Navigation Timing Studies of the ATLAS High-Level Trigger

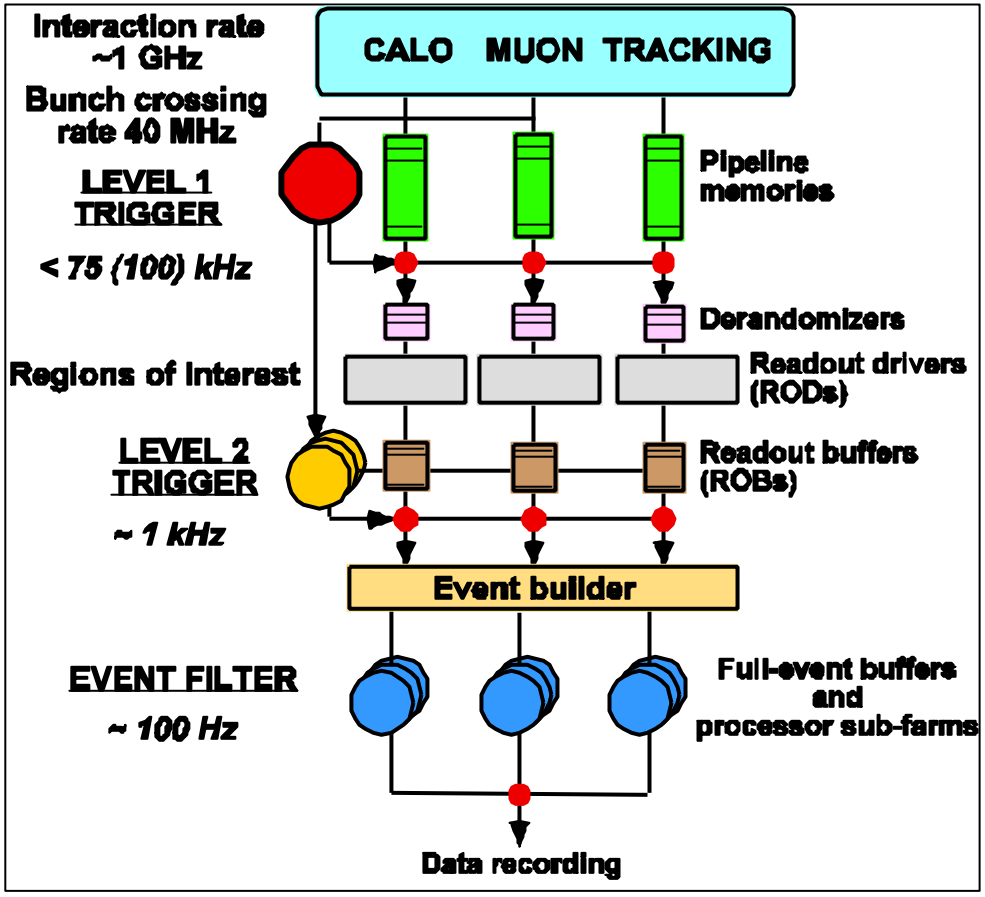
Andrew Lowe

Royal Holloway,

University of London

The ATLAS Trigger/DAQ System

- Three-level trigger architecture
- LVL1 acts on data from a subset of the detectors
- LVL2 uses full-precision data from most detectors, but only examines “RoI”
  - Latency = 10 ms
- EF uses full event data and decides which events are recorded for offline analysis
  - Latency = 1 s



The High-Level Trigger Software

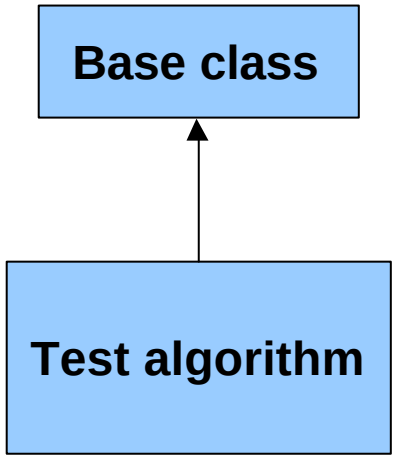
- The HLT (LVL2 + EF) operates on events that pass LVL1
- Boundary between LVL2 and EF is flexible to allow a trade-off of tasks between them, in order to optimise their roles
- HLT software will be the control and selection software which:
  - Runs in the online system for data-taking and testing
  - Runs in the offline software for development (particularly of algorithms) and efficiency/rate studies

The Navigation

- Navigation is part of the mechanism by which the reconstruction is guided to the event fragments needed from preparing the trigger decision
- Require that the time spent doing the navigation is less than 1% of the total time taken for algorithm execution (~10 ms)
- HLT selection algorithms are the data-processing components of the trigger software

Test Algorithm

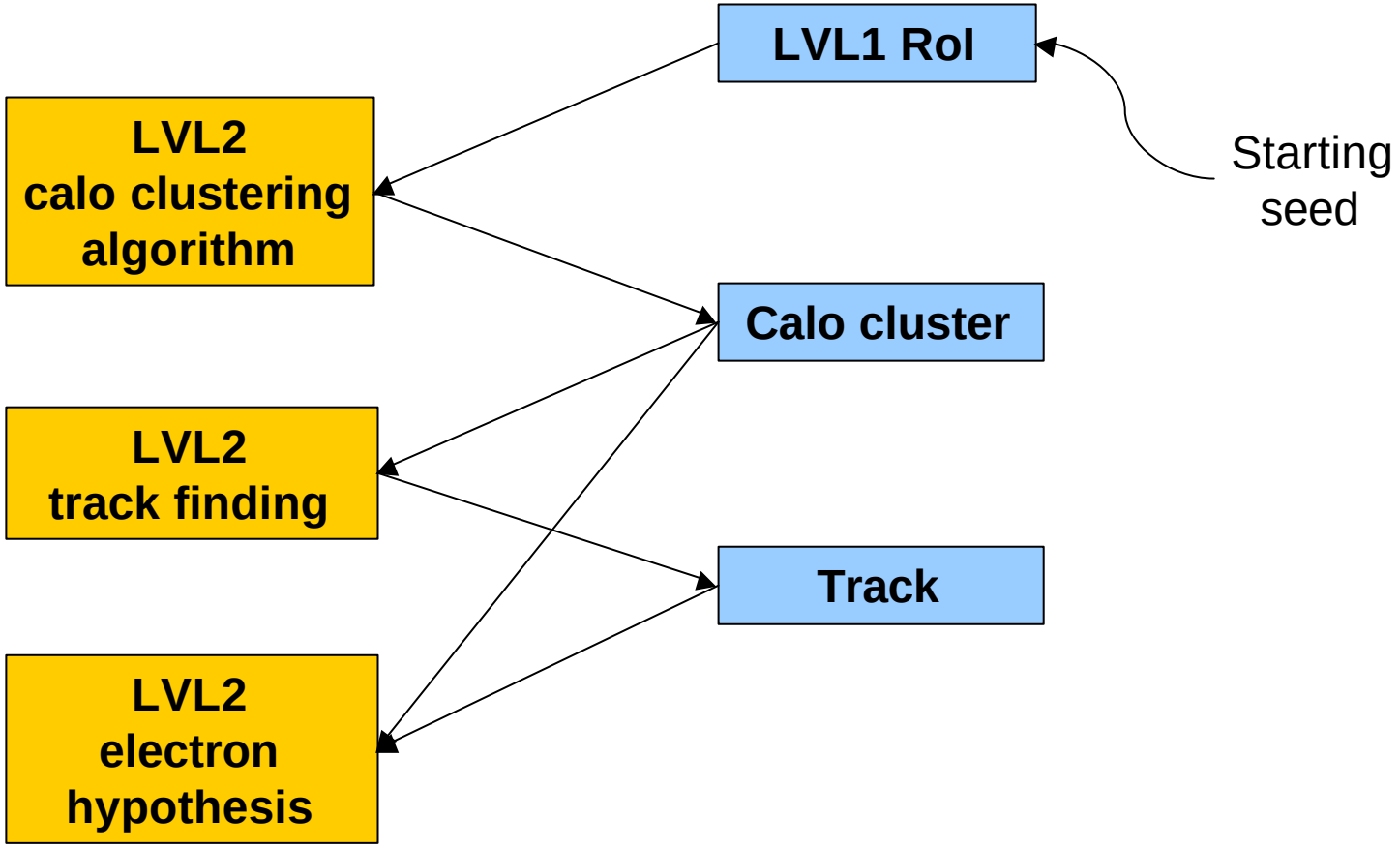
- Test algorithm inherits from a base class that is common to all HLT selection algorithms
- Base class contains “helper methods” that have been written to simplify navigation
- Methods are responsible for interactions with the store, which holds both run and event related data
- The test algorithm does most of the actions that real HLT selection algorithms do



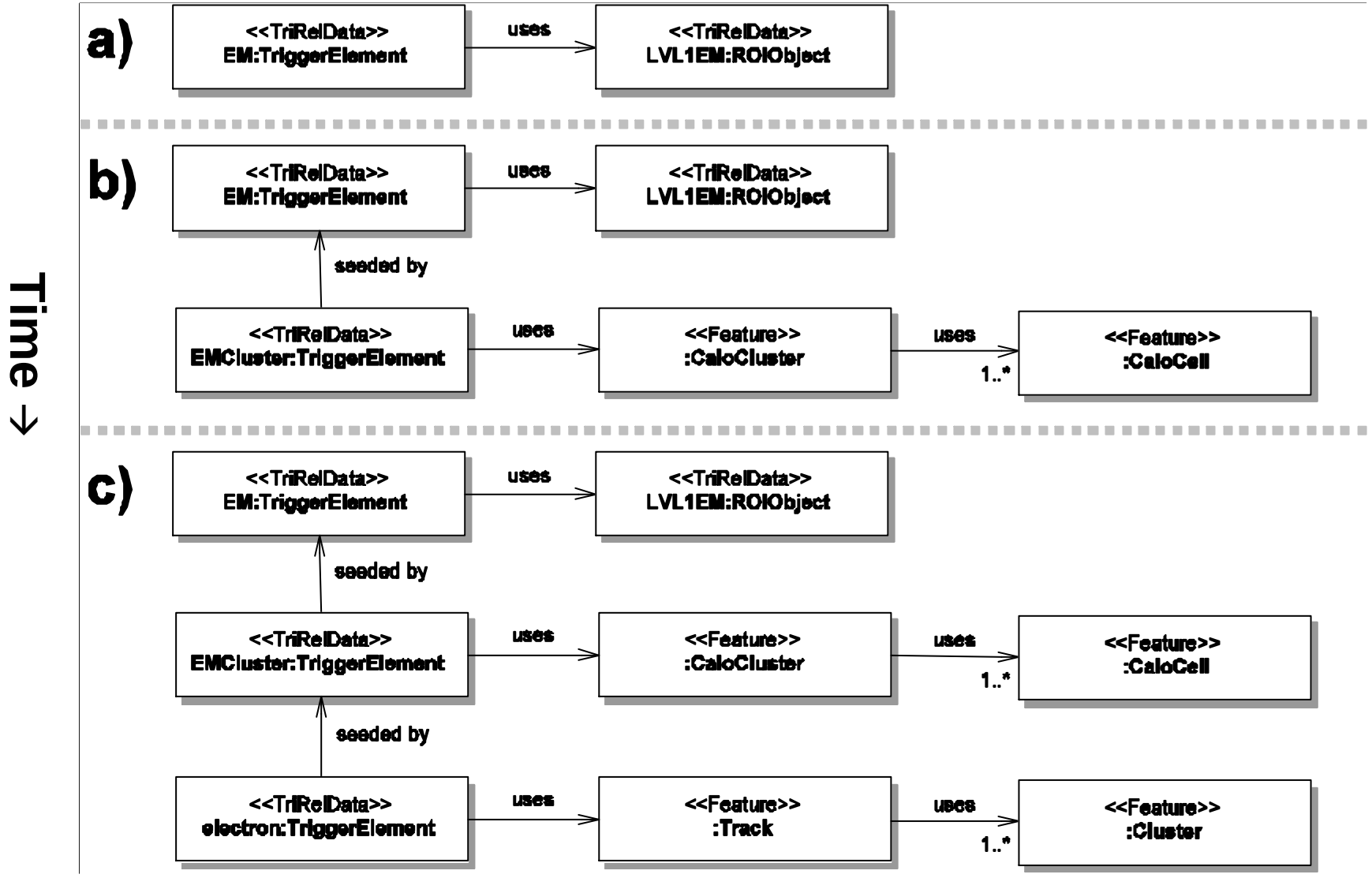
TriggerElements And Their History Objects

- Steering system responsible for ordering algorithm processing
- Objects of the class “TriggerElement” are used by the steering to steer the event processing
- Hold no data – but are related to data objects, and to each other, in a navigable way by means of relationship labels that are stored in the TriggerElement’s “history” object
- Implementation of history object:
  - Map relationship labels (string keys) to objects
  - “Hash multimap” provides optimal implementation:
    - Constant retrieval time
    - Typically faster than *sorted* containers that map keys to objects

Electron Trigger Example



The Seeding Mechanism



Navigation Helper Methods

- Important factors in determining the time taken to do the navigation:
  - Time taken to retrieve objects from a TriggerElement's history
  - Time taken to write objects to a TriggerElement's history
- Helper methods provided by HLT algorithm base class:
  - writeHistoryOfTE
    - Write an object, or a vector of objects, to a TriggerElement's history and attaches a label describing the relationship between the two, e.g., "uses", "seeded by", "excludes"
  - getVectorOfObject
    - Retrieves a vector of objects, that match a specific label, from a TriggerElement's history
- These methods were timed

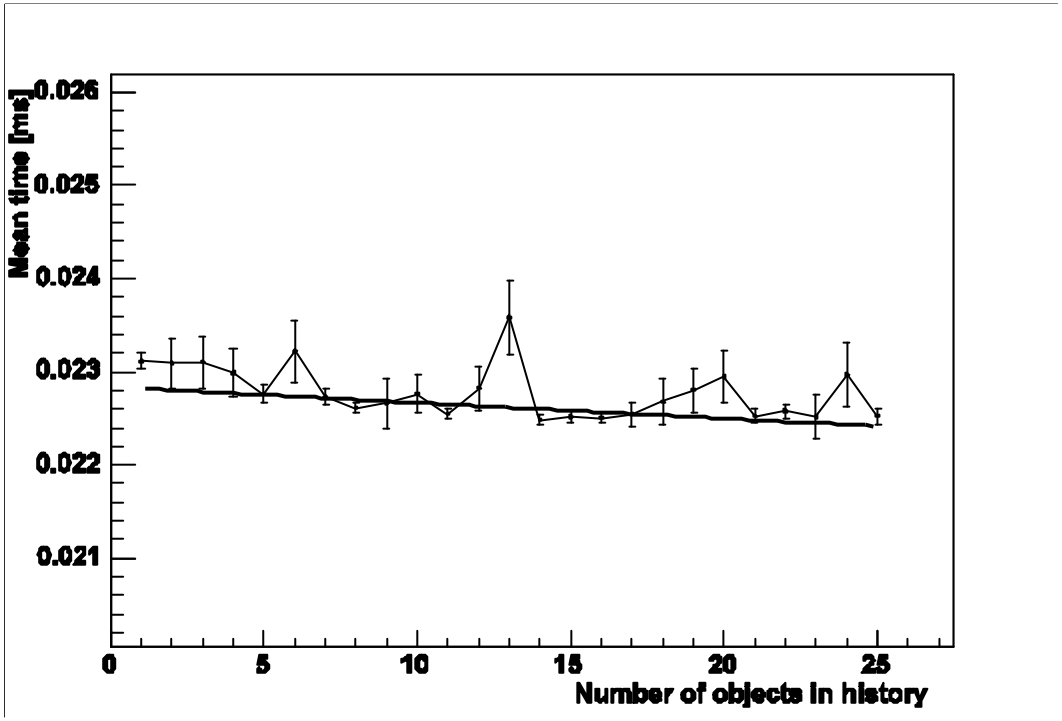
Factors Thought To Be Important In Determining Navigation Time

- Time taken to retrieve and to write objects to histories may depend on:
  - The size of the history object, i.e., the number of elements in the container
    - Are history objects with many objects slower to access than those with few objects?
  - The entry order of the object in the history
    - By definition, this type of container gives a constant retrieval time – expect size of history and entry order have no effect on retrieval time
  - The number of objects being retrieved
    - Obviously, one would expect the time taken to increase with the number of object being retrieved
- Vary these and make timing measurements

Tools And Method

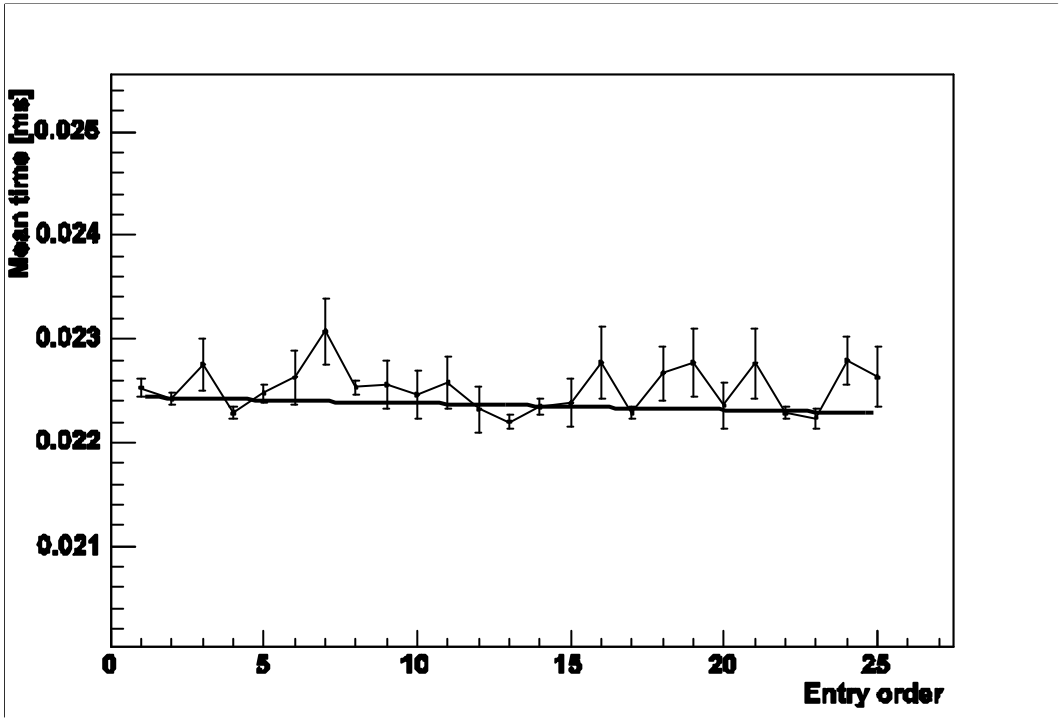
- Timing measurements were performed using a package that contains an algorithm that collects the time information, using the CPU clock, and enters it into an ntuple
- The units of the ntuples (and the plots presented here) are milliseconds
- Dedicated machine was used
  - Intel Xeon 2.40 GHz, 512 kb cache
- 100 events for each measurement
- To time a section of code, we need only add a statement before and after that starts and stops the timer
- We write 25 dummy TriggerElements to a history object, including one TriggerElement with a unique label

Retrieval Time: Size Of History



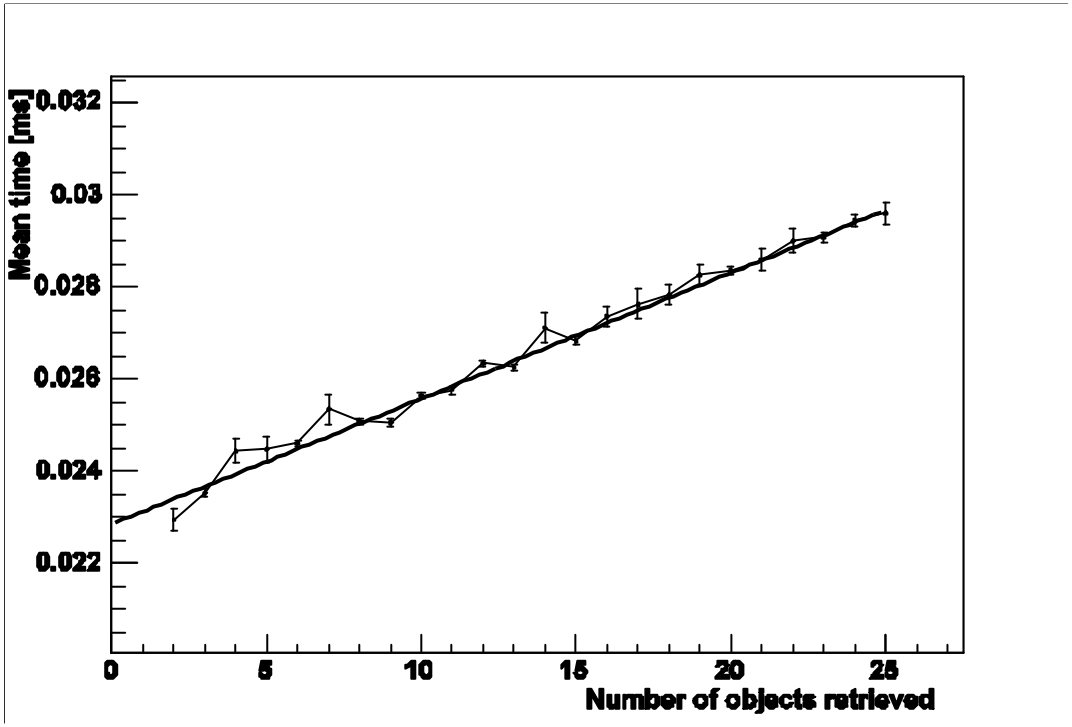
- As expected, the size of the history object (the number of elements within the hash multimap) has no effect on the retrieval time

Retrieval Time:
Entry Order In History



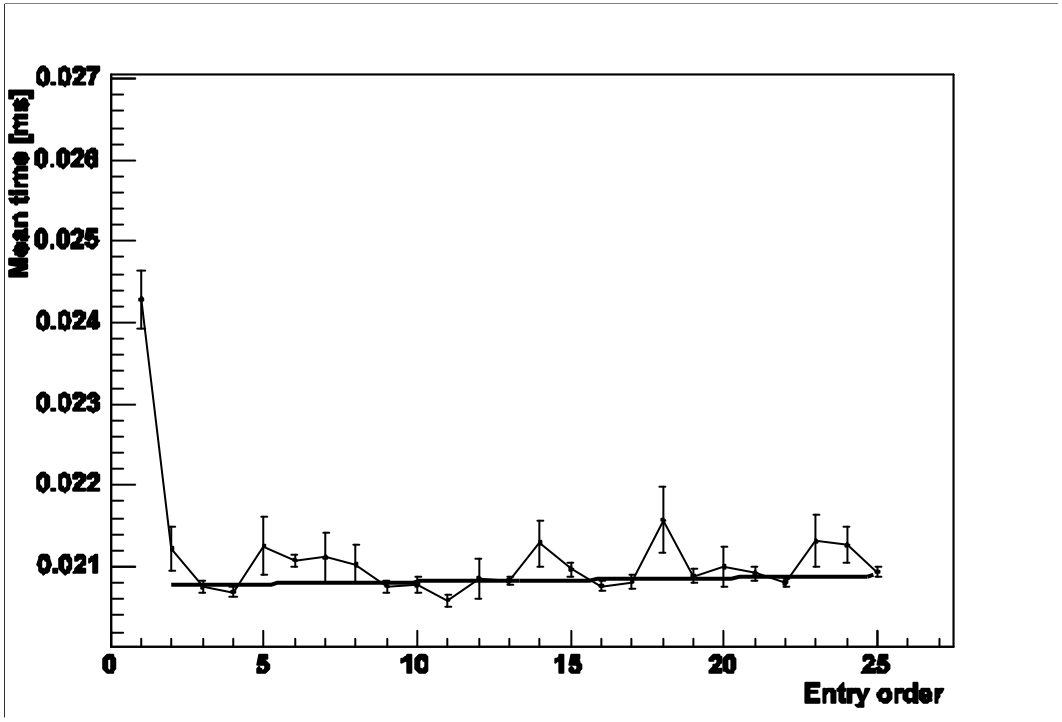
- The order in which the objects were written does not affect the time taken to retrieve the object with the unique label

Retrieval Time:
Number Of Objects Retrieved



- As one might expect, the time taken to retrieve objects increases linearly with the number of objects being retrieved

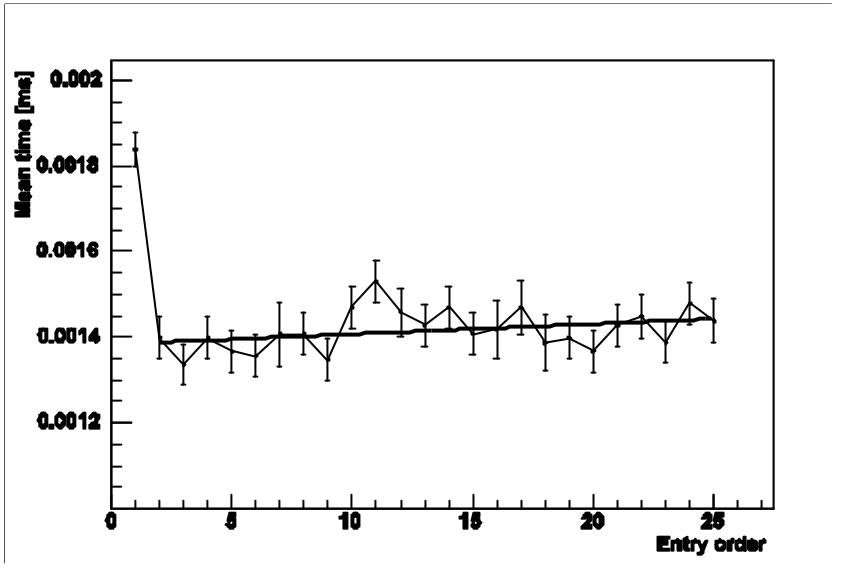
Writing Time:
Entry Order In History

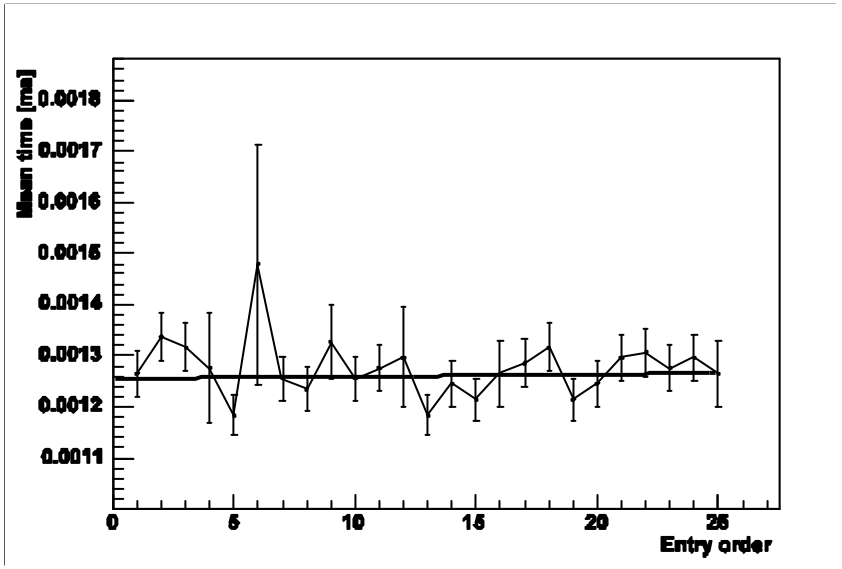


- There time taken to write an object is independent of the number of objects already written, but the first write takes longer

Effect Of Dynamic Cast

- This unusual feature was traced to a line of code that performs a dynamic cast
- If the dynamic cast is done beforehand, the feature disappears
- Probably due to caching





Results

- Mean retrieval time = 0.023 ms
- Mean writing time = 0.021 ms
- Typically we expect about two writes and retrievals during navigation
- Expect the navigation to take about 0.1 ms
- This is about 1% of the time taken to execute real HLT selection algorithms
- This indicates that the navigation imposes a very small overhead on algorithm execution
- This meets the goal stated previously