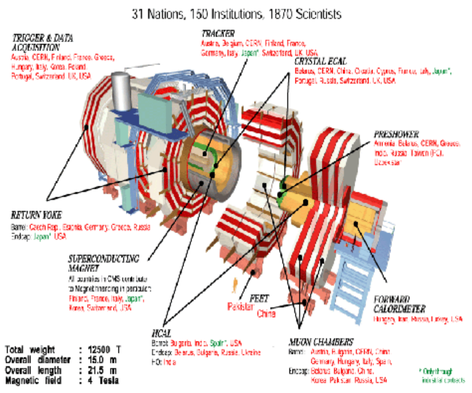
A Dynamic Workspace for CMS Simulations in Wisconsin

D. Bradley, P. Chumney, P. Couvares, S. Dasu, A. De Smet, R.G. Iyer, M. Livny, S. Rader, A. Roy, W.H. Smith
University of Wisconsin - Madison

The Compact Muon Solenoid



Monte-Carlo Event Simulation

Job Definition Scripts (Impala)
250 events per job, 10,000 jobs
Physics configuration from CMS database

→

Physics event generation (cmkin)
Job Runtime: 5 minutes
Output: 15 MB

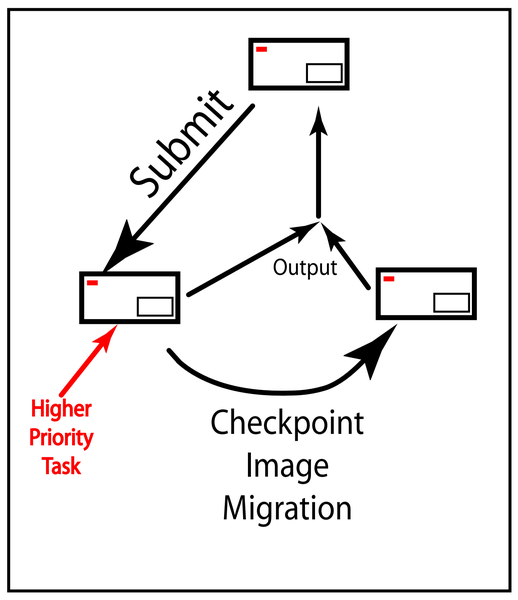
→

Detector Simulation (cmsim)
Job Runtime: 1-2 days
Output: 300 MB

Project Overview

- CMS physicists need millions of simulated events.
- We have access to unused CPU* cycles on commodity PCs in CS Department labs.
- However, jobs must vacate those machines when preempted by users or higher priority tasks.
- The typical time slot on a machine is less than the lifespan of a job, thereby requiring checkpointing of a job's state for later resumption, when resources become available.

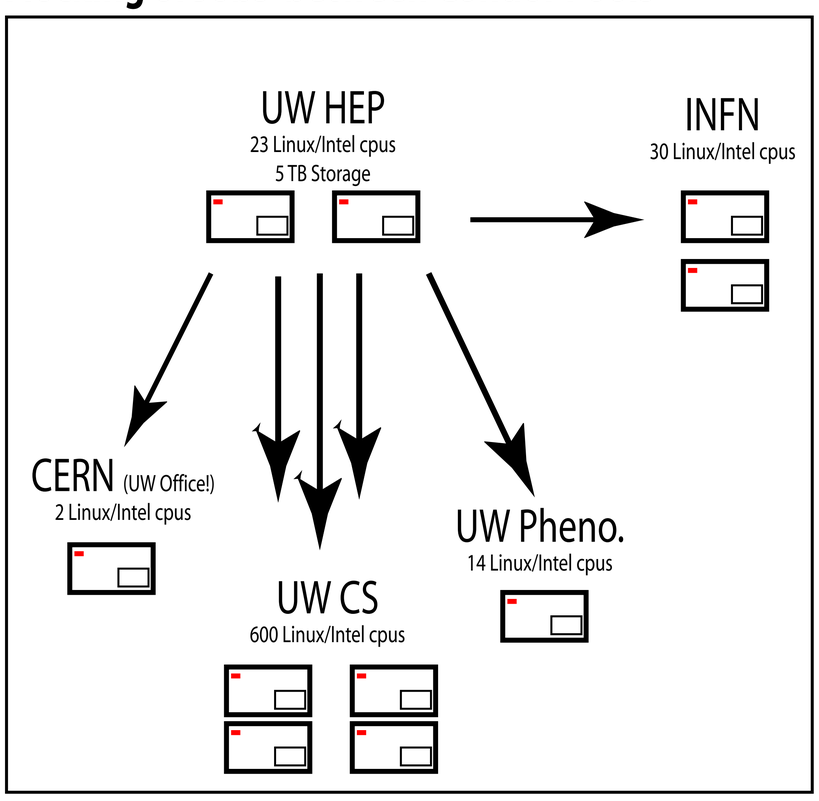
Resource Preemption

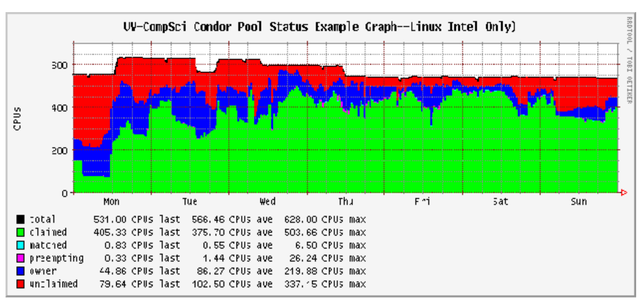


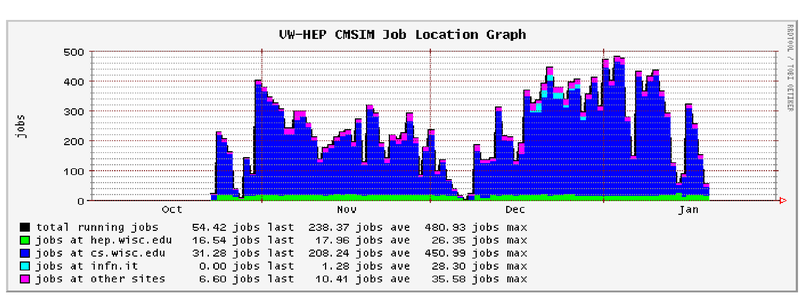
A Successful Solution

- Dynamic Workspace
  - Resources are allocated as and when available.
  - Resources are released whenever requested.
  - Normal completion of jobs irrespective of interruptions.
- Software is relinked with Condor "standard universe" libraries.
  - Transparent handling of remote system calls.
  - Fault tolerant streaming output.
  - Runtime image checkpointing.
  - Migration between machines.
- Condor "flocking" harnesses idle CPUs in collaborating pools.
  - Requirements of the job are met, but also
  - Priorities of resource owner are always enforced.

Flocking of Jobs between Condor Pools

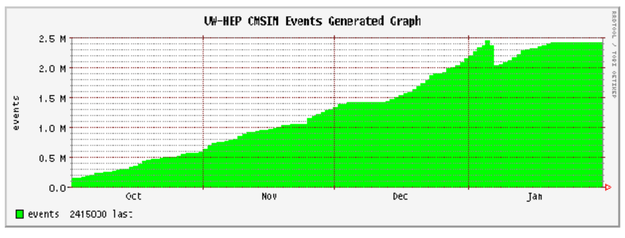


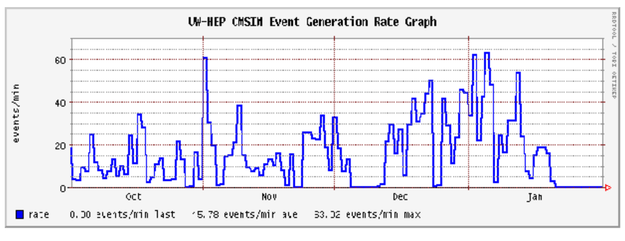


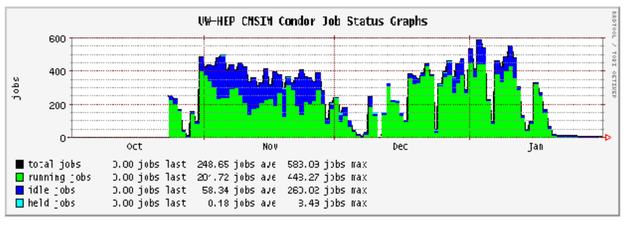


Results

- Over 2 million high multiplicity events simulated.
- Harnessed about 30 CPU-years of computing power.
- Each job was preempted 4 times, on the average.
- Checkpointing yielded additional benefit of robust handling of network outages.







Future Directions

- The next generation of CMS software is multi-threaded, which conflicts with current checkpointing libraries. (We will use restartable jobs with local file migration.)
- Remote system calls will have to be done explicitly, rather than "automagically". (Experimenting with an extension to ROOT/POOL that uses Condor's remote I/O protocol, Chirp.)
- Scalability and robustness was previously achieved by using multiple submission machines, but this process needs to be automated.
- Integration with Globus/Condor-G.

* One "CPU" approximately represents a 1 GHz Pentium III with 250 MB RAM.