INTERNATIONAL JOURNAL OF PURE AND APPLIED RESEARCH IN ENGINEERING AND TECHNOLOGY

A PATH FOR HORIZING YOUR INNOVATIVE WORK

UTILITY BASED LOAD BALANCING WITH GRID COMPUTING APPROACH

PROF. A. D. ISALKAR, PROF. S. T. KHANDARE

1. Department of Computer Science & Engg. Scholar, BNCOE, Pusad, India
2. Associate Professor, Department of Computer Science & Engg., BNCOE, Pusad, India

Accepted Date:

27/02/2013

Publish Date:

01/04/2013

Keywords

Grid,

Utility,

Computing,

ULB,

Load.

Corresponding Author

Mr. A. D. Isalkar

Abstract

Grid Computing is a high performance computing environment that allows sharing of geographically distributed resources across multiple administrative domains and used to solve large scale computational demands. In the grid environment, users can access the resources transparently without knowing where they are physically located. The commonly used load balancing policies implicitly shared resources on demand basis, giving more resources to the application that has a high demand and fewer resources to the application that has a low demand. However, a higher demand for resources does not always correlate with a higher performance from additional resources. It is beneficial for performance to invest the resources in the application that benefits more from resources rather than in the application that has more demand for the resources. This paper proposes Utility-Based Load Balancing Algorithm (ULB), dynamic, a low-overhead, runtime mechanism that balances the load among shared resources between multiple applications depending utility that each application is likely to obtain for a given amount of resources. The proposed mechanism monitors each application at runtime using a novel, cost-effective, hardware circuit. The information collected by the monitoring circuits is used by a load balancing algorithm to decide the amount of resources allocated to each application.

INTRODUCTION

As the number of nodes on a Grid increases, the pressure on the resource system to sustain the resource requirements of all the concurrently executing applications (or threads) increases. One of the keys to obtaining high performance from such architectures is to manage the load on same Grid efficiently so that resources from other Grids are reduced. This paper investigates the problem of balancing the load among shared resources [1].

A few static load balancing techniques are Round robin algorithm, Randomized algorithm, simulated annealing or genetic algorithms, and Dynamic load balancing algorithms make changes to the distribution of work among workstations at run-time; they use current or recent load information when making distribution decisions. Multicomputer with utility-based load balancing allocate/reallocate resources at runtime based on utility, which may determine when and whose tasks can be migrated. As a result, utility-based load balancing algorithms can provide a major improvement in performance over static algorithms. However, this comes at the additional cost of collecting and maintaining load information, so it is important to keep these overheads within reasonable limits [2,3,4].

To balance the load based on application's utility for the resource, we propose Utility-Based Load Balancing Algorithm (ULB). An important component of ULB is the monitoring circuits that can obtain the information about utility of resource for all the competing applications at runtime. For the ULB scheme to be practical, it is important that the utility circuit (UCKT) have low overhead. The information collected by UCKT is used by a

ULB algorithm to decide the amount of resources allocated to each competing application.

2. Background and Motivation

The load balancing problem is closely related to scheduling and resource allocation. It is concerned with all techniques allowing an evenly distribution of the workload among the available resources in a system. To minimize the time needed to perform all tasks, the workload has to be evenly distributed over all nodes which are based on their processing capabilities. This is why load balancing is needed. The main objective of a load balancing consists primarily to optimize the average response time of applications; this often means the maintenance the workload proportionally equivalent on the whole system resources. Load balancing is usually described as either load balancing or load sharing. Figure 1 shows the Job Migration.

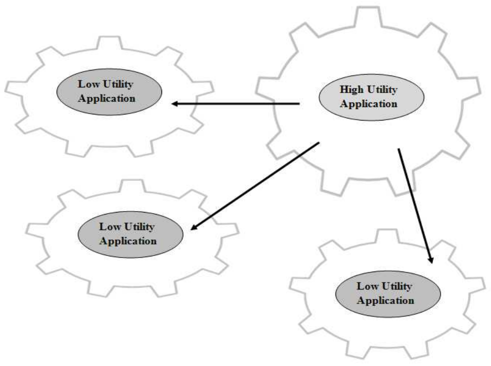

Figure 1: Application migration for Load Balancing over Grid

Load balancing improve performance by reducing the sharing of resources among the executing applications. Thus, the utility of resources for an application can be directly correlated improvement in performance of the application when the resource is shared. The utility of resource varies widely across applications. The applications are classified into three categories based on how much each of them benefits as the resource is increased. The applications which requires more resources is have low utility, whereas applications which requires less resources have high utility and others have saturating utility. If two applications having low utility are executed together on Grid, then their performance is not sensitive to the amount of resource available to each application. Similarly, when an application with high utility is run with any other application, its performance is highly sensitive to the amount of resource available to it. In such cases, it is important to balance the resources judiciously by taking utility information into account.

3. Utility-Based Load Balancing (ULB)

3.1 Framework

Figure 2 shows the framework to support ULB between two applications that execute simultaneously. One of the two applications execute on Grid1 and the other on Grid2. Each Grid is assigned a utility circuit (UCKT) that tracks the utility information of the application executing on it. The UCKT circuit is separated from the shared resource, which allows the UCKT circuit to obtain utility information about an application for all the ways in the Grid, independent of the contention from the application executing on the other Grid. The Load Balancing algorithm uses the information collected by the UCKT to decide the load balancing for each Grid. The migration engine of the shared resource is augmented to support the migration allocated by the Load Balancing algorithm.

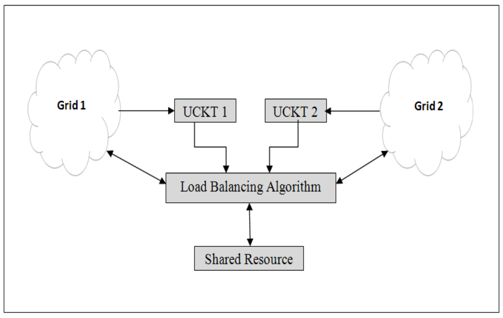

Figure 2: Framework of Utility Based Load Balancing technique.

3.2 Utility Circuits (UCKT)

Determining the utility information of an application requires a mechanism that tracks the number of resources required by each computing application. To compute the utility information for the baseline, the utility circuit is required to track resource information. A straight-forward, but expensive, method to obtain this information is to have resource requirement. The utility information from Grid is provided to Load Balancing algorithm which makes decision for balancing the load on demand basis.

3.3 Load Balancing Algorithm

The Load Balancing algorithm reads the utility information from all the UCKT circuits of each of the competing applications. The Load Balancing algorithm tries to minimize the load incurred by all the applications. The utility information in the UCKT directly correlates with the managing load for a given application. Thus, reducing the most balance is equivalent to maximizing the combined utility. If A and B are two applications with utility functions UA and UB respectively, then for load balancing decisions, the combined utility (U) of A and B is computed for all possible Grids (N).

$U_{tot} = UA_i^1 + UB^{(N-i)}_1$ … For i = 1 to N-1

The partition that gives the maximum value for U is selected. In our studies, we guarantee that the load balancing algorithm gives at least resource to each application, so starvation problem can be reduce. We invoke the balancing algorithm once every five million cycles (a design choice based on simulation results). After each balancing interval, the counters in all UCKTs are halved. This allows the UCKT to retain past information while giving importance to recent information.

3.4 Algorithm

Algorithm : Greedy Approach

```
balance = N /* Num resource to be allocated */
allocations[i] = 0 for each competing application i
while(balance) do:
    foreach application i, do: /* get utility for next 1 block */
        alloc = allocations[i]
        $U_{next}[i]$ = get_util_value(i, alloc, alloc+1)
    winner = application with maximum value of $U_{next}$
    allocations[winner]++
    balance = balance-1 return allocations
get_util_value(p, a, b):
    U = change in performance for application p when the number of resource assigned to it increases from a to b return U
```

4. Analysis

4. 1 Performance Metrics

There are several metrics to quantify the performance of a system in which multiple applications execute concurrently. We discuss the three metrics commonly used in the literature: weighted speedup, sum of IPCs, and harmonic mean of normalized IPCs for each computing application Let IPC be the IPC of the ith application when it concurrently executes with other applications and $SingleIPC_i$ be the IPC of the same application when it executes in isolation. Then, for a system in which N threads execute concurrently, the three metrics are given by:

$$\text{Weighted Speedup} = \sum \frac{IPCi}{SingleIPCi} \quad . . . (2)$$

$$IPC_{sum} = IPC_i \quad . . . (3)$$

$$\text{IPCnorm\_hmean} = N/\sum\left(SingleIPCi/IPCi\right)$$

The *Weighted Speedup* metric indicates reduction in execution time. The IPC metric indicates the throughput of the system but it can be unfair to a low IPC application. The IPC_{sum} metric balances both fairness and performance [9]. We will use *Weighted Speedup* as the metric for quantifying the performance of multicomputer configurations throughout the paper. Evaluation with the IPC metric will also be discussed for some of the key results in the paper.

5. Related Work

5.1. Related work in Load Balancing

Stone et al. [16] investigated optimal (static) load balancing of resources between two or more applications when the information about change in resource requirement for varying resource size is available for each of the competing application. However, such information is hard to obtain statically for all applications as it may depend on the input set of the application. The objective of our study is to dynamically balance the load by computing this information at runtime. Moreover, dynamic load balancing can adapt to the time-varying phase behavior of the competing applications, which makes it possible for dynamic load balancing to out perform even the best static load balancing [13].

Dynamic load balancing of shared resource was first investigated by Suh et al. [17][18]. Describes a low-overhead scheme that uses recency position of the resource for the lines in the Grid to estimate the utility of the cache for each application. However, obtaining the utility information from each application has the following shortcomings: (1) the number of lines in each set for which the utility information can be obtained for a given application is also dependent on the other application. (2) The recency position at which the application gets a resource is also affected by the other application, which means that the utility information computed for an application is dependent on (and polluted by) the concurrently executing application. ULB avoids these problems by separating the utility circuit from the main resource so that the utility information of the application is independent of other concurrently executing applications.

5.2 Related work in resource allocation

In the Grid systems domain, Zhou et al. [19] looked at resource allocation for competing applications using *miss ratio curve*. The objective of both their study and our study is the same, however, their study deals with the allocation of physical resource, whereas, our study deals with the allocation of resource for load balancing. The hardware solution proposed in [19] stores an extra tag entry for each resource in a separate hardware structure for each competing application.

6. Conclusion

Traditional designs for a shared resource balancing use static lad balancing scheme among competing applications on a demand basis. The application that accesses more unique resource in a given interval gets more resource than an application that accesses less unique resource in that interval. However, the benefit (reduction in resource) those applications get for a given amount of resources may not correlate with the demand. This paper proposes *Utility-Based Load Balancing (ULB)* to divide the resource among competing applications based on the benefit (utility) of resource for each application and makes the following contributions:

1. It proposes a low hardware overhead, utility circuit to estimate the utility of the resources for each application.

2. It proposes the Greedy Algorithm, as a scalable alternative to evaluating every possible load balancing decisions when there are a large number of applications sharing a resource.

REFERENCES

1. [Kumar,2011] U. Karthick Kumar "A Dynamic Load Balancing Algorithm in Computational Grid Using Fair Scheduling ", International Journal Of Computer Science and Informatics, Vol. 8, Issue 5, No 1, September 2011 ISSN (Online): 1694-0814.

2. [Vivekanandan & Ramyachitra, 2011] Dr. K. Vivekanandan, D.Ramyachitra, "A Study on Scheduling in Grid Environment", International Journal on Computer Science and Engineering (IJCSE), ISSN : 0975-3397 Vol. 3 No. 2 Feb 2011.

3. [Kumar & Singhal,2012] Sachin Kumar Niraj Singhal, " A Priority based Dynamic Load Balancing Approach in a Grid based Distributed Computing Network", International Journal of Computer Applications (0975 – 8887), Volume 49– No.5, July 2012

4. [Kaur & Singh, 2012] Pawandeep Kaur, Harshpreet Singh, "Adaptive dynamic load balancing in grid computing an approach," International journal of engineering science & advanced technology , ISSN: 2250–3676 Volume-2, Issue-3, 625 – 632, May-Jun 2012.

5. [Gogulan & Kavitha,2012] R.Gogulan, A.Kavitha, U.Karthick Kumar, "Max Min fair scheduling algorithm using in grid scheduling with load balancing", International Journal of Research in Computer Science, ISSN 2249-8265 Volume 2 Issue 3 (2012) pp. 41-49.

6. [Srivastava, 2011] Prabhat Kr. Srivastava, "Improving Performance in Load Balancing Problem on the Grid Computing System", International Journal of Computer Applications (0975 – 8887), Volume 16– No.1, February 2011

7. [Zheng,Tham &Veeravalli, 2010]Qin Zheng Chen-Khong Tham Bharadwaj Veeravalli, "Dynamic Load Balancing and Pricing Dynamic Load Balancing and Pricing in Grid Computing with Communication Delay, November 2, 2010

8. [Engelen ,2008] Prof. Robert van Engelen , "Concepts and Architecture Of

Grid Computing”, Advanced Topics Spring 2008.

9. [Yagoubi & Slimani, 2007] Belabbas Yagoubi and Yahya Slimani, “Dynamic Load Balancing Strategy for Grid Computing,” Journal of Information Technology and Applications ,Vol. 1 No. 4 Mar ch,2007, pp. 285-296

10. [Yagoubi & Slimani, 2006] Belabbas Yagoubi and Yahya Slimani, “Dynamic Load Balancing Strategy for Grid Computing,” World Academy of Science, Engineering and Technology 19 2006.

11. [Jacob & brown, 2005] Bart Jacob, Michael Brown, Kentaro Fukui, Nihar Trivedi ,“An Introduction to grid computing”, IBM/Redbooks December 2005.

12. [Dobber, Koole & van der Mei ,2004] Menno Dobber, Ger Koole, and Rob van der Mei, “Dynamic Load Balancing for a Grid Application”,pp. 342–352, 2004.

13. [Chang, 2004] Prof. Ruay-Shiung Chang , “Grid Architecture”, March 2004.

14. [Jacob , 2003] Bart Jacob , “Grid computing: What are the key components? Taking advantage of Grid computing for application enablement”, ITSO Redbooks Project Leader, IBM June 2003.

15. [Buyya ,Abramson & Giddy] Rajkumar Buyya, David Abramson, and Jonathan Giddy, “ A Case for Economy Grid Architecture for Service Oriented Grid Computing”.

16. “Introduction to Grid Computing “,The Globus Project™ Argonne National Laboratory USC Information Sciences Institute.

17. [Malik,2000] Shahzad Malik, “ Dynamic Load Balancing in a Network of Workstations”, November 29,2000

18. [Kokkinos & Christodoulopoulos]P. Kokkinos, K. Christodoulopoulos, N. Doulamis, and E. Varvarigos, “Quality of Service Scheduling of Computation and Communication Resources in Grid Networks”.

19. [Buyya ,Abramson & Giddy] Rajkumar Buyya, David Abramson, and Jonathan

Giddy, " A Case for Economy Grid Architecture for Service Oriented Grid Computing".