Resource Management and Selection

Christian Almazan

Sp'04 – cmsc818s – Grid Computing

Coverage

- Grid 2 – Chapter 18
  Resource and Service Management
  - Slides Based Mainly From Scott McMaster
- Design and Evaluation of a Resource Selection Framework for Grid Applications

Grid 2: Chapter 18 Resource and Service Management

Karl Czajkowski, Ian Foster,

Carl Kesselman

Defining Resource Management

- Resource
  - any capability that can be shared or exploited in a networked environment, specifically through a service
  - computing cycles, instruments, data, simulations, etc.
- Management
  - how resources are exposed and made available for use on Grid
- Resource Management
  - operations to control how resources are made available to others

Resource Management Requirements (I)

- Addresses the following aspects for Grid service agreements:
  - what, when, and how
- Task Submission
  - resource commits to a task
- Workload Management
  - guaranteed levels of capability
  - provisioning: reserve capability for use by a specific entity

Resource Management Requirements (II)

- On-Demand Access
  - resource capability made available at a specific time and for a specified duration (or “advance reservation”)
- Coscheduling
  - multiple resources available simultaneously
- Resource Brokering Scenarios
  - third party connects resource consumers and providers

Service Level Agreements (SLAs)

- “Contracts” between resource provider and client
- Addresses issues: resource capability, availability
- Tells consumers what to expect from negotiated resources without knowledge of resource policies
- Abstracts the actual resource
  - local usage policy, configuration, etc.
- Three Different Kinds
  - TSLA, RSLA, BSLA

SLA Types

- TSLA (Task Service Level Agreement)
  - agreement to perform a specific task
- RSLA (Resource Service Level Agreement)
  - agreement for the use of a resource
  - independent of the task
- BSLA (Binding Service Level Agreement)
  - ties a specific resource to a specific task

SLAs in Resource Management

- Resource management is implemented by a set of SLAs of various types
  - a single task may be governed by several SLAs addressing the various phases of resource management
    - planning
    - submission
    - acquisition
    - binding

Policy and Security Enforcement

- A given resource has rules about who can use it, how it can be used, and when it can be used.
- Must have facilities in place for authentication, authorization, and enforcing usage restrictions.

Resource Descriptions

- Help clients in finding resources and determining whether or not a given resources meets needs
- Resource Description Language Requirements
  - property names/value (to express things like bandwidth, space)
  - composition operators (to express need for multiple properties)
  - temporal rules for both of the above
  - be dynamically extensible

Resource Description Languages

- Two Major Languages
  - Globus Toolkit Resource Specification Language (RSL)
    - based on LDAP syntax (think LISP)
    - Schema-based (attribute meanings are well-defined)
  - ClassAds (from Condor) [will talk more about this later]
    - Semistructured (attribute meanings determined by convention)
- Primary Difference
  - ClassAds integrates task and resource characteristics while RSL treats them separately

Resource Discovery and Selection

- Resource Discovery
  - find a resource on the Grid which has characteristics and state that the resource consumer wants
- Resource Selection
  - selecting a resource (hopefully optimal) from a set of candidates returned by a resource discovery service

Task Management

- Need to be able to monitor long-running activities on resources obtained with SLAs
- May lead to various actions on SLAs
  - termination
  - extension
  - renegotiation
  - creation

Existing Systems

- Globus Grid Resource Allocation Manager (GRAM)
  - basic low-level resource management APIs
  - works with existing local resource management software
  - focused on computation resources
- General-Purpose Architecture for Reservation and Allocation (GARA)
  - more advanced, beyond just computational resources
  - supports advanced reservations
- Condor
  - provides TSLA management

Resource Brokers

- Also Known As “Metaschedulers”
- Middleman between resources and their users
- Provides a single location to submit tasks to
- Other Advantages
  - virtualization (simplified view of resources)
  - policy enforcement (various strategies possible for deciding where to route jobs to)
  - protocol conversation (helpful in exposing legacy applications)

Service Negotiation and Acquisition Protocol (SNAP)

- Supplies protocols for SLA-based resource management
- OGSA-based
- Underlies the next generation of GRAM
- Supports several emerging directions
  - service-oriented
  - management of all types of resources (not just hardware)
  - provisioning

Design and Evaluation of a Resource Selection Framework for Grid Applications

Chuang Liu, Lingyun Yang,

Ian Foster, Dave Angulo

Motivations for a Dynamic Resource Selection Framework

- What we have today:
  - high-speed networks (10Gb/s Ethernet, optical networking…)
  - distributed computation and storage resources
  - communication-intensive applications
- What we do not have:
  - dynamic discovery and configuration of required resources for specific applications for heterogeneous environments
  - there are options for homogeneous environments

Resource Selection Framework: Overview

- selection
  - select Grid resources appropriate for a particular application run
- configuration
  - organize resources
- mapping
  - appropriate application workload placement

Related Projects (I)

- NQE, PBS, LSF, I-SOFT, Load Leveler
  - user-submitted jobs find necessary resources identified by user
  - dynamic resource discovery not possible
- Globus, Legion
  - resource management architectures
    - resource discovery, dynamic resource status monitoring, resource allocation, and job control
  - simple, generic default scheduler in Legion
    - however, application knowledge aids scheduling performance

Related Projects (II)

- AppLeS framework
  - guides implementation of application-specific scheduler logic
- ScaLAPAK
  - more modular resource selector
  - however, application-specific details embedded in resource selection module and cannot be easily used by other apps
- MARS, SEA, DOME
  - for certain classes of applications

Condor

- Condor: general resource selection mechanism
- ClassAd language
  - Condor's resource description language
  - for resource requests and owners to describe resources
- Matchmaker
  - matches user requests with appropriate resources
  - when multiple resources, select best one with ranking

ClassAds and Matchmaking

- ClassAd (Classified Advertisement)
  - maps attribute names to expressions
    - can be constants or a function of other attributes
  - can evaluate an expression between two ClassAds (protocol)
    - other.size > 3 (check if has attribute “size” and greater than 3)
- Matchmaking
  - evaluates two ClassAds with respect to the other
    - match if each ClassAd has attribute “requirements” that evaluates to true in the context of the other
    - can also have attribute “rank” to quantify the quality of match

Extensions for Handling Multiple Resources

- Condor ClassAds and Matchmaking
  - designed for selecting a single machine, not for finding multiple resources for a single job
- Set-Extended ClassAds and Matchmaking
  - can specify aggregate resource properties
  - everything described through declarative statements

Set-Extended ClassAds

- Successful match definition
  - between a single set request and a ClassAd set
- Request is set-extended ClassAd
  - set expressions: collective properties of entire ClassAd set
  - individual expressions: properties for each entire ClassAd in set

Set-Extended ClassAds Syntax

- Type: identifies set-extended ClassAds
- Aggregation Functions: Max, Min, Sum
- Suffix(V,L): true if a member of L is suffix of V
  - Suffix(other.hostname, {“ucsd.edu”, “utk.edu”})
  - true if other.hostname = “torc1.cs.utk.edu”

Set-Matching Algorithm

- Two phases for evaluating set-extended ClassAds
  - filtering: remove ClassAds based on individual expressions
    - other.os == redhat6.1 && other.memory >= 100M
  - set construction: find best possible ClassAd for app req.
    - keep track of the “best” set (initially null)
    - from the pool, repeatedly remove “best” resource remaining
    - put that resource in the “candidate” set
    - if “candidate” set has higher rank than “best” set, it becomes “best”
- Set Construction: O(n^2)
  - n is the number of ClassAds after filtering

General-Purpose Resource Selection Framework

- Based on the set-matching technique.
  - accepts user resource requests
  - finds set of resources with highest rank based on resource information from a Grid information service
  - open interface allows users to customize resource selected by specifying an application-specific mapping module

System Architecture

- Grid Information Service
  - Monitoring and Discovery Service (MDS-2) of Globus Toolkit
  - uniform framework for discovery and accessing system configuration and status information
- Network Weather Service (MWS)
  - distributed system that periodically monitors and dynamically forecasts performance resources
- Grid Index Information Service (GIIS) and Grid Resource Information Service (GRIS)
  - resource availability and configuration information

RSS: Resource Selector Service

- Resource Monitor
  - acts as GRIS, queries MDS when necessary to update information, such as updating old values
- Set-Matcher
  - uses the set-matching algorithm
  - sometimes necessary to map resources before judging them because of tight application requirements
- Mapper
  - decides resource topology and allocation of application workload to resources
  - hard to find an efficient, general mapping algorithm for all applications

Resource Request

- Type of Service
  - synchronous or asynchronous
- Job Description
  - characteristic of the job to be run (i.e. performance model)
- Mapping
  - mapper program to use
- Constraint
  - user resource requirements (i.e. memory)
- Rank
  - criteria for ranking

Resource Selection Result

- Resource Selector Returns XML
  - indicates status, selected resources, and a mapping scheme

```
<virtualMachine>
    <result statusCode="200" statusMessage="OK" />
    <machineList>
        <machine dns="torcs2.cs.utk.edu" processor="2" x="20" />
        <machine dns="torcs3.cs.utk.edu" processor="2" x="15" />
        <machine dns="torcs6.cs.utk.edu" processor="2" x="15" />
    </machineList>
</virtualMachine>
```

Cactus Application

- Cactus Application
  - simulates 3D scalar field produced by two orbiting solutions
- Performance Model
  - describes required memory and execution time
- Mapping Algorithm
  - pick machine with highest CPU speed at first part of line
  - find machine with highest communication speed with last machine and put that at the end
  - continue second step until all machines are on the line

Experiments and Validations

- Conducted on the GrADS test bed
  - Univ. of Chicago, UIUC, UTK, UCSD, Rice University, USC/ISI
- Tests
  - execution time prediction function
  - Cactus mapping strategy
  - set-matching algorithm

Execution Time Prediction Test

- Computation Time Prediction Test
  - predict running time of Cactus on a single machine
  - their test showed they picked the machine with the most power
- Computation and Communication Time
  - predict execution time for Cactus that may involve one or more machines
  - prediction generally close, with error on average at 13.13%
    - possibly because the CPU load information used for prediction does not accurately reflect the real CPU load

Mapping Strategy Test

- Attempting to find a close to optimal mapping:
  - execution for different workload allocations on two machines
  - mapper was very close to optimal (1.2% higher)

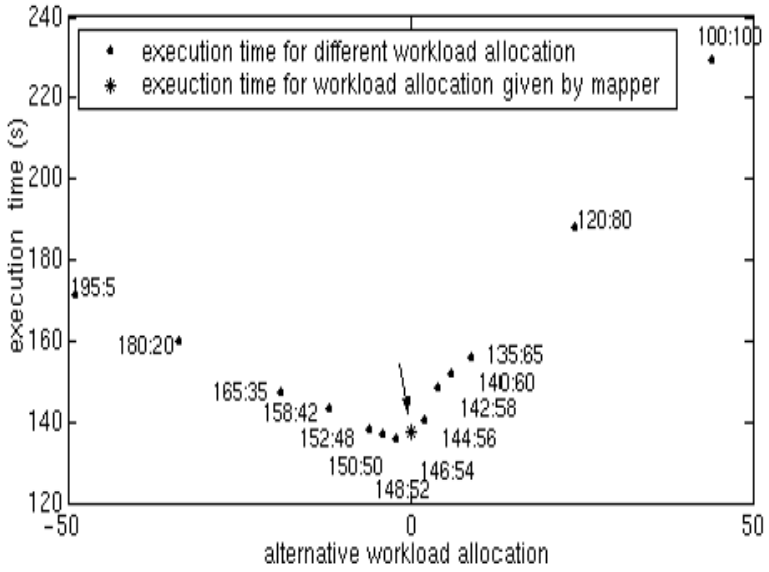

Figure 4. Execution times for different workload allocations on two machines.

Resource Selection Algorithm Test

- Asked resource selector to select a set of machines for Cactus from three candidates.
- Single Cluster Experiment
  - because communication has low penalty, selected all nodes
- Two-Cluster Experiment
  - because communication has high penalty, selected fastest node