International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:9, No:3, 2015

Performance Evaluation of Task Scheduling Algorithm on LCQ Network

Zaki Ahmad Khan, Jamshed Siddiqui, Abdus Samad

***Abstract*—**The Scheduling and mapping of tasks on a set of processors is considered as a critical problem in parallel and distributed computing system. This paper deals with the problem of dynamic scheduling on a special type of multiprocessor architecture known as Linear Crossed Cube (LCQ) network. This proposed multiprocessor is a hybrid network which combines the features of both linear types of architectures as well as cube based architectures. Two standard dynamic scheduling schemes namely Minimum Distance Scheduling (MDS) and Two Round Scheduling (TRS) schemes are implemented on the LCQ network. Parallel tasks are mapped and the imbalance of load is evaluated on different set of processors in LCQ network. The simulations results are evaluated and effort is made by means of through analysis of the results to obtain the best solution for the given network in term of load imbalance left and execution time. The other performance matrices like speedup and efficiency are also evaluated with the given dynamic algorithms.

***Keywords*—**Dynamic algorithm, Load imbalance, Mapping, Task scheduling.

I. Introduction

THE performance of parallel application running on multiprocessor system depends heavily on the mapping of tasks on a network of processor. This mapping of independent tasks is referred to as a task scheduling problem which plays a critical role to utilize the maximum benefits of parallel computing system. A parallel system without proper task scheduling algorithm may nullify the benefits of parallelization. Several researches have produced a number of algorithms to handle the problem of task assignment on multiprocessor systems [1]-[5]. Broadly the problem of task scheduling can be classified into two categories namely static task assignment and dynamic task assignment.

In static, the decision of task assignment takes place in advance whereas in dynamic task assignment algorithm the decision is taken on the fly, no prior knowledge is available. Static mapping does not involve overhead on the execution time, on the other hand dynamic mapping incurs more overload and are complex in nature. Dynamic task allocation is well advanced and can be applied to a number of real world applications [2]. Due to its key importance, the task assignment problem has been comprehensively studied and various methods have been reported in the literature [6]-[10].

Over the time, many scheduling policies were introduced which are designed to achieve their goals such as efficient utilization of process elements, minimization of resource idleness or decreasing the total execution time. Some techniques are specific to a particular type of multiprocessor architecture. These approaches are developed using different strategies such as Minimum Distance Strategy (MDS) [11], Hierarchical Balancing Method (HBM) [2], Two Round Scheduling Scheme (TRS) [12] and Multi-stage Scheduling Scheme [13]. There are algorithms which operate and optimize the task scheduling based on the prediction of process behavior. These algorithms consider the process behavior extraction, classification and prediction [14]. Iterative greedy approach is also a notable scheme to minimize the total execution time and communication cost [15]. The main idea in this algorithm is to improve the quality of the assignment in an iterative manner using results from previous iteration [15]. These schemes are applied on specific parallel system and the performance has not been extensively studied on a hybrid type of multiprocessor system. This paper is devoted to investigate the scheduling problem on a hybrid multiprocessor architecture [16]. The proposed network inherits the properties of cube based network as well as linear type of networks and named as Linear Crossed Cube (LCQ) network. It has smaller diameter, lesser number of nodes and complexity and linear extensibility. Two standard dynamic algorithms namely MDS and TRS algorithms which were designed originally for cube based multiprocessor networks are selected for implementation on the proposed LCQ network [11], [12]. Simulation results are evaluated and a comparative study based on various performance parameters is carried out on the results obtained by the algorithms.

The rest of the paper is organized as follows: Sections II describes the proposed multiprocessor architecture and its characteristics. In Section III, the dynamic algorithms are described. The simulation results are discussed in Section IV and the comparative study is made. Finally, the paper is concluded in Section V.

II. Proposed Multiprocessor Architecture and Its Characteristics

A. Linear Crossed Cube (LCQ)

The Proposed LCQ network is undirected graph and grows linearly in cube like shape. Let q be the set of designated processor of Q thus, $q = \{Pi\}$, $0 \leq i \leq N-1$. The Link functions E1 and E2 define the mapping from q to Q as.

Z. A. Khan is with the Aligarh Muslim University, Aligarh, India. He is now with the Department of Computer Science, Aligarh Muslim University, (Phone: +91-7417780086; e-mail: jmi.amu1@gmail.com).

J. Siddiqui is with the Aligarh Muslim University, Aligarh, India. He is now with the Department of Computer Science, Aligarh Muslim University, (e-mail: jamshed_faiza@rediffmail.com).

A. Samad is with the Aligarh Muslim University, Aligarh, India. He is now with University Women's Polytechnic, Aligarh Muslim University, (e-mail: abdussamadamu@gmail.com).

$$E1(Pi) = P(i+2) \text{ ModN} \; ; \; \forall \text{ Pi in q}$$
$$E2 (Pi) = P(i+3) \text{ ModN}$$

The two functions E1 and E2 indicate the links between various processors in the network.

Let Z be a set of N identical processors, represented as

$$Z= \{ P0, P1, P2..... P_{N-1}\}$$

The Total number of processor in the network is given by

$$N = \sum_{k=1}^{n} K,$$

where n is the depth of the network. For different depth, network having 1, 3, 6, 10, 15, 21 ... processors are possible.

In order to define the link functions, we denote each processor in a set K as P_{in}, n being the level/depth in LCQ where, the processor P_i resides. As per the LCQ extension policy, one or two processors exist at level/ depth n. Thus at level 1, P_0 and P_1 exit and similarly at level 2, P_2 and P_3 exist as shown in Fig. 1.

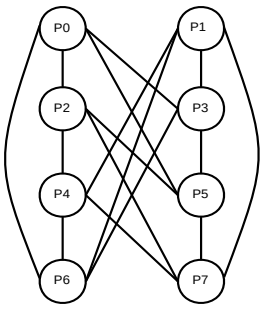

Fig. 1 Linear Crossed Cube with Eight Processors

B. Characteristics of LCQ

The following are the various topological properties of the LCQ network.

Number of Nodes: - The LCQ is an undirected graph, where the total number of nodes is:

$$N = \sum_{k=1}^{n} K \text{ e.g. } \{1, 3, 6, 15, 21.....\}$$

It shows that the network grows linearly, where $1 \leq K \leq n$, n is the level number up to which the network is designed.

Degree: - The degree of nodes is defined as the total number of edges (n-1) incident on each vertex. The degree of each vertex in the LCQ is remain constant i.e. 4 irrespective of the depth of network.

Diameter: - The diameter of network is the maximum eccentricity of any node in the network. It is the greatest distance between any pair of nodes. In LCQ, it is observed that the diameter does not always increase with the addition of a layer of processors. The diameter of LCQ is $\lfloor\sqrt{N}\rfloor$.

Cost of LCQ: - The cost of a network could be obtained as the product of the degree and diameter, hence for an LCQ the cost is equal to $4*(\lfloor\sqrt{N}\rfloor)$. Therefore, the cost is dependent on the value of diameter.

Extensibility: - The major advantage of proposed LCQ network is that its extension can be carried out in a linear fashion by adding one or two nodes in every extension. When single node is added, we call it odd extension and similarly an even extension can be made by adding two nodes in a particular extension. The important feature is that the proposed LCQ network does not have an exponential extension.

Among the better known architectures on which much work has been done in particular are HC, CQ and SCQ. The LEC is another architecture different from HC, however, possess some useful properties. Motivated by the properties of LEC, CQ and SCQ the proposed LCQ network has been designed. The above topological properties are analyzed with the above mentioned architectures. The values of various parameters are evaluated mathematically for comparison purpose. Table I gives a summary of various parameters for different type of multiprocessor networks.

TABLE I
SUMMARY OF VARIOUS PARAMETERS OF DIFFERENT MULTIPROCESSOR NETWORK

Parameter	HC	CQ	SC	SCQ	LEC	LCQ
Nodes	2^n	2^n	$n!2^m$	$n!2^m$	2n	$\sum K$
Diameter	n	n	$m+\lfloor 3(n-1)/2\rfloor$	$\lceil(m+1)/2\rceil + \lfloor 3(n-1)/2\rfloor$	$\lfloor n\rfloor$	$(\lfloor\sqrt{N}\rfloor)$
Degree	n	$\lceil n+1/2\rceil$	m+n-1	m+n-1	4	4
Cost	n^2	$n\lceil(n+1)/2\rceil$	$(m+n-1)(m+\lfloor 3(n-1)/2\rfloor)$	$(m+n-1)(\lceil(m+1)/2\rceil + \lfloor 3(n-1)/2\rfloor)$	$4\lfloor n\rfloor$	$4(\lfloor\sqrt{N}\rfloor)$

III. ANALYSIS OF VARIOUS EXISTING ALGORITHMS

The most common state-of-art techniques are based on reducing the communication overhead and consequently the total execution time. The goal is to maximize the overall CPU utilization while offering better performance by parallel execution of concurrent tasks. The performance of a parallel application running on a set of processors heavily depends on the mapping of tasks partitioned from the application onto the available processors in the system [15]. The performance evaluation can be made in term of comparisons metrics such as Load Imbalance Factor (LIF), execution time (ET), speed up (SP) and efficiency (E) of the parallel program execution on different set of processors [16]-[20]. The multiprocessor scheduling environment uses more than one processor to execute its processors. Therefore, the performance is evaluated and analyzed on different networks which consist of a number of identical processors or nodes. Though many approaches have been reported for mapping tasks on multiprocessors networks, we considered the two recently reported schemes which were originally designed for cube based system. These algorithms consider the basic approach of mapping and migration depending up on the level of connectivity. In the first approach the concept of minimum distance property has been incorporated which considers only the directly connected nodes among the network therefore named as minimum distance scheduling (MDS). The pseudo code for the algorithm is given in Fig. 2.

```
Assume the AL (Absolute Load), RAL (Round Absolute Load), PM (Total number of processors are given)
/* Generate the next level of task generation */
zs = = 2 * zs;
zs = = 3 * zs;
it_ count ++;
{
if (zs [it_count2] > RAL  {
/* migrate till load at nodes becomes equal to or less then RAL */
while (true) {
migrate (it_count2)
if (zs [it_count2] < = = RAL ) break;
} } }
lif = (M(zs) – AL) / AL;
} {
migrate (p_number)
temp[k++] =  =i;
k--;
}
```

Fig. 2 Pseudo Code of Minimum Scheduling (MDS)

Similarly, another algorithm considers the two level connectivity and named as Two Round Scheduling (TRS) [12]. The TRS scheme takes into consideration those acceptor nodes which are not connected directly to donor node. There may be more than one path between the donor and acceptor processors which require multi-hope. To maintain the efficiency of algorithm the TRS limits level of connectivity up to one intermediate node between donor and acceptor nodes. The Pseudo code for the TRS is going in Fig. 3.

```
/* Check the connectivity of node i with node j. Assume the level of connectivity is given (1 or 2)*/
int connected (int i, int j, int level) /* returns true if nodes i, j are connected */
{
  /* Display("\n node %d is connected to %d: %d", i, j, adj [i][j]); */
  if (level = = 1)
  return adj [i][j];
  for(int k = 0; k < no_proc; k++)
  {
    if(k = = i || k = = j) continue;
    if(connected (i ,k , 1) && connected (k, j, 1 ))
    {
   /* Display ("\n node %d is connected to %d through %d", i, j, k); */
   return 1;
    }
  }
  return 0;
```

Fig. 3 Pseudo Code of Two Round Scheduling (TRS)

A. Task Scheduling Model

The model of parallel system on which the task assignment is carried out consists of set of fully connected processors or nodes. There are no precedence relations between tasks and any task can be executed cost. The overall cost depends up on the mapping of application and the communication cost incurred in the network. In the proposed model the tasks are generated in a deterministic manner in the form of a regular tree. Each node of the tree represents a task, and executed in parallel in breadth-first manner starting from the root task which is assigned to some given nodes of the network. The total number of nodes in the task tree at level represents a particular stage of the load. We consider the two patterns of tree structure namely binary tree and ternary tree structure.

B. Performance Parameters

The performance analysis of task scheduling on the proposed network is carried out based on the fallowing parameters.

Load Imbalance factor (LIF):- The Load imbalance factor for K^{th} stages, denoted as LIF_k is defined as.

$$LIF_k = (\max \{load_k (P_i\} - \text{ideal-load}_k) / \text{ideal-load}_k,$$

where, $\text{ideal-load}_k = (load_k (P_0) + load_k (P_1) + \ldots load_k (P_{N-1})) /N$ and $\max(load_k (P_i))$ denotes the maximum load pertaining to stage K on a processor P_i, $0 \leq i \leq N-1$, and $load_k (P_i)$ stands for the load on processor P_i due to K^{th} stage. Each stage of the task structure (Load) represents a finite number of tasks.

Execution Time (ET):- The execution time of a given task is defined as the time spent by the system executing spent executing run-time or system services on its behalf. The execution time is the time during which a program is running (executing), in contrast to other phases of a program's lifecycle. The Execution time is used to calculate the runtime of scheduling algorithm which includes waiting time of job in queue and runtime in each resource.

Speedup (SP):- Speedup is the ratio of sequential execution time and parallel execution time.

$$SP = \frac{\text{Sequential Execution Time (SET)}}{\text{Parallel Execution Time (PET)}}$$

Efficiency (E):- The ratio of speedup and total number of processors (N) is called efficiency of a parallel program. Higher speedup or using lesser number of nodes makes the system efficient.

$$E = \frac{SP}{N}$$

IV. Result and Discussion

In this section the simulation results after implementing the dynamic task scheduling algorithm on LCQ network are discussed along with the comparative study of various results.

A. Simulation Results

In order to evaluate the performance of task scheduling on the proposed LCQ network the various parameters are evaluated. The criteria of an efficient scheduling algorithm depend up on the effective utilization of nodes by distributing tasks evenly. The effectiveness of node utilization could be measured in terms of LIF. The LIF represents the load imbalance left after a scheduling operation is performed on the available network. In the given work, the LIF's are computed with different algorithm by considering uniform task generation on a network of eight nodes. These results are shown in Figs. 4 and 5 by the curves plotted as LIF against the load stages.

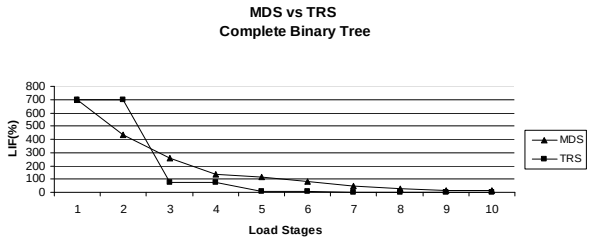

Fig. 4 Performance of MDS &TRS Scheme on LCQ Network

When MDS scheme is implemented on the LCQ network the task are scheduled with two types of tree type task structures. The mapping of task is performed at various levels of task structures and behavior is shown in the curves given Fig 4. It is clear from the curves that the LIF remains non zero even for higher stages of the load. On the other hand when TRS scheme is implemented the LIF is continuously reducing and become zero at 7th level of task structure.

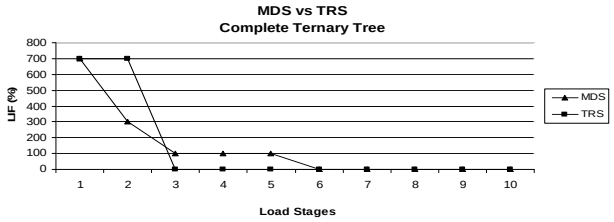

Fig. 5 Performance of MDS &TRS Scheme on LCQ Network

Similar results are evaluated with the same algorithm for a different type of task structure namely complete ternary tree and the curves are plotted and shown in Fig. 5. The results obtained indicate that both the scheduling are performing well when implemented on the given network, however, TRS scheme produced better performance as compare to MDS. In this case the performance of MDS is comparatively better as compare to binary task tree structure.

To further analyze the effectiveness of the scheduling algorithms we used other parameters like speedup and efficiency when parallel tasks are mapped on the LCQ network. Since the TRS algorithm performing well in terms of LIF we evaluated these parameters by implementing the TRS algorithm with two different type of task structure on different sets of processors. These results are given in Tables II and III. The results show that efficiency is not only dependent upon the number of nodes it also depends upon the type of task structure used. For instance, when task is generated evenly; the network with odd number of nodes results better efficiency and high speed up and vice-versa.

TABLE II
PERFORMANCE OF LCQ WITH BINARY TREE TASK STRUCTURE

Processor	Speedup	Efficiency
4	1.5	37.56
5	3.2	64.01
6	1.6	26.62
7	3.1	44.28
8	3.1	39.66

TABLE III
PERFORMANCE OF LCQ WITH TERNARY TREE TASK STRUCTURE

Processor	Speedup	Efficiency
4	1.6	40.00
5	1.6	32.00
6	1.6	26.66
7	1.5	21.42
8	1.5	20.00

V. COMPARATIVE STUDY

The comparative study is made on the basis of the minimum value of LIF obtained when MDS and TRS are implemented on the LCQ network with eight processors. The Results are shown in Figs. 4 and 5 which indicate that both the scheduling algorithms producing better results when sufficient number of task are available on the network. Similarly, the comparison is made on the maximum and minimum value of LIF with different sets of processor that's four, five, six, seven and eight processors. These results are shown in Figs. 6 and 7.

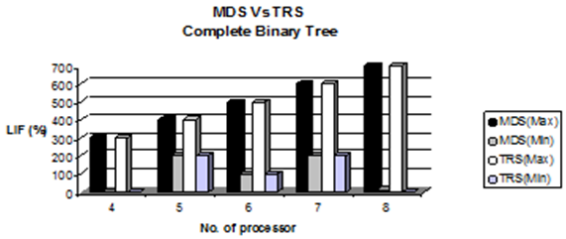

Fig. 6 Performance of MDS &TRS Scheme on LCQ Network

The maximum value of LIF is similar on LCQ network in both the algorithms on different sets of processor for binary type of task structures. However, The MDS scheme has non-zero values of LIF on LCQ with eight processors. Therefore, the MDS scheme is not performing better when numbers of processors are increased.

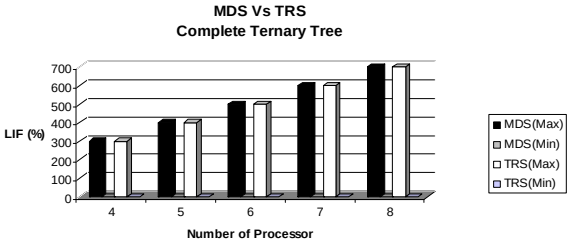

Fig. 7 Performance of MDS &TRS Scheme on LCQ Network

In case of complete ternary tree MDS and TRS are having same maximum and minimum value of LIF on four, five, six, seven and eight processors of LCQ multiprocessor network as depicted in Fig. 7.

As for as speedup and efficiency are concerned these parameters are evaluated keeping in mind the better performance of TRS. The results are evaluated by implementing TRS on different sets of processors of LCQ network. Table II shows that the maximum speed up and efficiency are obtained when TRS algorithm is implemented

with odd number of processors in LCQ network. However, these results are applicable only when complete binary tree task structure is taken into consideration. The results are changed when ternary tree task structure is considered. This feature is demonstrated in Table III.

VI. CONCLUSIONS

The overall performance of LCQ multiprocessor architecture is affected by a number of the factors, such as imbalance of load among the processor and scheduling overheads. To analyze the performance, two dynamic scheduling schemes Minimum Distance Scheduling (MDS) and Two Round Scheduling (TRS) are implemented on LCQ multiprocessor architecture. The performance is evaluated in terms of Load Imbalance Factor, Speedup and Efficiency. The comparative study shows that both the scheduling schemes are equally performing well on the LCQ network with lesser number of nodes. When number of nodes increase the TRS Scheme produces better results. The efficiency and speedup are not only dependent on the types of scheduling scheme, they are also depending on to the type of task pattern used. When the tasks are generated in binary pattern the algorithm producing better results when odd numbers of processors are considered. On the other hand, when task are generated in the multiple of three better efficiency is obtained with even number of processors by considering the appropriate task structure. The standard dynamic scheduling scheme can be applied to map the parallel task on the proposed LCQ network. In future, we intend to design to design the task structure independent algorithm for the proposed LCQ network.

REFERENCES

[1] I. Ahmad and A. Ghafoor, "Semi-Distributed Load Balancing for Massively Parallel Multicomputer Systems," IEEE Transactions on Software Engineering, vol. 17, no. 10, pp. 987-1004, 1991.
[2] M. H. W. LeMair and A. P. Reeves, "Strategies for dynamic load balancing on highly parallel computers," IEEE Transactions on Parallel and Distributed Systems, vol. 4, no. 9, pp. 979-993, 1993.
[3] M. J. Zaki, W. Li and S. Parthasarathy, "Customized Dynamic Load Balancing for a Network of Workstations," Journal of Parallel and Distributed Computing, no. 43, pp. 156-162, 1997.
[4] S. Sharma, S. Singh and M. Sharma, "Performance Analysis of Load Balancing Algorithms," in proceeding of World Academy of Science, Engineering and Technology, vol. 2 , pp. 02-21, 2008.
[5] Z. Zeng and B. Veeravalli, "Design and Performance Evaluation of Queue-and-Rate-Adjustment Dynamic Load Balancing Policies for Distributed Networks," IEEE Transactions on Computers, vol. 55, no. 11, pp. 1410-1422, 2006.
[6] K. Lakshmanan, D. D. Niz and R. Rajkumar, "Coordinated Task Scheduling, Allocation and Synchronization on Multiprocessors," in proceeding of 30th IEEE Real-Time Systems Symposium, pp. 469-478, 2009.
[7] A. Chandra and P. Shenoy, "Hierarchical Scheduling for Symmetric Multiprocessors," IEEE Transactions On Parallel And Distributed Systems, vol. 19, no. 3, pp. 418-431, 2008.
[8] J. Jia, B. Veeravalli and J. Weissman, "Scheduling Multiprocessor Divisible Loads on Arbitrary Networks," IEEE Transactions On Parallel And Distributed Systems, vol. 21, no. 4, pp. 520-531, 2010.
[9] M. Guzek, J. E. Pecero, B. Dorronsoro and P. Bouvry, "Multi-objective evolutionary algorithms for energy-aware scheduling on distributed computing systems," Applied Soft Computing, vol. 24, pp. 432–446, 2014.
[10] F. A. Omara and M. M. Arafa, "Genetic algorithms for task scheduling problem," Journal Parallel Distributed Computing, vol. 70, pp. 13–22, 2010.
[11] A. Samad, M. Q. Rafiq and O. Farooq, "A Novel Algorithm For Fast Retrival Of Information From A Multiprocessor Server," in proceeding of 7th WSEAS International Conference on software engineering, parallel and distributed systems (SEPADS '08), University of Cambridge, UK, pp. 68-73, 2008.
[12] A. Samad, M. Q. Rafiq and O. Farooq, "Two Round Scheduling (TRS) Scheme for Linearly Extensible Multiprocessor Systems," International Journal of Computer Applications, vol. 38, no. 10, pp. 34-40, 2012.
[13] A. Samad, M. Q. Rafiq and O. Farooq, "Multi-stage scheduling scheme for massively parallel systems," in proceeding of International Conference on Software Engineering and Mobile Application Modelling and Development (ICSEMA), pp. 168-176, 2012.
[14] E. Dodonov and R. F. d. Mello, "A novel approach for distributed application scheduling based on prediction of communication events," Future Generation Computer Systems, vol. 26, pp. 740–752, 2010.
[15] Q. Kang, H. He and H. Song, "Task assignment in heterogeneous computing systems using an effective iterated greedy algorithm," The Journal of Systems and Software, vol. 84, pp. 985–992, 2011.
[16] N. Rajak, A. Dixit and R. Rajak, "Classification of list task scheduling algorithms: A short review paper," Journal of Industrial and Intelligent Information, vol. 2, no. 4, pp. 320-323, 2014.
[17] R. Kaur and R. Kaur, "Multiprocessor scheduling using task duplication based scheduling algorithms: A review paper," International Journal of Application or Innovation in Engineering and Management, vol. 2, no. 4, pp. 311-317, 2013.
[18] R. Hwang, M. Gen and H. Katayama, "A comparison of multiprocessor task scheduling algorithms with communication costs," Computers and Operations Research, vol. 35, pp. 976-993, 2008.
[19] S. Bansal, B. Kothari and C. Hota, "Dynamic Task-Scheduling in Grid Computing using Prioritized Round Robin Algorithm," International Journal of Computer Science Issues, vol. 8, no. 2, pp. 472–477, 2011.
[20] Z. A. Khan, J. Siddiqui and A. Samad, "Linear Crossed Cube (LCQ): A New Interconnection Network Topology for Massively Parallel System," International Journal of Computer Network and Information Security, vol. 7, no. 3, pp. 18-25, 2015.

Prof. Jamshed Siddiqui received his Ph.D degree from IIT Roorkee, India. His research areas and special interests include Information Systems, MIS, Systems Analysis & Design, Knowledge Management Systems, E-Business, Data Mining and Parallel Computing. His areas of teaching interest includes Analysis and design of Information system, Software Engineering, Performance evaluation of computer systems, Computer oriented Numerical methods. He has published various papers in international journals and journals of international repute such as Journal of Information Technology, TQM Magazine, (Emerald Group Publishing Ltd.), Business Process Management Journal, (Emerald Group Publishing Ltd.), Journal of Information, Knowledge, and Management, Journal of Systems Management, International Journal of Services and Operations Management etc**.**

Dr. Abdus Samad received his Ph.D degree in Computer Engineering from Aligarh Muslim University, Aligarh, India in 2010. He completed Bsc. Engg and M.Tech. From Z. H. College of Engineering & Technology, Aligarh Muslim University, Aligarh in the year 1997 and 1999 respectively. The research areas are parallel and distributed systems, algorithm design, microprocessor and parallel system design. Contributed and attended various national and international conferences in India and abroad, and published papers in reputed journals. Member of various professional organizations. Presently working as Assistant Professor in Computer Engineering at University Women's Polytechnic, AMU, Aligarh and having teaching experience of more than 16 years.

Mr. Zaki A. Khan pursuing Ph.D from Aligarh Muslim University Aligarh, India. He recived Msc.Tech (industrial math's with computer Applications) and Bsc (H) Mathematics degree from Jamia Millia Islamia New Delhi, India in 2010 and 2007 respectively. The research areas are parallel and distributed systems, Scheduling Algorithm, Load Balancing, Information Retrieval, Multiprocessor System, and Green Computing. Contributed and attended

national and International conferences and workshops in India. He has published various research papers in reputed International journals. Member of CSTA, IAENG, IACSIT and UACEE.