Pricing for Utility-driven Resource Management and Allocation in Clusters

Chee Shin Yeo and Rajkumar Buyya

Grid Computing and Distributed Systems (GRIDS) Lab.
Dept. of Computer Science and Software Engineering
The University of Melbourne, Australia

www.gridbus.org/
Presentation Outline

- Motivation
- Computation Economy
- Economy-based Admission Control, Resource Allocation & Job Control
- Pricing Function
- Performance Evaluation
- Conclusion and Future Work
Motivation

- Cluster-based systems have gained popularity and widely adopted
  - 75% of Top500 supercomputers world-wide based on Cluster architecture.
  - Clusters are used in not only used in scientific computing, but also in driving many commercial applications.
  - Many Corporate Data Centers are cluster-based systems.
Problem and our Proposal

- However, RMS responsible for managing clusters and allocating resources to users
  - Still adopts system-centric approaches such as FCFS with some static pariorities.
    - Maximize CPU throughput & CPU utilization
    - Minimize average waiting time & average response time
  - They provide no or minimal means for users to define Quality-Of-Service (QoS) requirements.
- We propose the use of user-centric approaches such as computational economy in management of cluster resources.
Computational Economy

- Management of shared resources with economic accountability is effective:
  - Regulates supply and demand of cluster resources at market equilibrium
  - User-centric management of clusters
    - Users express Quality Of Service (QoS) requirements
    - Users express their valuation for the required service
  - Economic incentives for both users and cluster owner as a means of feedback
Utility-driven Cluster RMS Architecture

User Population

Manager Node

Resource Manager

Job Manager

Node Status Monitor

Pricing

Economy-based Admission Control

Economy-based Resource Allocation

Utility-driven Job Scheduler

Computation Nodes

Computation Node 1

Computation Node n
Economy-based Admission Control & Resource Allocation

- Uses the pricing function to compute cost for satisfying the QoS of a job as a means for admission control
  - Regulate submission of workload into the cluster to prevent overloading
  - Provide incentives
    - Deadline -- $\uparrow$
    - Execution Time -- $\downarrow$
    - Cluster Workload -- $\downarrow$
- Cost acts as a mean of feedback for user to respond to
Economy-based Admission Control & Resource Allocation

- Accept or reject based on 3 criteria (consider required QoS)
  - resource requirements that are needed by the job to be executed
  - deadline that the job has to be finished
  - budget to be paid by the user for the job to be finished within the deadline
- Requires estimated execution time
- Allocates job to node with least remaining free processor time
Job Control: Economy-based Proportional Resource Sharing

- Monitor and enforce required deadline.
  - Time-shared
  - Allocate resources proportional to the needs of jobs based on the estimated execution time and required deadline
  - Update processor time partition periodically
Essential Requirements for Pricing

- Flexible
  - Easy configuration
- Fair
  - Based on actual usage
- Dynamic
  - Not static
- Adaptive
  - Changing supply and demand of resources
Pricing Function

\[ P_{ij} = (\alpha \times PBase_j) + (\beta \times PUtil_{ij}) \quad (3) \]

\( \alpha \) - factor for static component based on the base pricing rate \( PBase_j \) for utilizing the resource on computation node \( j \)

\( \beta \) - factor for dynamic component based on the utilization pricing rate \( PUtil_{ij} \) of that resource that takes into account job \( i \).

\( PBase_j \) - fixed base pricing rate per unit of cluster resource specified by cluster owner.

\[ PUtil_{ij} = \frac{RESMax_j}{RESFree_{ij}} \times PBase_j \quad (4) \]

\( RESMax_j \) is the maximum units of the resource on computation node \( j \) from time \( AT_i \) to \( DT_i \).
Pricing Function

\[ RESFree_{ij} = RESMax_{j} - \left( \sum_{k=1}^{n_{accept}_{j}} RES_k \right) - RES_i \]  

(5)

\[ n_{accept}_{j} - n_{accept} \] jobs (accepted by admission control) that are executing on computation node \( j \) from time \( AT_i \) to \( DT_i \).

\[ RESFree_{ij} = RESMax_{j} - \left( \sum_{k=1}^{n_{accept}_{j}} EE_k \right) - EE_i \]  

(6)
Processing Cost Functions for Different Scheduling Algorithms

- **First-Come-First-Served (FCFS)**

  \[ C_i = EE_i \times PBase_j \]

- **Economy based Proportional Resource Sharing (Libra)**

  \[ C_i = \gamma \times EE_i + \delta \times EE_i / D_i \]

- **Libra with dynamic pricing (Libra+$$)**

  \[ P_{ij} = (\alpha \times PBase_j) + (\beta \times PUtil_{ij}) \]

  \[ C_i = EE_i \times P_{ij} \]
Performance Evaluation: Simulation

- **Simulation Model**
  - Simulated scheduling for a cluster computing environment using the GridSim toolkit (http://www.gridbus.org/gridsim)

- **Simulated Cluster**
  - manjra.cs.mu.oz.au (13 single-processor nodes with Pentium4 2-GHz CPU)
Experimental Methodology

Models a high demand for cluster resources where the majority of jobs have short deadlines:

- 200 jobs with exponentially distributed job inter-arrival time of mean 0.5 hours and exponentially distributed job execution time \( E_i \) of mean 10 hours

- 80% of the 200 jobs belongs to a high urgency job class with a low \( D_i/E_i = 1.5 \) and a high \( B_i/f(E_i) = 6 \), where \( f(E_i) \) is a function to compute the minimum budget required for job execution time \( E_i \)

- 20% of the 200 jobs belongs to a low urgency job class with a high \( D_i/E_i = 6 \) and a low \( B_i/f(E_i) = 1.5 \)

- \( D_i \) and \( B_i \) are normally distributed within each high/low \( D_i/E_i \) and \( B_i/f(E_i) \)

- The high urgency and low urgency job classes are randomly distributed in arrival sequence

- For Libra+$, static pricing factor \( \alpha = 1 \) and dynamic pricing factor \( \beta = 0.01 \)
Evaluation Metrics

- Job QoS Satisfaction
- Cluster Profitability
- Average Waiting Time
- Average Response Time
Normalised Comparison of FCFS, Libra & Libra+$

![Comparison bar chart showing Job QoS Satisfaction, Cluster Profitability, Average Waiting Time, and Average Response Time for FCFS, Libra, and Libra+$ with β = 0.01.](chart.png)
Varying Cluster Workload

- Scheduling policies
  - First-Come-First-Served (FCFS)
  - Economy based Proportional Resource Sharing (Libra)
  - Libra with dynamic pricing (Libra+$)
- An increasing mean job execution time
  - 6, 7, 8, 10, 15 and 30 hours
Impact of Increasing Job Execution Time on Job QoS Satisfaction

![Graph showing the impact of increasing job execution time on job QoS satisfaction. The x-axis represents the mean job execution time (hours), ranging from 6 to 30 hours. The y-axis represents the job QoS satisfaction percentage, ranging from 0% to 90%. Three lines are shown: FCFS, Libra, and Libra+$, $\beta = 0.01$. The FCFS line shows the highest job QoS satisfaction, followed by Libra, and then Libra+$, $\beta = 0.01$. As the mean job execution time increases, the job QoS satisfaction decreases for all three methods.]
Impact of Increasing Job Execution Time on Cluster Profitability

![Graph showing the impact of increasing job execution time on cluster profitability. The graph compares FCFS, Libra, and Libra+$. β = 0.01. The x-axis represents Mean Job Execution Time (hours), and the y-axis represents Cluster Profitability (%). The graph demonstrates that Libra+ with $= 0.01$ maintains a higher profit margin compared to FCFS and Libra as the execution time increases.]
Varying Pricing Factor for Different Level of Sharing

- Scheduling policies
  - Libra with dynamic pricing (Libra+)$
- An increasing dynamic pricing factor $\beta$
  - 0.01, 0.1, 0.3, and 1
Impact of Increasing Dynamic Pricing Factor on Job QoS Satisfaction

![Graph showing the impact of Dynamic Pricing Factor on Job QoS Satisfaction]
Impact of Increasing Dynamic Pricing Factor on Cluster Profitability

Cluster Profitability (%) vs. Dynamic Pricing Factor $\beta$

- FCFS
- Libra
- Libra+$
Tolerance against Estimation Error

- **Under-estimated execution time** $EE_i$
  - e.g. job whose execution time $E_i = 60$ hours has $EE_i = 30$ hours for estimation error = 50%

- **Scheduling policies**
  - Libra – Economy based Proportional Resource Sharing (Libra)
  - Libra with dynamic pricing (Libra+$)

- **An increasing estimation error for estimated execution time** $EE_i$
  - 0%, 10%, 30% and 50%
Impact of Increasing Estimation Error on Job QoS Satisfaction

![Chart showing the impact of increasing estimation error on job QoS satisfaction.](chart.png)
Impact of Increasing Estimation Error on Cluster Profitability

![Graph showing the impact of increasing estimation error on cluster profitability. The graph compares different versions of Libra and Libra+ with varying parameters. The x-axis represents the estimation error for estimated execution time (EEi), while the y-axis shows cluster profitability. The legend includes different markers and colors for each version: orange for Libra, blue for Libra+, and various symbols and colors for versions Libra+ with parameters β = 0.01, 0.1, 0.3, and 1.0.](image-url)
Conclusion & Future Work

- Importance of effective pricing function (demand exceeds supply of resources)
- Satisfy four essential requirements for pricing
- Serves as means of admission control
- Tolerance against estimation errors
- Higher benefits for cluster owner
- Future work
  - Explore different pricing strategies
  - Examine different application models
Backup
Related Work

- **Traditional cluster RMS**
  - Load Sharing Facility (LSF) – Platform
  - Load Leveler – IBM
  - Condor – University of Wisconsin
  - Portable Batch System (PBS) – Altair Grid Technologies
  - Sun Grid Engine (SGE) – Sun Microsystems

- **Market-based cluster RMS**
  - REXEC
  - Libra
User-level
Job Submission Specification

\[ job_i([Segment_1][Segment_2]...[Segment_s]) \] (1)

\[ job_i([JobDetails][ResourceRequirements][QoSConstraints][QoSOptimization]) \] (2)

- **Job details**
  - eg. Estimated execution time

- **Resource requirements**
  - eg. Memory size, Disk storage size

- **QoS constraints**
  - eg. Deadline, Budget

- **QoS optimization**
  - eg. Time, Cost
Performance Evaluation Metrics

- **Job QoS Satisfaction**

*Job QoS Satisfaction* measures the level of utility for satisfying job requests. A higher Job QoS Satisfaction represents better performance. Computed as the proportion of $n_{QoS}$ jobs whose required QoS (deadline and budget) are fulfilled out of $n$ jobs submitted:

\[
\text{Job QoS Satisfaction} = \frac{n_{QoS}}{n}
\]  

(7)

$n_{QoS}$ is $n_{\text{accept}}$ jobs (accepted by the admission control) where $FT_i \leq DT_i$ and $C_i \leq B_i$. 
Performance Evaluation Metrics

- **Cluster Profitability**

*Cluster Profitability* measures the level of utility for generating economic benefits for the cluster owner. A higher Cluster Profitability denotes better performance. Computed as the proportion of profit earned by the cluster out of the total budget of jobs that are accepted for execution:

\[
\text{Cluster Profitability} = \frac{\sum_{i=1}^{n_{\text{accept}}} C_i}{\sum_{i=1}^{n_{\text{accept}}} B_i}
\]

(8)
Performance Evaluation Metrics

- **Average Waiting Time**

*Average Waiting Time* is the average time a job waits in the cluster before it starts execution. A lower Average Waiting Time indicates better performance.

\[
\text{Average Waiting Time} = \frac{1}{n_{\text{accept}}} \sum_{i=1}^{n_{\text{accept}}} ST_i - AT_i
\]  

(9)
Performance Evaluation Metrics

- **Average Response Time**

  Average Response Time is the average time a job is completed and results returned to the user. A lower Average Response Time signifies better performance.

  \[
  \text{Average Response Time} = \frac{1}{n_{\text{accept}}} \sum_{i=1}^{n_{\text{accept}}} FT_i - AT_i
  \]  

  (10)