FTK high level simulation & the physics case

The FTK simulation problem

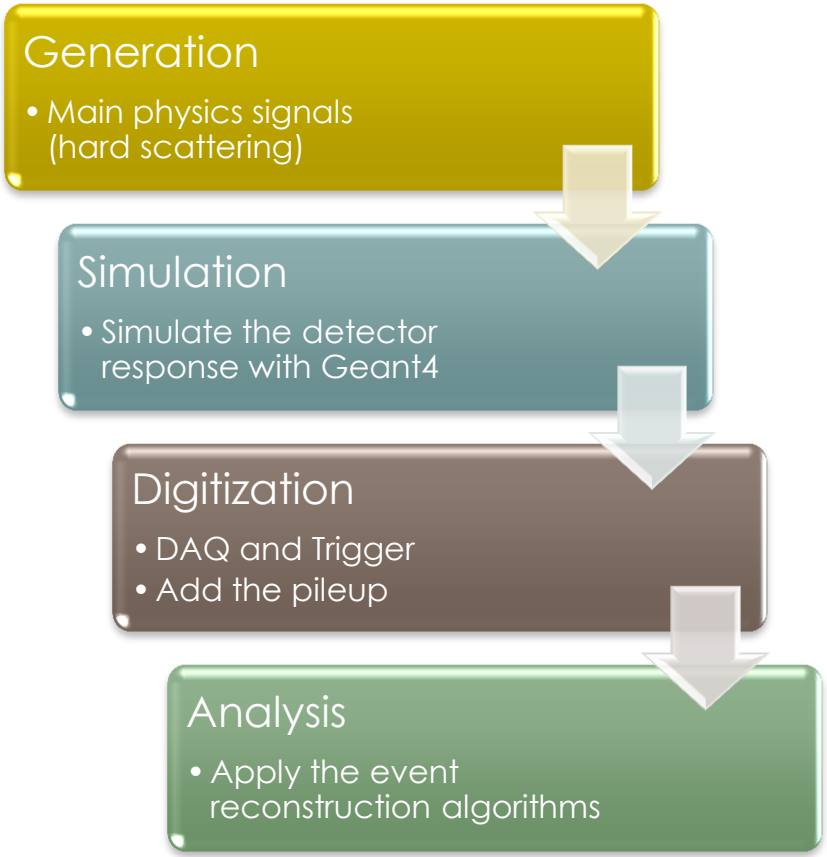
G. Volpi
Laboratori Nazionali Frascati, CERN Associate
FP07 MC Fellow

Introduction

- The FTK (high-level) simulation it is a set of C++ programs
  - Designed to interface with simulated and real data
  - It is able to reproduce the logic of the main electronic components within the pipeline
  - Allows to evaluate the load of the different boards
    - Can allow to evaluate the size of FIFOs and predict the bottlenecks
- This simulation has been extensively used as test bed for the algorithm and makes decisions on the design
  - In the past years was the place where the algorithms have been decided or studied: AM, track fitting, filtering techniques, …

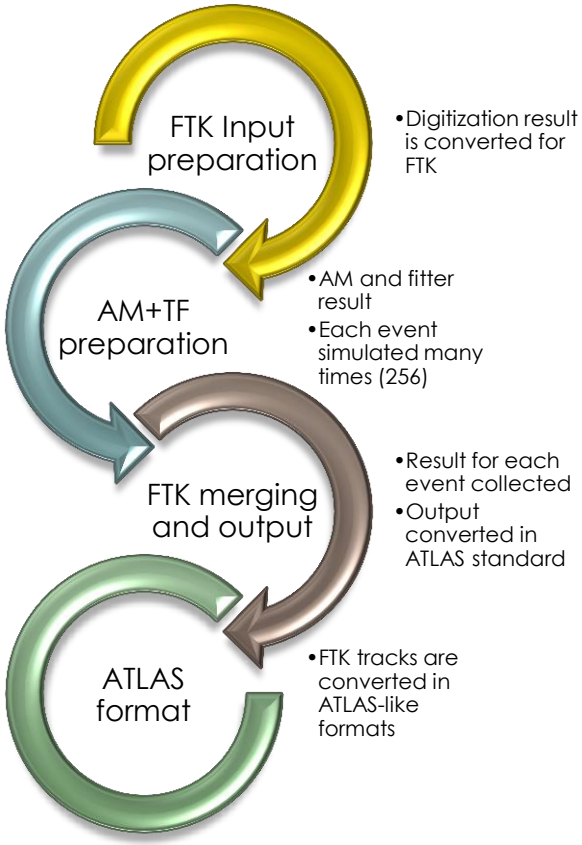
Physics case: ATLAS simulation

- The ATLAS experiment has developed a multi step MC to simulate the detector response
- The response of the full detector is simulated with great accuracy
  - Physics and electronic effects are included: pileup, noise, inefficiency
- FTK result should be added during digitization

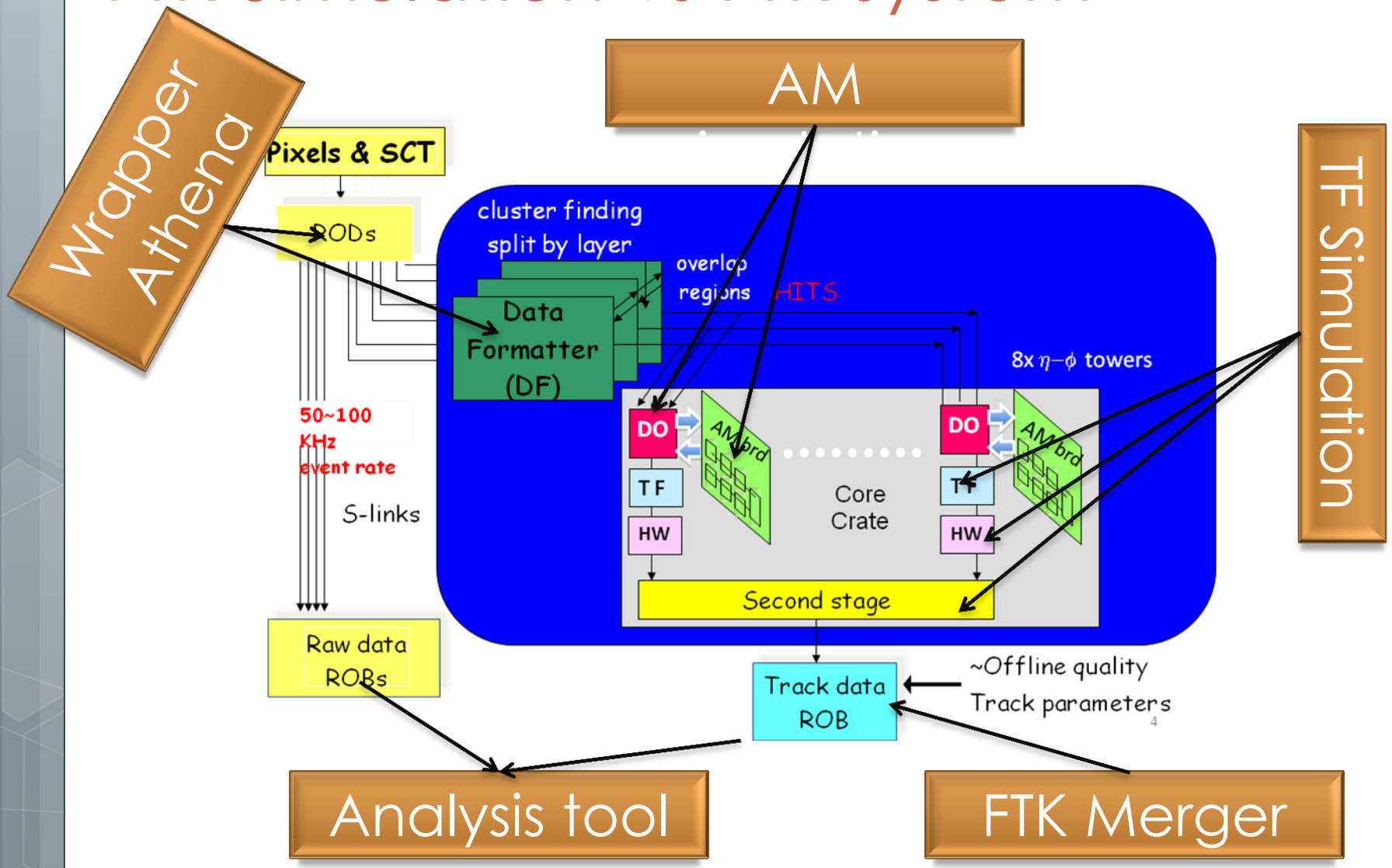


Physics case: FTK Simulation

- FTK simulation reads data from ATLAS after digitization
  - Only silicon tracker information is retrieved
  - Using a special format
  - Same procedure can be used to read RAW data
- The data are fed in a logical simulation of the FTK processor
- The output can merged back to the original ATLAS stream



FTK Simulation vs FTK system



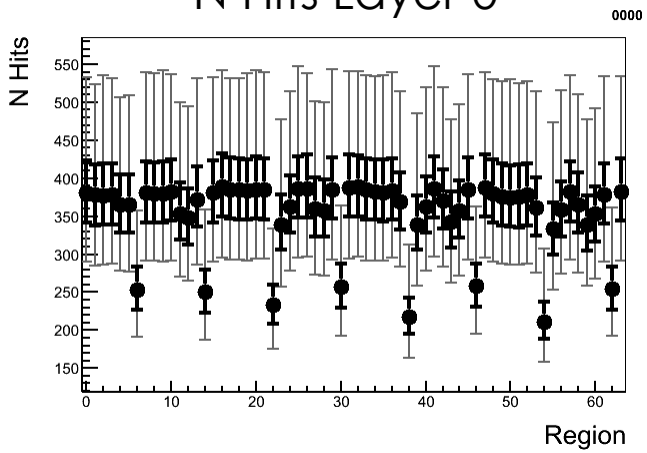
Why the simulation is a core component?

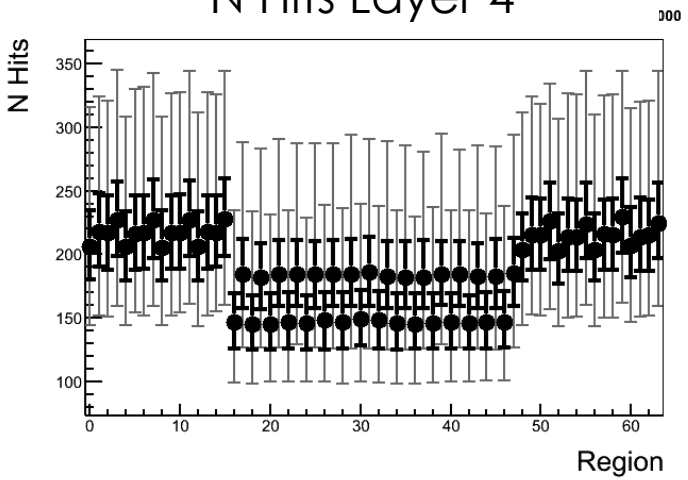
- The FTK simulation allows to study and to extend the Silicon Vertex Trigger idea to a different experiment
  - ATLAS inner track wasn't designed for trigger (CDF tracker was)
    - The simulation allows to define the units where the linear fit procedure can work (the **sectors**)
- The simulation prepares the configuration files
  - Pattern banks generation and setup of the "don't care bits"
  - Linearized fit constants preparation
- It is the most important test-bench for the algorithms
  - Issues in performance or data flow can be predicted and solution studied.

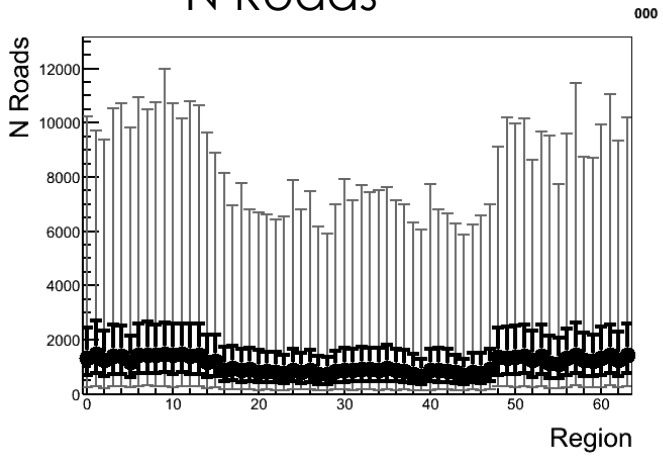
Simulation outcomes for the current architecture

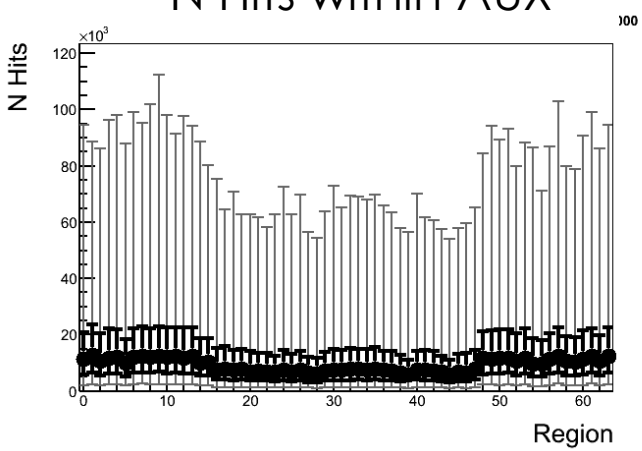
- Improved majority logic track fitting w.r.t SVT, limiting the number of constants
- Duplicate tracks suppression method (hit-warrior)
- Decision to drop the duplicate roads removal algorithm (a.k.a. road-warrior)
  - Simulation studies on high pileup sample revealed a modest benefits compared to the additional complexity
- Detail comparison among two main architectures:
  - two stages road finding and track fitting
  - single stage of partial road finding and two stage fits
    - Identical performance but simpler implementation
- Use of the "don't care" (DC) feature in the AM chip
  - Initial study on a system with AM chip + "Tree search processor"
    - DC feature was found more effective

Result: Dataflow

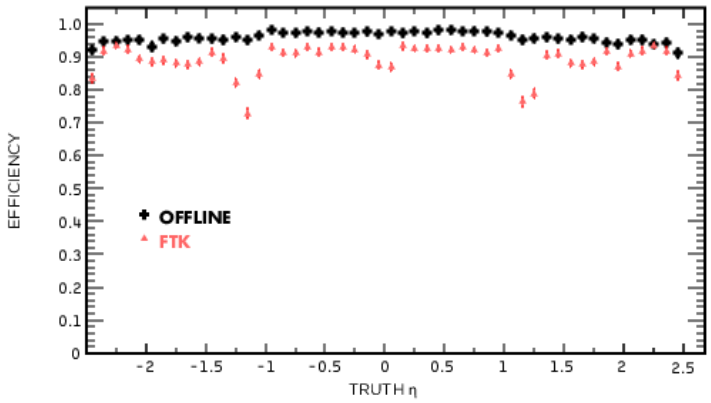


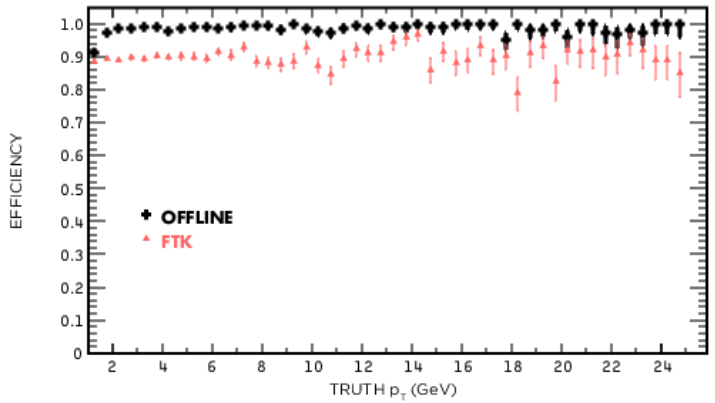


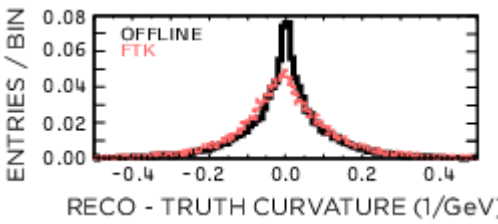


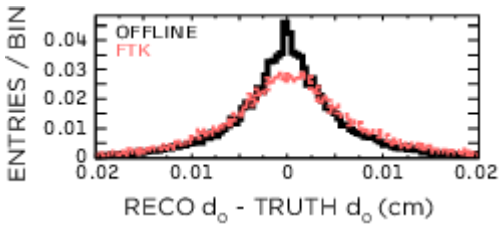


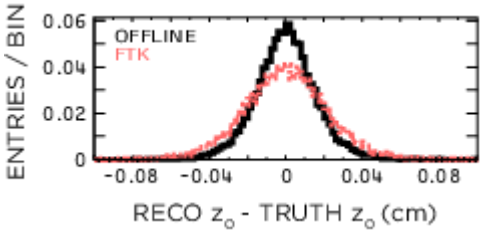
Result: Single Track performance

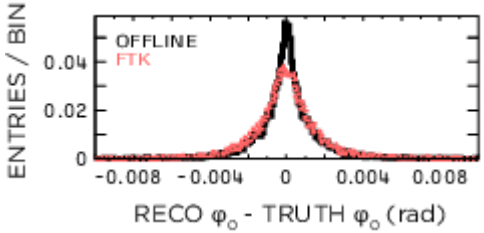


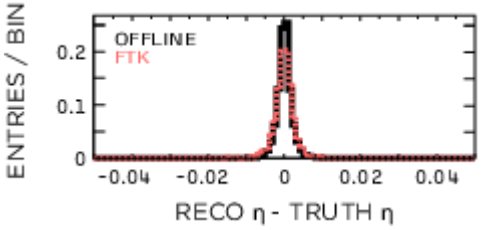




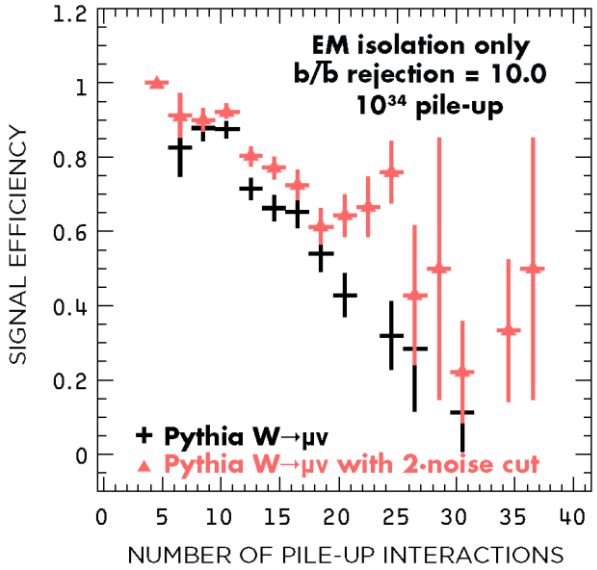


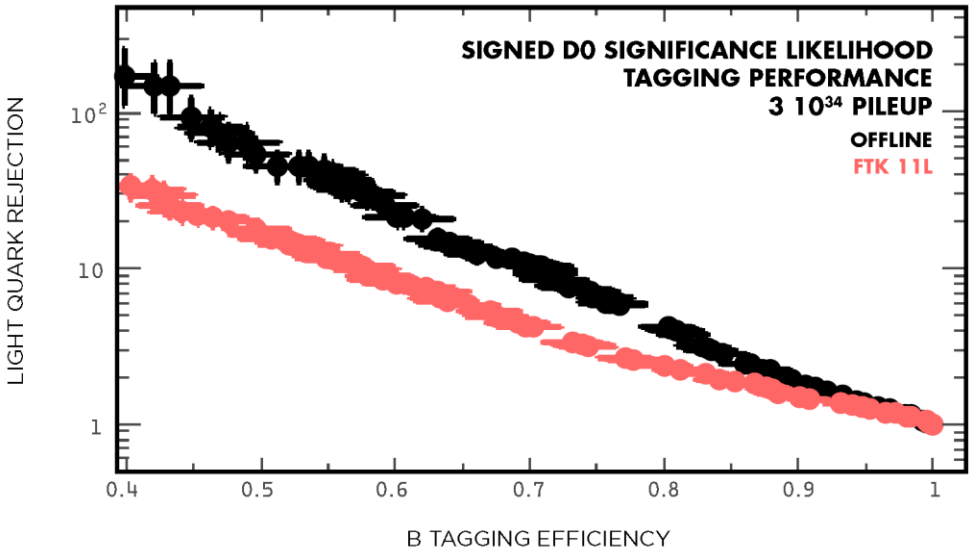


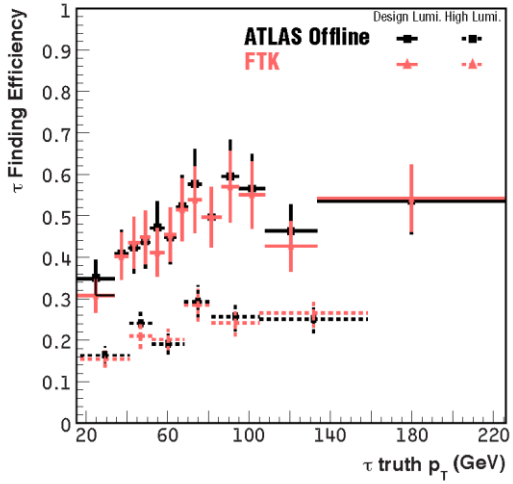


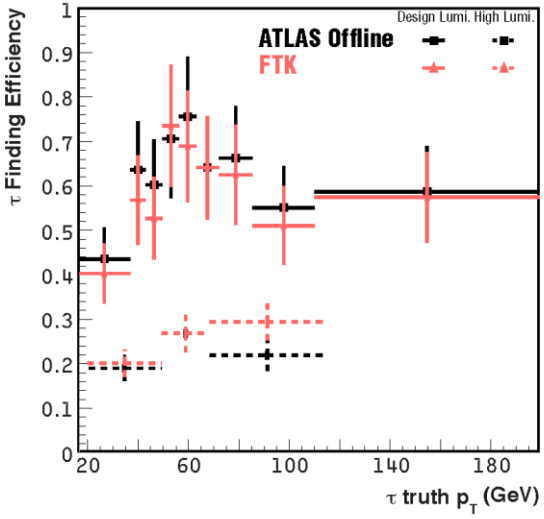


Result: Basic Performance





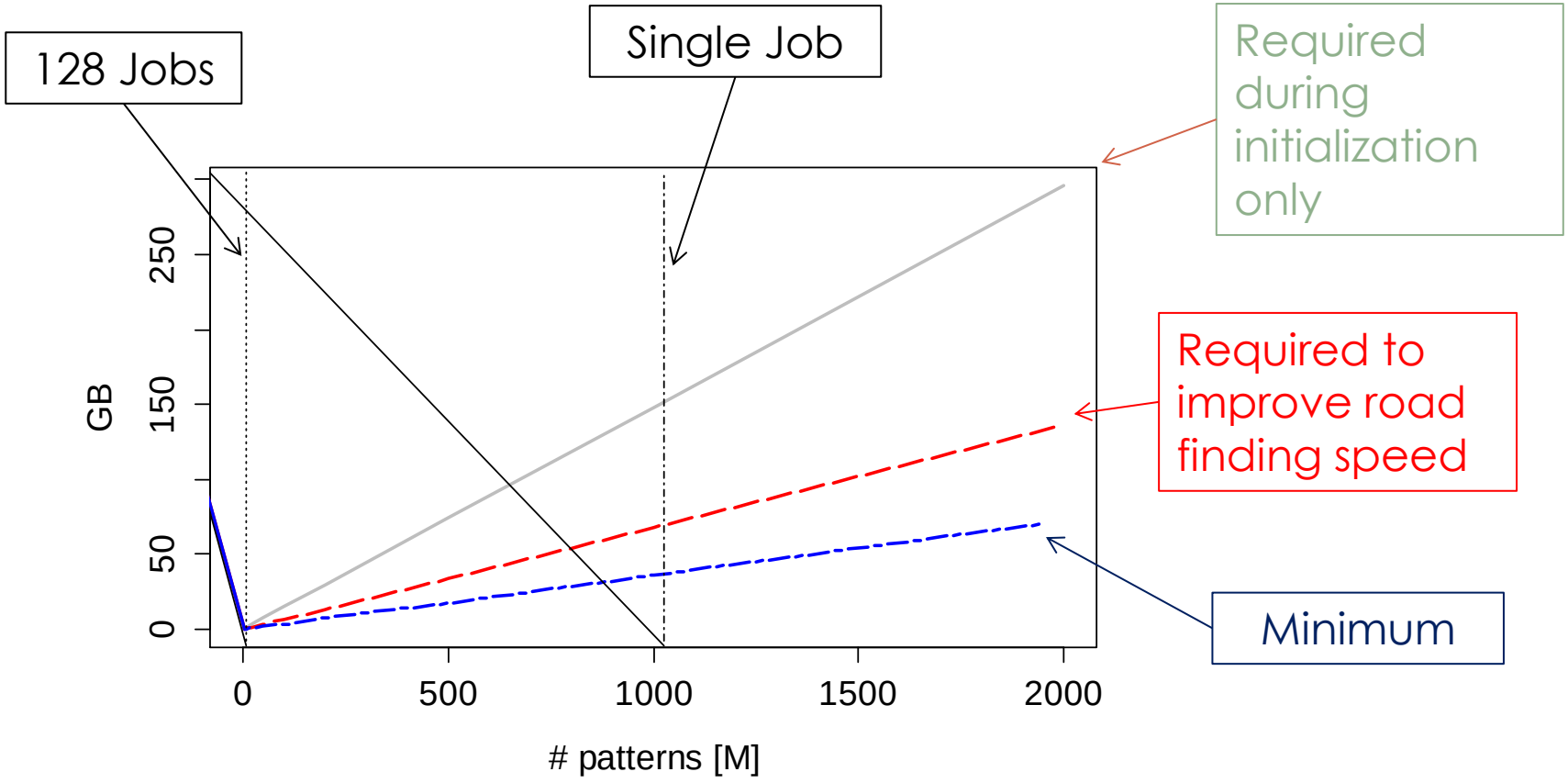




FTK Simulation challenge

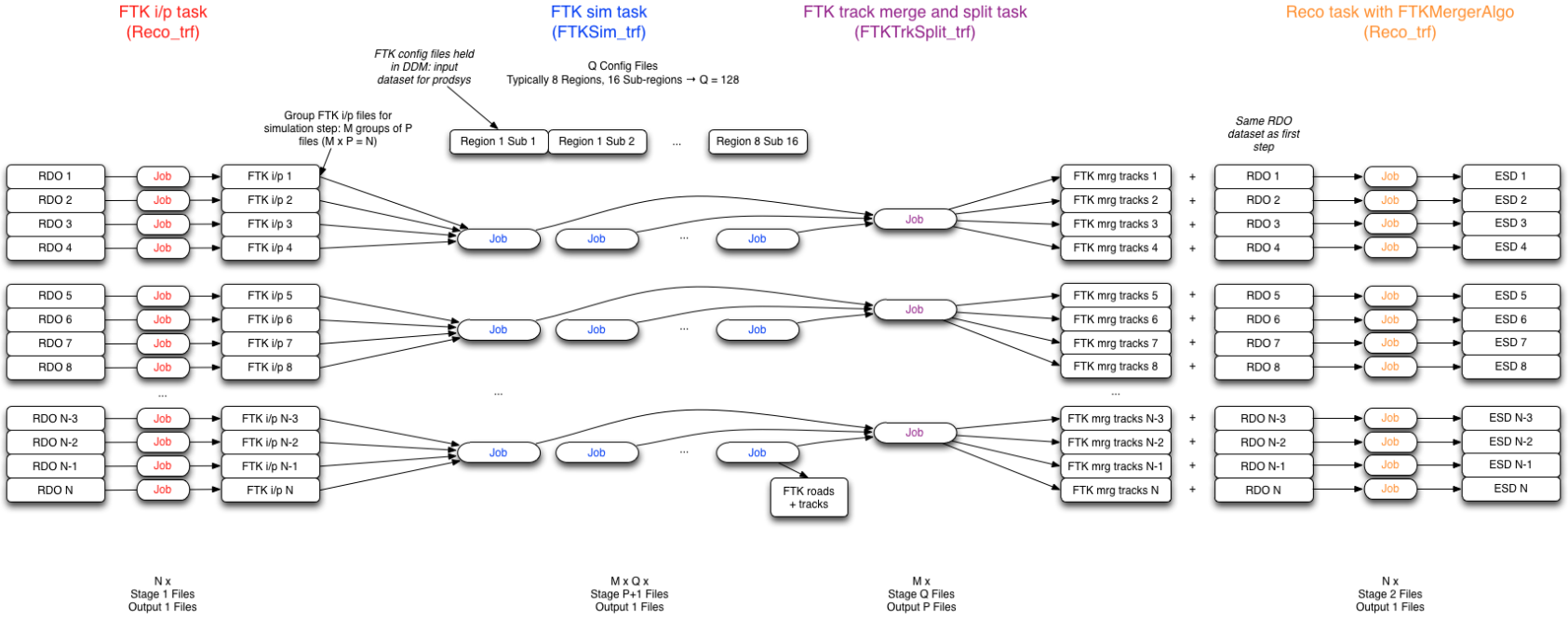
- The algorithms used by FTK are very inefficient on a CPU
  - Designed to make intensive use of memory and I/O
- Pattern matching algorithm is particularly tricky
  - The final hardware will be able to contain 1 Gpatterns
  - Each pattern is now described using 36 bytes
    - Additional indexing structures are required to improve the performance (almost double the memory/pattern)
  - Target HW is the typical grid WN: 3 GB/core
- We developed a strategy to simulate each event against a part of the pattern bank
  - Each job remains within the memory limit
  - Many parallel jobs allow a more effective use of the GRID
  - There is the need to merge the output

Memory budget



Not including the memory required to store the hits, other algorithms, libraries …

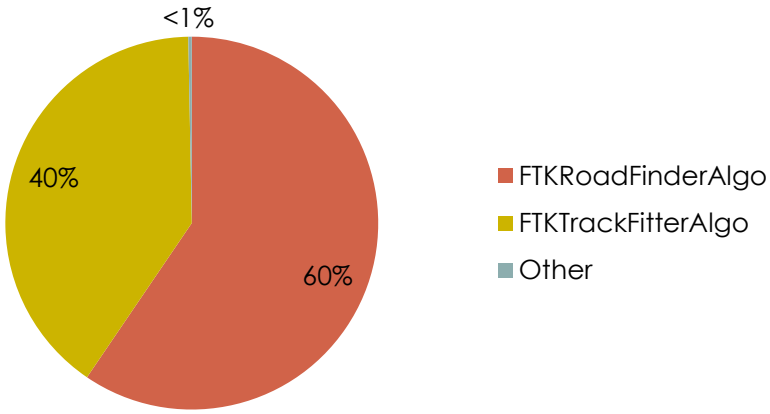
The grid simulation flow



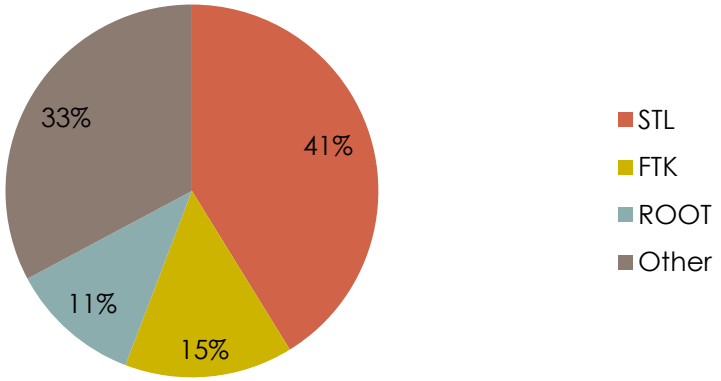
- The job segmentation results in a complicated submission scheme
- The average processing time per event is 12 minutes
  - A substantial time is spent in initialization and book keeping

1.9 GP (x2, pessimistic)
PU 70
No code optimization (yet)

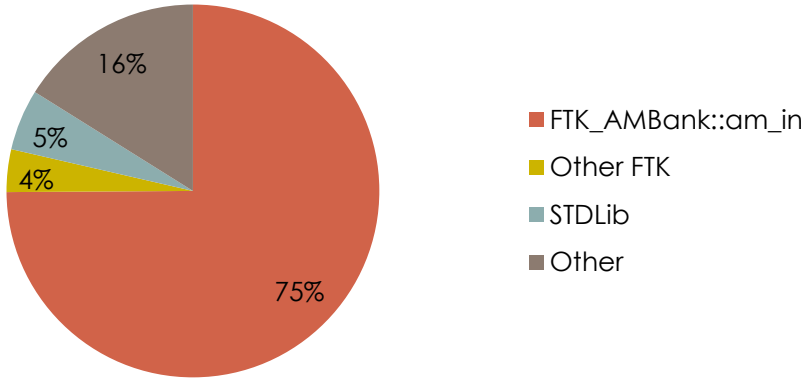
FTK Simulation consumers



Track fitting consumers



Pattern matching consumers



Possibilities with coprocessors

- Having a mixed hardware (CPU + ???) solution can improve the simulation performance
  - FTK algorithm are designed to be parallel
  - Porting the algorithms has technical difficulties but there aren't logical problems
- Commercial parallel processors exist and are currently on the spot: GPU and MIC more
  - Designed to make easy the porting of C/C++ code
    - Small proof of porting of FTK algorithms done
- Special cards with AM chips to be installed in a PC
  - SIP or other cards proposed in European funding requests

Future of the simulation

- FTK simulation will change from a prediction only to a **monitoring** (and prediction) tool
  - It will be possible to feed RAW data and compare the hardware and simulation output
    - The Vertical Slice experience was the first example
- Better matching between hardware and simulation configuration or algorithms required
  - Software implementation of HW techniques can be considered. Need to carefully check the performance
- Complexity of the simulation can limit how frequently of the monitoring
  - Any improvements in the simulation will be good also for the HW monitoring

Monitoring with FTK Simulation

- The VS slice experience shows that we need to improve our debugging tools
- The combination between the online simulation, by Federico, and the offline simulation allowed to spot issues
  - A problem in the SCT decoding was suggested
- The two codes need to be merged or better integrated
  - In the short term some algorithms may need to be duplicated in two versions: HW and logic version
  - The redundancy will allow to spot issues
- For the long term the simulation should be integrated in the Point-1 system running a low rate (~1 Hz) and check differences between real FTK and simulated result

Conclusions

- The FTK simulation it is a core component of the system
  - It allows to predict and tune the performance
  - It allows to study the benefits for the experiment in using a similar coprocessor
- In a working system it can help monitoring the system and be part level diagnostic
  - Can integrate some features of the online detailed simulation
- Simulating a highly parallel system with commercial computing units is challenging
  - The AM chip allows an enormous memory at very high bandwidth
    - CPU implementation pays huge penalties
    - Memory allocation has enormous limits