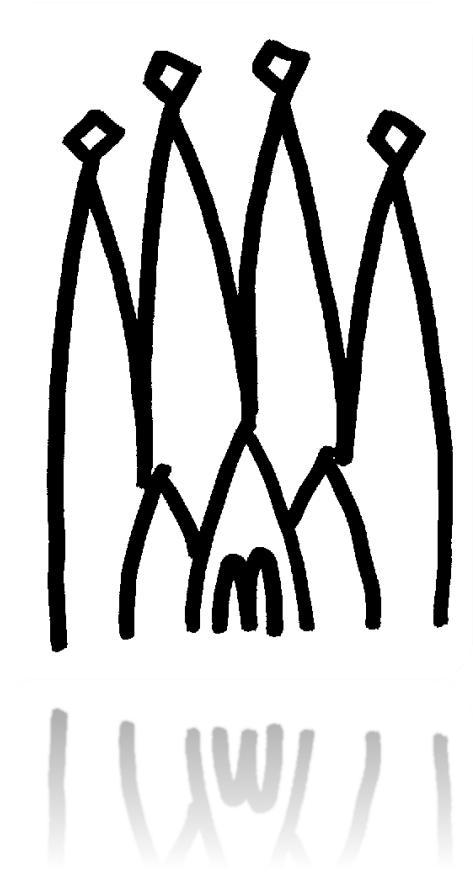
B. Hegner, P. Mato, D. Piparo

UPDATE ON WHITEBOARD AND SCHEDULER: INTEGRATION INTO GAUDI



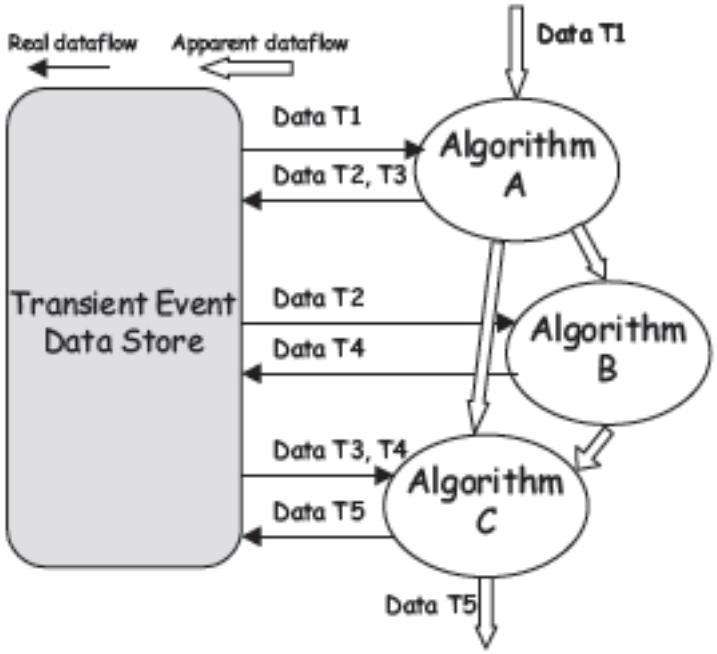
Outline

- Our goal
- Multicore Gaudi prototype: Design Cornerstones
- An Example: the LHCb Reconstruction Application
- Conclusions

Many thanks to M. Clemencic, M. Frank and I. Shapoval for the useful discussions and contributions and to A. Nowak & Openlab for the machine on which we ran.

Our Goal

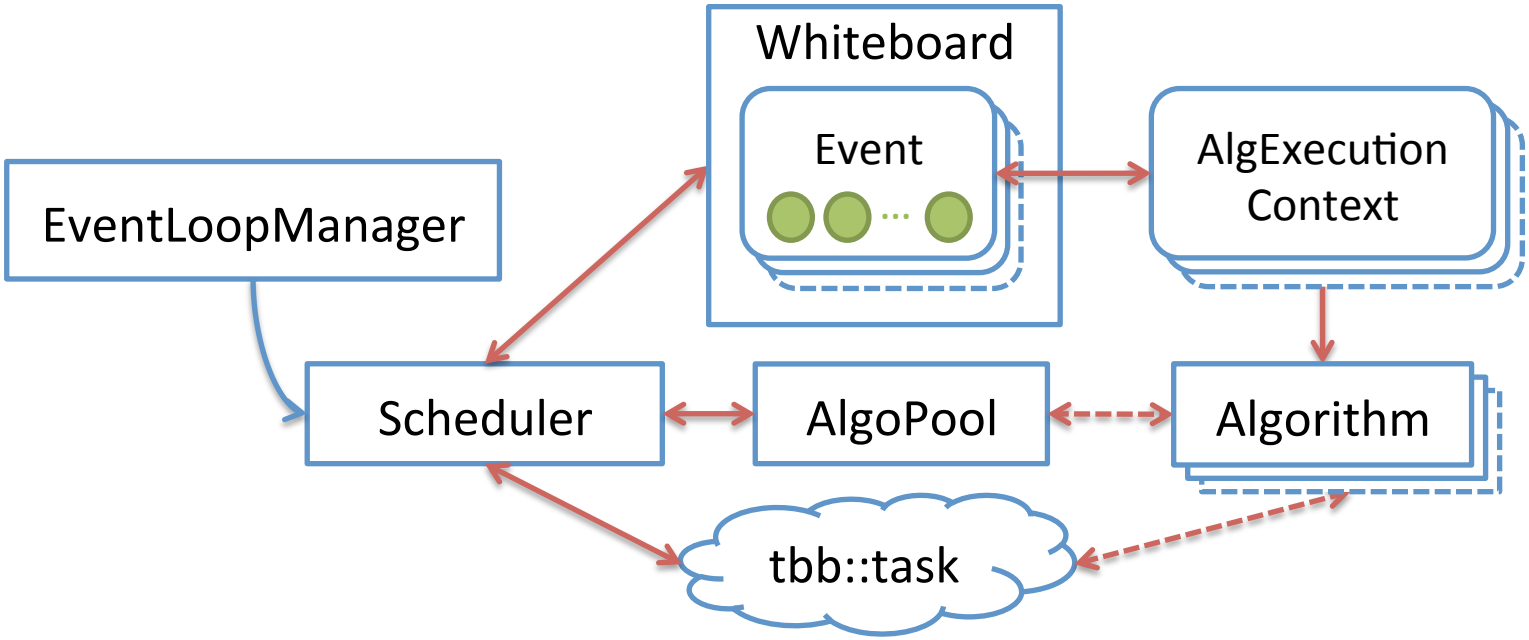
- *Gaudi is serial* as several (all) well known other HEP software frameworks
- The idea is to introduce concurrency
  - Several algorithms running concurrently
  - Several events processed simultaneously
- Starting point: the whiteboard demonstrator
- Presence of data triggers the scheduling of algorithms



Gaudi2CMSSW translation: Algorithm == Module

Prototype Design Cornerstones

What we started to call *Gaudihive*



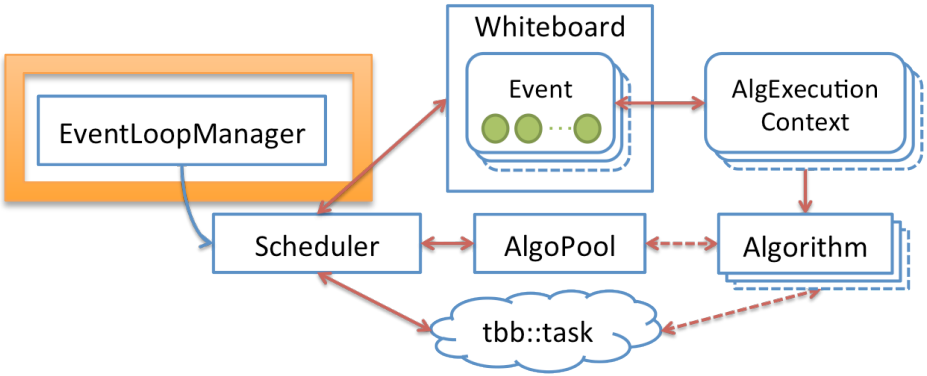
Gaudi2CMSSW translation: Algorithm == Module

Prototype Design Cornerstones

EventLoopManager:

- Responsible for event looping
- Adjust number of in-flight events at runtime
  - Add a new dimension to parallelism
- Integrated in Gaudi as a derived class of `MinimalEventLoopManager`

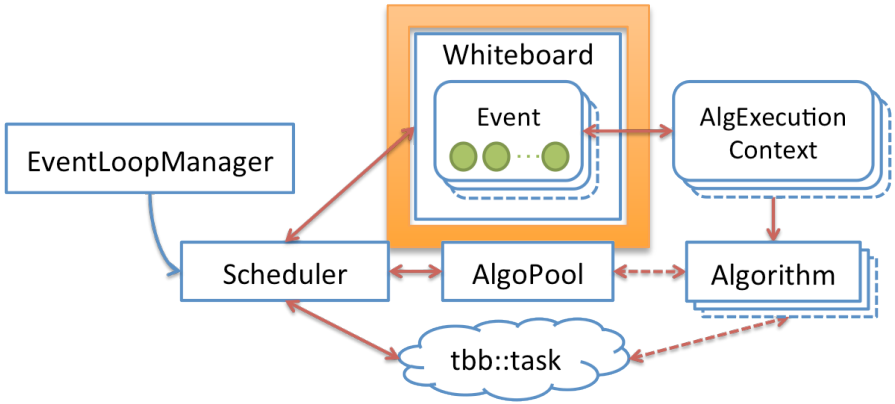
Implemented as: loop until events are finished containing a loop over events in flight containing a loop on algorithms.



Prototype Design Cornerstones

Whiteboard:

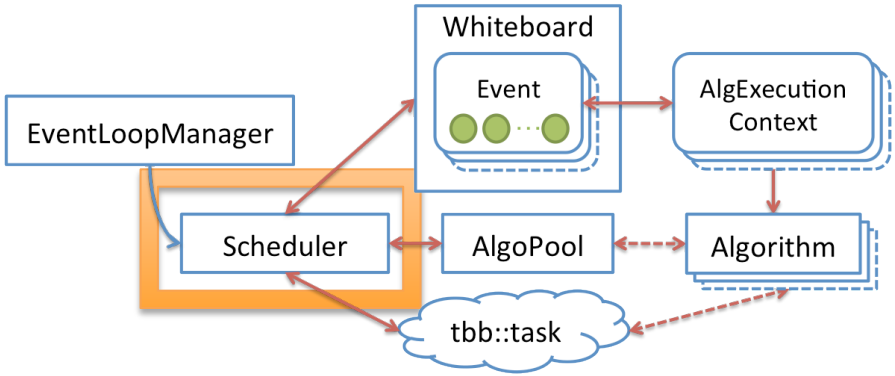
- Can contain several events
- Integrated within the existing Gaudi `DataSvc`
  - Which was made thread safe



Prototype Design Cornerstones

Scheduler:

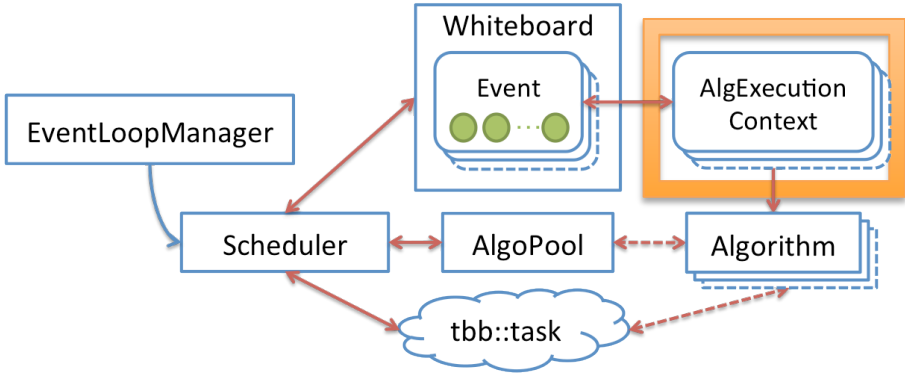
- Controls submission of algorithms according to data dependencies
  - Input data available → algorithm scheduled
- Can manage multiple events simultaneously
  - Increase probability for an algorithm to be "schedulable"



Prototype Design Cornerstones

Algorithm execution context:

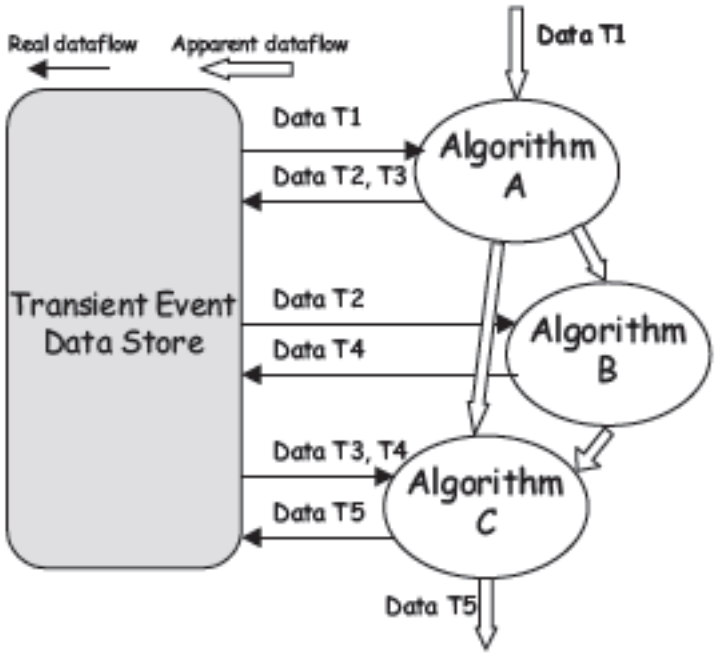
- Puts the algorithm in the right "context" for the execution
- Event specific data

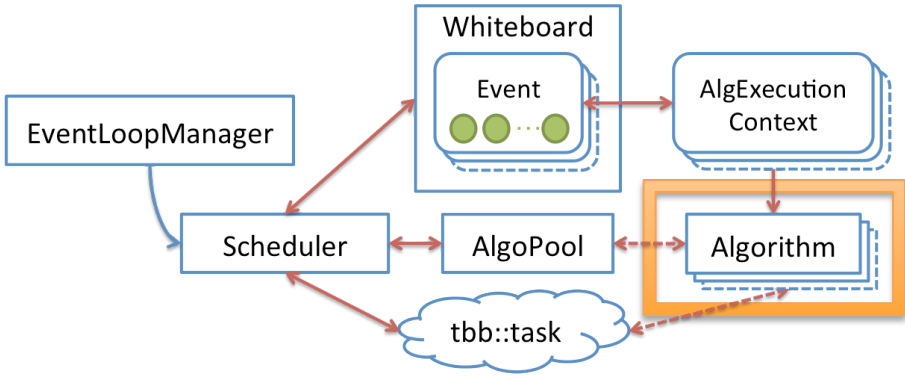


Prototype Design Cornerstones

Algorithm:

- Declare input data

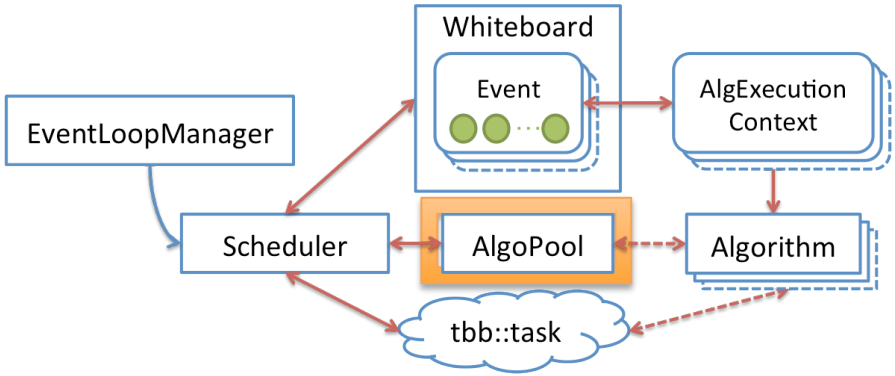




Prototype Design Cornerstones

AlgoPool:

- Manages the lifetime of algorithms
- Can create clones of algorithms at runtime if needed
  - Increase parallelism with multiple instances of the same algorithm running simultaneously



Technologies

TBB

- Task model
- Concurrent containers
  - concurrent_vector, concurrent_queue

C++11

- Atomics
  - Reference counting, counters
- Confortable syntax
  - Range based for loops, auto, tuples

PARTIAL BOTTOMLINE

A prototype of a multicore Gaudi version has been created with which:

- Algorithms can be executed concurrently
- Multiple events can be handled simultaneously

Performance Measurements

The Setup

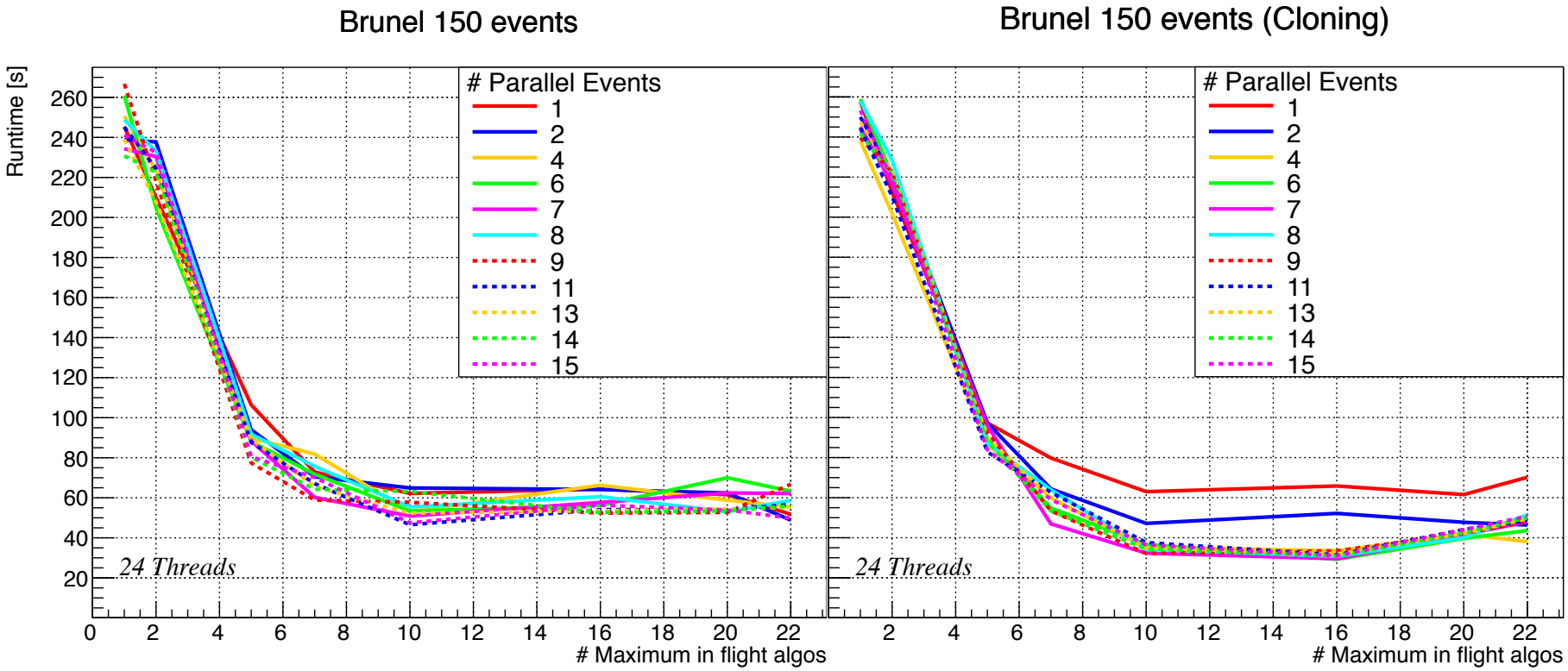
Machine:

- Intel Xeon X5650, 2 x 6 cores (12 threads) ≅ 24

Application:

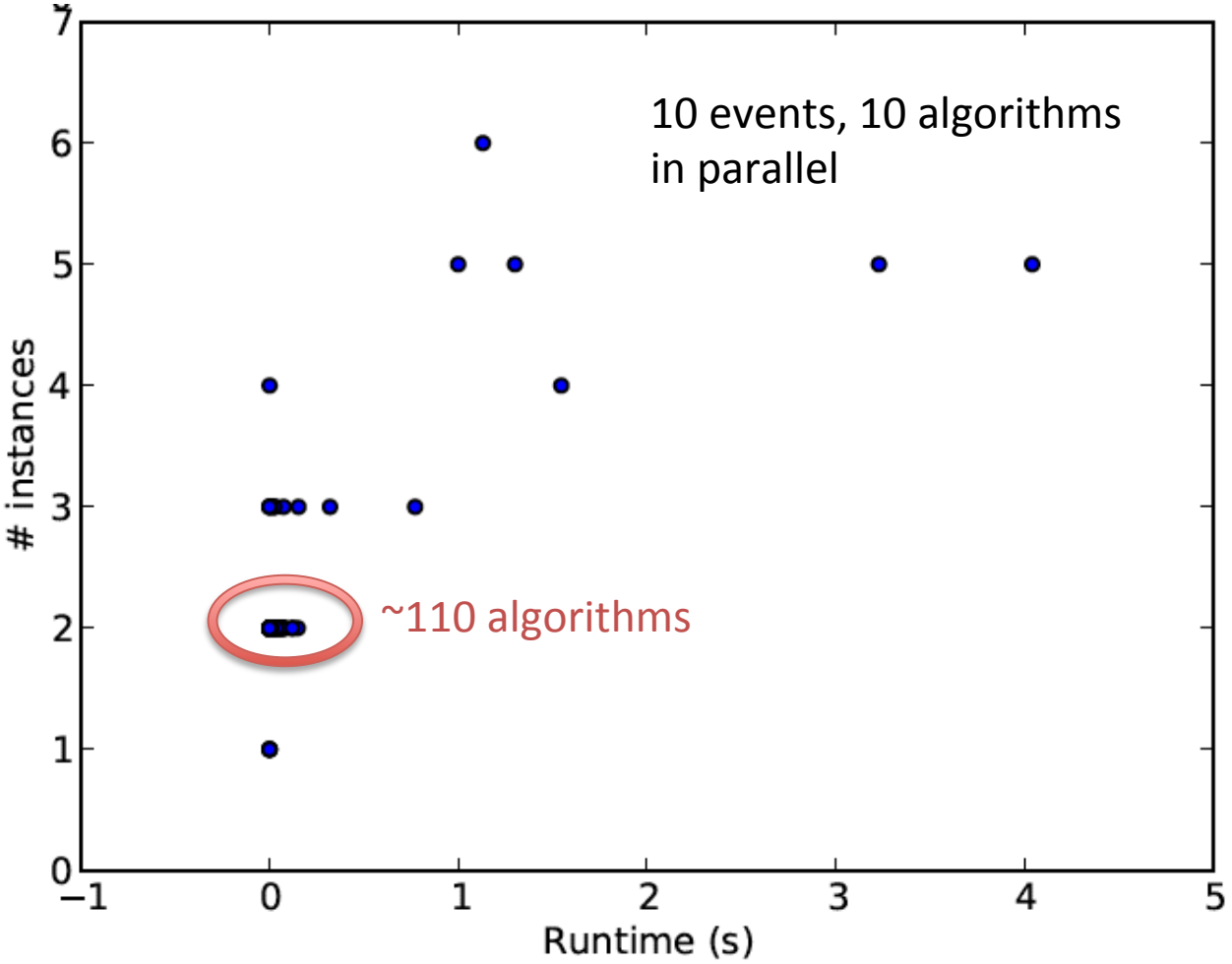
- Typical LHCb reconstruction chain (Brunel)
  - 214 algorithms involved
- Use a placeholder algorithm
  - Keeps CPU busy
  - Running time is a parameter
  - Dependencies + runtime of Brunel's algorithms maintained

Execution Speed



- Speedup depends strongly on the workflow chosen
  - Data dependencies of algorithms vary
- Cloning algorithms allows to increase parallelism (even with a moderate number of events in flight)

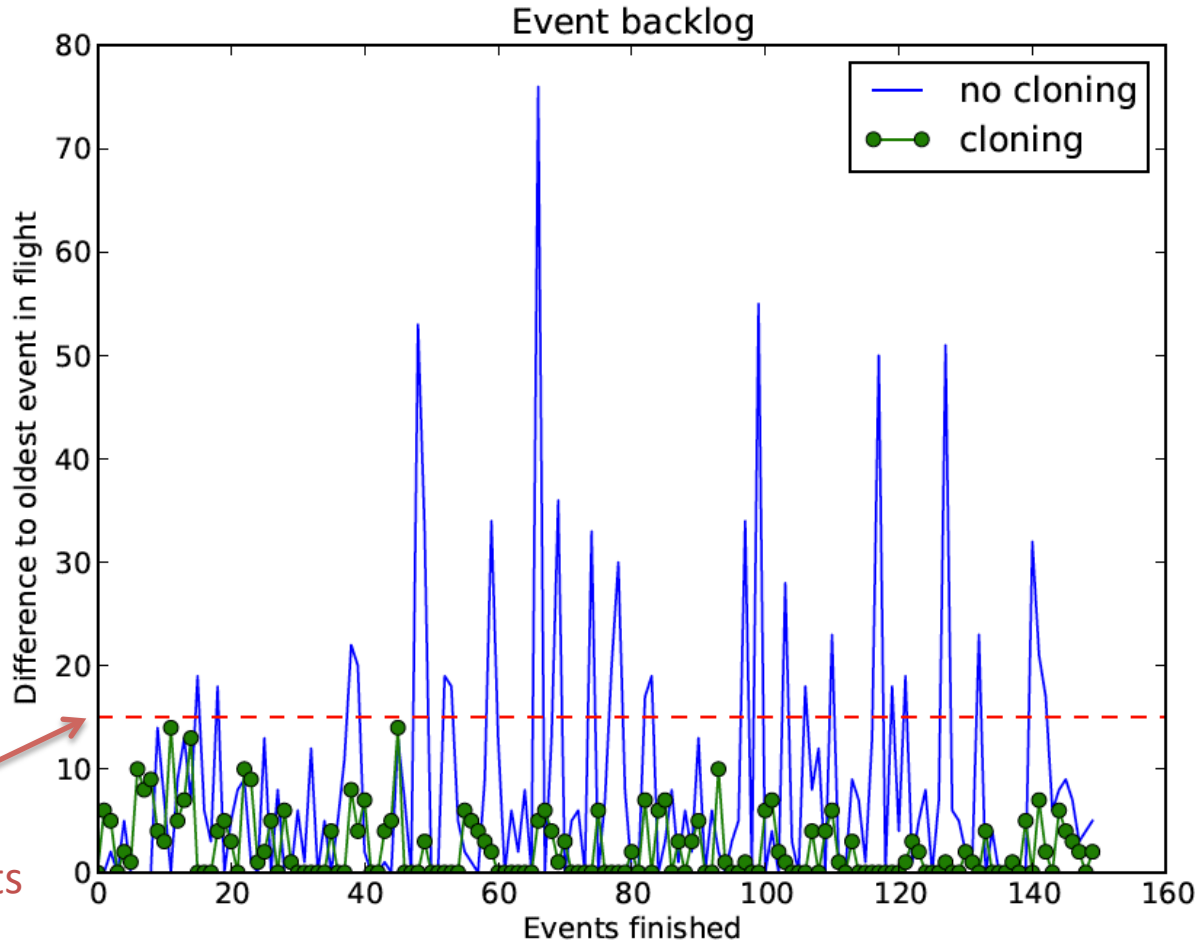
Clones – Runtime Relation



- Long running algorithms have multiple clones
- A high number of short algorithms have 2 copies:
  - Forbid multiple copies for those and inspect runtime

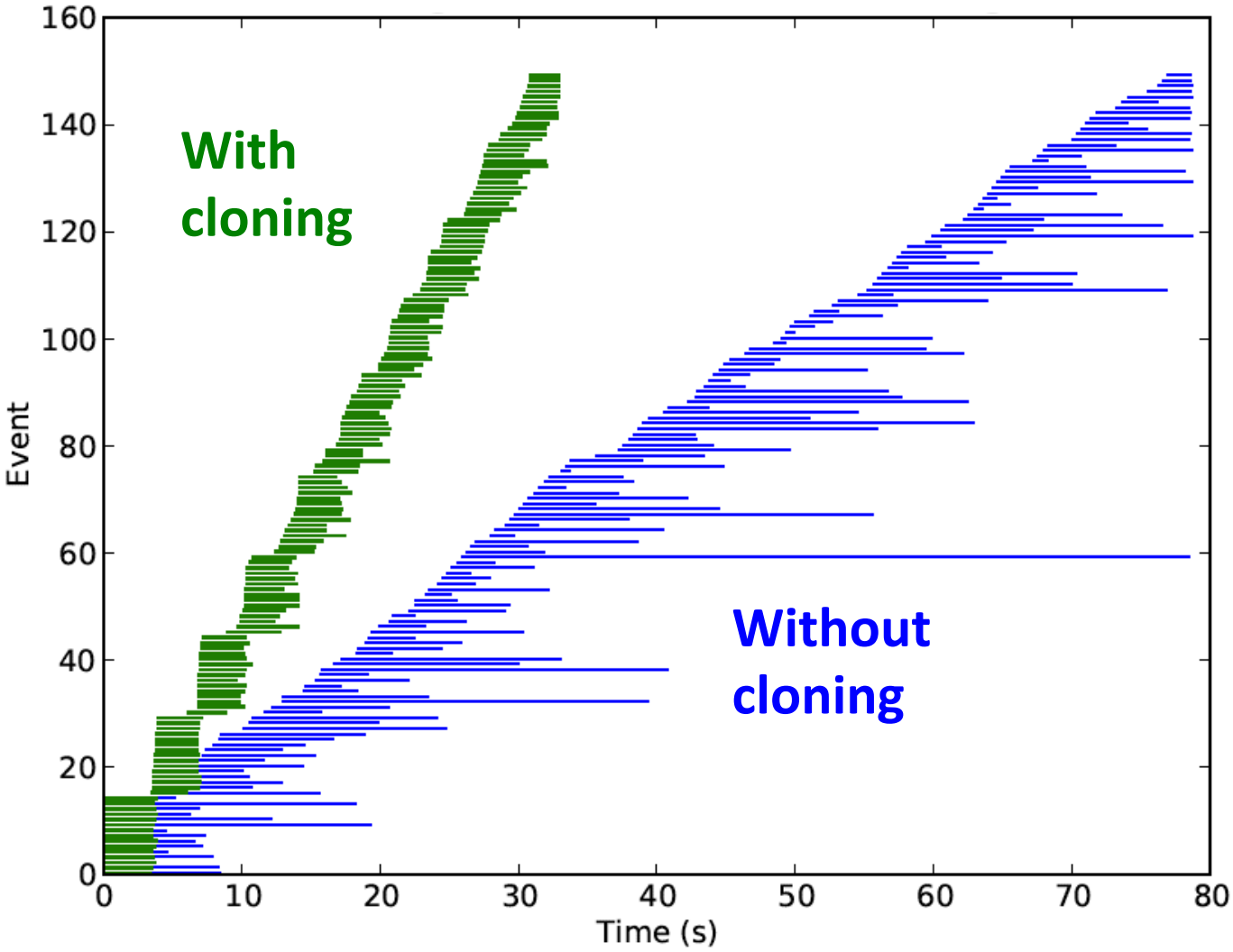
"Event Backlog"

- Event backlog: difference between latest event put in flight and oldest event being processed.
- Cloning helps maintaining a little event backlog.



Number of events in flight

A Snapshot Of The Events In Flight



Conclusions

- A prototype of a multicore Gaudi has been developed
  - Manage algorithms concurrently
  - Manage multiple events
- Reproducing an LHCb reconstruction job:
  - Important speedup
  - Cloning increases parallelism, keeps events backlog low

- All ingredients are there to repeat the exercise and compare results using CMS input
- Perform the same study for Atlas?

BACKUP

The Gaudi Framework (In a Nutshell)

- Experiment independent software framework
- Adopted at LHC by Atlas and LHCb
  - But also GLAST, HARP, DayaBay,MINERvA
- Designed with flexibility and reusability in mind
  - Loose coupling among components
  - Abstract interfaces and plug-ins
- Solid base for all applications in a HEP experiment
  - Simulation, reconstruction, event display, user analysis…

Gaudi is serial: the idea is to introduce concurrency starting from the Whiteboard demonstrator