Core software aspects for the LHCb upgrade

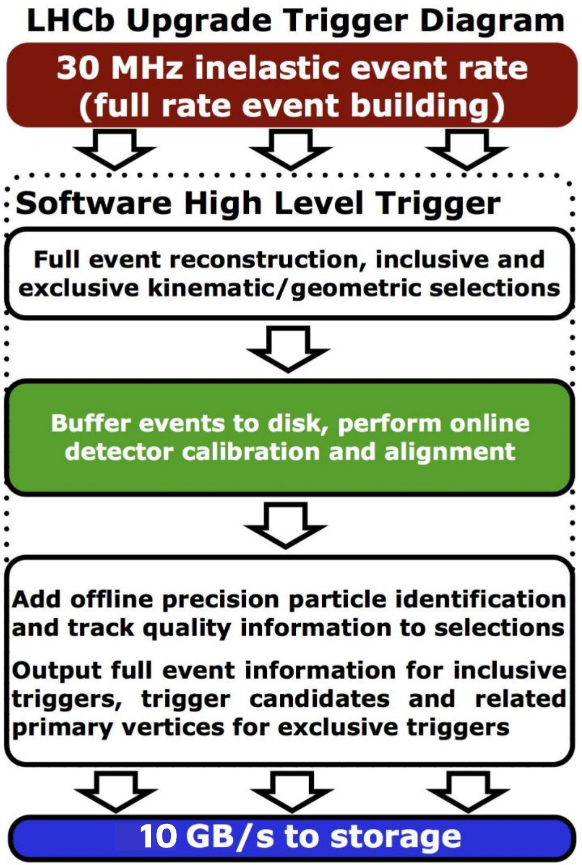
Niklas Nolte on behalf of the LHCb collaboration
ACAT 2019 - Track 1 - 12.03.2019

Bundesministerium
für Bildung
und Forschung

Introduction

- ➢ Major LHCb detector upgrade
  - i. New detector components
  - ii. Removal of hardware trigger stage “L0” → 30 MHz of pp collisions to be processed by a software trigger (HLT)

- ➢ The framework and the trigger need to have greatly improved performance



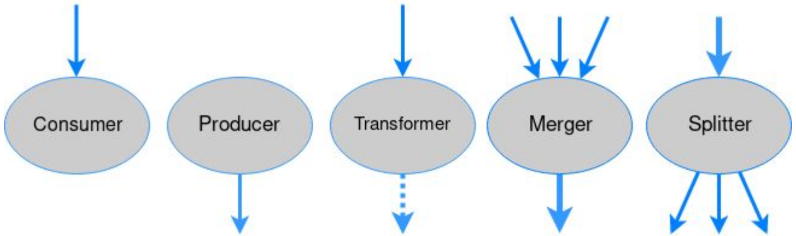
https://lhcb.web.cern.ch/lhcb/speakersbureau/html/TriggerScheme.html

LHCb software

- Development started in early 2000s, some code even older
- Based on the software framework “Gaudi”
  i. shared with ATLAS and others
  ii. Was initially not laid out for MultiThreading (MT), only process forking → weak scalability in RAM usage
  iii. Now: MT in “GaudiHive” as part of Gaudi
  iv. Gaudi::Functional: MT friendly by construction
- LHCb software: reconstruction + selection algorithms are ported to Gaudi::Functional

Gaudi::Functional

- Functional stateless wrapper around the base class “Algorithm”



- Algorithms store and load their data from an event store, Gaudi::Functional handles this for the user
- Inputs and Outputs are defined before processing
  → Easier to trace data dependencies
- User only implements the operator() **const** → MT friendly and reentrant

Event Store

- Each event has own event store
- Data are identified by string “/some/location/data”
- Implemented as tree structure
- Retrieving elements is slow
  - tree is traversed
  - lookup in a map for each step

- New implementation (not yet fully integrated):
  - Hashmap { full locations → data }, no more tree structure
  - Speeds up the lookup process greatly

Data Layout

- ➢ Earlier: collection of pointers “KeyedContainer<Track>”, essentially std::vector<Track*> with much additional logic
  - → Many small memory allocations
  - → Much pointer chasing
- ➢ Now: std::vector<Track>
  - → Preallocate as good as possible, larger allocations, but fewer
  - → Still not the most efficient layout, although much better already

Removal of KeyedContainer

Function	CPU
operator new	16.7%
_int_free	8.3%
PrPixelTracking::bestHit	5.7%
PrForwardTool::collectAllXHits	5.7%
PrStoreFTHit::storeHits	4.5%
PVSeed3DTool::getSeeds	2.7%

Function	CPU
PrPixelTracking::bestHit	19.2%
PrForwardTool::collectAllXHits	9.9%
operator new	8.9%
PrStoreFTHit::storeHits	7.9%
PVSeed3DTool::getSeeds	5.5%
_int_free	5.1%

Top 6 CPU consumers in the upgrade HLT1 stage before (left) and after (right) the removal of KeyedContainer

CERN-LHCC-2018-007. LHCB-TDR-017

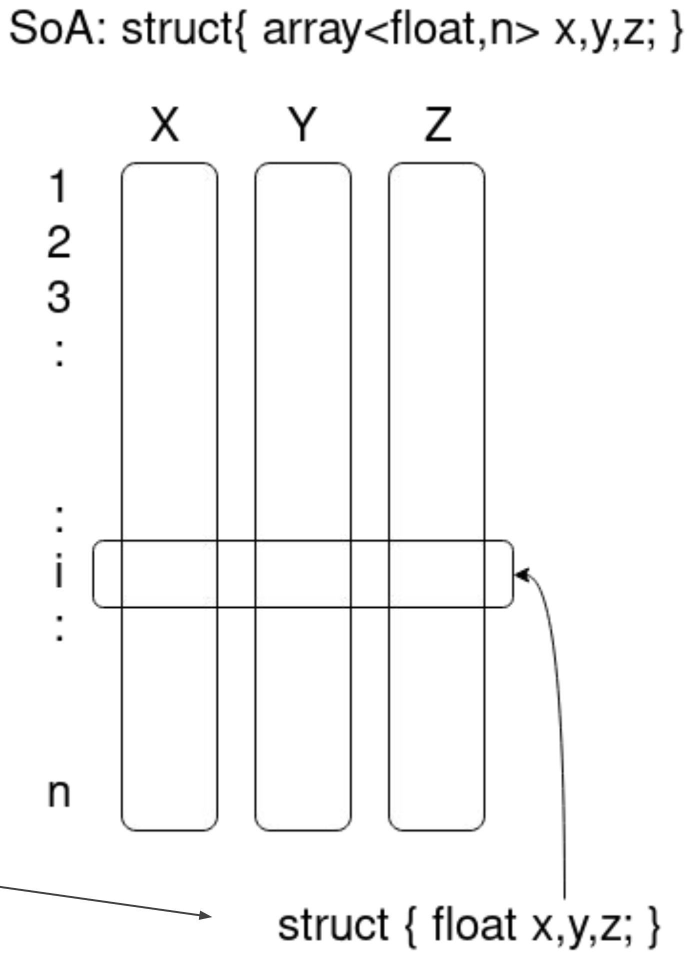
Data Layout

➢ Investigation ongoing:
switch to a SoA data layout as opposed to current AoS approach

➢ Possibly wrap into “SOAContainer”[1]

→ merges advantages of AoS and SoA

i. allows underlying SoA structure
ii. still able to ask for the underlying scalar structure



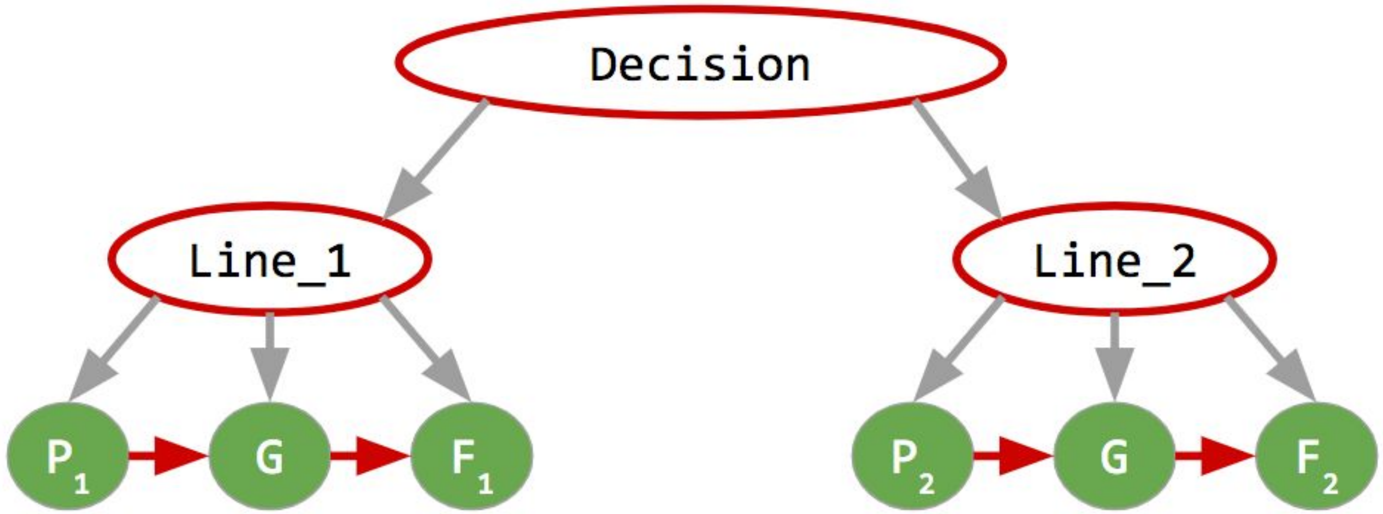
1: https://indico.cern.ch/event/736105/contributions/3036391/attachments/1687650/2717271/talk.pdf

Scheduling Algorithms

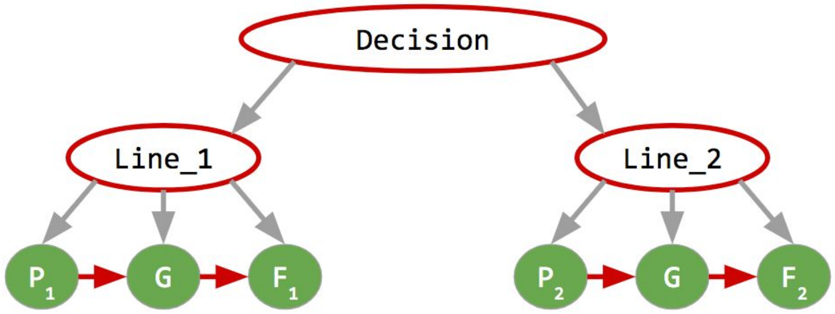
- First MT scheduler in Gaudi: AvalancheScheduler CERN-THESIS-2016-028
  i. Enables granular levels of parallelisation (within Events, or even within Algorithms)
  ii. Too complex and much too slow for LHCb Run III trigger use case

- New scheduler with following goals:
  i. Minimal runtime overhead
  ii. Support arbitrary control & data flow
  iii. Parallelize over events (1 event/task)

Trigger control flow

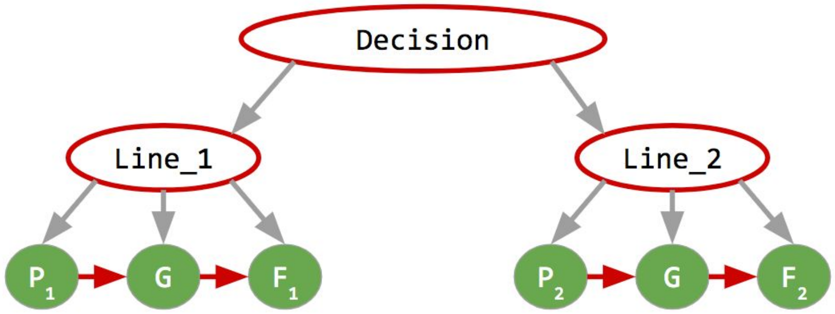


Control flow nodes



- **Composite Nodes**
  1. Logic (AND | OR | NOT)
  2. Children
  3. Execution policy: “allow short-circuiting” or “execute all children”
  4. Ordering constraint on children (control flow edges)
- **Basic Nodes**
  1. Manage one algorithm
  2. List of data dependencies
  3. Distributes decision provided by the underlying algorithm

Configuration

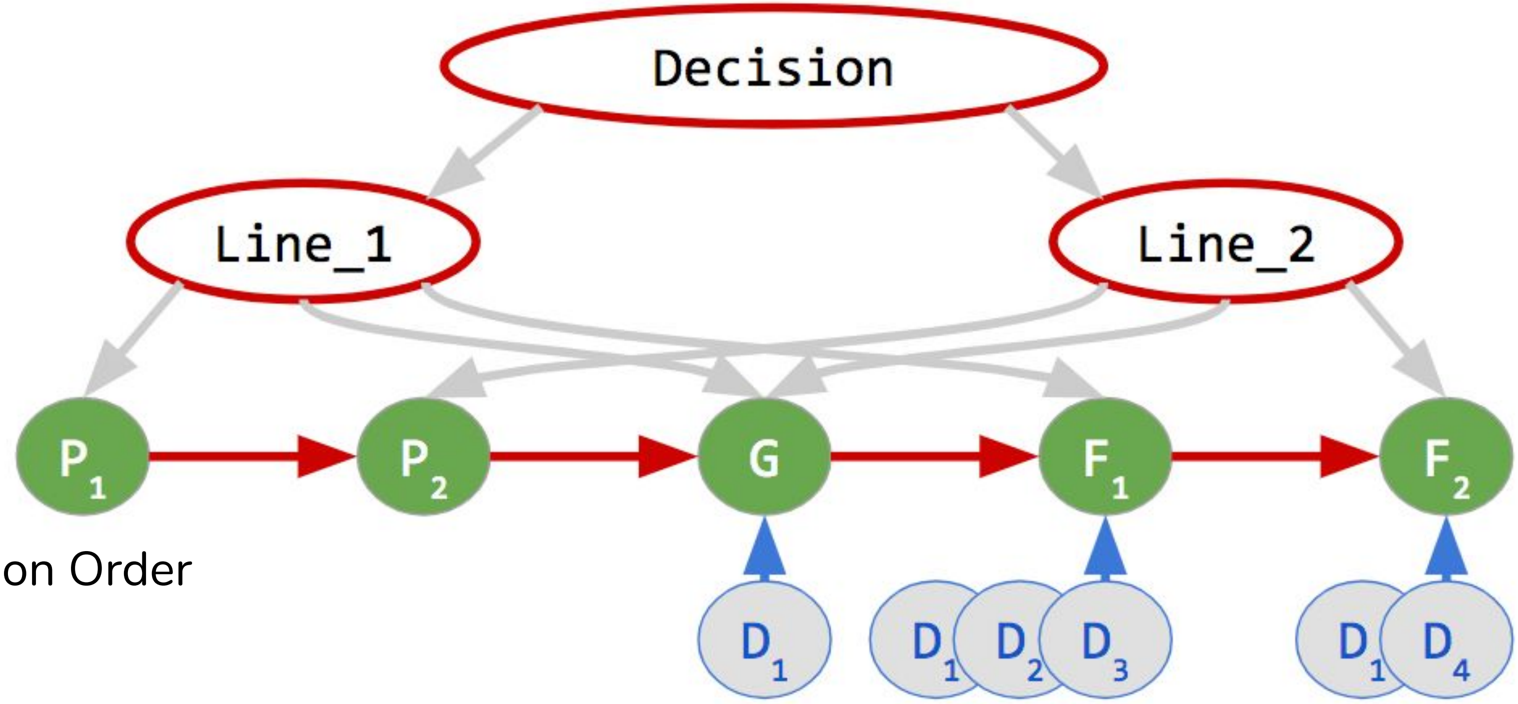


- The user configures..
  i. .. control flow via a set of composite and basic nodes
  ii. .. data flow by defining algorithm inputs and outputs

- Construct data dependency list for each basic node by matching in- and outputs

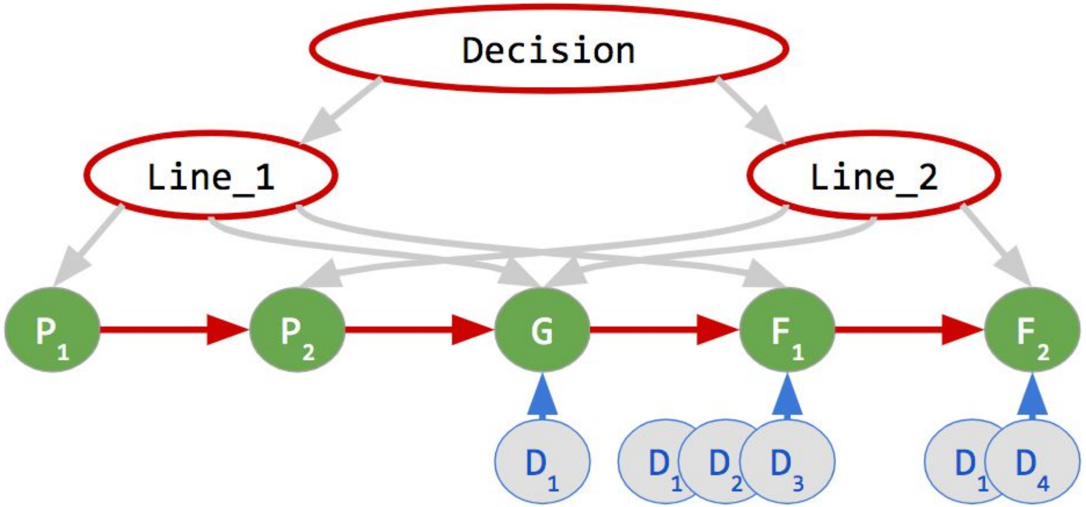
- Order basic nodes into flat list respecting ordering constraints

Ordered execution



Execution Order
→→

Runtime



Execution Order →→

Execution:

- ➢ Go through ordered list of basic nodes
- ➢ If node **requested** by any parent node, i.e. the parent is not yet evaluated..
  a. **Execute** all needed **data producers** if not yet executed
  b. **Execute** Basic Node
  c. **Notify parents** about decision
  d. **Parents evaluate** if execution policy permits it

Scheduling algorithms - technical aspects

- Master thread prepares self-contained tasks (full events) for a threadpool → no thread synchronization
- Performance overhead:
  1. no measurable overhead in the upgrade HLT1 reconstruction
  2. less than 1% overhead in a mock HLT1 with 20 trigger lines processing at 30 kHz per computing node
  3. less than 2% on a mock HLT2 with 1000 trigger lines

Further ongoing work

- ➢ Optimization of reconstruction algorithms

- ➢ DD4Hep as detector description (DD) is being integrated
  1. Current DD is quite outdated and lacks some features
  2. Better maintainability because DD4Hep is not LHCb-exclusive

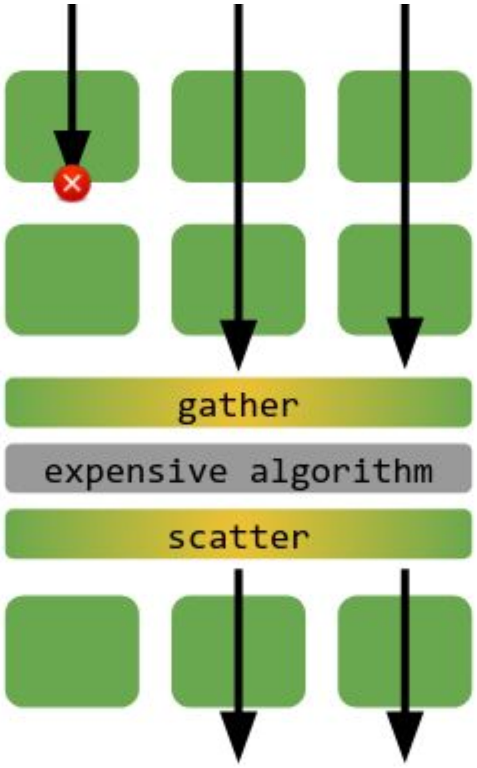
- ➢ Adaptation and optimization of the “conditions database” (time-varying conditions) and the access to it
  1. Make it compatible with the upgrade code
  2. speed up the framework startup time

Thanks

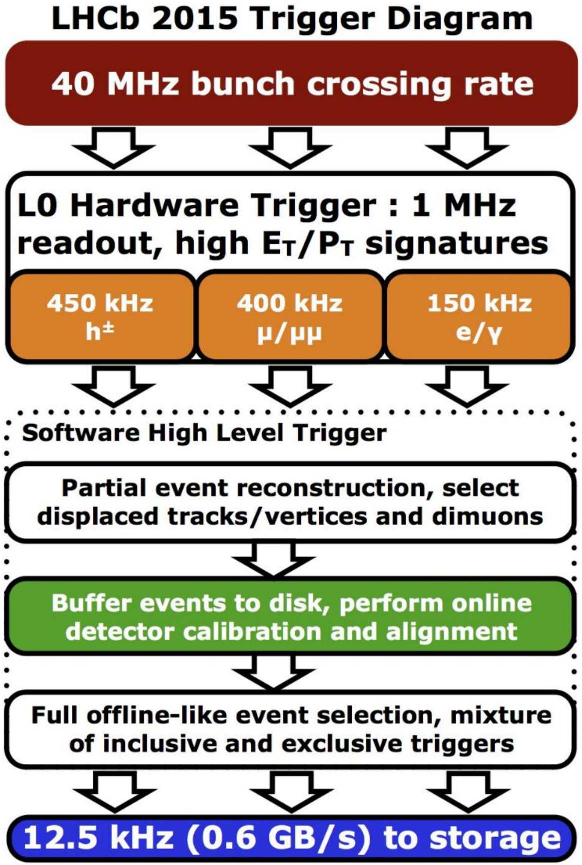
Bundesministerium
für Bildung
und Forschung

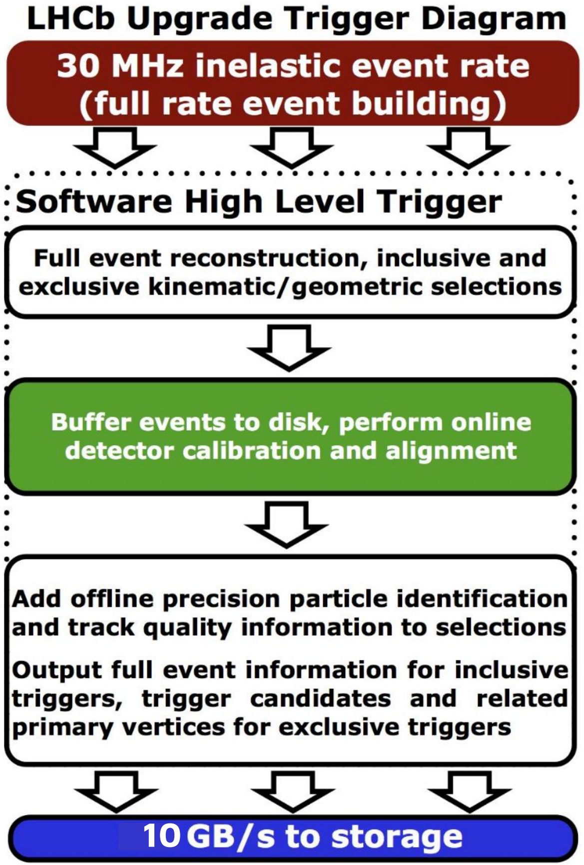
Barrier - share the work

- Scenario: Multiple algorithms select intersecting subsets of the same collection
- A following, expensive algorithm can be invoked once on the union of all selections → avoid duplication of work
- This requires **optional** data dependencies
- Imposes additional order constraints on basic nodes



LHCb Trigger Run II vs Run III





https://lhcb.web.cern.ch/lhcb/speakersbureau/html/TriggerScheme.html

References

- LHCb collaboration, Technical design report Trigger, March 2018, CERN-LHCC-2018-007. LHCB-TDR-017
- LHCb Collaboration, Trigger schemes website, visited 24.02.2019, modified 2-5 Gb → 10 Gb
  https://lhcb.web.cern.ch/lhcb/speakersbureau/html/TriggerScheme.html
- Manuel Schiller, “SoAContainer”, visited 25.02.2019,
  https://indico.cern.ch/event/736105/contributions/3036391/attachments/1687650/2717271/talk.pdf
- Illya Shapoval, Adaptive Scheduling Applied to Non-Deterministic Networks of Heterogeneous Tasks for Peak Throughput in Concurrent Gaudi, CERN-THESIS-2016-028