

A Cluster-based Dynamic Load Balancing Middleware Protocol for Grids

Kayhan Erciyeş^{1,2} and Reşat Ümit Paylı³

¹ Izmir Institute of Technology
Computer Eng. Dept., Urla, Izmir 35430, Turkey

² California State University San Marcos,
Computer Science Dept. San Marcos CA 92096, U.S.A.
`kerciyes@csusm.edu`

³ Computational Fluid Dynamics Laboratory
Purdue School of Engineering and Technology
Indiana University-Purdue University Indianapolis
Indianapolis, Indiana 46202, U.S.A.
`rpayli@iupui.edu`

Abstract. We describe a hierarchical dynamic load balancing protocol for Grids. The Grid consists of clusters and each cluster is represented by a coordinator. Each coordinator first attempts to balance the load in its cluster and if this fails, communicates with the other coordinators to perform transfer or reception of load. This process is repeated periodically. We show the implementation and analyze the performance and scalability of the proposed protocol.

1 Introduction

Computational Grids consist of heterogeneous computational resources, possibly with different users, and provide them with remote access to these resources [1], [2], [3]. The Grid has attracted researchers as an alternative to supercomputers for high performance computing. One important advantage of Grid computing is the provision of resources to the users that are locally unavailable. Since there are multitude of resources in a Grid environment, convenient utilization of resources in a Grid provides improved overall system performance and decreased turn-around times for user jobs [4]. Users of the Grid submit jobs at random times. In such a system, some computers are heavily loaded while others have available processing capacity. The goal of a load balancing protocol is to transfer the load from heavily loaded machines to idle computers, hence balance the load at the computers and increase the overall system performance. Contemporary load balancing algorithms across multiple/distributed processor environments target the efficient utilization of a single resource and even for algorithms targeted towards multiple resource usage, achieving scalability may turn out difficult to overcome.

A major drawback in the search for load balancing algorithms across a Grid is the lack of scalability and the need to acquire system-wide knowledge by the

nodes of such a system to perform load balancing decisions. Scalability is an important requirement for Grids like NASA's Information Power Grid (IPG) [5]. Some algorithms have a central approach, yet others require acquisition of global system knowledge. Scheduling over a wide area network requires *transfer* and *location* policies. Transfer policies decide *when* to do the transfer [6] and this is typically based on some threshold value for the load. The location policy [7] decides where to send the load based on the system wide information. Location policies can be *sender initiated* [8] where heavily loaded nodes search for lightly loaded nodes, *receiver initiated* [9] in which case, lightly-loaded nodes search for senders or *symmetrical* where both senders and receivers search for partners [10].

We propose a protocol to perform load balancing in Grids dynamically where an ordinary node does not need to have a global system wide knowledge about the states of other nodes in the Grid. The protocol is semi-distributed due to the existence of local cluster center nodes called the *coordinators*. We show that the protocol designed is scalable and distributed as the coordinators communicate and synchronize asynchronously.

The paper is organized as follows: In section 2, the proposed protocol including the coordinator and the node algorithms is described with the analysis. In Section 3, the implementation of the protocol using an example is explained and Section 4 contains the concluding remarks along with discussions.

2 The Protocol

For load balancing in grids, we propose the architecture shown in Fig. 1 where nodes form clusters and each cluster is represented by a coordinator similar to [11]. Coordinators are the interface points for the nodes to the ring and perform load transfer decisions on behalf of the nodes in their clusters they represent. They check whether load can be balanced locally and if this is not possible, they search for potential receivers across the grid.

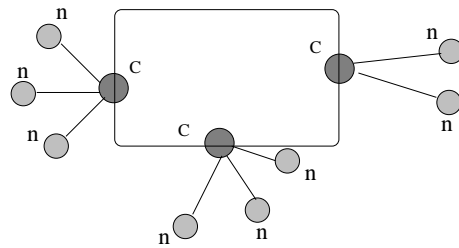


Fig. 1. The Load Balancing Model for a Grid

2.3 Analysis

Let us assume k , m , n and d are upperbounds on the number of clusters, nodes in a cluster in the network, nodes in the ring of coordinators and the diameter of a cluster respectively.

Theorem 1. *The total time for a load transfer is between $4dT + L$ and $(4 + k)dT + L$ where T is the average message transfer time between adjacent nodes and L is the actual average load transfer time.*

Proof. A node transfers its state to the coordinator in d steps in parallel with the other nodes and assuming there is a match of LOW-HIGH nodes in the local cluster, the coordinator will send *Coord_Xfer* message to the HIGH node in d steps. Then there will be L time for the actual load transfer. The HIGH and LOW nodes also perform a final handshake to confirm delivery of load in $2d$ steps. The total minimum time for load transfer is then the sum of all of these steps which is $4dT + L$. In the case of a remote receiver, the messages for load transfer have to pass through k hops resulting in $(4 + k)dT + L$ time.

Corollary 1. *The total number of messages exchanged for load transfer is $O(k)$*

Proof. As shown by Theorem 1, the maximum total number of messages required for a remote receiver will be $(4 + k)d$. Assuming d is approximately unity, the message complexity of the algorithm is $O(k)$.

Corollary 2. *The protocol described achieves an order of magnitude reduction in the number of messages exchanged for load transfer with respect to a similar protocol that does not use clusters.*

Proof. Assuming $k=m$, that is, the maximum number of clusters in the Grid equals the maximum number of nodes in a cluster, the total number of messages exchanged would be in the order of $O(k^2)$ for a similar protocol that does not use any hierarchical cluster structure. Therefore, the number of messages using our approach provides an order of magnitude decrease in the number of messages transferred.

3 An Example Operation

An example operation of the model is depicted in Fig.5. The following are the sequence of events :

1. All of the nodes in clusters 1, 2 and 3 inform their load states to their cluster coordinators C_1 , C_2 and C_3 . There are no LOW nodes in Cluster 1, there is one HIGH and one LOW node in Cluster 2 and 1 LOW and two MED nodes in Cluster 3. This is shown in Fig.5(a).
2. The coordinator for Cluster 1, C_1 , forms a request message *Xfer_Req* and sends it to the next coordinator on the ring, C_2 .

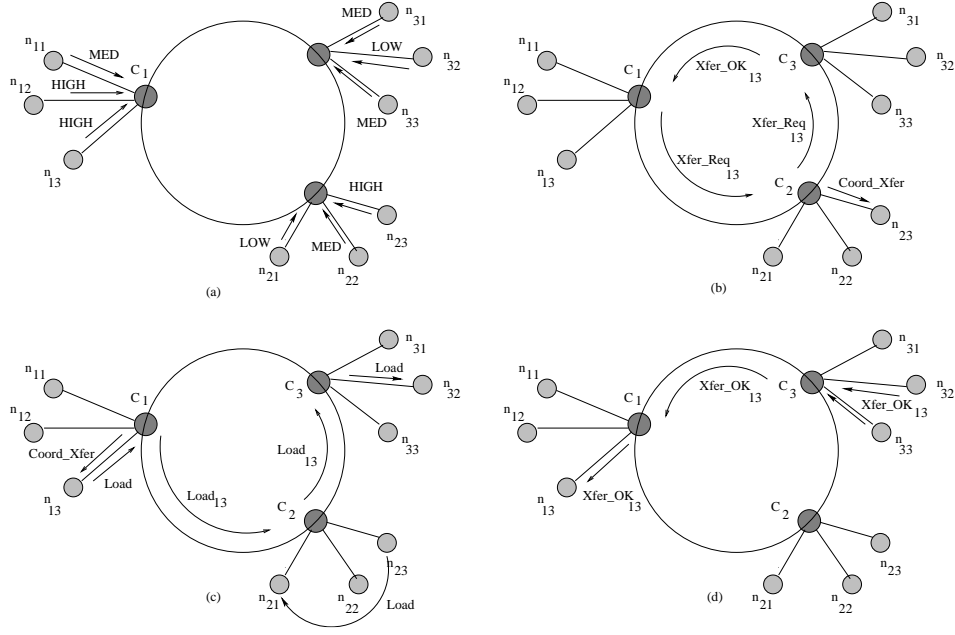


Fig. 5. An Example Operation of the Protocol

3. C_2 has a local match between its two nodes (n_{23} and n_{21}) and has no other receivers. It therefore passes the message immediately to its successor C_3 . It also sends *Coord_Xfer* message to n_{21} to initiate local load transfer.
4. C_3 has a potential receiver (n_{32}) which has reported LOW load. It therefore replies by changing the message *Xfer_Req* to *Xfer_OK* and sends this to C_1 . Steps 2,3 and 4 are shown in Fig.5(b).
5. C_1 receives the reply message for R_{13} and sends the *Coord_Xfer* message to n_{13} which then sends its load to C_1 .
6. C_1 now transfers the load to C_3 which transfers the load to n_{32} . Steps 5 and 6 are shown in Fig.5(c).
7. n_{32} sends *Xfer_ACK* message to its coordinator C_3 which passes this to C_1 which then forwards it to n_{13} . This step is shown in Fig.5(d).

4 Discussions and Conclusions

We proposed a framework and a protocol to perform dynamic load balancing in Grids. The Grid is partitioned into a number of clusters and each cluster first tries to balance the load locally and if this is not possible, a search for potential receivers is performed across the Grid using a sender-initiated method. We showed that the proposed protocol is scalable and has significant gains in the number of messages and the time perform load transfer.

The coordinators have an important role and they may fail. New coordinators may be elected and any failed node member can be excluded from the cluster. The recovery procedures can be implemented using algorithms as in [12] which is not discussed here. Our work is ongoing and we are looking into implementing the proposed structure in a Grid with various load simulations. Another reserach direction would be the investigation of the proposed model for real-time load balancing across the Grid.

References

1. I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the Grid: Enabling scalable virtual organizations. *Intl. Journal of High Performance Computing Applications*, 15(3), (2001), 200–222
2. Foster, I., What is the Grid ? A Three point Checklist, *Grid Today*, (1)6, (2002)
3. Foster, I., Kesselman, C., eds., *The Grid: Blueprint for a New Computing Infrastructure*, MorganKaufmann, San Fransisco, CA 1999.
4. Arora, M., Das, S., K. : A De-centralized Scheduling and Load Balancing Algorithm for Heterogeneous Grid Environments, *Proc. of Int. Conf. Parallel Processing Workshops*, (2002).
5. Johnston, W. E., Gannon, D., Nitzberg, B. : Grids as Production Computing Environments : The Engineering Aspects of NASA's Information Power Grid. *Proc. 8th Int. Sym. High Performance Distributed Computing*, (1999), 197-204.
6. Eager, D. L., Lazowska, E. D., Zahorjan, J., A Comparison of Receiver-initiated and Sender-initiated Adaptive Load Sharing, *Performance Evaluation*, 6(1), (1986), 53-68.
7. Kumar, V. , Garma, A., Rao, V. : Scalable Load Balancing Techniques for Parallel Computers, *Journal of Parallel and Distributed Computing*, (1994), 22(1), 60-79.
8. Liu, J., Saletore, V. A., Self-scheduling on Distributed memory Machines, *Proc. of Supercomputing*, (1993), 814-823.
9. Lin, H., Raghavendra : A Dynamic Load-balancing Policy with a Central Job Dispatcher, *IEEE Trans. on Software Engineering*, 18(2), (1992), 148-158.
10. Feng, Y., Li, D., Wu, H., Zhang, Y. : A Dynamic Load Balancing Algorithm based on Distributed Database System, *Proc. 8th Int. Conf. High Performance Computing in the Asia-Pacific Region*, (2000), 949-952.
11. Erciyes, K, Marshall, G., A Cluster Based Hierarchical Routing Protocol for Mobile Networks, *SV Lecture Notes in Computer Science*, 2004, ICCSA(3):518-527.
12. Tunali, T, Erciyes,K., Soysert, Z., A Hierarchical Fault-Tolerant Ring Protocol For A Distributed Real-Time System, *Special issue of Parallel and Distributed Computing Practices on Parallel and Distributed Real-Time Systems*, 2(1), 2000, 33-44