International Journal of Science & Informatics
Vol. 1, No. 1, Fall, 2011, pp. 28-33
ISSN 2158-835X (print), 2158-8368 (online),

DISTRIBUTED ALGORITHMS FOR PROVIDING FAIRNESS IN HETEROGENEOUS COMPUTER SYSTEMS

Satish Penmatsa (spenmatsa@umes.edu)
University of Maryland Eastern Shore, USA

ABSTRACT

Distributed computing systems often consist of heterogeneous computing resources managed by different administrators. Due to the distributed nature, effective management of the resources may become difficult and the performance of the system may be affected. Also, the users of distributed systems may be self-interested whose goal would be to maximize their own utility. Such self-interested agents may adversely affect the performance of the system. Here, we present performance optimization algorithms whose objective is to provide fairness to all the users of a distributed system involving selfish users i.e. all the users will experience approximately the same expected response time for the execution of their jobs or will have to pay approximately the same expected price for the execution of their jobs. Distributed heterogeneous systems with various system and node models are considered. Experimental results with various system configurations are presented comparing the performance of the presented algorithms with other existing schemes.

Keywords*: Distributed algorithms, Heterogeneous systems, Fairness.*

INTRODUCTION

Distributed computing systems often consist of heterogeneous computing and communications resources. The computing resources of these distributed systems are typically connected using wide area networks, can be under different administrative domains, and can be managed by different resource owners. Also, the users of these systems or their agents may be self-interested whose goal would be to optimize their own utility. For example, a user might want to use most of the available resources for minimizing his transaction or job completion time or the price that has to be paid for executing his/her jobs. This selfish nature can affect the fairness that the system has to provide to all the users. Hence, efficient scheduling algorithms or schemes are essential for providing fairness in a distributed computing system. We say that an allocation is fair if all the users experience approximately the same expected response time or have to pay approximately the same expected price for the execution of their jobs which are approximately of the same size independent of the computers allocated for their execution.

Here, we present a few performance optimization algorithms (job allocation schemes or scheduling schemes) whose objective is to provide fairness to all the users of a distributed computing system. Distributed systems with various system and node models are considered. Non-cooperative game theory (Nash (1951), Fudenberg et al. (1994)) is used in formulating the optimization problems and the concept of Nash equilibrium (Nash (1951) is used for providing a solution to the optimization problems. A pricing model based on bargaining game theory (Ghosh et al. (2005) is used to obtain the prices charged by the resource owners for executing the users' jobs. The performance of the algorithms was evaluated using simulations with various heterogeneous system models.

The following schemes are presented in this paper: NASH scheme (Grosu et al. (2005)) provides fairness to all the users in terms of their expected response time without considering any communication delays; NASHP algorithm (Penmatsa et al. (2005)) provides fairness to all the users in terms of their expected price without considering any communication costs; NASHPC algorithm (Penmatsa et al. (2006)) provides fairness to all the users in terms of their expected price for a system by considering communication costs; and NCOOPC-CS algorithm (Penmatsa et al. (2010)) provides fairness to all the users in terms of their expected response time for a system with central-server node model by considering communication costs.

Many studies exist on load balancing or job allocation in distributed systems considering various system models. For example, Ghosh et al. (2005), Kameda et al. (1997), Campos et al. (2000) and references there-in. However, most of the above studies considered the minimization of the overall system response time or cost as their objective. Fairness is also very important in modern computer systems which involve selfish agents. Preliminary survey of the above schemes that provide fairness can be found in Penmatsa (Dec. 2010).

FAIR JOB ALLOCATION DISTRIBUTED ALGORITHMS

In the following we present the job allocation / load balancing algorithms for distributed computing systems that provide fairness to all the users. Distributed system models with *n* computers (nodes) and *m* users are considered. The load balancing problem is formulated as a non-cooperative game among users and the concept of Nash equilibrium (Nash (1951)) is used as the solution of the game. Nash equilibrium provides a user-optimal operation point for the distributed system.

A *non-cooperative job allocation game* (Grosu et al. (2005)) consists of a set of players, a set of strategies, and preferences over the set of strategy profiles. The users are the players. Each user j $(j = 1 \dots m)$ must find the workload that is assigned to computer i such that the expected response time or price of his/her own jobs is minimized. The strategy of user j depends on the job allocation strategies of the other users. Each user's preferences are represented by his/her expected response time or price. At the Nash equilibrium, a user j cannot further decrease his/her expected response time or price by choosing a different job allocation strategy when the other users' strategies are fixed. The equilibrium strategy profile can be found when each user's job allocation strategy is a *best response (best reply)* (Grosu et al. (2005), Penmatsa et al. (2005)) to the other users' strategies.

NASH Job Allocation Scheme

NASH load balancing algorithm for providing fairness to all the users in terms of their expected response time is proposed by Grosu et al. (2005). Each computer is modeled as an M/M/1 queuing system (i.e. Poisson arrivals and exponentially distributed processing times) (Jain (1991)). Each user finds his/her workload that should be assigned to computer i $(i = 1 \dots n)$ such that the expected response time of his/her jobs is minimized. This response time does not include any communication delays because of job transfers. The best reply for user j is given by Theorem 2.1 in Grosu et al. (2005).

The computation of Nash equilibrium may require some coordination between the users. This is necessary in the sense that users need to coordinate in order to obtain the load information from each computer. From the practical point of view decentralization is needed and this can be obtained by using distributed greedy best reply algorithms (Grosu et al. (2005)). In these algorithms each user updates from time to time his/her load balancing strategy by computing the best reply against the existing load balancing strategies of the other users. Based on the BEST-REPLY algorithm in Grosu et al. (2005), a greedy algorithm, NASH, is devised for computing the Nash equilibrium for the non-cooperative load balancing game. In this algorithm users are synchronized such that they update their strategies in a round-robin fashion.

The execution of this algorithm is restarted periodically or when the system parameters are changed. Once the Nash equilibrium is reached, the users will continue to use the same strategies and the system remains in equilibrium. This equilibrium is maintained until a new execution of the algorithm is initiated.

NASHP Job Allocation Scheme

NASHP load balancing algorithm for providing fairness to all the users in terms of their expected price is proposed by Penmatsa et al. (2005). Based on the allocation provided by NASHP, all the users pay approximately the same price for the execution of their jobs independent of the computers allocated for the execution of their jobs which are approximately of the same size. Each computer is modeled as an M/M/1 queuing system. Each user finds his/her workload that should be assigned to computer i $(i = 1 \dots n)$ such that the expected price of his/her jobs is minimized. Similar to NASH scheme, NASHP also does not consider the communication costs in making allocation decisions. A pricing model based on bargaining game theory is used to obtain the prices that have to be paid by the users for using the system resources. The best reply for user *j* is given by Theorem 3 in Penmatsa et al. (2005). Similar to NASH algorithm, a greedy algorithm, NASHP, is used for computing the Nash equilibrium for the non-cooperative load balancing game.

NASHPC Job Allocation Scheme

NASHPC load balancing algorithm for providing fairness to all the users in terms of their expected price is proposed by Penmatsa et al. (2006). NASHPC takes the communication costs into account in making allocation decisions. The computers and the communications network are modeled as M/M/1 systems. A job arriving at node *i* may be either processed at node *i* or transferred to a neighboring node *j* for remote processing through the communications network. The decision of transferring a job does not depend on the state of the system and hence is static in nature. A pricing model based on bargaining game theory is used to obtain the prices that have to be paid by the users for using the system resources.

NASHPC categorizes the computers (nodes) into the following (Penmatsa et al. (2006)): Idle source nodes, Active source nodes, Neutral nodes, and Sink nodes. Each user finds his workload that should be assigned to computer *i* $(i = 1 \ldots n)$ such that the expected price of his/her jobs is minimized. This expected price includes the node and communication costs. The best response for user *j* is given by Theorem 3.1 in Penmatsa et al. (2006). A greedy algorithm, NASHPC, is used for computing the Nash equilibrium for the non-cooperative load balancing game.

NCOOPC-CS Job Allocation Scheme

NCOOPC-CS load balancing algorithm for providing fairness to all the users in terms of their expected response time is proposed by Penmatsa et al. (2010). NCOOPC-CS takes the communication costs into account in making allocation decisions. Each computer in the system is modeled as a central-server model. A central-server computer model consists of a processor and one or more I/O devices.

Similar to NASHPC, NCOOPC-CS categorizes the computers (nodes) into the following (Penmatsa et al. (2010)): Idle source nodes, Active source nodes, Neutral nodes, and Sink nodes. Each user finds his workload that should be assigned to computer *i* $(i = 1 \ldots n)$ such that the expected response time of his/her jobs is minimized. This expected response time includes the node and communication delays. The best response for user *j* is given by Theorem 3.1 in Penmatsa et al. (2010). A greedy algorithm, NCOOPC-CS, is used for computing the Nash equilibrium for the non-cooperative load balancing game.

EXPERIMENTAL RESULTS

In this section, we present some experimental results that show the fairness provided by the above schemes. Various heterogeneous system models are used for evaluating the performance of the presented schemes. The performance of the presented schemes is compared with other existing schemes. For example, OPTIM (Kim et al. (1992)) scheme provides a system optimal solution that minimizes the expected response time of all the jobs in the system. PROP (Chow et al. (1979)) scheme allocates the jobs in proportion to the processing speeds of the computers. In the following we present some results from Penmatsa et al. (2010) for the NCOOPC-CS scheme. The results obtained for the other schemes NASH, NASHP, and NASHPC are similar.

A system configuration used for the experiments is given in Table 1. The system has computers with four different service rates. The first row shows the relative service rates of the different computer types in the system. The second row shows the number of computers in the system corresponding to each computer type. The third row shows the service rate of each computer type in the system.

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

Figure's 1 and 2 present the expected response time obtained by NCOOPC-CS and OPTIM for each user at 50% and 90% system loads respectively. It can be observed that in the case of OPTIM there are large differences in the users expected response times where as in the case of NCOOPC-CS the users expected response times are approximately the same. Similar kind of behavior of NCOOPC-CS and OPTIM has been observed at all system loads. Hence, the allocation provided by NCOOPC-CS is considered fair to all the users *i.e.* all the users experience approximately the same expected response time independent of the computers allocated for the execution of their jobs which are approximately of the same size.

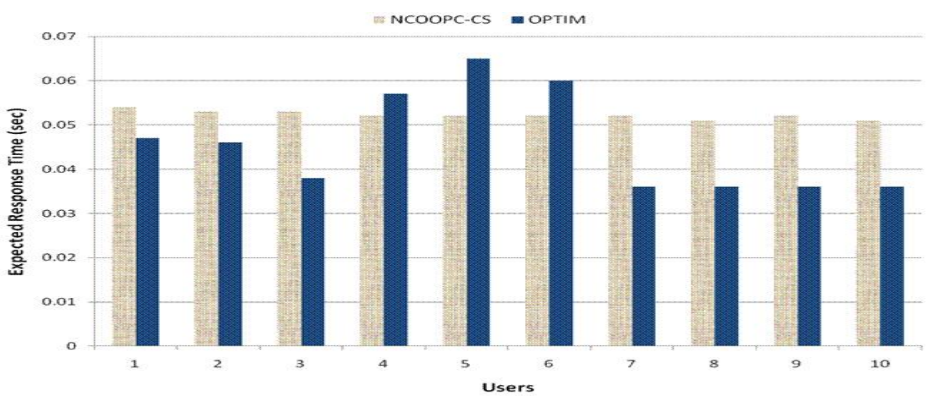

Figure 1: Expected Response Time for each User (system load = 50%)

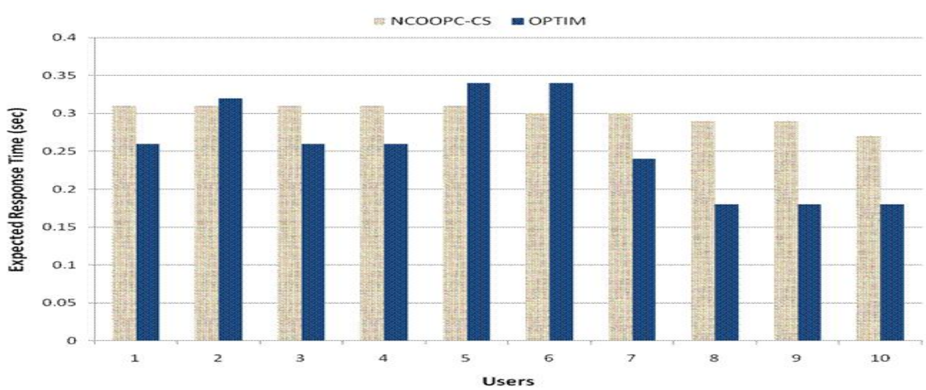

Figure 2: Expected Response Time for each User (system load = 90%)

Figure's 3 and 4 show the expected response time obtained by NCOOPC-CS and OPTIM for each user for heterogeneity levels (Penmatsa et al. (2010)) 8 and 16 respectively. It can be observed that in the case of OPTIM there are large differences in the users expected response times where as in the case of NCOOPC-CS the users expected response times are approximately the same.

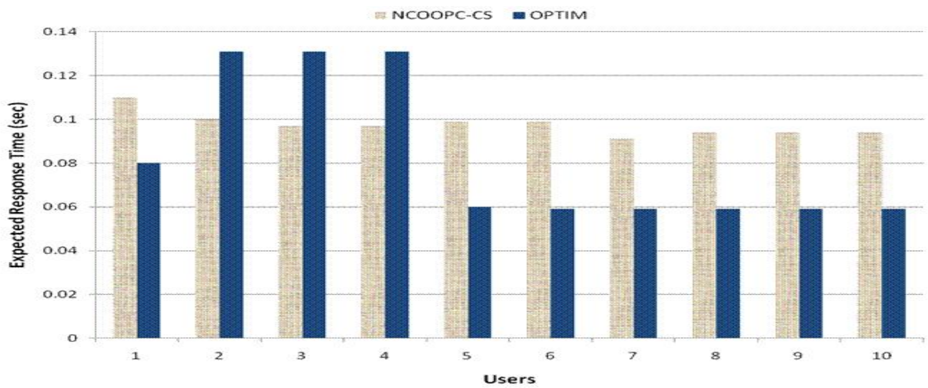

Figure 3: Expected Response Time for each User (heterogeneity level = 8)

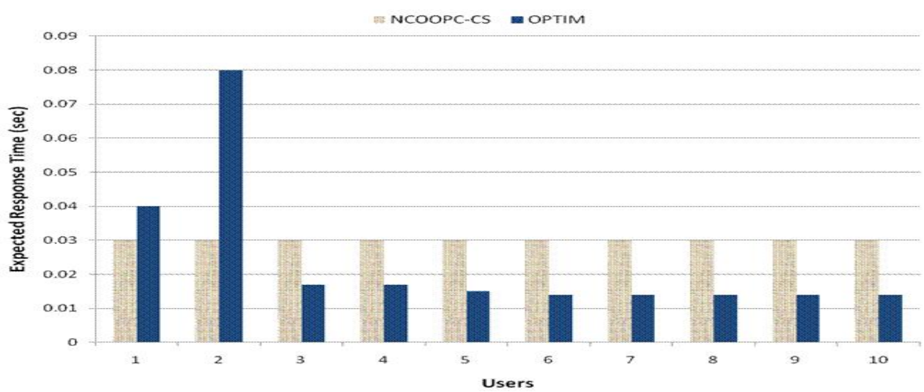

Figure 4: Expected Response Time for each User (heterogeneity level = 16)

Similar kind of behavior of NCOOPC-CS and OPTIM has been observed for other levels of heterogeneity. The expected response times of all the users are approximately the same in the case of NCOOPC-CS where as there are large differences in the case of OPTIM. Hence, the allocation provided by NCOOPC-CS is considered fair to all the users.

CONCLUSIONS

In modern distributed computing systems, fairness to the users is a very important characteristic. In this paper, job allocation algorithms (NASH, NASHP, NASHPC, and NCOOPC-CS) for heterogeneous distributed computing systems whose objective is to provide fairness to all the users are presented. The schemes are proposed for various system and node models. Non-cooperative game theory was used in formulating the optimization problems and the concept of Nash equilibrium is used for providing a solution to the optimization problems. Experimental results with various system configurations are presented comparing the performance of the presented algorithms with other existing schemes. Results showed that the performance of the presented schemes is close to that of system optimal schemes. Moreover, the presented schemes provide almost equal response times or prices for all the users and hence provide fairness in a selfish environment.

REFERENCES

Campos L. M. and Scherson I. (2000). Rate of change load balancing in distributed and parallel systems, *Parallel Computing*, 26(9):1213–1230, July 2000.

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

Fudenberg D. and Tirole J. (1994). Game Theory, *The MIT Press*, 1994.

Ghosh P., Roy N., Das S. K., and Basu K. (2005). A pricing strategy for job allocation in mobile grids using a non-cooperative bargaining theory framework. *Journal of Parallel and Distributed Computing*, 65(11):1366–1383, 2005.

Grosu D. and Chronopoulos A. T. (2005). Noncooperative load balancing in distributed systems, *Journal of Parallel and Distributed Computing*, 65(9):1022–1034, Sep. 2005.

Jain R. (1991) The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling, *Wiley-Interscience*, 1991.

Kameda H., Li J., Kim C., and Zhang Y. (1997). Optimal Load Balancing in Distributed Computer Systems, *Springer Verlag*, London, 1997.

Kim C. and Kameda H. (1992). An algorithm for optimal static load balancing in distributed computer systems, *IEEE Trans. on Computers*, 41(3):381–384, March 1992.

Nash J. (1951). Non-cooperative games. *Ann. Math*., 54(2):286295, Sept. 1951.

Penmatsa S. and Chronopoulos A. T. (2005). Job allocation schemes in computational grids based on cost optimization. In *Proc. of the 19th IEEE International Parallel and Distributed Processing Symposium*, Denver, Colorado, USA, 2005.

Penmatsa S. and Chronopoulos A. T. (2006). Price-based user-optimal job allocation scheme for grid systems, *Proc. of the 20th IEEE Intl. Parallel and Distributed Proc. Symposium*., Rhodes Island, Greece, April 25-29, 2006.

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

Penmatsa S. (Dec. 2010). Performance Optimization Algorithms for E-Commerce Systems Involving Selfish Agents: A Brief Survey. *Proceedings of the 2010 MTMI-NIT International Conference on Global Issues in Business & Technology*, ChungLi, Taiwan, Dec. 22-24, 2010.