Using Grid for Micro-Architecture Research

Liqiang He, Christiana Ioannou, Marios Kleanthous, Yiannakis Sazeides
Ξ – Computer Architecture Group
University of Cyprus/HiPEAC
cioannou@cs.ucy.ac.cy
www.cs.ucy.ac.cy/carch/xi

Ξ - Computer Architecture Research Lab

- Conduct computer architecture research aiming to develop technology to address current and future challenges of Computer Systems
- Experimental Methodology
  - We perform thousands of simulation experiments weekly to investigate the potential of new techniques we develop
- EGEE infrastructure
  - Provides a high throughput and powerful computing infrastructure that matches our simulation computing needs

We use Grid for…

- Superscalar Out-of-Order Processor Simulation
  - Research on memory hierarchy optimizations
  - Research on reliability
- Thermal Aware Multi-Core Scheduler
  - Thermal simulator is used to compute temperature every 1 μs
  - Temperature estimation is computationally intensive
- Detailed Multi-Core Simulations
  - Multiple threads running parallel to simulate a multi-core processor
  - Relatively long simulation time
  - Requires powerful machines (memory size, CPU type, multithreading support, etc)

Using Grid for Micro-Architecture Research

- Our simulations
  - Due to interacting parameters, every set of experiments may require several hundreds to thousands of simulations
  - Each individual run requires from several hours to few days
- Grid helps to perform a more quick and comprehensive design space exploration
  - Crucial for determining good solution points
- Grid functionality enables to select computational nodes that match our needs
  - Also specifying criteria based on each job category can increase throughput i.e.
    - Memory Size
    - CPU type

Our Experiences and Lessons with Grid

Using Storage Element (SE)

+ Allows to store input and output data on the SE
+ In case the user interface machine fails, it will not affect the jobs that are currently running on grid
- It requires to periodically delete unused files to avoid flooding SE

Using gLite

+ Multiple jobs can be submitted in less time than edg
- Usually only 80% of the jobs submitted were finished in less than 12 hours

* Redundant submission of the same job set can increase the percentage of finished jobs in 12 hours but it is an ad hoc solution

How to become more efficient

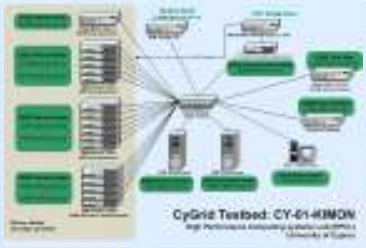
- Job Management
  - A good job manager is necessary for:
    - Job submission and Status checking
    - Resubmitting failed jobs and
    - Retrieving results
  - Currently we are developing scripts to manage our jobs
  - A more complete suit of such scripts/services can be very useful to increase efficiency
- Failures
  - A more detail report on the cause of failure will help to understand and deal with it
- Proxy
  - Initializing the proxy on every submission avoids any unexpected proxy expiration and job failures

Conclusions

- Grid has the potential to be a good match for our computational needs

Grid Wish List:

- Refinement of the job management scripts will improve the efficiency of grid usage
- 100% of the jobs submitted to finish on time



This research is supported by University of Cyprus and EGEE-II