A Framework for Scheduling Scientific Computing Tasks on Heterogeneous Clouds

Brian Peterson, Gerald Baumgartner, Qingyang Wang
Division of Computer Science and Engineering
School of Electrical Engineering and Computer Science
Louisiana State University

Motivation

- Supercomputer too expensive for many scientific applications
- Rented dedicated clusters are also expensive
- Rented VMs are cheap, but only work well for map-reduce
  - Two VMs might share same physical node
  - High communication latency

Goal

- Use cheap rented nodes
- Measure their performance characteristics
- Place tasks on nodes based on performance requirements

Proposed Cloud Framework

- Cloud or physical nodes measure their own resource availability
- Resource availability is advertised along an overlay network

- Applications are structured as task graphs
- Application selects appropriate set of nodes for each task
- Nodes have the option to reject tasks based on application priorities
- Idle nodes steal work from overloaded nodes
- If resource availability changes, tasks can migrate to other nodes

Deployment Scenarios

- In-house cloud for load balancing purposes
  - Alternative to Condor Pool with task migration for load balancing
- PAAS cloud with API that allows apps to query resource availability
  - Allows tasks other than map-reduce on cheap virtual resources
- User-defined infrastructure for very compute-intensive applications
  - Runs on top desktop nodes, supercomputers, commercial clouds

Planned Implementation

- Use Java for easier deployment and experimentation
- Mobile agents carry their computational payload to idle nodes
- Strong mobility allows task migration
- Use quantum chemistry tensor computations as demo application

Experiments

- Test performance measurement tools
- Test distributed algorithms for propagating performance information
- Test task placement and migration strategies
- Compare centralized & decentralized placement/migration algorithms
- Measure scientific application performance in different environments

Experimental Needs

- Variety of different cloud resources
- Both dedicated nodes and nodes with other workload
- Access to storage for storing intermediate results of application
- Linux and Java
- Ability to install other software (ProActive, measurement tools, NWChem)

Related Work

- Organic Grid
- IBM’s Air Traffic Control
- BOINC-based desktop grids

Summary

- Framework for matching application needs with available resources
- Resource measurements and propagation of resource information
- Decentralized task placement with work stealing
- Modest experimental needs