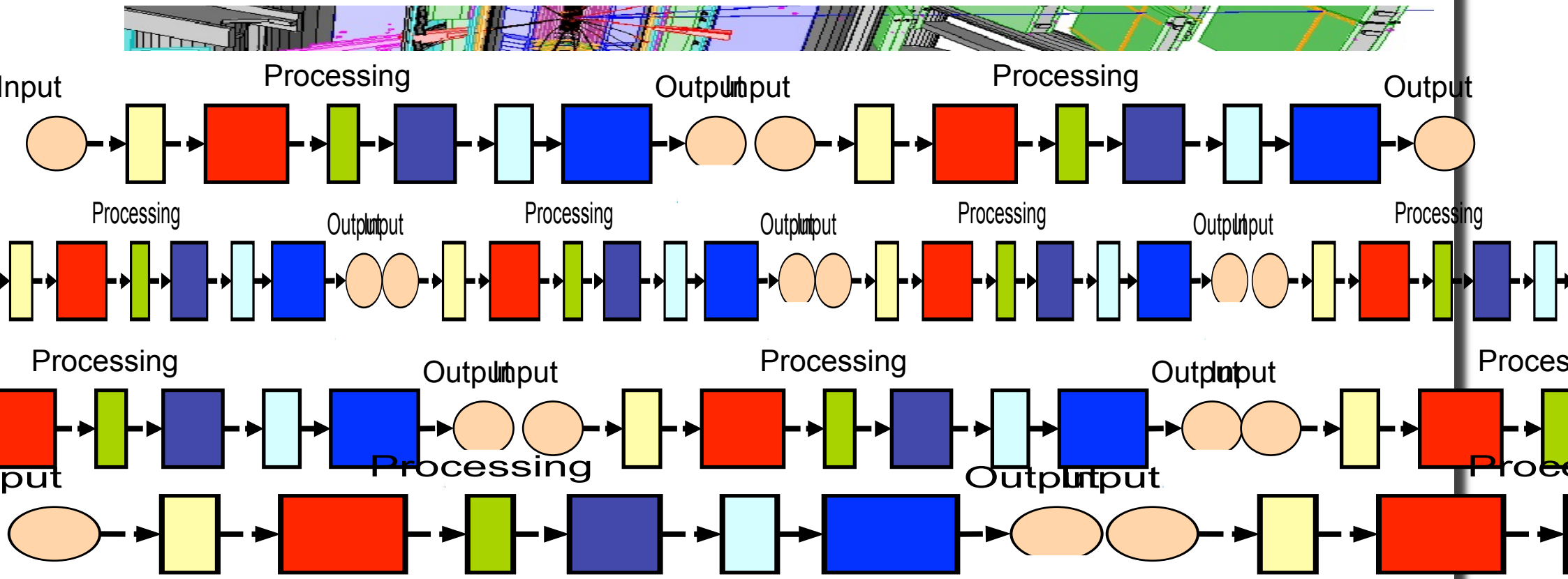
Struct of arrays



➔ More suitable for vectorisation

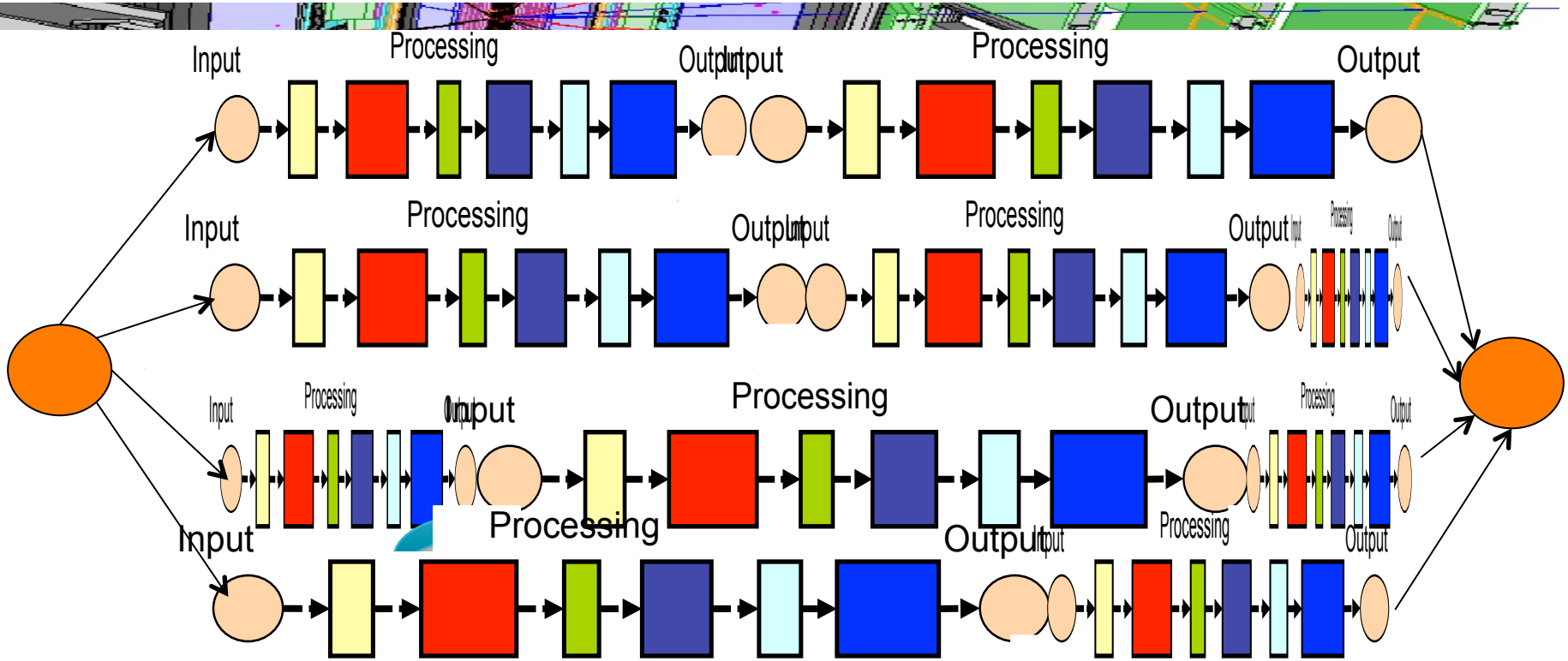
- Data organisation often need to be completely revisited prior to algorithm vectorisation
- (may improve performance even without vectorisation due to better locality (less cache misses))

One core one job



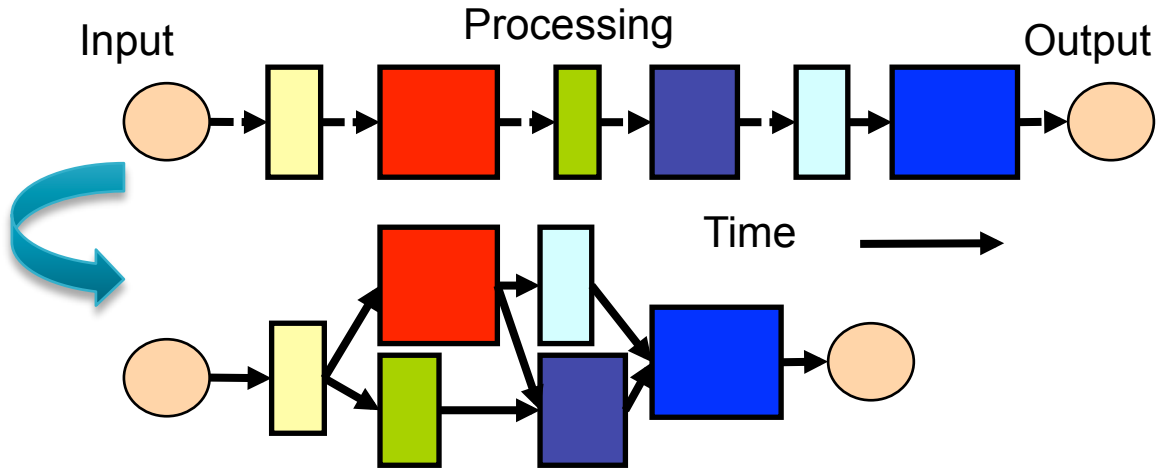
- Today, typical grid workhorse is a 16GB memory, 8 core CPU (2GB/core) (3GB/core is now common, but not sustainable in the future)
- Each core is adressed by the batch system as a separate processor
- Each job process event one by one, running one by one a finite number of algorithms
- One processor may handle simultaneously e.g. one Atlas reco job, 3 CMS simulation job, and 4 LHCb analysis jobs
- This works (today), however disorganised competition for resources like memory, I/O

One processor one job



- Available today (GaudiMP, AthenaMP) but not used in production yet
- One job goes to one processor (which is completely free)
- The framework distributes event processing to all cores, while sharing common memory (code, conditions,…) using Copy-on-Write
- No change to algorithmic code required (in principle)
- ~50% reduction of memory achieved (w.r.t. independent jobs)

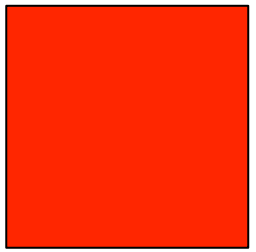
Event level parallelism

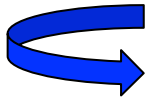


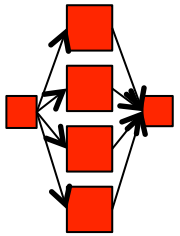
- framework schedules intelligently the algorithms from their dependency graph
- e.g. run tracking in parallel with calorimeter, then electron ID
- in practice too few algorithms can run in parallel (amdahl's law)
- ➔most cores remain idle

Note on multi-threading

- One possible answer is to use parallel multi-threading within algorithms
  - E.g. the tracking algorithm spawns multiple threads, each one reconstructing tracks in an eta-phi region
  - Test jobs on empty processor will effectively run (much) faster
  - However, in a grid environment, with one job per core, the multiple threads will compete with the other jobs running on the same processor➔no good!



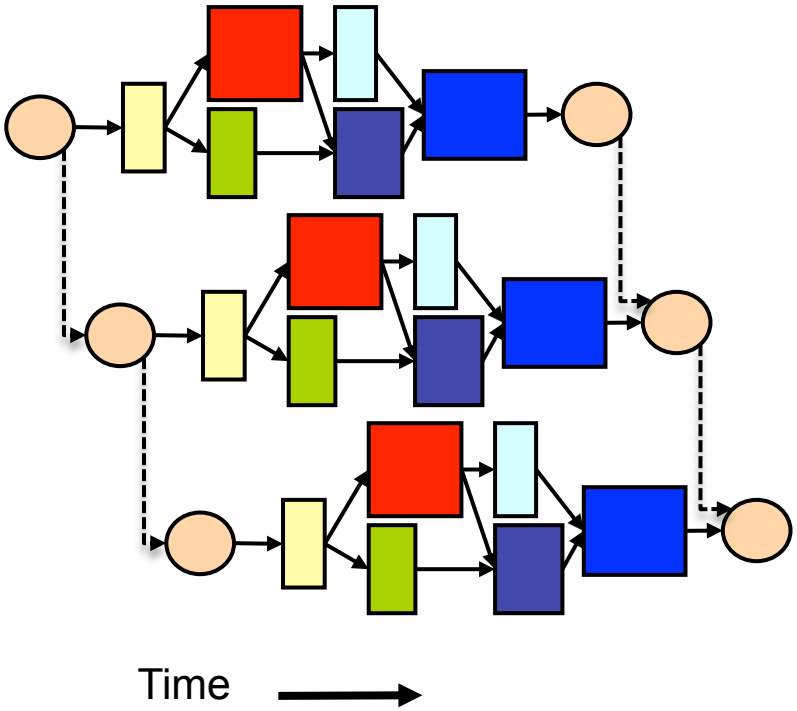




- Multi-threading useful if done at framework level, in an organised way (➔)

Event level concurrent event processing

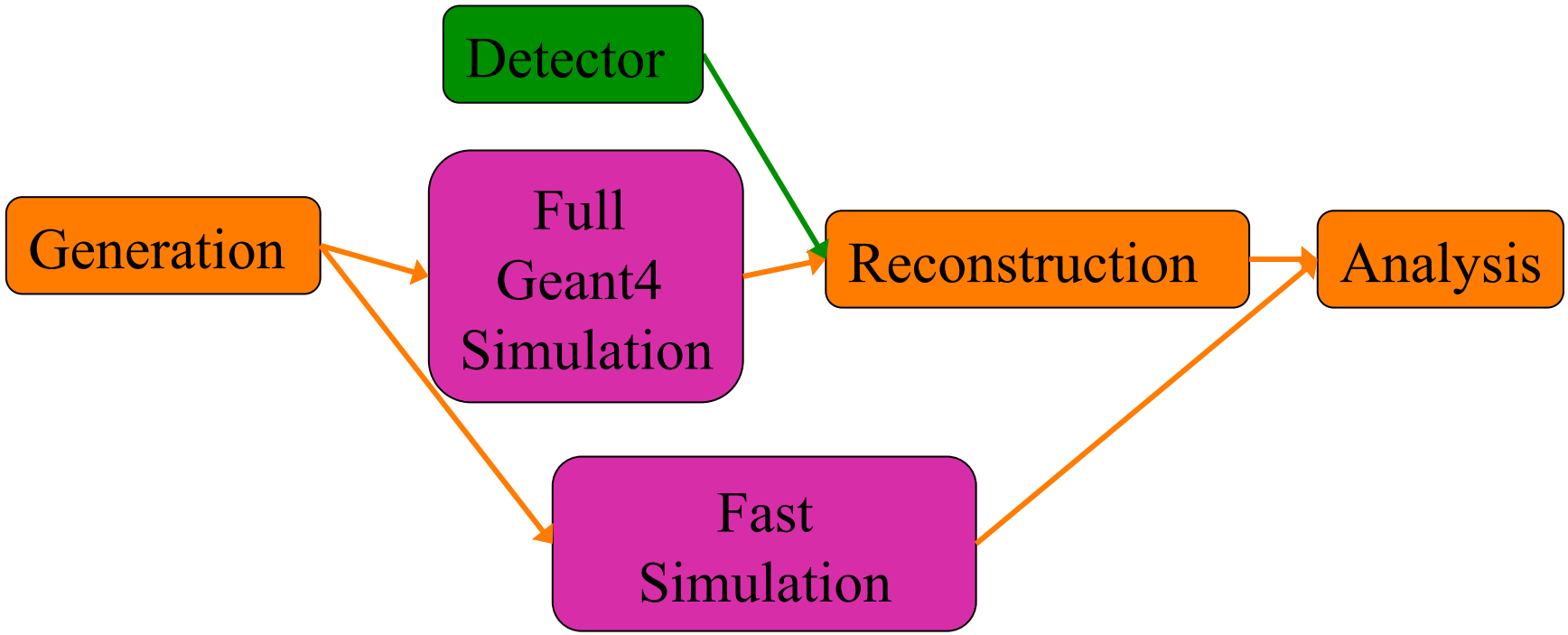
a.k.a the Holy Grail



- The framework processes several events simultaneously…
- …distributes intelligently algorithms to cores
- can allocate more cores to slowest algorithms
- can optimise use of specialised cores

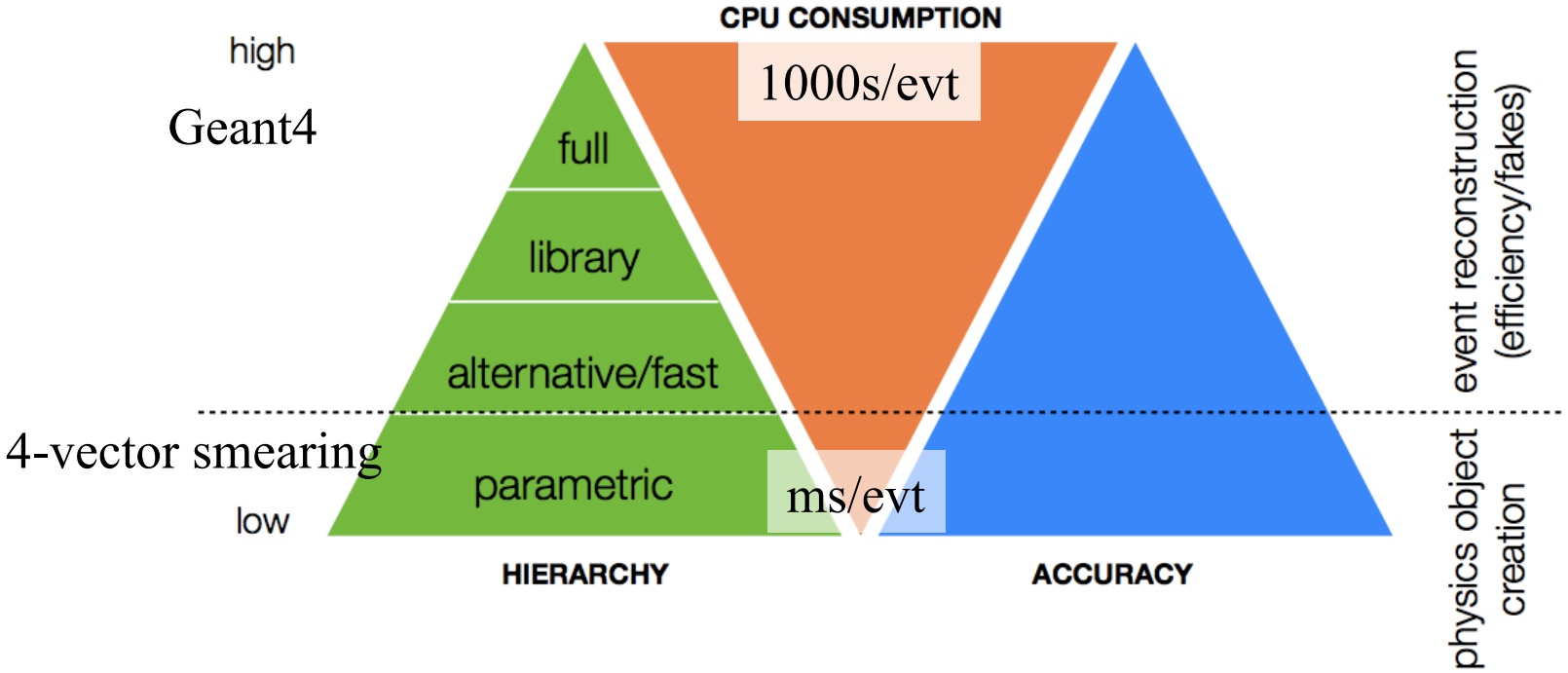
- In addition to algorithm scheduling, the framework provides services to pipeline access to resources (I/O, conditions, message logging…)
- Algorithms should be thread safe : no global object (except through the framework), only use thread safe services and libraries
- Algorithms do not need to handle threads themselves
- ➔regular software physicist with proper training can (re)write algorithms

Processing steps



Simulation

- Dominates CPU consumption on the grid
- HL-LHC : 10x read-out rate➔10x n events simulated ? Even more due to increased requirement on precision
- Continue effort on Geant4 optimisation:
  - G4 10.0 multi-threaded released Dec 2013
  - Re-thinking core algorithms with vectorisation in mind
- Rely on blend of G4/Fast sim/Parametric. Challenge : the optimal blend is very analysis dependent. But only one pot of resources.

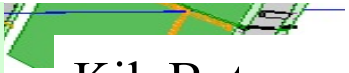


Geant4

- Geant4 10.0 livré le 6 décembre 2013
- A la demande, possibilité de simuler les évènements en parallèle.
- Implémenté à l'aide de multi-threading de type POSIX.
- Les données en lecture seule sont partagées (géométrie, cross-sections...)
- Reproductibilité des résultats
  (un générateur aléatoire par thread + un germe prédéfini par évènement).
- Excellente efficacité et passage à l'échelle, en multi-cœurs et sur Xeon Phi.
- Possibilité de croiser avec MPI.
- Prototypes en cours utilisant TBB, les GPUs...
- Tous les contributeurs doivent maintenant faire du code « Thread-Safe »

Analysis cycle

Creativity in physics analysis : having a lot of ideas…and…being able to test them quickly



RAW

1-3 months
Every
4-12 months

Petabytes

How much time does it take to redo a plot ?
(new cut, new variable,…)
How much time does it take to redo a full analysis ? (proper reweighted plots and values after all corrections)

plot

Few seconds
every minute
root

Analysis objects

Difficulty : ~200 analysis streams/teams, 2000 users sharing same resources

dedicated Ntuple

Giga/MegaBytes

One or more dedicated intermediate datasets with event selection and information reduction.
Balance between completeness, **resource** usage, speed to use and speed to reproduce the dataset.

Analysis software

- Order kB per event, a small fraction of events used, not CPU intensive➔I/O bounded jobs
- In the future, x10 larger disks, x10 larger bandwidth but disk access rate unchanged to a few 100Hz in data centers (SSD deployment limited)
- ➔even more sophisticated data organisation/access methods will be needed

Software for GPU

- Graphic co processor massively parallel, up to x100 speed-up on paper
- In practice, task must be prepared by traditional CPU and transferred to GPU
- Successfully used in HEP for very focussed usage e.g. Alice trigger tracking (gain factor 3 in farm size), now also in other experiments
- Code need to be written from scratch using libraries such as Cuda, etc…
- …and largely rewritten/retuned again for different processors, generation
- Physics performance not as good as original code
- Usage on the grid unlikely/difficult due to the variety of hardware
- ➔Need to maintain a second, traditional, version of the code
- In the future, expect progress in generic libraries (e.g. OpenCL) which would ease maintenance (one code for all processors) at an acceptable loss in performance

Common software

- Can we have more common software ? (beyond flagships Geant4, Root)
- One monster software with `if (CMS) do_this(); if (LHCb) do_that();` ? certainly not…
- Still, we can most likely do more that what we are doing right now
- Note that we are largely running on the same Grid (even the same processor can run at one time some cores with Atlas jobs, some with CMS, some with LHCb)
- Three angles:
  - Framework : introducing parallelism at different levels
  - Foundation libraries
  - Highly optimised HEP Toolboxes
- More and more common development needed

Foundation libraries

- Study (semi) drop-in replacement of low level libraries like (examples):
  - Arithmetic functions
  - Memory management (e.g. tmalloc)
  - Random number generators
  - Geometry (e.g. CLHEP) and 4-vectors (e.g. TLorentzVector)
    - E.g. ATLAS just migrated to Eigen (vectorised small matrix handling), 30% overall CPU gain after touching 1000/2500 packages
- Sometimes, even the right set of compiler/linker options can bring a few percent for free

GridCL

- Objectif premier : Evaluer l'utilisation de matériel many-core hétérogène, via OpenCL, au sein d'une grille.
- Financé 2012 par P2IO (labex Physique des 2 Infinis et des Origines)
- Partenaires : LLR, IAS, LAL, IMNC, IRFU, IPNO, LPT...
  - La porte est ouverte, contacter David Chamont (LLR)
- Ressources partagées:
  - 2 noeuds sandy-bridge, dotés chacun de 2 NVidia K20, connectés en Infiniband.
  - 2 noeuds sandy-bridge, dotés chacun de 2 Intel 5110P, connectés en Infiniband.
  - Bientôt : 1 noeud ivy-bridge, doté de 6 NVidia Titan.
  - Logiciel : OpenCL, CUDA 5, Intel Cluster Studio XE, CAPS OpenACC.
- Expérimentations
  - Pour CMS : reconstruction des traces dans les collisions d'ions lourds. o Pour CTA : traitement de signaux de télescope.
  - Pour SDO : traitement d'images satellitaires sur matériel hétérogène. o Evaluation et enrichissement du banc d'essai SHOC.
- Dissémination
  - Atelier OpenMP/MPI/OpenCL aux JDEVs 2013
  - A venir : formation sur les outils de profilage Intel

LPaSo

- Candidature ANR « défi de tous les savoirs », préselectionnée 2014
- Objectif : Gagner en performance en exploitant les cœurs multiples, les instructions vectorielles, les accélérateurs, ..., afin de pouvoir absorber la montée en luminosité du LHC.
- Partenaires:
  - LAL (ATLAS, LHCb), LLR (CMS), IRFU/SPP (ATLAS)
  - LRI, LIMOS
  - LPNHE (LHCb), LPC-Clermont (ATLAS)
  - LPTHE
- Thématiques
  - Parallélisme de tâches avec GaudiHive (ATLAS, LHCb).
  - Déclenchement haut-niveau avec GPUs (LHCb).
  - Traitement de données avec accélérateurs (CMS).
  - Parallélisation/vectorisation des outils d'analyse statistique.
  - Parallélisation/vectorisation de FastJet.

HEP sw collaboration

- Motivation
  - Pour exploiter efficacement les nouveaux matériels, et garder le contact avec les autres communautés scientifiques, notre patrimoine logiciel vieillissant a besoin d'une refonte profonde (C++11, parallélisme sous toute ses formes).
- Proposition
  - Créer une collaboration formelle mondiale, afin d'apporter plus de reconnaissance aux contributeurs, de solliciter des fonds auprès de H2020 et NSF/DOE, d'être plus attractif auprès de l'industrie.
- Work Packages
  - Etudes R&D courtes sur les alternatives matérielles et logicielles.
  - Remaniement des bibliothèques et boites à outils existantes, maintenance à long terme
  - Développement de nouveaux composants logiciels d'intérêt général.
  - Constitution d'une infrastructure d'essai matérielle (Xeon/Phi, AMD, NVidia, ARM, ...) et logicielle (compilateurs, déboggeurs, profileurs,...).
  - Déploiement d'outils et processus communs (dépôts, système d'intégration continue, ...). o Expertise, consultance et accompagnement auprès des expériences.
- Réunion de lancement au CERN 3-4 avril
- Collecte de « white papers » en mai
- Accord violent, d'où une organisation légère devrait émerger

Résumé

- Futur du LHC à l'horizon 2025 : complexité x10, N événements x 10 ➔ besoin x 100
- Budget plat : gain en resource x 10

- ➔un facteur 10 à gagner

- Introduction micro et macro parallelisme, dans 5 millions ligne de code, maintenues par qq centaines de physiciens