Fast Retrieval of Information from Server by Developing Novel Multiprocessor Architecture

Mukul Varshney
Sharda University, Gr. Noida, India

Anand Sharma
Manav Bharti University, Solan, India

Ankita Varshney
NIET, Gr. Noida ,India

ABSTRACT

Achieving Parallelism is now a necessity to improve the performance of computer system. One of the main Issues is how to effectively utilized parallel computer that have become increasingly complex. It is estimated that many modern supercomputers and parallel processor deliver only 10 percent or less of their peak performance potential in a variety of applications. Yet high performance degradation are many. Performance losses occur because of mismatches among applications software and hardware. Research is active in the direction of developing new multiprocessor architectures and schedule the partitioned program on to it to achieve higher performance. [1, 3, 4]

In this paper, effort concentrated on the Study of all factors for developing a novel multiprocessor Architecture and after developing we will evaluate the performance of a multiprocessor architecture for server and networking with simulation study and to schedule the arriving load on to it in order to achieve higher performance. The other important issue are accessing delay and downloading the information while using multiprocessor technique. comparable with similar architecture. In addition to designing an appropriate network, the efficient management of parallelism on the network involves optimizing performance needs , like the minimization of communication and scheduling over head. A simulation studies are carried out to compare the performance of different multiprocessor architecture (such as LEC, LET, Hypercube, Debruinju etc) with the various standard dynamic scheduling algorithms like Sender initiated diffusion (SID), receiver initiated diffusion(RID) etc .The simulation result will show that our Multiprocessor architecture (linearly extensible triangle) gives better performance as compare to other existing multiprocessor architecture with low cost and reduce the load balancing time and scheduling over head.

1. INTRODUCTION

In past some years, the internet has become a necessity of every human life in any field. And internet has been a popular source of data access day by day. On the other hand user's expectations have increased in terms of the data should be downloaded in the very shortest possible time. For this many multiprocessor system has been developed for accessing data from the internet or through the local database.

Exploiting parallelism is now a necessity to improve the performance of computer systems One of the most important issues is how to effectively utilized parallel computers that have become increasingly complex to improve the performance. Such systems are constructed by different processor connected with communication link to operate in parallel with relatively low cost known as multi processor system[3,5].

The parallel computer is one of the remarkable developments of methodology and technology in computer science in recent years. Due to multiprocessor structure of the computer architecture, this computer has a capability to execute multiple instructions or multiple data simultaneously. The parallel computer not only provides support for efficient computation of mathematical, economical, industrial, and ecological problems but also aims new computer architecture beyond the traditional von Neumann type.

Multiprocessor system be very efficient at solving problems that can be partitioned into tasks with uniform computation and commun- ication patterns. However, there exists a large class of nonuniform problems with uneven and unpredictable computation and communication requirements. Therefore Dynamic load balan-ceing (DLB) schemes[1,2,7] are needed to efficiently solve non-uniform problems on multiprocessor systems.

The evaluation of multiprocessor architecture has been influenced by several factors such as

(i) speedup (ii) scalability and (iii) flexibility.

In this paper we evaluate the performance of different multiprocessor architecture on the basis two factors (i) load imbalance factor and (ii) load balancing time using different standard dynamic load balancing strategies.

Load balancing involves assigning the work of each processor in such a manner so that we minimize the execution time of program and increase the performance. Static load balancing strategies perform by predetermined policy without consideration of status of the system. On the other hands dynamic load balancing strategies makes the load balancing in accordance of status of the system such as no. of jobs , arrival rate of jobs etc.

So In this paper a simulation study are carried out to compare the performance of different multiprocessor architecture (such as LEC, LET, HYPERCUBE, DEBRUINJU etc). And schedule the arriving load on different architecture so that user can access data in the possible shortest time.

So we use different dynamic scheduling strategies for simulation study such as sender initiated diffusion, receiver initiated diffusion, hierarchal load balancing methods etc.

2. RELATED WORK

A. Different Multiprocesor Architecture

Over the year, many different multiprocessor architecture, having different topological structure and different interconnection, have been used in different commercially available parallel systems. An enormous amount of research has centred on the design and their interconnection. Major architecture are found in ring network ,hypercube, debruijn network, LET and LEC network.[3,4,8](Fig 1 to Fig 6).A number of multiprocessor architecture have been reported in literature.[6,7,8]

In case of internet, a web server is used to store the data and retrieve the data. A web (URL) request comes to switch or router, responsible for routing the request to the web server according to scheduling strategies responsible for reducing the scheduling over head and load balancing time. Basically scheduling overhead deals with the processing and communication overhead both. The load balancing overhead includes the communication cost of getting load information and cost of informing to the overload processor, processing cost of calculating load information to load transfer.

For achieving this various dynamic load balancing strategies has been proposed by number of researchers. In this paper we simulate these dynamic load balancing strategies onto different multiprocessor architecture having different properties.

Schedule the arriving load on different architecture so that user can access data in the possible shortest time.

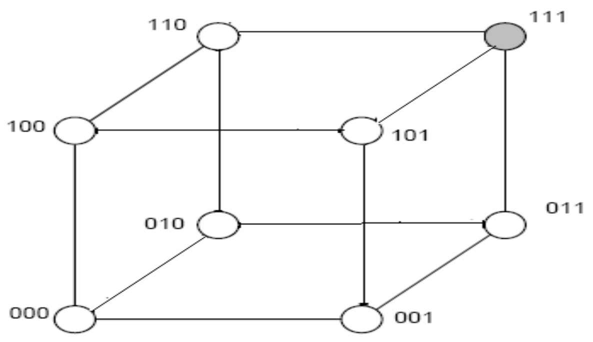


Fig. 1. Hypercube architecture

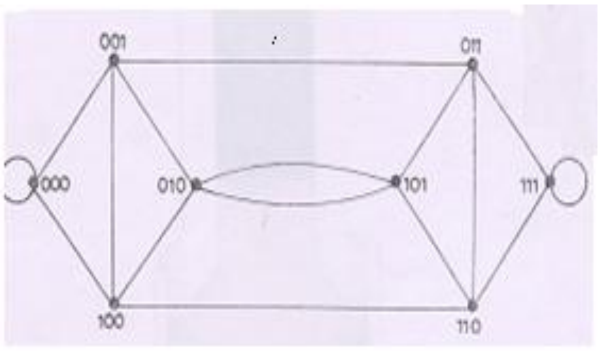


Fig. 2. Debruijn architecture

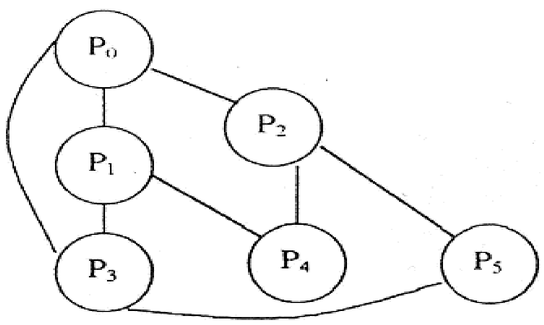


Fig. 3. Linearly extensible tree

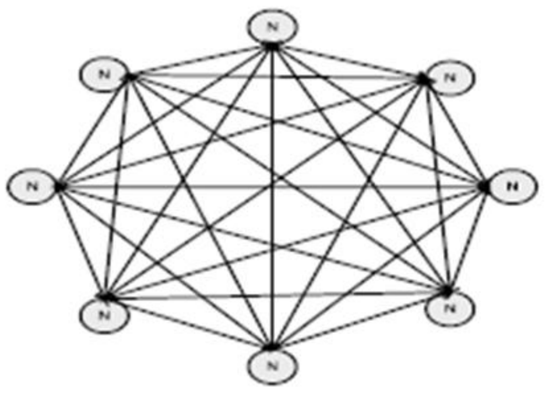


Fig. 4. Completely connected architecture

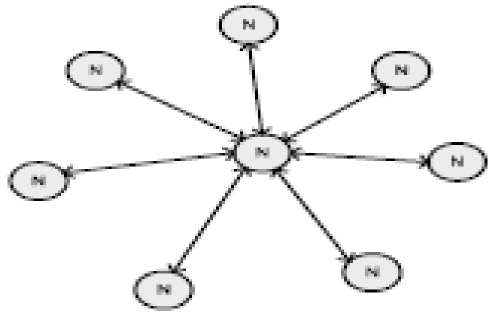


Fig. 5. Star connected architecture

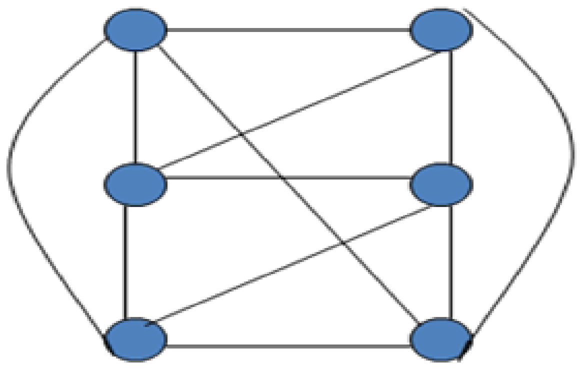


Fig. 6. Linearly extensible cube

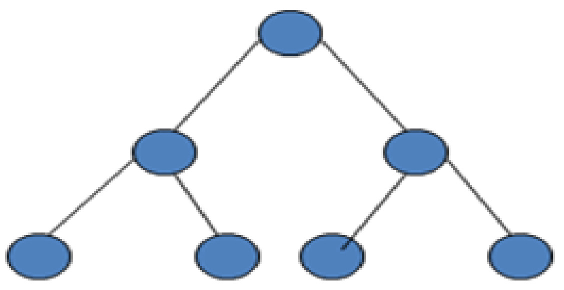


Fig. 7. Binary tree of depth two

B. Properties of These Architecture and their Comparision

1) Number of Node: It specify total No. of processing element in the multiprocessor architecture If no. of node increases the complexity as well as cost also increases whereas performance is decreases. So no of node should be minimum so that our system is less complex [1].

Note: Standard existing multiprocessor

architecture has maximum N=8

nodes (processors) (fig 1).

2) Degree of Node: It specify number of connections in each node .Higher the degree more complex the system. It is best if the degree or connectivity on each node is constant and independent of the network size implies that the processor organization scales more easily to systems with large no. of nodes.

Note Degree [1] should not be greater than four. [Deg $\ngtr$ 4]

3) Diameter: It is the measure of Maximum Inter-node distance in a network means the largest distance between two nodes.

It specify the communication delay between nodes as the size increases the diameter will also be increases.

Low diameter is better because low the diameter the communication delay between nodes is less.

4) Expandability: It is the property which facilitates constructing the large size system out of small one system with minimum change in the configuration of the architecture. It should be linear not exponential.

5) Scalability: It specify the ability of network to be modularly expandable with minimum change in the configuration [8].

6) Bisection Width: It specify minimum No. edge that must be removed to divide the network into two halves. High bisection width is better because

In algorithm requiring large amount of data movement, the size of the data set divided by the bisection width puts a lower bound on the complexity of the parallel system [1].

7) Complexity: Complexity is directly proportional to no of node and deg. of node in the network. Less complex system gives the higher performance. So less no. of node ,lesser the complexity of the network [7].

C. Characteristics of Existing Architecture (comparative study)

TABLE I

COMPARISION OF PARAMETER

Network	No. of Link	Node degree.	Diameter	Bisection Width
Star	N-1	N-1	2	N-1
Completely connected	N(N-1)/2	N-1	1	$(N/2)^2$ $\sqrt{N}$
Binary Tree	N-1	3	$2(\text{Log}_2N - 1)$	1
Illiac Mesh	2N	4	$\sqrt{N} - 1$	$2\sqrt{N}$
Hypercube	$N\text{Log}_2N/2$	N	Log_2N	N/2
De-bruijn	2N-1	4	Log_2N	N/Log_2N
LET	N+2	4	$\sqrt{N}$	2
LEC	$(N/2)^2 + 3$	4	N	N

D. Limitation on Speedup of Multi-Processor System

There is some limitation on enhancing the speed of processing on multiprocessor system in terms of

- Interprocessor communication
- Synchronization
- Load Balancing

1) Interprocessor Communication: Whenever one processor generates (computes) a value that is needed by the fraction of the program running on another processor,that value must be communicated to the processors that need it, which takes time [4].

On a uniprocessor system,entire program runs on one processor, so there is no time lost to interprocessor communication

2) Synchronization: It is often necessary to synchronize the processors to ensure that they have all completed some phase of the program before any processor begins working on the next phase of the program [3,6].

3) Load Balancing: In many parallel applications, difficult to divide the program across the processors.

When each processor working the same amount of time not possible, some of the processors complete their tasks early and are then idle waiting for the others to finish

3. DYNAMIC LOAD BALANCING ALGORITHM

We have developed a general model for dynamic load balancing.

This model is organized as a four phase process:

- Processor load evaluation
- Load balancing profitability Determination
- Task Transfer strategy
- Task selection strategy

The following DLB strategies are designed to support highly parallel systems.

- Sender Initiated Diffusion (SID)
- Receiver Initiated Diffusion (RID)
- Hierarchical Balancing Method (HBM)

A. *Sender Initiated Diffusion (SID)*

The SID strategy is a, local, near-neighbor *diffusion* approach which employs overlapping balancing domains to achieve global balancing. for an *N* processor system with a total system load *L*, a diffusion approach, such as the SID strategy, will cause each processor's load to converge to *L/N.* [1,7,8]

Balancing is performed by each processor whenever it receives a load update message from a neighbor indicating that the neighbors load, 1_i<Ideal Load , where *Ideal Load* is a preset threshold. Each processor is limited to load information from within its own domain, which consists of itself and its immediate neighbors.

B. *Receiver Initiated Diffusion (RID)*

- First, the balancing process is initiated by any processor whose load drops below a pre specified threshold ***(L_{Low}) . [7,8]***
- Second, upon receipt of a load request, a processor will fulfill the request only up to an amount equal to half of its current load
- The RID strategy differs from its counterpart **SID** in the task migration phase. Here, an underloaded processor first sends out requests for load and then receives acknowledgment for each request

C. *Hierarchical Balancing Method (HBM)*

- It is an asynchronous global, approach which organizes the system into a hierarchy of subsystems. [1,7]
- Load balancing is initiated at the lowest levels in the hierarchy with small subsets of processors and ascends to the highest level which encompasses the entire system.
- Specific processors are designated to control the balancing operations at different levels of the hierarchy.

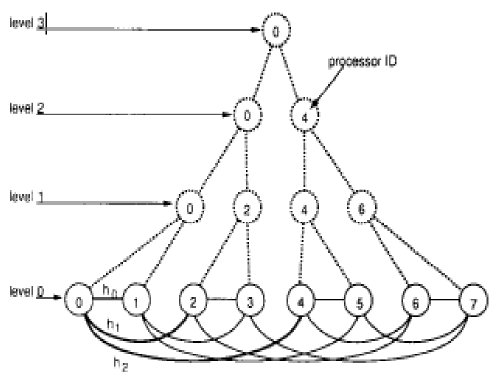


Fig. Hierarchical organization of an eight-processor system with hyper-cube interconnections, where h_k is the connection to the neighbor at the *k*th level.The processor IDs at intermediate nodes in the tree represent those processors delegated to manage the balancing of corresponding lower-level domains.

Fig. 7.

- The hierarchical balancing scheme functions asynchronously.
- The balancing process is triggered at different levels in the hierarchy by the receipt of load update messages indicating an imbalance between lower level domains.
- All load levels are initialized with each processor sending its load information up the tree.

4. RESULT & ANALYSIS (COMPARATIVE STUDY)

- In this work, We use Sender Initiated Diffusion(SID) dynamic load balance Strategy for performance evaluation of existing Architecture.
- We use two performance parameter for performance evaluation:
  - ➢ Load Imbalance Factor (LIF)
  - ➢ Load Balancing Time
- We use four existing multiprocessor architecture for comparative study such as:
  - ➢ Linearly Extensible Cube (LEC)
  - ➢ Linearly Extensible Tree (LET)
  - ➢ Hypercube
  - ➢ Debruijn
- LIF (Load Imbalance Factor) =

  [(max load on a processor after balancing –Ideal Load)/Ideal Load)]*100
- Ideal Load = Total load/ no. of processor
- Load Balancing Time = (Time after balancing – Time before balancing)

Table and Graph shown below in [Fig 8,9,10] represent Time taken by different multi-processor architecture when different sample of load is given to each architecture.

In our work we will develop a new architecture

Which have higher processing speed in comparison to existing architecture. And have low cost and reduces the access time of data.

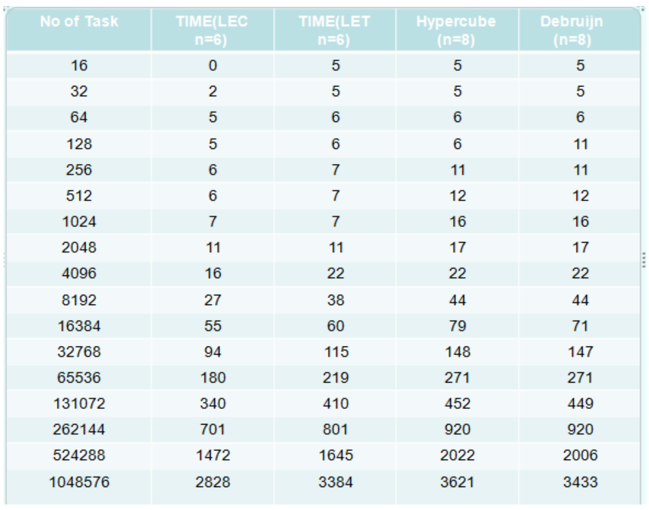
TABLE II

TASK VS LIF

No of task	LIF (LEC n=6)	LIF (LET n=6)	Hypercube (n=8)	Debrujin (n=8)
16	200	200	0	0
32	40	40	0	0
64	40	40	0	0
128	9.52	9.52	0	0
256	9.52	9.52	0	0
512	2.35	2.35	0	0
1024	2.35	2.35	0	0
2048	0.59	0.59	0	0
4096	0.59	0.59	0	0
8192	0.15	0.15	0	0
16384	0.15	0.15	0	0
32768	0.04	0.04	0	0
65536	0.04	0.04	0	0
131072	0.01	0.01	0	0
262144	0.01	0.01	0	0
524288	0	0	0	0
1048576	0	0	0	0

TABLE III

TASK VS TIME



No of Task	TIME(LEC n=6)	TIME(LET n=6)	Hypercube (n=8)	Debruijn (n=8)
16	0	5	5	5
32	2	5	5	5
64	5	6	6	6
128	5	6	6	11
256	6	7	11	11
512	6	7	12	12
1024	7	7	16	16
2048	11	11	17	17
4096	16	22	22	22
8192	27	38	44	44
16384	55	60	79	71
32768	94	115	148	147
65536	180	219	271	271
131072	340	410	452	449
262144	701	801	920	920
524288	1472	1645	2022	2006
1048576	2828	3384	3621	3433

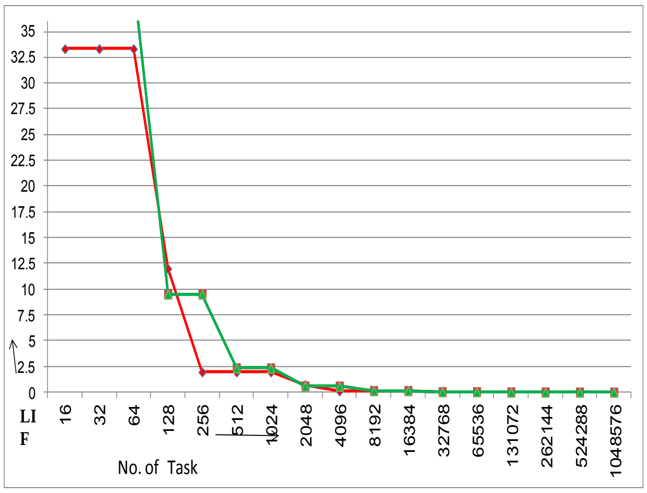


Fig. 8. Task Vs LIF

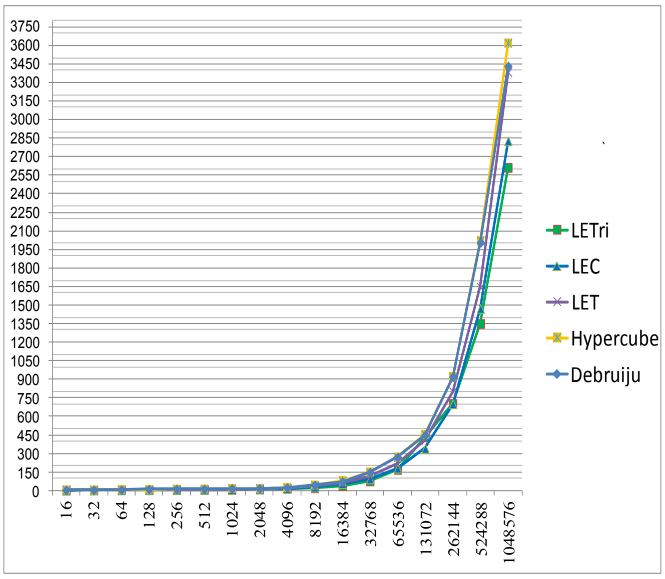


Fig 9: Task Vs Time

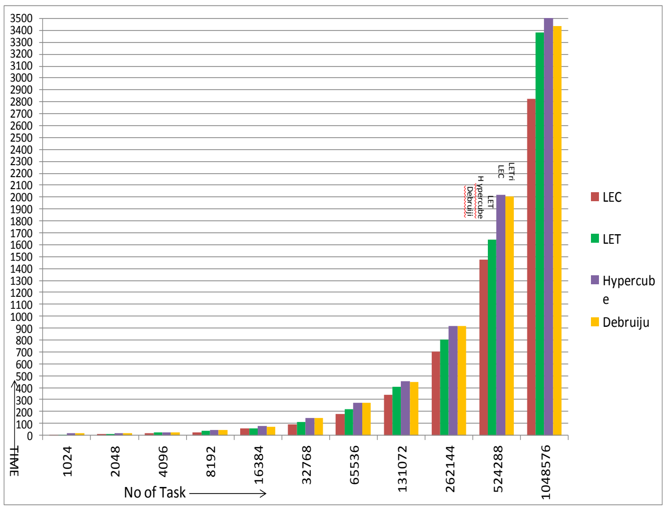


Fig. 10. Task Vs Time

5. FUTURE WORK

We will develop a new multiprocessor architecture and evaluate the performance of this architecture by simulation on the basis of some existing dynamic load balancing algorithm. This architecture reduces the access time for data and have higher speed and low cost. This architecture can be used for image processing, web search, data and web mining applications. Use for fast processing and developing very high speed system with low cost high speed.

6. REFERENCES

[1] A.samad, M.Q. Rafiq and Omar Farooq "A novel algorithim for fasT retrival of information from a multiprocessor server" Proc. Intl. on parallel and distributed system, U.K. Feb 20 to feb 22, 2008

[2] Z. Zeng and B.Veeravalli "Design and Performance of Queue and Rate Adjustment Dynamic load Balancing Polices for Distributed Network". IEEE trans. On computer, vol 55 ,no. 11, pp. 1410-1422,nov 2006.

[3] Abdel A E and Khaled d," The Hyperstar interconnection Network "journal of Parallel and distributed computing no. 48, pp 175-199,1998

[4] Nizadeh M., and Sarbazi-Azad, H., The necklace hypercube: a well scalable hypercube-based interconnection network for multiprocessors, ACM SAC 2005, pp.729-733, 2005.

[5] Razi-Azad H., Constraint-based performance comparison of multi-dimensional interconnection networks with deterministic and adaptive routing strategies, Journal of Computers and Electrical Engineering, vol. 30, pp.167-82, 2004.

[6] Meraji S., Sarbazi-Azad H., Nayebi A., Message routing and performance issues in necklace hypercubes, Technical Report, School of Computer Science, IPM, Tehran, Iran, 2006.

[7] Marc H. Willebeek-LeMair, Member, IEEE, and Anthony P. Reeves "Strategies for Dynamic Load Balancing on Highly Parallel Computers", Senior Member, IEEE, IEEE Transactions on Parallel and Distributed Systems, VOL. 4, NO. 9, September 1993

[8] Rafiq, M. Q.; Padam kumar and gupta J. P. "A Novel Tree –structured multiprocessor Network." Proc. Int'l conf on robotics vision and parallel processing for automation, Malaysia, VOL. 2 , pp 576,585,1999