Design and development of a Scalable Multiprocessor Architecture

[1]Ambreen Ahmad, [2]M. Q. Rafiq
Aligarh Muslim University, Aligarh, 202002, U.P, India
[1]ambreenah@gmail.com, [2]mqrafiq@hotmail.com

ABSTRACT : *The goal of load balancing is to assign to each node a number of tasks proportional to its performance. Load balancing techniques play a very important role in developing high performance cluster computing platforms. Many load balancing polices achieve high system performance by increasing the utilization of CPU, memory, or a combination of CPU and memory. In this work, load balancing scheme for the Linearly Extensible Tree (LET) and Linearly Extensible Triangle (LEÄ) have been proposed that modifies the Minimum Distance Scheduling (MDS) reducing Load Imbalance Factor (LIF) and also reduce response time of parallel jobs.*

General Terms : *Load balancing, time, Linearly Extensible Tree, Linearly Extensible Triangle, Load Imbalance Factor.*

Keywords : *Load Balancing, LET, LEÄ, LIF, MDS*

1. INTRODUCTION

Multiprocessing is the simultaneous execution of task on parallel asynchronous computer system. There are two basic multiprocessor models: shared-memory and message passing system. The shared memory model provides a globally shared physical address space, which is highly desirable from the programmer's point of view. However, simultaneous access to shared memory in many processors complicates the design of such systems. In contrast message-passing model provides memory associated with each processor and data between processors is passed through messages [6].

There are number of techniques and methodologies for scheduling processes of a distributed system. These are task assignment, load-balancing, load-sharing approaches. In task assignment approach, each process is viewed as a collection of related tasks and these tasks are scheduled to suitable nodes so as to improve performance. In load sharing approach simply attempts to assure that no node is idle while processes wait for being processed. In load balancing approach, processes are distributed among the nodes of the system so as to equalize the workload among the nodes at any point of time.

Load balancing strategies may be static or dynamic [1,3]. In static scheduling, the assignment of the tasks to the nodes is done before the execution of the program. A task is always executed on the node to which it is assigned. Dynamic scheduling is based on the re-distribution of processes among the processors during execution time. This redistribution is performed by transferring tasks from heavily-loaded processors to lightly-loaded processors with an aim to minimize the processing time of the application. The flexibility inherent in dynamic load balancing allows for adaptation to unforeseen application requirements at run-time. In general, load-balancing algorithms can be broadly categorized as centralized or decentralized, dynamic or static, periodic or non-periodic, and those with thresholds or without thresholds [3].

The rest of the paper is organized as follows. Section 2 discusses the related work. In section 3 we refer to the various multiprocessor interconnection networks. Section 4 describes the proposed load balancing strategy for LET and LEÄ network. We have simulated the behavior of load balancing algorithms for these network architectures. Section 5 and 6 are the result and discussion, and conclusion. Section 7 contains the references.

2. RELATED WORK

There are so many variants involved, which are responsible for optimizing the mapping and load balancing and hence finally improve the performance of the multiprocessor system. The difficulty of solution varies with inclusion or exclusion of pre-emption, network topology, the number of parallel processors, cardinality, communication overhead, precedence constraints etc [6]. Various metrics for comparing the load balancing algorithms have been identified in [1]. It also discusses the components of dynamic load balancing algorithms and various dynamic load balancing algorithms. The algorithm adopted for load balancing is closely related to the type of network, number of nodes, number and weight of links which connect the nodes and job size. In order to balance the load uniformly over a grid one has to choose a mix of centralized, decentralized, sender- initiated and receiver- initiated approach. Communication overhead and load balancing time depend upon the approach selected in the algorithm. [3] Considers a cluster computing platform of heterogeneous system in which a set of N nodes are connected via a high speed network. Each node in this model composed of a combination of various resources including processor, memory, disk, network connectivity. Here, a load manager or master node is responsible for load balancing and monitoring available resources of node. [2] Carries out an overview of a six node multiprocessor server, Linearly Extensible Cube, to achieve both load balancing and downloading information efficiently. It implements a new proposed algorithm on the server which uses store and forward like technique that reduces the resource download time. [4] Carries out the study and comparison of six load balancing algorithms, various parameters are used to check the results. It concludes that static load balancing algorithms are more stable in comparison to dynamic and it is also easy to predict the behavior of static, but the dynamic distributed algorithms are always considered better than static algorithms. [5] Concludes that the load balancing algorithm developed leans on a structure of data of network type WAN, what guarantees its portability on any grid computing. The distribution of loads indeed assures the convergence of the algorithm in acceptable time.

3. MULTIPROCESSOR INTERCONNECTION NETWORKS

3.1 Network Characteristics

Some important characteristics of a multistage interconnection network are its mode of operation, switching technique, routing technique and interconnection network.

Operation modes: there are two basic modes of network operation; synchronous and asynchronous. In the synchronous mode, the network is centrally supervised, connection paths established simultaneously. In the asynchronous mode, connection paths are setup or disconnected on an individual basis. The asynchronous mode is of operation is more appropriate for multiprocessor system.

Switching techniques: there are three basic switching techniques: circuit switching, packet switching and wormhole switching. Circuit switching sets up the switches and ports and establishes a dedicated path between an input-output pair, efficient for large transmissions. Packet switching refers to a technique in which messages between any two terminals are broken into several shorter, fixed-length packets which are routed independently to their destination using store- and- forward procedures. Compared with circuit switching, packet switching is efficient for shorter and more frequent transmissions.

Routing techniques: it is the method of establishing communication paths and resolving conflicts. Three basic routing techniques have been considered: centralized, distributive and adaptive. In the centralized routing, a central control makes all the logic decisions needed to set up communication paths. This scheme is more flexible for small to medium scale systems. In the distributed scheme, logical decisions are made locally based on the current conditions. In the adaptive scheme, information about the network is collected globally, but routing decisions are made locally.

3.2 Interconnection Network Topology

The network topology is the way in which the switches are interconnected. The topology is perhaps the most important factor in determining network performance [6].

Number of Nodes: the number of nodes affects the complexity of the system and hence the cost of the system.

Degree of Node: it is the number of connections required at each node. It determines the complexity of the network, so it should be as low as possible.

Diameter: it is the measure of the maximum internodes distance in the network. It determines the distance involved in communication and hence the performance of multiprocessor system.

Extensibility: it is a property, which facilitates constructing large sized systems out of small ones with minimum changes in the configurations of the nodes.

Fault Tolerance: in multiprocessor network if one or more components fail then it should work correctly although perhaps with reduced performance.

4. PROPOSED ALGORITHM FOR LOAD BALANCING

4.1 Linearly Extensible Tree (LET)

The LET network combines linear extensibility with small number of processing elements per extension. In LET network the number of nodes at level j is (j+1) [8] as shown in Figure 1.

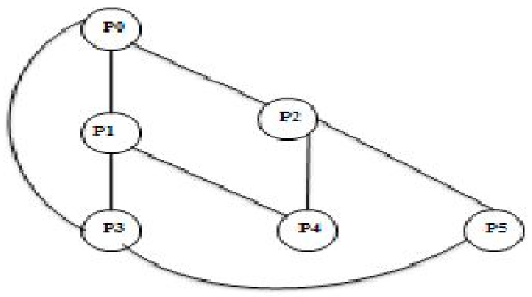

Fig 1: LET Network with 6 Processors

Load balancing starts out at the node with smallest degree i.e. node which has minimum number of links. The load balancing algorithm is as shown in Figure 2.

```
1.  N = Number of processors
    L = Number of tasks
    iL = Ideal Load
    n = Number of levels
    k = Number of processors at level j
    Number of processors at level j is ( j + 1 )
    $Q = [ P_0 , P_1 , \dots , P_{N-1} ]$
    $N=S^{n}_{k=1} k$
    $iL = L/N$
    Z = Set of processors with minimum degree which are allocated iL load*
    $Z = [ P_{n-i} : i = 2, 3 , \dots , n ]$
    rem_l = load* supplied to the ( 2 * | Z | ) links
    $rem\_l = L - S^{n}_{i=2} P_{N-i}$
    Repeat for (n–1) times to obtain W1: where (n–1) is the number of levels to be iterated and W1 is the set of processors on which load balancing is to be performed next rl = rem_l / ( 2 * | Z | )
    $X1=[P¢_{n-1+j} : j = 1, 2, \dots , n]$: Set of processors assigned rl
    $X2=[P¢ ¢_{n-1+j} : j = 1, 2, \dots , n]$: Set of processors in the next subsequent level
     $P¢ ¢_{n-1+j} = P¢_{n-1+j} + iL$
    rem_l = rl
    end
```

```
$W1 = [ P¢_{0} , P¢_{N-n} , P¢_{N} ]$
$P_{0} = P¢_{0} - \{ (2 / ( N/2 ) )* P¢_{0} \}$
$P¢ ¢_{N-n} = P¢_{n-N} + \{ (2 / ( N/2 ) )* P¢_{0} \}$
$P_{N-n} = P¢ ¢_{N-n} - \{ (1 / ( N/2 ) )* P¢ ¢_{N-n} \}$
$P_{N} = P¢_{N} + \{ (1 / ( N/2 ) )* P¢_{N-n} \}$

Repeat till LIF can be minimized no further
Find the node $P_m$ with maximum load: $P_m = iL$
Assign $P_m$ ideal load $iL$ and distribute remaining load rl equally on the $j$ links from $P_m$
$rl = P_m - iL$
$M [ P_i : i=1, 2, \dots , j ]$ : set of processors attached to $P_m$
$P_i = P_i + (rl / j)$
end
*load refers to the number of tasks
```

Fig. 2 Load Balancing Algorithm for LET

4.2 Linearly Extensible Triangle (LEÄ)

This triangle-based multiprocessor network has concept of simple geometry and its interconnections topology exhibits the properties of linearly extensible multi processor architecture [6]. An LEÄ network with four processors is shown in Figure 3.

Load balancing starts at the (N-1)th node. The load balancing algorithm for LEÄ network is shown in Figure 4.

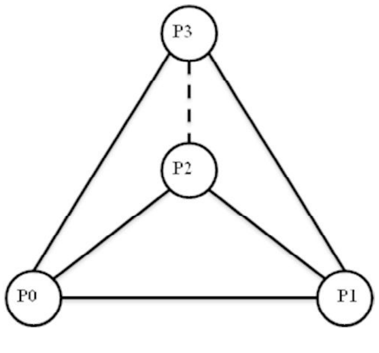

Fig. 3 LEÄ Network with 4 Processors

```
$N$ = Number of processors
$L$ = Number of tasks
$iL$ = Ideal Load
$n$ = Number of levels
$Q = [ P_0 , P_1 , \dots , P_{N-1} ]$
$N= k + 1 ; k >= 3$
$iL = L/N$
The root node is assigned ideal load $iL$
$Q[P0] = iL$
$S1$ = Set of processors assigned $iL$ load* to obtain
$P_0$ and $P¢_{z+j}$
$S1 = [ P_0 , P_{z+j} : z = (k + i ) ; i = 1, 2, \dots$ and $j = 1, 2 ]$

Repeat for $m = 1, 2, \dots , (N-1)-2$ times to obtain
$P¢_{N-2}$ and $P¢_{N-1}$
$l¢$ = load on $P_m$
$P_m = l¢ / (k + i) ; i = 0, 1, 2, \dots$
$rem\_l = l¢ - P_m$
$l¢ = rem\_l$
end

$W1 = [ P¢_{N-2} , P¢_{N-1} ]$
$P_{N-2} = P¢_{N-2} - ( P¢_{N-2} / 2)$
$P_{N-1} = P¢_{N-1} + ( P¢_{N-2} / 2)$

*load refers to the number of tasks
```

Fig. 4 Load Balancing Algorithm for LEÄ

The above stated algorithms were implemented in Matlab and the curves obtained were compared with various other algorithms for load balancing. In Figure 5 the results of the algorithm for LET are plotted along with results for Gradient Method (GM), Hierarchical Balancing Method (HBM), Minimum Distance Scheduling (MDS), Two Round Scheduling (TRS) and the following results were obtained.

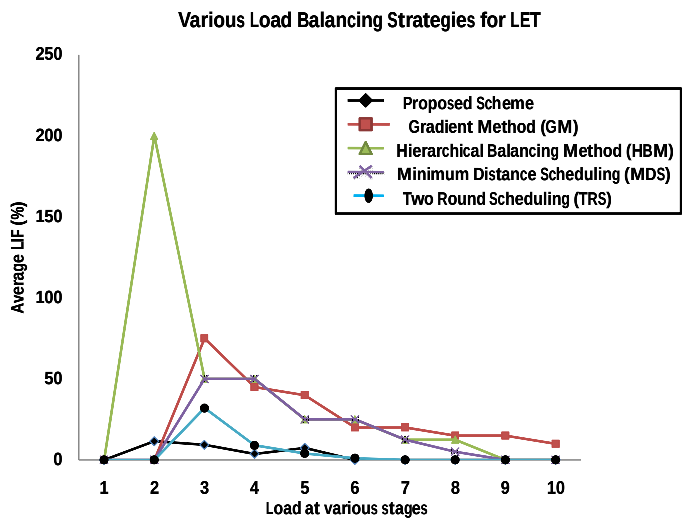

Fig. 5 Comparison of Various Load Balancing Strategies for LET

In this Figure, the load at various stages of load balancing for an LET network is plotted against the average LIF (in %). It can be seen that the proposed scheme results in lesser LIF at each stage in comparison to most of the other schemes. Figure 6 shows that it also takes less time for balancing than most of the other schemes.

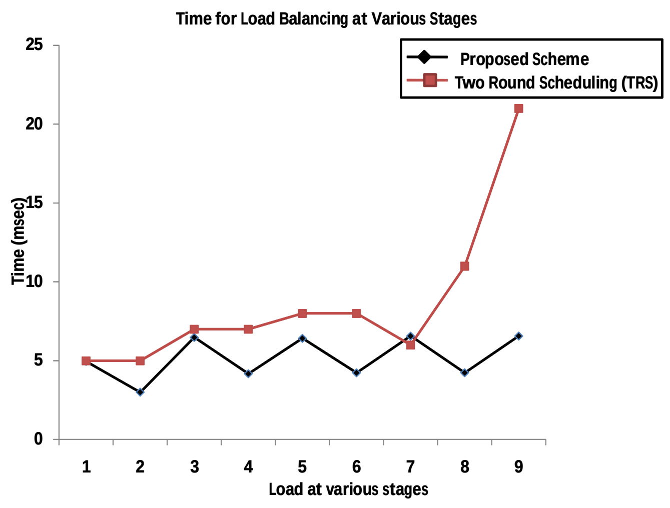

Fig. 6 Time for various Load Balancing Strategies for LET

Another algorithm was implemented for the LEÄ network and the results obtained are shown in Figure 7 and Figure 8. Figure 7 shows the average LIF (%) at various stages of load balancing plotted against load at various stages. It can be seen that in comparison to the LET network, LEÄ network gives better results. The average LIF (%) at various stages shows a regular pattern i.e. it remains constant from the next stage after load balancing starts. In case of LET, average LIF (%) reaches zero after certain number of stages. Figure 8 shows the time (in milliseconds) required for balancing a particular number of tasks in LET and LEÄ networks.

6. CONCLUSION

From the above results and discussion it is shown that an efficient scalable algorithm for load balancing in the LET and LEÄ networks has been designed. It also reduces the response time of tasks running in parallel. It also reduces the Load Imbalance Factor (LIF) to less than 30 %.

7. REFERENCES

[1] Janhavi B., Sunil Surve, Sapna Prabhu, "*Comparison of Load Balancing Algorithms in a Grid*", *2010 International Conference on Data Storage and Data Engineering*, pp 20-23, 2010.

[2] Abdus Samad, M. Q. Rafiq and Omar Farooq, "A Novel Algorithm for Fast Retrieval of Information from a Multi processor Server", *7th WSEAS Int'l Conf. on SOFTWARE ENGINEERING, PARALLEL AND DISTRIBUTED SYSTEMS (SEPADS '08)*, University of Cambridge, UK, pp 68-73, 2008.

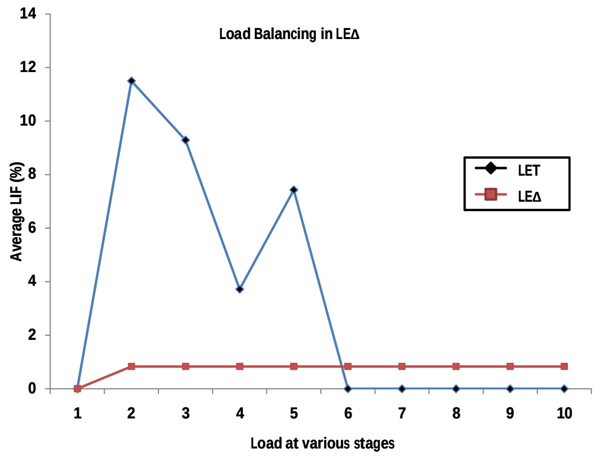

Fig. 7 Load Balancing Strategy for LEÄ

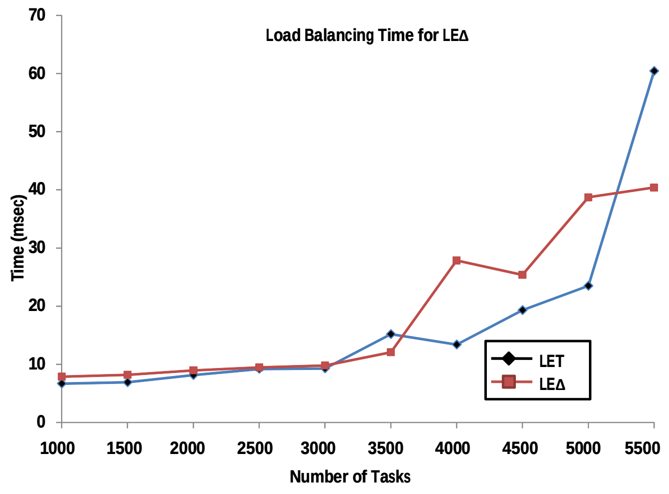

Fig. 8 Time for Load Balancing Strategies for LET and LEÄ

[3] Chandra, Pushpendra Kumar, Sahoo, Bibhudatta, "Dynamic Load Distribution Algorithm Performance in Heterogeneous Distributed System for I/O Intensive Task", *TENCON 2008 - 2008, TENCON 2008, IEEE Region 10 Conference,* pp 1-5, 2008.

[4] Sandeep Sharma, Sarabjit Singh, and Meenakshi Sharma, "Performance Analysis of Load Balancing Algorithms", *World Academy of Science, Engineering and Technology 38,* pp 269-271, 2008.

[5] Abdallah Boukerram, Samira Ait Kaci Azzou, "Implementation of Load Balancing Algorithm in a Grid Computing", *American Journal of Applied Sciences*, 2006.

[6] Manaullah, "Performance Evaluation of Multiprocessor Architectures", *Ph.D. thesis,* Jamia Millia Islamia, 2002.

[7] Abdus Samad, "Performance Evaluation of Linearly Extensible Multiprocessor Archi-tectures for Networking", *Ph.D. thesis,* Aligarh Muslim University, 2009

[8] M. Q. Rafiq, "Studies on the Performance Evaluation of a Linearly Extensible Multiprocessor Network", *Ph.D. thesis,* Univ. of Roorkee, 1995.

[9] D. Acker, S. Kulkarni, "A Dynamic Load Dispersion Algorithm for Load-Balancing in a Heterogeneous Grid System", *Sarnoff Symposium IEEE*, pp 1- 5, 2007.

[10] A. Chhabra, G. Singh, "Qualitative Parametric Comparison of Load Balancing Algorithms in Distributed Computing Environment", *14th International Conference on Advanced Computing and Communication, IEEE,* pp 58 – 61, 2006