Unifying Heterogeneous Resources
Moab Con 2009

Scott Jackson – Engineering
Overview

- Introduction
- Heterogeneous Resources w/in the Cluster
- Disparate Clusters -- Multiple Resource Managers
- Disparate Clusters -- Multiple Schedulers
- Q&A
Introduction
Introduction

- Fast Pace of Technological Innovation Induces Heterogeneous Environments
  - Technology Refreshes, Phased Procurements, New Procurements
- Productive During Transition Period
- Maintain High Utilization across Heterogeneous Resources
  - Avoid partitioning, fragmentation
Scheduling Jobs Across Heterogeneous Nodes
Heterogeneity

- **Hardware**
  - Architectures (Cray XT, SGI Altix, IBM Cell...)
  - Processor Type, Chipsets
  - Consumable resources (Processors, Memory, Disk)
  - Node Specialization (Licenses, Network, Features)

- **Software**
  - Operating System
  - Clustering Software
  - Resource Manager
  - Application Versions
  - Filesystems
Four Resource Selection Cases

1. **Nodes of Specified Type**  
   - [Specified Same]  
   - Give me nodes with 8 gigabytes of memory

2. **Nodes of Similar Type**  
   - [Unspecified Same]  
   - Give me all nodes with same amount of memory

3. **Nodes of Different Type**  
   - [Specified Different]  
   - Give me one node with 8 GB memory and 10 nodes with 2 GB memory

4. **Nodes of Any Type**  
   - [Unspecified Different]  
   - Give me whatever you can find
1. Nodes of Specified Type

A job may request nodes of a specified type
-- i.e. Quad core only, or only nodes with 8 GB memory

- Enabling Technologies
  - Node Features
  - Node Allocation Policies (e.g. MINRESOURCE)

- Example Syntax
  - qsub –l procs=8:quad hello.job
moab.cfg:

NODEALLOCATIONPOLICY MINRESOURCE
2. Nodes of Similar Type

A job may require the nodes to be of the same type, but it does not care which. For example, we may want the job to run entirely across quad core nodes or dual core nodes, but not across both simultaneously.

- Enabling Technologies
  - Node Sets

- Example Syntax
  - `qsub -l procs=8,nodeset=oneof:feature:dual:quad hello.job`
Default Node Set Policy

moab.cfg:

# By default, jobs will be allocated nodes of a single core size
NODESETPOLICY ONEOF
NODESETATTRIBUTE FEATURE
NODESETLIST DUAL,QUAD

# Try to keep jobs within similar resource types, but have the flexibility
# to run earlier if a preferred resource type is not available
NODESETISOPTIONAL TRUE
3. Nodes of Different Types

A job may specifically request disparate chunks of nodes of multiple varieties. For example, the user may want the job to run a single master task on one quad core node having 8 GB memory, and 20 slave tasks on 10 dual core nodes.

- Enabling Technologies
  - Multi-chunk jobs
  - Select Syntax

- Example Syntax
  - qsub –l select=1:mem=8gb:quad+20:dual hello.job
4. Nodes of Any Type

A job may not care if it allocated across heterogeneous node types. This gives the scheduler the greatest flexibility in maximizing utilization of the resources and avoiding fragmentation. The user’s job is likely to run sooner. For example, a job might request to run on 8 cores.

- **Enabling Technologies**
  - Moab heterogeneous node scheduling

- **Example Syntax**
  - `qsub -l procs=8 hello.job`
Scheduling Jobs Across Disparate Systems

Ahh, but what if the new technology comes in the form of a separate system? Can you schedule jobs between different clusters?

- Of Course! There are two ways to do this:
  - Single Moab with Multiple Resource Managers
  - Multiple Moabs in Grid relationship

- Motivation
  - Single point of submission
  - Load balancing
  - Unified Job Accounting/Policies
Multiple Resource Managers

- Independent Login Node
  - Moab CLI

- Independent Head Node
  - Moab Server
  - Torque 1 CLI
  - Torque 2 CLI

- Cluster1 Head Node
  - Torque Server 1
  - Moab CLI

- Cluster2 Head Node
  - Torque Server 2
  - Moab CLI

- Cluster1 Compute Nodes

- Cluster2 Compute Nodes
Multiple Resource Managers

- **Pros**
  - Single Point of Submission
  - Simple Load Balancing
  - Unified Statistics, Job Accounting, Logs

- **Cons**
  - Difficult to enforce different priority schemes, throttling policies for the separate systems
  - Single Point of Failure
  - Have to resolve nodeid collisions
Configuration Files

**moab.cfg:** 
# Example For Cray XT4

RMCFG[cluster1] TYPE=NATIVE:XT4 SERVER=cluster1-pbs SUBMITCMD=/opt/torque-cluster1/bin/qsub

RMCFG[cluster2] TYPE=NATIVE:XT4 SERVER=cluster2-pbs SUBMITCMD=/opt/torque-cluster2/bin/qsub

JOBMIGRATEPOLICY IMMEDIATE

**config.xt4.pl:** 
# Cray XT4 only

$alpsUser = "root";

%alpsHost = ( cluster1 => "cluster1-login", cluster2 => "cluster2-login" );

%torquePath = ( cluster1 => "/opt/torque-cluster1/bin", cluster2 => "/opt/torque-cluster2/bin" );

**torque.cfg:**

QSUBSENDDUID true  # On all submit nodes

VALIDATEPATH false  # on independent moab node
Multiple Schedulers

Grid Login Node
  Moab CLI

Grid Head Node
  Moab Server
  Torque 1 CLI
  Torque 2 CLI

Cluster1 Login Node
  Torque1 CLI
  Moab1 CLI

Cluster1 Head Node
  Moab Server 1
  Torque Server 1

Cluster1 Compute Nodes

Cluster2 Login Node
  Torque2 CLI
  Moab2 CLI

Cluster2 Head Node
  Moab Server 2
  Torque Server 2

Cluster2 Compute Nodes
Multiple Schedulers (Grid)

- **Pros**
  - Single Point of Submission
  - Load Balancing across systems
  - Sovereign policies and control
  - Handles credential mapping, nodeid mapping

- **Cons**
  - Grid license
  - Extra layer of translation/conveyance
  - Interactive jobs
Configuration Files

moab.cfg: # Grid Head (also moab-private.cfg)
SCHEDCFG[gridhead] SERVER=grid-head
RMCFG[cluster1] TYPE=MOAB SERVER=cluster1-head SUBMITCMD=/opt/torque-cluster1/bin/qsub OMAP=file:///var/moab-grid/cluster1.omap.dat
RMCFG[cluster2] TYPE=MOAB SERVER=cluster2-head SUBMITCMD=/opt/torque-cluster2/bin/qsub OMAP=file:///var/moab-grid/cluster2.omap.dat
JOBMIGRATEPOLICY IMMEDIATE

moab.cfg: # Cluster Heads (also moab-private.cfg)
SCHEDCFG[cluster1] SERVER=cluster1-head
RMCFG[cluster1] ...
RMCFG[gridhead.INBOUND] FLAGS=LOCALWORKLOADEXPORT,CLIENT
JOBMIGRATEPOLICY IMMEDIATE

torque.cfg:
VALIDATEPATH false # on all cluster torque server nodes
Managing Leadership Systems w/ Moab

ORNL

Jaguarpf: Cray conjoined XT4/XT5
~181,000 cores
1.64 Petaflops

World’s Most Powerful Computer.
For Science!

*The Jaguar system at ORNL provides immense computing power in a balanced, stable system that is allowing scientists and engineers to tackle some of the world’s most challenging problems.*

—2008, Kelvin Droegemeier, Meteorology Professor, University of Oklahoma.
Managing Leadership Systems w/ Moab

Sandia – Red Storm

Red Storm: Cray XT3
12,960 nodes
38,400 cores

- 284 teraOPS theoretical peak performance
- 135 racks
- AMD Opteron™
- 78 terabytes of memory
- 1.7 petabytes of disk storage
- Linux/Catamount OS
- 2.5 megawatts power & cooling

Design: Sandia
Managing Leadership Systems w/ Moab

Other Leading Government Site

Cray XT4
Over 38,000 cores
352 TFlops/sec
• AMD Opteron™
Market Usage

• **Billions of Dollars** worth of Hardware run Moab
• **Worlds Largest computer** runs Moab (1 Petaflop – over 100,000 processor cores used)
• **Future Largest Systems** (w/ planned Moab use):
  • Another 1 Petaflop System
  • 2 Petaflop System
  • 5 Petaflop System
  • 25 Petaflop System
• ~25% of the resources of the **Top 100** systems in the world use Moab (Using Top500.org - 2008)
• **98+% Customer Retention** (By Revenue)
Conclusion
Conclusion

- Moab and Torque can be used on HPC systems to:
  - Improve utilization
  - Enforce site policies

- Moab’s Intelligent Integration with ALPS and CPA Allow:
  - Support for heterogeneous resources
  - Unification of disparate XT systems into a grid resource

This means better utilization and easier transitions during the life cycle of the system as you update, enhance and expand your HPC systems.
For more information

Contact: Scott Jackson
Cluster Resources, Inc.
scottmo@adaptivecomputing.com
(801) 717-3708
http://www.adaptivecomputing.com
Appendix
Moab Grid Suite™

What it is:
A workload management solution that provides simple web-based job submission and controls, graphical grid administration and management reporting tools for a group of high performance computing environments unified into a grid.

What it does:
- Enables rapid unification of multiple clusters into a managed grid environment
- Intelligently applies policies which enforce guidelines provided by owners of the resources
- Optimizes resource usage for timing, best fit resource usage and location
- Tracks usage for billing purposes

Why you should care:
- Improves utilization of resources by 10 to 30% and provides access to unique resources
- Enables collaboration between teams without the complexity of interacting manually with multiple systems and overcoming the politics of sharing
- Aids organizations to share costs of infrastructure investment and to properly apply the investment to projects and needs in a timely and controlled basis