Libra: An Economy driven Job Scheduling System for Clusters

Jahanzeb Sherwani\textsuperscript{1}, Nosheen Ali\textsuperscript{1}, Nausheen Lotia\textsuperscript{1}, Zahra Hayat\textsuperscript{1}, \textbf{Rajkumar Buyya}\textsuperscript{2}

1. Lahore University of Science and Management (LUMS), Lahore, Pakistan

2. Grid Computing and Distributed Systems (GRIDS) Lab., University of Melbourne, Australia

www.gridbus.org
Agenda

- Introduction/Motivations
- The Libra Scheduler Architecture & Cost-based Scheduling Strategy
- Implementation
- Performance Evaluation
- Conclusion and Future Work
Introduction

- Clusters (of “commodity” computers) have emerged as mainstream parallel and distributed platforms for high performance, high-throughput and high-availability computing.

- They have been used in solving numerous problems in science, engineering, and commerce.
Adoption of the Approach
Cluster Resource Management System: Managing the Shared Facility

Parallel Applications

Sequential Applications

Parallel Applications

Parallel Programming Environment

Cluster Management System
(Single System Image and Availability Infrastructure)

PC/Workstation

Communications Software

Network Interface Hardware

PC/Workstation

Communications Software

Network Interface Hardware

PC/Workstation

Communications Software

Network Interface Hardware

PC/Workstation

Communications Software

Network Interface Hardware

Cluster Interconnection Network/Switch
Some Cluster Management Systems

- Commercial and Open-source Cluster Management Software
- Open-source Cluster Management Software
  - DQS (Distributed Queuing System)
  - Condor
  - GNQS (Generalized Network Queuing System)
  - MOSIX
  - Load Leveler
  - SGE (Sun Grid Engine)
  - PBS (Portable Batch System)
Cluster Management Systems Still Use System Centric Approach

- Traditional CMSs focus has essentially been on maximizing CPU performance, but not on improving the value of utility delivered to the user and quality of services.

- Traditional system-centric performance metrics
  - CPU Throughput
  - Mean Response Time
  - Shortest Job First
  - FCFS
  - Some Static Priorities
  - ...
The Libra Approach: Computational Economy Paradigm for Management & Job Scheduling
Cost Model: Why are they needed?

- Without cost model any shared system becomes un-manageable.
- It supports QoS based resource allocation and help manage supply-and-demand for resources.
- Improves the value of utility delivered.
- Also, improves the resource utilization.
- Cost units (G$) may be:
  - Rupees/Dollars (real money)
  - Shares in global facility
  - Stored in bank
Cost Matrix

- Non-uniform costing
  - Different users are charged different prices that vary with time.

Resource Cost = Function (cpu, memory, disk, network, software, QoS, current demand, etc.)

Simple: price based on peaktime, offpeak, discount when less demand, ..
Computational Economy Parameters

- Job parameters most relevant to user-centric scheduling
  - Budget allocated to job by user
  - Deadline specified by user
Libra Architecture

User Application

Cluster Management System (PBS)

Job Input Control

Job Dispatch Control

Budget Check Control

Best Node Evaluator

Deadline Control

Node Querying Module

Scheduler (Libra)

Server: Master Node

Cluster Worker Nodes with node monitor (pbs_mom)

(node 1)

(node 2)

(node N)

User Application

(jobs, deadline, budget)
Libra with PBS

- Portable Batch System (PBS) as the Cluster Management Software (CMS)
  - Robust, portable, effective, extensible batch job queuing and resource management system
  - Supports different schedulers
  - Job accounting
  - Allows Plugging of Third-Party Scheduling Solution
The Libra Scheduler

- **Job Input Controller**
  - Adding parameters at job submission time
    - *deadline*
    - *budget*
    - *Execution Time*
  - Defining new attributes of job
- **Job Acceptance and Assignment Controller**
  - Budget checked through cost function
  - Admission control through deadline scheduling
  - Execution host with the minimum load and ability to finish job on time selected
  - **Node Resource Share Allocation**: Proportional to the needs of multiple User Jobs QoS needs.
The Libra Scheduler

- **Job Execution Controller**
  - Job run on the best node according to algorithm
  - Cluster and node status updated
    - runTime
    - cpuLoad

- **Job Querying Controller**
  - Server, Scheduler, Exec Host, and Accounting Logs
Pricing the Cluster Resources

- \( \text{Cost} = \alpha \times (\text{Job Execution Time}) + \beta \times (\text{Job Execution Time} / \text{Deadline}) \)
  - \( \text{Cost} = \alpha \times E + \beta \times E / D \) (where \( \alpha \) and \( \beta \) are coefficients)

- Cost of using the cluster depends on job length and job deadline: the longer the user is prepared to wait for the results, the lower his cost

- Cost formula motivates users to reveal their true QoS requirements (e.g., deadline)
PBS-Libra Web --- Front-end for the Libra Engine

PBS-Libra Web Login Page

Login

User Name: ____________________________
Password: ____________________________
Submit

Welcome to the Web front-end of Libra -- an Economy-Driven Cluster Scheduler. The Libra team is:

Project Owner/Client: Rajkumar Buyya (rajkumar@esse.monash.edu.au)
Jahangshah Sherwani (2002-02-0058)
Nosheen Ali (2002-02-0113)
Nausheen Lotia (2002-02-0111)
Zahra Hayat (2002-02-0189)

Lahore University of Management Sciences

Department of Computer Science
in collaboration with

Monash University
School of Computer Science and Software Engineering

Send questions and comments to spro13@lums.edu.pk. You can find help here.
### PBS-Libra Web Script Submission

**Navigation:** Start Page | Tar File Upload | Console Uploaded Files | Script Generation and Submission | PBS Queue Information | View Job Status | View Job Output | View Home Drive | Login | Logout | Change PBS-Libra Web Password | Erase all submissions

---

#### Execution Commands:

```bash
date
/usr/local/bin/povray
-4/suede/povray11/scenes/advanced/sunsechf.pov +fp +w640 +h480
bgm0 tempting sunsechf.ppm > /home/j/public_html/sunsechf.jpg
date
```

---

#### Job Options

**Job name:**
```
[proj]
```

**Estimate (in seconds):** 10

**Deadline (in seconds):** 20

**Budget (in Expires):** 16

**Queue to submit job:** [default]

---

**Number of processors:** 1

**Maximum time:** 00:00:00 (00:00:00 = no time limit)

**Merge STDERR to STDOUT:** [ ]

**Send message when job:**
- [ ] Aborts
- [ ] Ends
- [ ] Starts

**Address to send messages to:** [sproj@turns.edu.pk]

---

**File Staging (data files only; executable automatically staged)**

**Stagein**
- From here:
- To there:

**Stageout**
- From here:
- To there:

---

**Clear file staging**

---

**Submit Job**

---

**Send questions and comments to:** [sproj@turns.edu.pk]
Job Status for Job 393

Job Id: 393.mspc37.lums.edu.pk
Job_Name = sproj3
Job_owner = j0mspc37.lums.edu.pk
resources_used.cput = 00:00:00
resources_used.mem = 2856kb
resources_used.vmem = 64044kb
resources_used.walltime = 00:00:00
job_state = R
queue = dque
server = mspc37.lums.edu.pk
Checkpoint = u
ctime = Thu May 9 02:29:51 2002
Error_Path = mspc37.lums.edu.pk:/home/j/pbweb/libra/sproj3.e393
exec_host = mspc37/0
Hold_Types = n
Join_Path = n
Keep_Files = n
Mail_Points = web
Mail_Users = sproj3@lums.edu.pk
mtime = Thu May 9 02:29:51 2002
Output_Path = mspc37.lums.edu.pk:/home/j/pbweb/libra/sproj3.o393
Priority = 0
qtime = Thu May 9 02:29:51 2002
Rerunnable = True
Resource_List.ncpus = 1
Resource_List.walltime = 01:00:00
session_id = 20094
Shell_Path_List = /bin/sh
Variable_List = PBS_O_HOME=/home/j,PBS_O_LOGNAME=j,
PBS_O_PATH=/usr/local/bin:/usr/bin:/bin:/usr/bin:/usr/bin:/usr/bin:/usr/bin:
PBS_O_MAIL=/var/mail/j,PBS_O_SHELL=/bin/bash,
PBS_O_HOST=msp037.lums.edu.pk,PBS_O_WORKDIR=/home/j/pbweb/libra,
PBS_O_QUEUE=dque
ctime = Thu May 9 02:29:51 2002
budget = 15
deadline = 20
estimate = 10
Performance Evaluation: Simulations

- **Goal:**
  - Measure the performance of Libra Scheduler

- **Performance = ?**
  - Maximize user satisfaction
  - Maximise value delivered by the utility

- **Simulation Platform: GridSim**
  - Simulated scheduling using the GridSim toolkit
  - [http://www.gridbus.org/gridsim](http://www.gridbus.org/gridsim)
Simulations

- Methodology
  - Workload
    - 120 jobs with deadlines and budgets
    - Job lengths: 1000 to 10000 (MIs)
  - Resources
    - 10 node, single processor (MIPS rating: 100) (homogenous) cluster
Simulations

- **Scheduler simulated as a function**
  - Input: job size, deadline, budget
  - Output: accept/reject, node #, share allocated
Simulations

- **Compared:**
  - Proportional Share (Libra)
  - FIFO (PBS)

- **Experiments:**
  - 120 jobs, 10 nodes
  - Increasing workload to 150 and 200
  - Increasing cluster size to 20
Simulation Results

- 120 jobs, 20 did not meet budget
100 Jobs, 10 Nodes
FIFO: 23 rejected - Proportional Share: 14 rejected

 PBS FIFO

 Libra Proportional

Deadline / Completion time.
Simulation Results

- Increase workload to 200 jobs on the same 10 node cluster
200 Jobs, 10 Nodes
FIFO: 105 rejected - Proportional Share: 93 rejected
Simulation Results

- Scale the cluster up to 20 nodes
200 Jobs, 20 Nodes
FIFO: 35 rejected - Proportional Share: 23 rejected
# PBS FIFO & Libra Strategy

<table>
<thead>
<tr>
<th>No. of Jobs</th>
<th>No. of Nodes</th>
<th>No. of Jobs Accepted</th>
<th>No. of Jobs Rejected</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>PBS FIFO</td>
<td>Libra</td>
</tr>
<tr>
<td>100</td>
<td>10</td>
<td>77</td>
<td>86</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>90</td>
<td>86</td>
</tr>
<tr>
<td>200</td>
<td>10</td>
<td>95</td>
<td>102</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>165</td>
<td>177</td>
</tr>
</tbody>
</table>
Conclusion & Future Work

- Successfully developed a Linux-based cluster that schedules jobs using PBS with our economy-driven Libra scheduler, and PBS-Libra Web as the front end.
- Successfully tested our scheduling policy
- Proportional Share delivers more value to users
- Exploring other pricing mechanisms
- Expanding the cluster with more nodes and with support for parallel jobs
- Implement Libra for SGE (Sun Grid Engine)
  - Sponsored by Sun!
Thank you

Copyright © 1999 United Feature Syndicate, Inc.
Redistribution in whole or in part prohibited