Towards reactive grids through multi objective reinforcement learning

Julien Perez, Cécile Germain-Renaud, Balazs Kégl, Charles Loomis

LRI, Université Paris-Sud / CNRS, Laboratoire de l'Accélérateur Linéaire

December 16, 2008

Outline

Motivations

Proposed methodology

Experimental results

Perspectives

Outline

Motivations

Proposed methodology

Experimental results

Perspectives

EGEE, Enabling Grid for E-sciencE (1/4)

EGEE, Enabling Grid for E-sciencE

- 50k CPUs
- 20 Petabytes of disk
- 150k jobs / day
- 259 **independant** sites in 52 countries
- **Heterogeneous** utilization, from Astronomy to Life Sciences

Aims

- Develop methods **independently** applicable at the site level of the grid.
- Enable **differentiated** QoS w.r.t **jobs** and **users** characteristics.
- Provide **High Level goal driven** scheduling policy

EGEE, Enabling Grid for E-sciencE (2/4)

Issues of the scheduling task in the site level

- Scheduling of interactive jobs
- Workload is hardly predictable
- Minimizing overhead without sacrificing fairness and utilization

EGEE, Enabling Grid for E-sciencE (3/4)

Figure: short jobs in EGEE grid

EGEE, Enabling Grid for E-sciencE (4/4)

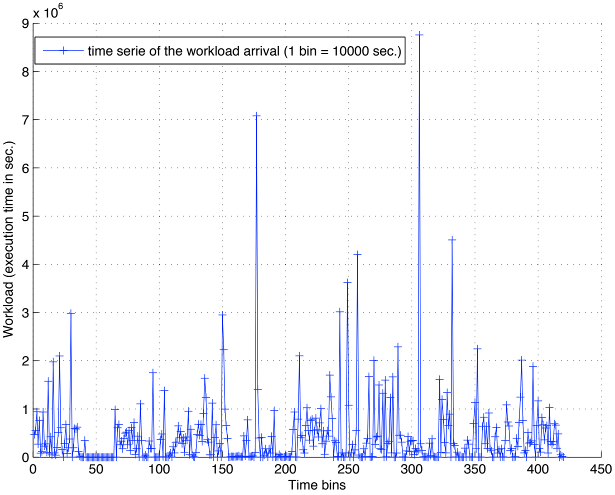

Figure: Workload time serie

Related work (1/2)

Goal-oriented scheduling policy

- **Greedy policies**[1]: Unable to ensure trade-off between several objectives in the long term.
- **Predictive models**[2,3]: Complex models may be required to obtain good performances in real grid systems that are dynamic and non-steady.

---

[1] E.D. Jensen, 1985, **A time driven scheduling model for real-time operating systems**

[2] R.Doyle and al, 2003,**Model-based resource provisioning in a web service utility**

[3] R.Gavalda and al, 2008,**Self-Adaptive Utility-based Web Session Management**

Related work (2/2)

Previous work: ressource allocation using Reinforcement Learning

- Reinforcement learning based scheduling for single machine. [4]
- Power consumption management for server infrastructure.[5]
- Dynamic Channel allocation in Cellular telephone systems[6]

[4] D. Vengerov, 2005, **A Reinforcement Learning Framework for Utility-Based Scheduling in Resource-Constrained Systems**

[5] G. Tesauro, 2007, **Managing Power Consumption and Performance of Computing Systems Using Reinforcement Learning**

[6] D. Bertsekas, 1997,**Reinforcement Learning for Dynamic Channel Allocation in Cellular Telephone Systems**

Outline

Motivations

Proposed methodology

Experimental results

Perspectives

Overview

- **Markov Decision process formalization** of the cluster scheduling problem
- **Reinforcement learning** approach to develop an effective control policy for job scheduling.
- **Multi-criteria objective function** to assure a trade-off between overhead minimization and fairshare.

Scheduling as Markov Decision Process

- Local cluster scheduling problem is considered as a **MDP** :
  - **States**: set of real valued variables measured in the system, s_t.
  - **Actions**: set of real valued variables describing each waiting jobs in the queue j_k.
  - **Rewards**: $R(s_t, j_k) = w_{overhead}.R_{overhead} + w_{fairshare}.R_{fairshare}$
- where $w_{overhead} + w_{fairshare} = 1$
- **Goal**: Find a policy $\pi : S \rightarrow A$, that maximizes the reward accumulation in the long run.

Rewards definitions

- **Relative overhead**:

$$R_{overhead} = \frac{Time\ spent\ in\ the\ system\ -\ Execution\ time}{Execution\ time}$$

- **Fairshare index**: difference between **allocated** resources, F_k, and actually **used** resources, S_k, for each group of users, k, also called Virtual Organizations (VO).

$$R_{fairshare} = 1 - \frac{\underset{k}{\operatorname{argmax}}\,(F_k - S_k)_+}{\underset{k}{\operatorname{argmax}}\,(F_k)}$$

Principle of RL for online policy learning

On-policy Temporal-Difference Control Learning

Require: $Q(s, a) \leftarrow$ *arbitrarily*
repeat
  Initialize s
  Choose action a in s **using policy derived from** Q
  repeat
    Take action a, observe r, s'
    Choose a' from $\mathbb{A}(s')$ **using policy derived from** Q
    $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma Q(s', a') - Q(s, a)]$
    $s \leftarrow s'$; $a \leftarrow a'$
  until s is terminal
until forever

Regression of the Q-Function

Principle

- Use of a feed-forward back propagated **neural net** to regress Q via **SARSA** algorithm[7].
- Neural Models:
  - Standard Feed Forward Network
  - Echo State Network[8]

[7] G. A. Rummery, M. Niranjan, 1994, **On-Line Q-Learning Using Connectionist Systems**

[8] H. Jaeger, 2003, **Adaptive nonlinear system identification with echo state networks**

Outline

Motivations

Proposed methodology

Experimental results

Perspectives

EGEE workload

- **Environment**:
  - Simulation plateform developed in Matlab
  - multi machine, multi queue
  - plugin scheduler
- **Jobs**: extracted from **Torque logs** of the **LAL**.
  - 7 VOs
  - Fairshare target : (20%, 12%, 12%, 6%, 6%, 9%, 35%)
- **Schedulers**:
  - Reinforcement Learning based algorithm
  - gLite

Overhead performance

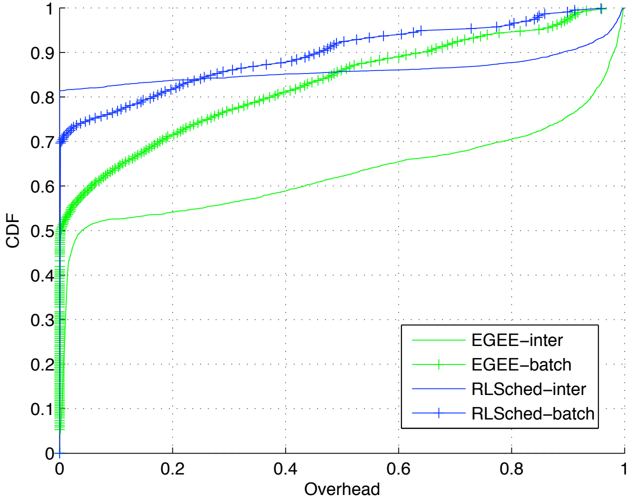

Figure: Overhead comparison

Waiting time performance

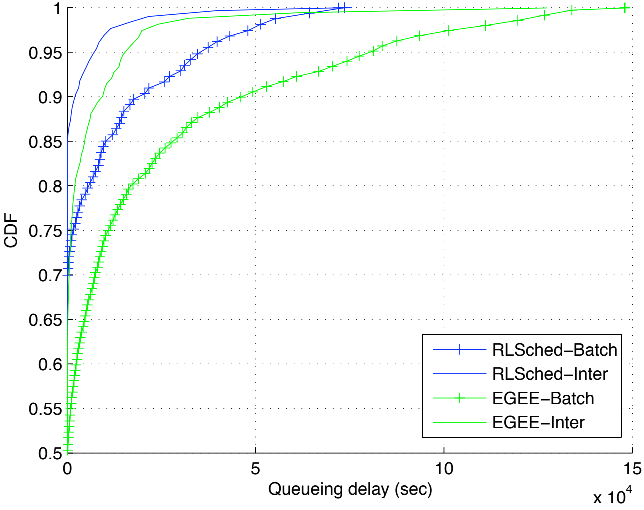

Figure: Waiting time comparison

fairshare performance

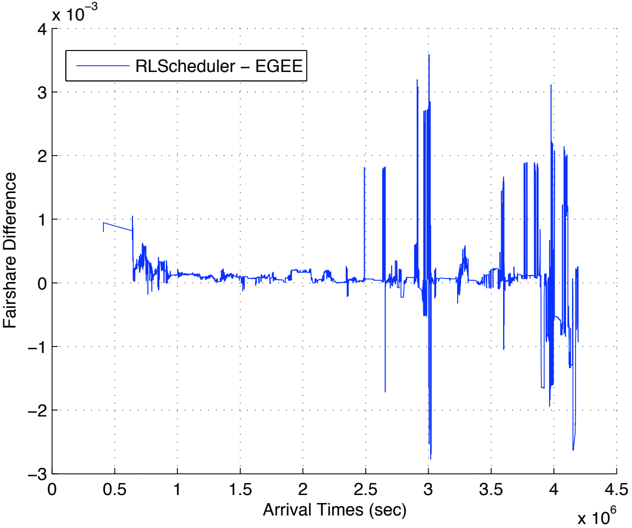

Figure: Fairshare distance to requirement for a RL based scheduler, $D_{EGEE} - D_{RLSched}$

Outline

Motivations

Proposed methodology

Experimental results

Perspectives

On going works and Perspectives

Grid Modeling

- Job description: better use of utility functions
- State description: grid measurement

Reinforcement Learning

- Hierarchical Reinforcement learning[9]
- Apprenticeship Learning[10]

[9] A. Barto and al, 2004, **Recent Advances in Hierarchical Reinforcement Learning**

[10] P. Abbeel and A. Y. Ng, 2005, **Exploration and apprenticeship learning in reinforcement learning**