A Simulation Study of Cooperative Load Balancing in Central-Server Node Distributed Systems

Satish Penmatsa and Jiromu Amioku

Department of Mathematics and Computer Science
University of Maryland Eastern Shore
Princess Anne, MD 21853, USA

Abstract - *Load balancing is very important for achieving high performance in distributed computer systems which often consist of heterogeneous computing and communications resources. In this paper, we study a cooperative load balancing scheme for central-server node distributed systems (CCOOP-IO) and evaluate its performance using simulations. The objective of CCOOP-IO is to minimize the mean response time of jobs in a heterogeneous distributed computing system and also to provide fairness to all the jobs in the system. We consider a heterogeneous computing system model connected by a single-channel communications network. A central-server model is used to model the computers in the system. The performance of CCOOP-IO is evaluated using simulations with various system loads and configurations.*

Key words: Load balancing, Resource Allocation, Distributed Systems, Fairness.

1. Introduction

The computing resources (computers or nodes) in distributed computing systems are often heterogeneous. Jobs may arrive with different job arrival rates to these nodes. Also, the communications networks that connect the computing resources may have different bandwidths. The above factors may degrade the performance of distributed systems if load is not properly balanced among the computers. Hence, load balancing is very important for achieving high performance in distributed computer systems.

The problem of load balancing in distributed systems has been studied extensively. For example, in [4, 11, 15, 16, 10], static load balancing schemes for single-class and multi-class job distributed systems were proposed and analyzed by considering various network topologies. Various models for dynamic load balancing were studied in [1, 6, 14]. A macroeconomic model for resource allocation in distributed systems was studied in [12]. Most of the past work on load balancing in distributed systems considered the minimization of the overall system expected (mean or average) response time (job execution time) as their main objective. However, some jobs might experience much longer response time than the others in such allocations. Providing *fairness* to all the jobs in the system is to find an allocation of jobs to computers that yields an approximately equal expected response time for all the jobs of approximately the same size. Fairness is a major issue in many modern computing systems.

Load balancing in distributed systems based on game theory with the objective of providing fairness has been studied ([2, 5, 7, 17] and references there-in). However, in most of the above studies, the computer model considered has only a processor. Game-theoretic scheduling in cognitive radio systems has been studied in [13] and references there-in. Here, we consider a central-server computer model which is very common in modern computer systems. A central-server computer model consists of a CPU (processor) and one or more input/output (I/O) devices. A central-server node model for static job allocation in E-commerce systems and utility-computing systems has been studied in [8, 9] and for dynamic job allocation in [6].

In this paper, we study a cooperative load balancing scheme for central-server node distributed systems (CCOOP-IO). CCOOP-IO is derived from the cooperative scheme studied in [7]. The objective of CCOOP-IO is to minimize the mean response time of jobs in a heterogeneous distributed computing system and also to provide fairness to all the jobs in the system. We consider a heterogeneous computing system model connected by a single-channel communications network. Jobs arrive at each computer according to a time-invariant exponential process. We achieve load balancing by transferring some jobs from the heavily loaded nodes to the nodes that are idle or lightly loaded.

The performance of CCOOP-IO is evaluated using simulations with various system loads and configurations. *Expected response time* (execution time) and *fairness index* are used as the performance metrics. For comparison, we also implemented two representative load balancing schemes. These schemes are: OPTIM-IO (which minimizes the expected response time of all the jobs in a system) and PROP-IO (which allocates the jobs to the computers in proportion to their processing speeds in the system).

2. Cooperative Load Balancing

A distributed computing system model having *n* nodes connected by a single channel communications network is considered. The nodes in the system are typically heterogeneous having different processing speeds. Each node is modeled as a central-server model as shown in Figure 1 similar to [6]. The terminology and notations used similar to [3, 6, 7] are as follows:

- t_{IO}: The service time of an input/output (I/O) device.
- μ_i: The service rate of node *i*.
- ϕ_i: The external job arrival rate at node *i*.
- $\boldsymbol{\Phi}$: The total external job arrival rate of the system. So, $\boldsymbol{\Phi} = \sum_{i=1}^{n} \phi_i$.
- βi : The job processing rate (or load) allocated by the load balancing algorithm for node i.
- x_{ij}: The job flow rate from node i to node j (i.e. the number of jobs sent from i to j per unit time).
- t: Mean communication time for sending or receiving a job from one node to another.
- P_0: Probability that a job after departing from the processor finishes.
- P_1: Probability that a job after departing from the processor requests I/O service.
- P_1/P_0: Average number of I/O requests per job.

Each node is assumed to have a single computing resource (processor) with a round-robin service discipline and jobs arrive in a single queue. The nodes and the communications network have an exponential service-time distribution [3] and the external jobs arriving at each node and jobs being transferred by the communications network follow a Poisson distribution [3].

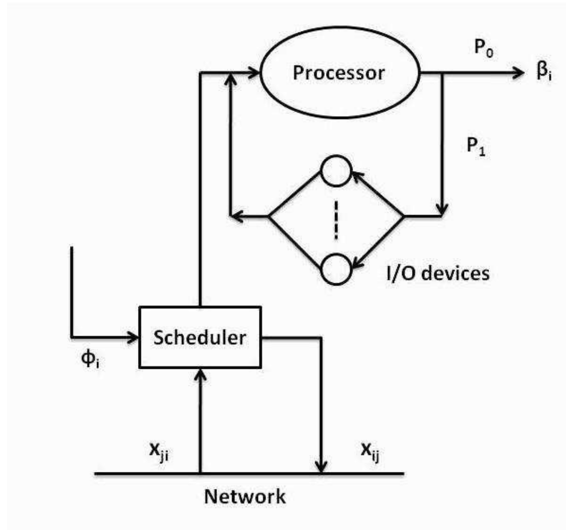

Figure 1. Node Model

A job arriving at node *i* may either be processed at node *i* or transferred to node *j* through the communications network for remote processing. The mean communication delay from node *i* to node *j* is independent of the source destination pair (*i*, *j*) but depends on the total traffic through the network denoted by λ where $\lambda = \sum_{i=1}^{n} \sum_{j=1}^{n} x_{ij}$. Based on the above assumptions and assumptions similar to [7], the mean node delay (mean response time or total execution time) of a job at node *i* is given by:

$$d_i(\beta_i) = \frac{1}{\mu_i - \beta_i} + \frac{P_1}{P_0} t_{IO}, \quad i = 1, \ldots, n.$$

The mean communication delay for a job is given by:

$$g(\lambda) = \frac{t}{1 - t\lambda}, \ \lambda < \frac{1}{t}.$$

The inverse of the node delay is given by:

$$d^{-1}(x) = \mu_i - \frac{1}{(x - \frac{P_1}{P_0} t_{IO})} \ \ if \ x > \frac{1}{\mu_1} + \frac{P_1}{P_0} t_{IO}$$

$$d^{-1}(x) = 0 \ \ if \ \ x \le \frac{1}{\mu_1} + \frac{P_1}{P_0} t_{IO}$$

We also assume that the communication delay incurred as a result of sending a job directly from node *i* to node *j* is less than or equal to the sum of the delays from node *i* to node *k* and from node *k* to node *j*. Based on this assumption, nodes are classified into Sinks (S), Idle Sources (R_d), Active Sources (R_a), and Neutrals (N) similar to [7].

The load balancing problem for providing fairness to all the jobs in the system is formulated as a cooperative game among the computers and the communications subsystem similar to [7]. Based on the Nash Bargaining Solution (NBS) which provides a *pareto* optimal and fair solution, we provide an algorithm (CCOOP-IO) for computing the NBS for our cooperative load balancing game.

In the following, we present the CCOOP-IO algorithm. The cooperative load balancing game among the computers and the communication subsystem, theorems, and properties which are the basis for the below algorithm are similar to the ones described in [7] by replacing *d*, *g*, and d^{-1} in [7] by *d*, *g*, and d^{-1} presented in the previous section.

CCOOP-IO Algorithm:

Input: Node job service rates: $\mu_1, \mu_2, \ldots, \mu_n$
Node job arrival rates: $\emptyset_1, \emptyset_2, \ldots, \emptyset_n$
Mean communication time: t
Service time of an I/O device: t_{IO}
Probabilities: P_0, P_1

Output: Load allocation to the nodes: $\beta_1, \beta_2, \ldots, \beta_n$

1. Initialize the loads of all the nodes to their job arrival rates and label all the nodes as *Neutrals*.
2. Sort the computers in increasing order of their node delays.
3. Categorize the nodes into Sinks (S), Idle Sources (R_d), Active Sources (R_a), and Neutrals (N) using a binary search (for finding an optimal point (say, α) that categorizes) similar to Step 3 of the CCOOP algorithm in [7].
4. Determine the loads on the computers as follows:
   - $\beta_i \leftarrow 0$, if node *i* is an Idle Source.
   - $\beta_i \leftarrow d^{-1}(\alpha + g(\lambda))$, if node *i* is an Active Source.
   - $\beta_i \leftarrow d^{-1}(\alpha)$, if node *i* is a Sink.
   - $\beta_i \leftarrow \emptyset_i$, if node *i* is a Neutral.

3. Experimental Results

In this section, we evaluate the performance of the CCOOP-IO scheme. The performance metrics that are used in the experiments are the *expected response time* and the *fairness index*. The fairness index [7] is used to quantify the fairness of load balancing schemes. We also implemented the Overall optimal load balancing scheme (OPTIM-IO) [4] and the Proportional load balancing scheme (PROP-IO) [1] for comparison purposes.

System utilization represents the amount of load on the system and is defined as the ratio of the total arrival rate to the aggregate service rate of the system. A heterogeneous distributed system consisting of 16 computers was simulated (as shown in Table 1) to study the effect of system utilization. The system has computers with four different service rates. For each experiment, the total job arrival rate in the system is determined by the system utilization and the aggregate service rate of the system. We had chosen fixed values for the system utilization and determined the total job arrival rates. The mean communication time is assumed to be 0.001 sec. We assumed that I/O operations were evenly spread throughout the execution of each job (similar to [6]) and that each disk I/O request took 0.06 milliseconds. The number of I/O requests for each job was chosen from a normal distribution with a mean of 12 and a standard deviation of 10 and was assumed to be greater than 0.

Table 1. System Configuration.

Relative service rate	1	2	5	10
Number of computers	6	5	3	2
Service rate (jobs/sec)	10	20	50	100

In Figure 2, we present the expected response time of the system for different values of system utilization ranging from 10% to 90%. The performance of CCOOP-IO is very similar to OPTIM-IO for system utilizations ranging from 10% to 40% and is around 50% better than PROP-IO for system utilizations ranging from 50% to 60%. CCOOP-IO approaches PROP-IO for high system utilizations.

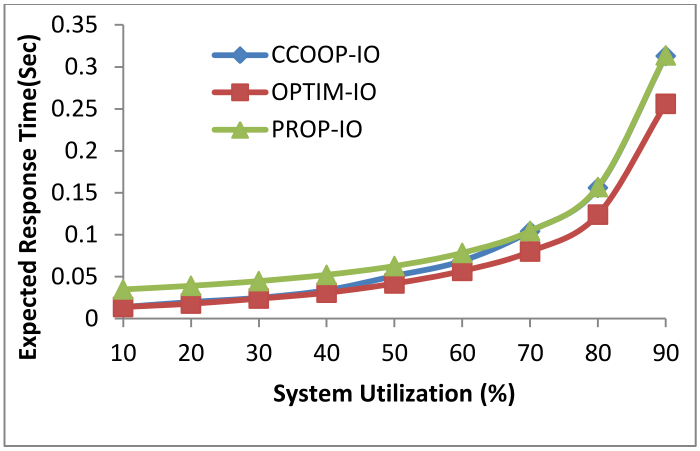

Figure 2. Expected Response Time v/s System Utilization

The effect of the system utilization on the fairness index of the various schemes is presented in Figure 3. It can be observed that the fairness index of CCOOP-IO is almost 1 for any system utilization and the fairness index of OPTIM-IO drops from 1 to around 0.85. PROP-IO has a constant fairness index which is around 0.72. This shows that CCOOP-IO provides fairness to all the jobs in the system independent of the computers to which they are allocated.

Figure 4 presents the expected response time at each computer for all the schemes at a system utilization of 70%. It can be observed that CCOOP-IO guarantees almost equal expected response times for all the computers. This means that all the jobs will have almost the same expected response time independent of the allocated computers. In the case of OPTIM-IO and PROP-IO, the expected response times are less balanced than CCOOP-IO.

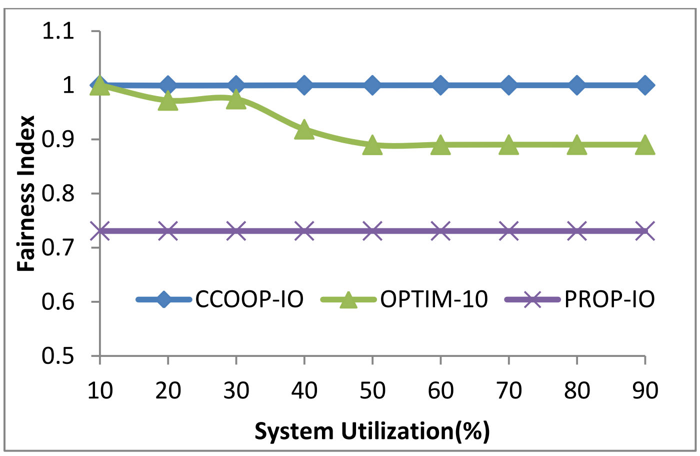

Figure 3. Fairness Index v/s System Utilization

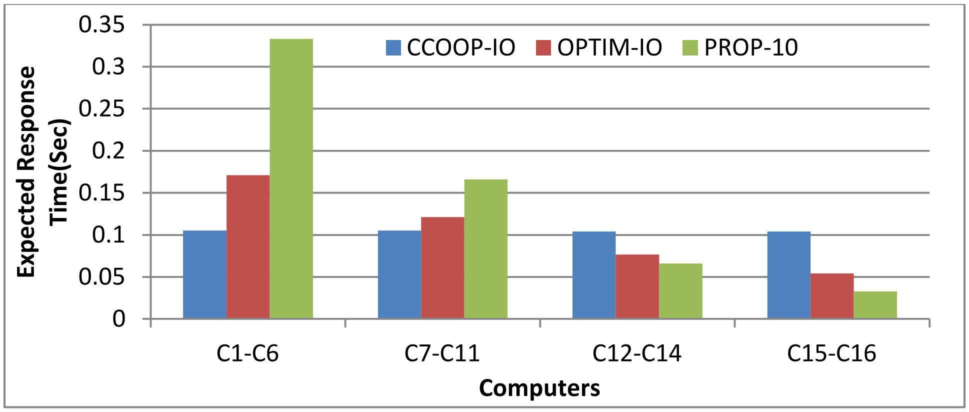

Figure 4. Expected Response Time at each Computer (System Utilization = 70%)

In the following, we study the effect of heterogeneity (speed skewness) [7] on the performance of CCOOP-IO. Speed skewness is defined as the ratio of maximum service rate to the minimum service rate of the computers in the system. A heterogeneous distributed system of 16 computers (2 fast and 14 slow) was simulated to study the effect of heterogeneity. Slow computers have a relative processing rate of 1 and the relative processing rate of the fast computers is varied from 1 (homogenous system) to 20 (highly heterogeneous system).

Figure 5 presents the effect of speed skewness on the performance of CCOOP-IO. For low skewness, the performance of CCOOP-IO is similar to PROP-IO. However, as the skewness increases, the performance of CCOOP-IO approaches to that of OPTIM-IO. Figure 6 presents the effect of speed skewness on the fairness index of CCOOP-IO. It can be observed that CCOOP-IO has a fairness index of almost 1 over all range of speed skewness. The fairness index of OPTIM-IO and PROP-IO falls from 1 at low skewness to 0.95 and 0.9 respectively at high skewness. This shows that CCOOP-IO provides fairness in highly heterogeneous systems for all the jobs in the system.

Figures 7 and 8 present the expected response time at each computer for all the schemes at medium system utilization for a skewness of 8 and 12. CCOOP-IO guarantees almost equal expected response times for all the computers. This means that CCOOP-IO provides a fair and load balanced allocation compared to OPTIM-IO and PROP-IO where the jobs are treated unfairly.

4. Conclusions

In this paper, a cooperative load balancing scheme (CCOOP-IO) for heterogeneous distributed systems was studied and evaluated. A distributed system with central-server nodes was considered. The performance of CCOOP-IO is evaluated by varying the system utilization and heterogeneity. Experimental results showed that CCOOP-IO is not only fair but also is comparable with that of the system optimal scheme in terms of the mean response time.

Acknowledgements

This research is supported in part by the Department of Mathematics and Computer Science at the University of Maryland Eastern Shore.

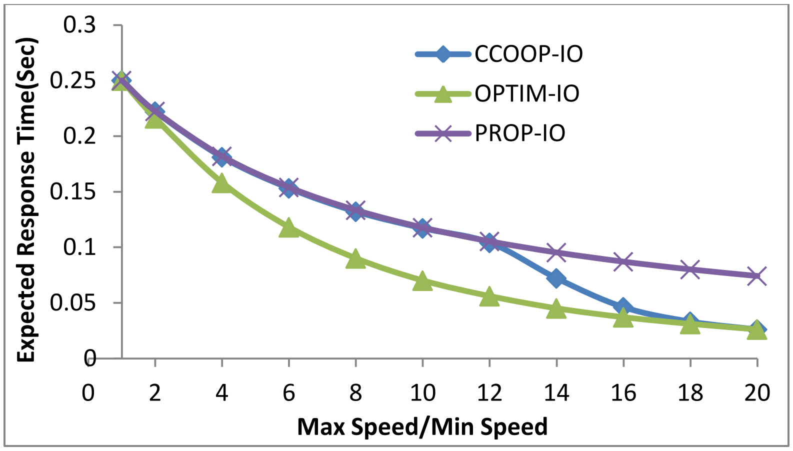

Figure 5. Expected Response Time v/s Heterogeneity

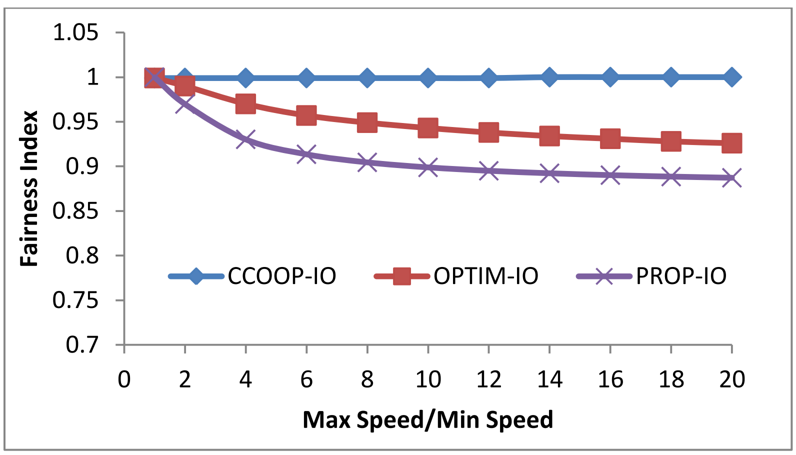

Figure 6. Fairness Index v/s Heterogeneity

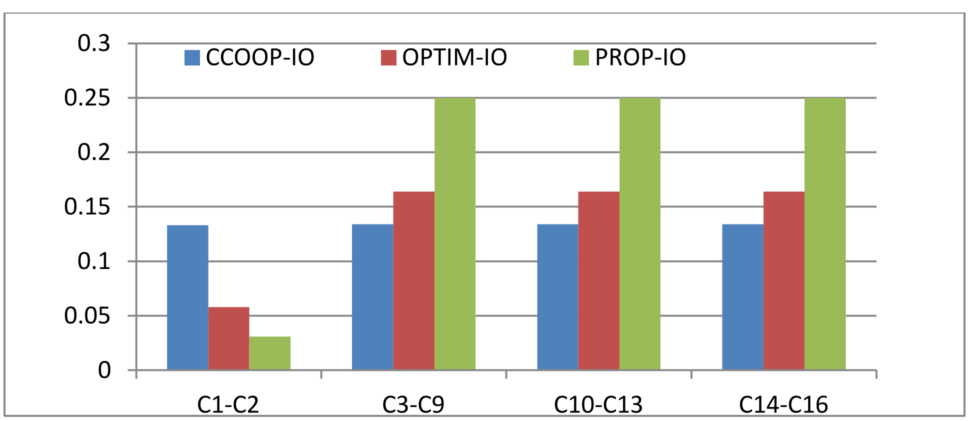

Figure 7. Expected Response Time at each Computer (Speed Skewness = 8)

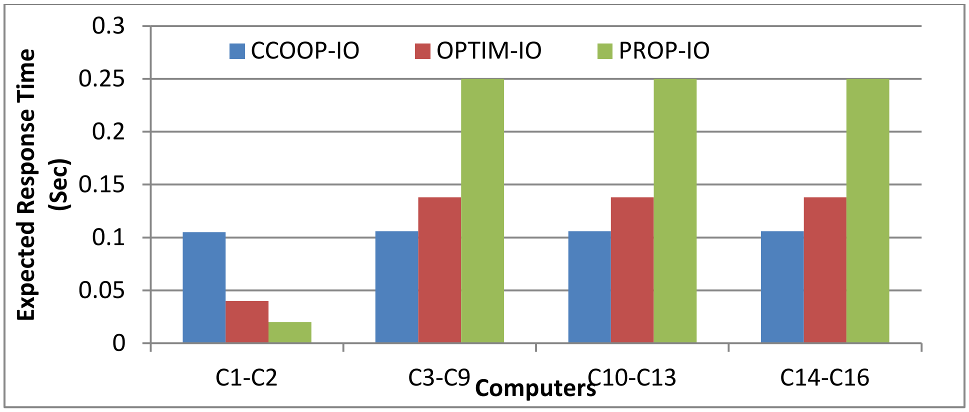

Figure 8. Expected Response Time at each Computer (Speed Skewness = 12)

References

[1] Y.C. Chow and W.H. Kohler. Models for dynamic load balancing in a heterogeneous multiple processor system. *IEEE Trans. On computers*, C-28(5):354-361, May 1979.

[2] D. Grosu and A.T. Chronopoulos. Noncooperative load balancing in distributed systems. *Journal of parallel and distributed computing*, 65(9):1022-1034, Sep. 2005.

[3] R. Jain. *The Art of computer systems performance analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. Wiley-Interscience, 1991.

[4] H. Kameda, J. Li, C. Kim, and Y. Zhang. *Optimal Load balancing in distributed computer systems.* Springer Verlag, London, 1997.

[5] R. Subrata, A. Zomaya, and B.Landfeldt. Game theoretic approach for market-like computational grids. *IEEE Trans. On parallel and distributed systems*, 19(1):66-76, Jan. 2008.

[6] Y. Zhang, K. Hakozaki, H. Kameda, and K. Shimizu. A performance comparison of adaptive and static load balancing in heterogeneous distributed systems. In proc. *IEEE 28th Annual Simulation Symp*., pages 332-340, Phoenix, AZ, 1995.

[7] S. Penmatsa and A.T. Chronopoulos. Game theoretic static load balancing for distributed systems. *Journal of Parallel and Distributed Computing*, 71 (2011), 537-555.

[8] S. Penmatsa and Gurdeep S. Hura. Job Allocation in E-Commerce Systems Involving Self-Interested Agents. *Journal of Global Information Technology*, 5(1), pp 1-10, 2010.

[9] S. Penmatsa and Gurdeep S. Hura. Cost Minimization in Utility-driven Distributed Computing Systems with Central-Server Model. *Proceedings of the 23rd International Conference on Computer Applications in Industry and Engineering (CAINE 2010)*, Las Vegas, NV, USA, Nov. 8-10, 2010.

[10] S. Penmatsa and V. P. Mantena. Comparison of Static Load Balancing Schemes for Utility-driven Distributed Computing. Proceedings of the 16th International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 2010), Las Vegas, NV, USA, July 12 15, 2010.

[11] A.N. Tantawi, D. Towsley, Optimal static load balancing in distributed computer systems, J. ACM 32 (2) (1985) 445–465.

[12] X. Bai, D.C. Marinescu, L. Boloni, H.J. Siegel, R.A. Daley, I.-J. Wang, A macroeconomic model for resource allocation in large-scale distributed systems, J. Parallel Distrib. Comput. 68 (2) (2008) 182–199.

[13] Qiang Ni and C. Zarakovitis, Nash Bargaining Game Theoretic Scheduling for Joint Channel and Power Allocation in Cognitive Radio Systems, IEEE J. on Selected Areas in Communications, 30(1), 2012, pp. 70-81.

[14] Y. Zhang, H. Kameda, S.L. Hung, Comparison of dynamic and static load balancing strategies in heterogeneous distributed systems, IEE Proc. Comput. Digit. Tech. 144 (2) (1997) 100–106.

[15] N.G. Shivaratri, P. Krueger, M. Singhal, Load distributing for locally distributed systems, Comput. 25 (12) (1992) 33–44.

[16] T.D. Braun, H.J. Siegel, A.A. Maciejewski, Y. Hong, Static resource allocation for heterogeneous computing environments with tasks having dependencies, priorities, deadlines, and multiple versions, J. Parallel Distrib. Comput. 68 (11) (2008) 1504–1516.

[17] K. Rzadca, D. Trystram, A. Wierzbicki, Fair game-theoretic resource management in dedicated grids, in: Proc. of the 7th IEEE Intl. Symp. on Cluster Computing and the Grid, Brazil, May 2007, pp. 343–350.