Contribution ID: **194** Type: **Parallel**

Study of a Fine Grained Threaded Framework Design

Monday, 21 May 2012 16:35 (25 minutes)

Traditionally, HEP experiments exploit the multiple cores in a CPU by having each core process one event. However, future PC designs are expected to use CPUs which double the number of processing cores at the same rate as the cost of memory falls by a factor of two. This effectively means the amount of memory per processing core will remain constant. This is a major challenge for LHC processing frameworks since the LHC is expected to deliver more complex events (e.g. greater pile-up events) in the coming years while the LHC experiment's frameworks are already memory constrained. Therefore in the not so distant future we may need to be able to efficiently use multiple cores to process one event. In this presentation we will discuss a design for an HEP processing framework which can allow very fine grained parallelization within one event as well as supporting processing multiple events simultaneously while minimizing the memory footprint of the job. The design is built around the libdispatch framework created by Apple Inc. (a port for Linux is available) whose central concept is the use of task queues. This design also accommodates the reality that not all code will be thread safe and therefore allows one to easily mark modules or sub parts of modules as being thread unsafe. In addition, the design efficiently handles the requirement that events in one run must all be processed before starting to process events from a different run. After explaining the design we will provide measurements from simulating different processing scenarios where the CPU times used for the simulation are drawn from CPU times measured from actual CMS event processing. Our results have shown this design to be very promising and will be further pursued by CMS.

Primary author: Dr JONES, Christopher (Fermi National Accelerator Lab. (US))

Presenter: Dr JONES, Christopher (Fermi National Accelerator Lab. (US))

Session Classification: Event Processing

Track Classification: Event Processing (track 2)