Classification of Task Partitioning and Load Balancing Strategies in Distributed Parallel Computing Systems

Rafiqul Zaman Khan
(Associate Professor)

Javed Ali
(Research Scholar)

Department of Computer Science,Aligarh Muslim University,Aligarh.

ABSTRACT
Distribution of the tasks amongst the various computing nodes is itself an intellectually challenging problem in the high performance distributed computing systems. To choose the appropriate strategy for the required system is difficult without the meaningful comparison of the existing task partitioning and load balancing strategies. The effectiveness of the strategy depend on the number of factors-efficiency, interconnection topology, communication mode, program structure, throughput and computing capabilities of the structure. A number of task partitioning and load balancing strategies have been proposed, each of which perform remarkable results under different circumstances. The main goal of the paper is to unravel the mystery of strategies and to classify when and where each strategy is appropriate. In this paper, taxonomy of task partitioning and load balancing is presented in an attempt to provide a common terminology and classification mechanism.

KEYWORDS:Task Scheduling, Dynamic, Preemptive, Non-Preemptive, Parallel Computing etc.

1. INTRODUCTION
Scheduling is a function that assign job to the different processors [6]. It's a two step process, namely processor allocation and assignment. Jobs are autonomous program that execute in its domain. Resource allocation is done by the scheduler over two dimensions, time and space, and at two level jobs and threads. In running state, job constitutes threads to reduce the overhead. If software package is used to execute the parallel jobs instead of threads, it increased load of the parallel computing system. Interacting entities are called process if they are independent under the limitation of the operating system. Threads and communication between them may be static and dynamic [10].For example, in MIMD architecture [1], number of threads, and the communication pattern between the threads can change dynamically during the execution in the parallel computing systems. If multiple executing entities are parts of the same application then we treat entities as a threads and application as a job. Parallel computing is used to solve the large problems in the scientific and systematic manner. Effective task partitioning and load balancing strategies of large task is crucial to achieve high performance in parallel and distributed system. The increasing demand of the high performance computing systems amongst the various field of the science is the key feature of interest in the parallel computing. The selection of appropriate strategy for the particular system is the deciding factor for the successful execution of the tasks. A parallel job is the collection of the tasks having some precedence relationship. A task can be identified as a executable fragment that must be sequentially executed without partial parallel execution [2].All parallel job cannot be fully parallelized. In a homogeneous architecture the serial fraction of the computation has to be executed at the speed of any of the identical processor, limiting the speed that can be obtain due to parallel execution of the jobs.The sequential bottleneck can be greatly reduced by executing the tasks on the heterogeneous parallel computers by running the critical tasks on a faster processor. The efficiency analysis of the heterogeneity presented by Polychronopoulos and Andrews [4] and Almeida and Menasce [5]. Processor allocation in the efficient manner deals with the determination of the number of processor allocated to a job; the assignment phase means the distribution of the job's task to the allocated processor [7]. Time complexity analysis tool for the task partitioning developed by Pugh and Nirkhe [8] which accurately estimates the execution time of general real time program employing on high level structures. Towsley and Nelson[9] present a analytical model for partitioning the independent tasks on the different processors.Due to the large variety of tools, computing architectures and different requirement there is no ideal task partitioning strategy. Impact of heterogeneity in loosely coupled systems analyzed by Freund in different processing paradigms (MIMD,SIMD,SISDetc.).

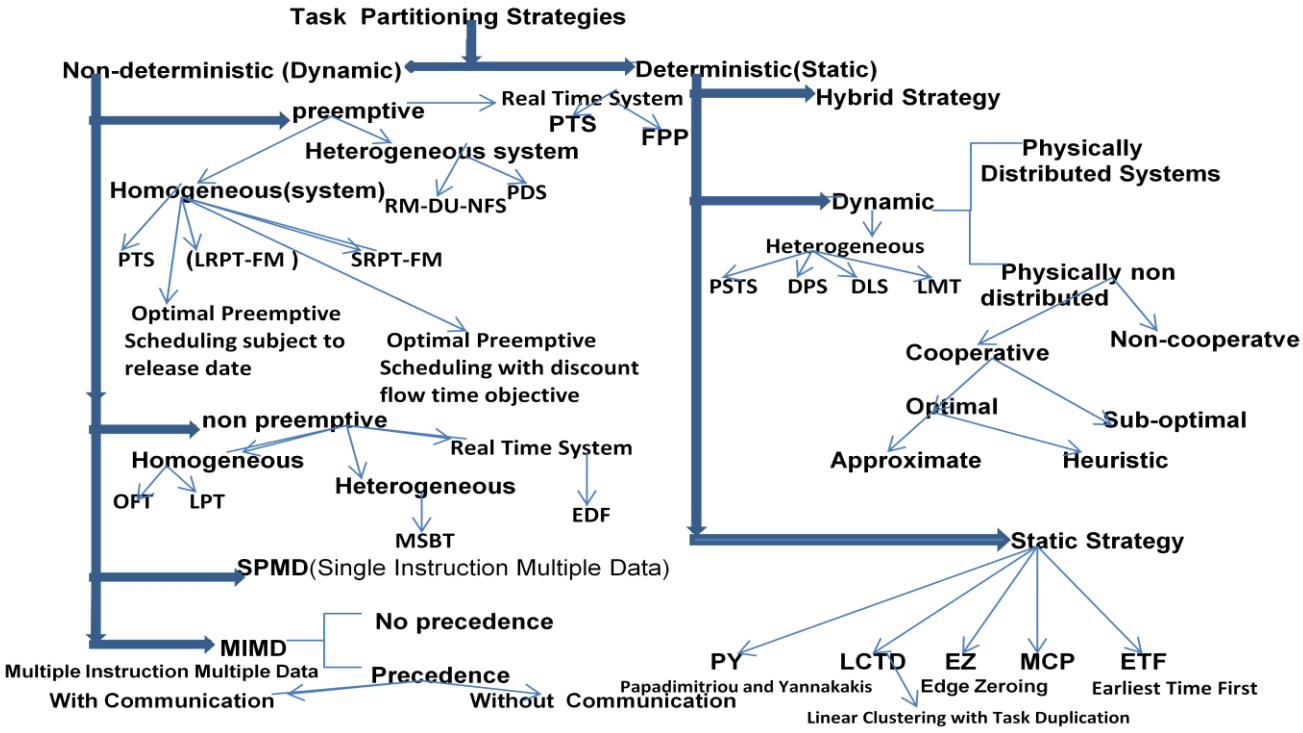

Figure: The hierarchal view of the different task partitioning strategies in different platforms.

2. DETERMINISTIC TASK (STATIC) PARTITIONING STRATEGIES:

In deterministic task partitioning the characteristics of an application such as communication cost, execution time, synchronization, and data dependency are known in advance [11].It assign the tasks to the different processors before execution, and allocation remain same during the execution of the process. Dynamic scheduling performs scheduling at runtime, and allocation of the process may change during the execution of tasks. Static scheduling cannot support load balancing and fault tolerance while dynamic scheduling support these parameters. Deterministic scheduling on Network of Workstation (NOW) is NP-complete in strong sense while nondeterministic is not strong NP-complete. Purely static task partitioning with OpenCL, determines the best partitioning across the processors in a system and divides the tasks into as many chunks as there are processor with each processor receiving the appropriate chunks [13].

PY (Papadimitriou and Yannakakis)[34] scheduling approximate the absolute achievable lower bound of the start time of a node by using attribute e-value. From the first node to exit node e-value is computed recursively to assign the priority to the node. After calculating the e-value of the node, each node is inserted into a cluster in such a way that ancestors have data arrival times larger than the e-value of the node.

LCTD (Linear Clustering with Task Duplication) [35] scheduling identify the edges among clusters that determine the completion time after the linear clustering of DAG. After that it tries to duplicate the parents corresponding to these edges to reduce the start times of some nodes in the cluster.

EZ (Edge Zeroing) Scheduling [33] select computing environment by merging edge weights. This scheduling strategy finds the edge with the largest weight in every step. If merging does not increase the completion time, the two cluster incident by the edges will be merged (zeroing the largest weight).The ordering of the node is define in the resulting cluster based on the static b-level of the nodes.DAG edges are sorted in a descending order of edge weight.

MCP (Modified Critical Path)[31] scheduling decide the priority of the nodes on the basis of ALAP. It compute the ALAP of the existing node, and construct a list of node in ascending order of ALAP times.MCP looks for the processor idle time slot for a given node. Select the first node from the node list and insert it into the processor of earliest execution time. It cannot guarantee an optimal schedule for the fork and join structure.

ETF (Earliest Time First) scheduling [32] technique, select the nods of the smallest start time from the node of the earliest start time in the ready queue at each step. Earliest start time is computed by examine the start time of the node on all processor exhaustively. When two node have the same EST then ETF breaks the tie by selecting the node with the higher static priority. The priorities are assigning to the node by computing the static b-level. This strategy is not guaranteed a optimal schedule for the fork and join structure.

3. NON DETERMINISTIC TASK (DYNAMIC) PARTITIONING STRATEGIES:

Position Scan Task Scheduling (PSTS) is pure dynamic load balancing, highly parallel and efficient, and can be used in centralized and decentralized manner [14]. It is based on divide and conquers principle in which hyper-grids of dimension k are divided into grids of dimension k-1, until the dimension is 1.Position Scan Load Balancing (PSLB) technique is two phase technique. In first phase the system is modeled as a hypercube only at one time. In the second phase

load balancing is performed by obtaining necessary information for each system node. On the basis of the prior information PSLB strategy calculate its future load, and load is migrated according to the capability of the nodes. An important aspect of this node is that, in the case of load migration each node knows exactly where to send its extra work load. The communication is take place only between the neighbor nodes.

Evolutionary Task Scheduling [26] upon the previous scheduling stages, in static and dynamic environment. The use of heuristic in the initialization phase and specific mutation operator are two parameters which provide the beneficial result for the effective scheduling in the static environment. In the consistent environment, moving the tasks from high level processors to the processor of smaller makespan by using "rebalancing" operator provide the better result. But in the case of inconsistent environment less greedy operator provide the good scheduling. The information collected from the previous state is used, to select the environment by the scheduler.

DPS(Dynamic Priority Scheduling) [27] assign the priorities to the tasks based on the difference of bottom level (b-level) and top(t-level), to schedule the minimum schedule length in Directed Acyclic Graph(DAGs) [3].Compile time scheduling is used in the heterogeneous environment .Dynamic process assigning select more important tasks before less important ones. The mapping and scheduling strategies are depend upon the processor scheduler, network architecture and the DAG structure[28].The t-level is the length of longest path (execution cost+ communication cost) between target task and entry task, while b-level is the largest path between target task and exit task. In this way, t-level determine the earliest start time, b-level of a task is bounded by the critical path of the DAG [12].

DLS[29] used a GDL(generalized dynamic level) to determine dynamic priorities, for all tasks in the ready queue, over all processors at every scheduling step.GDL give the priority to the processor according to their speed, and many other factors are taken into account. Top level and bottom level are loss their meaning in the case of the heterogeneous architecture due to the different execution cost on each processor. It consider the various factors, median of the execution cost over all processors, average execution cost and maximum or minimum cost for the computation of the schedule levels.

4. PREEMPTIVE TASK PARTITIONING STRATEGIES:

Preemptive Deterministic Scheduling (PDS) ensure replica behavior while preserving concurrency. Replication is achieved by the threads and there is no communication between the threads. Performance improvement is achieved due ti multithreading by exploiting concurrency in thread execution. Multithreaded replica shows non deterministic behavior. Only one physical thread is schedule at given time, so it shows poor scalability and performance. PDS schedule multiple threads at same time so its throughput five times from Preemptive Nondeterministic Scheduler (NPDS) [11].PDS remove the need for inter replica communication yet preserves a large degree of replica concurrency. Optimal preemptive schedule subject to release date are taken to minimize the execution time on the homogeneous platforms'. This strategy performs the two step, in first step, minimize maximum completion time on the desired number of machines. In second step, maximum lateness is minimizes with respect to due dates for the jobs [12] on the arbitrary number of machines.

Fast Preemptive Scheduling (FPS) [17] strategy simulates preemptive task execution at a very low overhead and requires small runtime support in heterogeneous and homogeneous parallel computing environment. Preemptive Task Scheduling (PTS) [18],a list heuristic scheduling strategy is used homogeneous distributed memory systems.

RM (Rate Monotonic)-DU (Decrease Utilization)-NFS (Next Fit Scheduling) [19] strategy does not suffer from execution time and period anomalies. It's a scheduling of periodically arriving tasks on multiprocessor environment. This strategy partitioning tasks by using a variant of next-fit-bin-packing. If the following assumption does not hold; (1) assume that a task meet its deadline if it did so when all tasks are executed at their maximum execution time, or (2) assume that a task meets its deadline if it did so when all tasks arrived frequently, then this situation is known as scheduling anomalies. RM-DU-NFS is the combination of DU and NFS which is based on high system utilization bound.

Optimal Preemptive Scheduling [20] is used to schedule n jobs on, m parallel uniform machine. By assigning shortest remaining processing time jobs to the fastest available machine, discounted flow time is minimized by serving jobs preemption in increasing order of their remaining processing time. Shortest Remaining Processing Time on Fastest Machine (SRPT-FM) rule is also optimal for the discounted flow time criteria. Flow time is minimized by scheduling jobs according to the shortest remaining processing time on the fastest machine. Minimization of the makespan is achieved by scheduling the (LRPT-FM rule) Longest Remaining Processing Time on Fastest Machine [21].

PTS (Preemption Threshold Scheduling) [23] disable the preemptions up to a specific priority level, called preemption threshold. Regular priority assign to the arriving tasks, and the tasks may preempt only if the priority of the arriving task is higher than the threshold of running task.

In FPP (Fixed Preemption Point) scheduling[24] strategy, a task is assign in a non-preemptive mode, and a preemption can take place at a specific point of the code which is known as preemption point. In this way, a task is divided into subtasks. If the higher priority task is arrive then preemption is postpone until the next preemption point. So the tasks cooperate to each other by the preemption point.

5. NON-PREEMPTIVE PARTITIONING STRATEGIES:

LSA (Loose Synchronization Algorithms) use non-preemptive deterministic scheduler to maintain multithreaded replica consistency. Its control concurrency by the leader (mutex) thread, which enforced the leader dictated order on the execution of their thread (followers) [13]. OFT (Optimal Finish Time) non preemptive strategy is known as NP-complete with many uniform processors [14].LPT (Largest Processing Time First) scheduling is near optimal to non-preemptive OFT.

MSBC (Multiple Strict Bound Constraints) scheduling is non-preemptive static strategy for heterogeneous computing

systems. It perform alternative task priority schedule instead of Heterogeneous Earliest Finish Time (HEFT) scheme. It's also used to raised the performance of parallel computing applications on the heterogeneous platforms due to macro data flow graph implementation [16].

EDF (Earliest Deadline First) Strategy, schedule periodic or sporadic tasks on uniprocessor without preemption and without inserted idle time [25].Periodic tasks are invoked after a certain time interval while sporadic tasks are invoked in arbitrary time but within the limited time constraints.EDF is universal for the set of periodic or sporadic tasks.

Comparative analysis of various task partitioning and scheduling strategies is discussed below. This analysis is highly beneficial to choose the appropriate strategy under the different requirement and architecture:

S.N.	Name of Strategy	Type of Strategy	Architecture	Remark	Ref
1	PY (Papadimitriou and Yannakakis) Scheduling	Static	Homogeneous	DAG based scheduling. Not Optimal.	34
2	LCTD (Linear Clustering with Task Duplication)	Static	Homogeneous	DAG based scheduling. Not Optimal	35
3	EZ (Edge Zeroing)	Static	Homogeneous	DAG based scheduling	33
4	MCP (Modified Critical Path)	Static	Homogeneous	DAG based scheduling Optimal Scheduling	31
5	ETF (Earliest Time First)	Static	Homogeneous	DAG based scheduling. Optimal Scheduling.	32
6	Position Scan Task Scheduling (PSTS)	Dynamic	Heterogeneous	It can be used in centralized and decentralized manner.	14
7	Evolutionary Task Scheduling	Static and Dynamic	Heterogeneous	Previous state information decide the scheduling environment	26
8	DPS(Dynamic Priority Scheduling)	Dynamic	Heterogeneous	Difference of top level and bottom level of node decide the priority of the tasks, Its DAG based scheduling.	27
9	DLS (Dynamic Level Scheduling)	Dynamic	Heterogeneous	DAG based scheduling. Priority assign to the tasks on the basis of GDL (Generalized Dynamic Level) in the ready list at every scheduling step.	29
10	LMT(Levelized Min Time) Scheduling	Dynamic	Heterogeneous	DAG based scheduling. During first phase level sorting is used and in second phase greedy heuristic is applied for assigning priority.	30
11	FPS(Fast Preemptive Scheduling)	Preemptive	Heterogeneous and Homogeneous	Scheduling cost is very low	17
12	PTS(Preemptive Task Scheduling)	Preemptive	Homogeneous	Cannot be implemented in Heterogeneous	18
13	PDS(Preemptive Deterministic Scheduling)	Preemptive	Heterogeneous	Multiple threads, and there is no inter replica communication	11
14	Optimal Preemptive Scheduling subject to release date	Preemptive	Homogeneous	Many step process	12
15	RM-DU-NFS Scheduling	Preemptive	Heterogeneous	Free from scheduling anomalies	19
16	Optimal Preemptive Scheduling with discount flow time objective	Preemptive	Homogeneous	SRPT-FM rule is also optimal for discounted flow time criteria	20
17	SRPT-FM Scheduling	Preemptive	Homogeneous	Minimize flow time	21
18	LRPT-FM Scheduling	Preemptive	Homogeneous	Minimize makespan	21
19	PTS(Preemptive Threshold Scheduling)	Preemptive	Real Time System	It's used to reduce unnecessary preemption	23
20	FPP(Fixed Preemption Point) Scheduling	Preemptive	Real Time System	Non preemptive task make preemptive with the help of preemption point in code of the job	
21	LSA(Loose Synchronization Algorithm) Scheduling	Non-preemptive	Heterogeneous	Its multithreaded replica strategy in which leader(thread) control to(follower) thread by using mutex (locking/unlocking).	13
22	OFT(Optimal Finish Time) strategy	Non-preemptive	Homogeneous	NP-Complete	14
23	LPT(Largest Processing Time	Non-preemptive	Homogeneous	Its near optimal to non-preemptive	15

	First) strategy			OFT	
24	MSBT(Multiple Strict Bound Constraints) strategy	Non preemptive, static	Heterogeneous	Its use macro data flow graph implementation	16
25	EDF(Earliest Deadline First) Scheduling	Non-preemptive	Real Time System	Universal for the set of periodic or sporadic tasks	25

Table 1: Comparison of Different Task Partitioning Strategies under the various architectures

6. FUTURE WORK

Advancement of the technologies and the architecture is a key factor for the mapping heuristics of the task partitioning strategies. A researcher decides his direction of the research problem according to the existing strategies. Comparative analyses of the recent scheduling strategies provide the standard of the existing work in the field of the parallel computing systems. In the static scheduling the most challenging direction is to extend DAG scheduling to the heterogeneous environment. A heterogeneous computing environment is a collection of heterogeneous machines consisting of communication protocols, high speed interconnections, interfaces and efficient parallel computing tools. One research avenue in the research direction is that new model and optimized algorithms should be designed which cover the major fields of the parallel computing systems in the efficient manners. A common terminology is used assigning the tasks onto the different type of the parallel computing architectures.

7. CONCLUSION

We have studied problems, discussed in the literature from the last twenty years. The main intension of the paper is to provide a suitable framework for comparing past work in the area of distributed parallel computing systems. Ideal performance of the strategy is depending upon the requirement and the architecture used for the distributed parallel processing systems. From the brief discussion of scheduling strategies it's clear that there is no ideal strategy for all parallel computing system The fruitful comparative analysis of strategies provides easiest comparisons of the existing systems. In this paper, Dynamic, preemptive and non-preemptive task partitioning and load balancing strategies are briefly discussed.

8. REFERENCES:

[1] Gary. E. Christensen, MIMD vs. SIMD parallel processing: A case study in 3D medical image registration, Parallel Computing 24 (9/10) (1998) ,pp. 1369–1383.

[2] Feitelson D.G., "A Survey of Scheduling in Multiprogrammed Parallel Systems", Research Report RC 19790 (87657), IBM T.J. Watson Research Center, August 1997.

[3] Jia-X. Z.; Wei-M. Z.; , "A DAG-based partitioning-reconfiguring scheduling algorithm in network of workstations,"High Performance Computing in the Asia-Pacific Region, 2000. Proceedings. The Fourth International Conference/Exhibition on , vol.1., 2000, pp.323-324

[4] Andrews J. B. and Polychronopoulos C. D.. "An analytical approach to performance/cost modeling of parallel computers". Journal of Parallel and Distributed Computing, 12(4):Aug. 1991, pp.343–356.

[5] Menasce, D. A., Saha, D., Porto, S. C. D. S.,Almeida, V. A. F., and Tripathhi, S. K. "Static and dynamic processor scheduling disciplines in heterogeneous parallel architectures". J. Parallel Distrib. Comput. 28, 1 (July 1995), pp .3-6.

[6] [Gajski,D., and Peir, J., "Essential Issue in Multiprocessor", IEEE Computer Vol 18, No.6 ,1985,PP. 1-5.

[7] Menasce, D. A., Porto, S. C.,and Tripathi, S. K.. Static heuristic processor assignment in heterogeneous message passing architectures. Int. J. High Speed Computing. 1994, PP.114–135.

[8] Shmoys D. B., Wein J., and Williamson D.P., "Scheduling parallel machines on-line". SIAM Journal on Computing,, December 1995.

[9] Nelson R. and Towsley D., "Comparison of threshold scheduling policies for multiple server systems," IBM, Research. Report,1985.

[10] Feitelson D.G. and Rudolph L.. Parallel job scheduling: Issues and approaches. In IPPS'95 Workshop: Job Scheduling Strategies for Parallel Processing,Springer–Verlag, Lecture Notes in Computer Science LNCS 949, 1995,pp.1-3.

[11] Kwok Y. and Ahmad I., "Static Scheduling Algorithms for Allocating Directed Task Graphs to Multiprocessors", ACM Computing Surveys 31(4) (December 1999) 406–471

[12] Yu-Kwong K. and Ishfaq A., "Efficient Scheduling of Arbitrary Task Graphs to Multiprocessors Using a Parallel Genetic Algorithm",Journal of Parallel and Distributed Computing, 1997, pp.1-3.

[13] Grewe D.and M. F. O'Boyle. A static task partitioning approach for heterogeneous systems using OpenCL. In CC'11, Mar.-Apr. 2011.

[14] Savvas K and Tahar Kechadi, M. "Dynamic Task Scheduling in Computing Cluster Environments," Proceedings of the ISPDC/Heterogeneous Parallel Computing, IEEE conference, 2004, pp.121–154.

[15] S. Ali, H.J. Siegel, M. Maheswaran, D. Hensgen and S. Ali. "Task Execution Time Modeling for Heterogeneous Computing System. Proceedings of Heterogeneous Computing Workshop", 2000, pp. 184-199 .

[16] Chen H.. "On the Design of Task Scheduling in the Heterogeneous Computing Environments". IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, 2005.

[17] Ahmed M., S.M.H. Chowdhury, M. Hasan, Fast preemptive task scheduling algorithm for homogeneous and heterogeneous distributed memory systems, in: Ninth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2008, pp. 721– 726.

[18] Radulescu A. and A. J. C. van Gemund. "On the complexity of list scheduling algorithms for distributed memory systems". ACM Int'l Conf. on Supercomputing,Rhodes Greece, 1999.

[19] Andersson B., Jonsson J., "Preemptive multiprocessor scheduling anomalies," Proceedings of IPDPS 2002, pages: 12-19, 2002

[20] Pandelis] D. G., "Optimal Preemptive Scheduling on Uniform Machines with Discounted Flow Time Objectives," European Journal of Operational Research, Vol. 177, No. 1, 2007, pp. 630-637.

[21] Gonzalez T., "Optimal Mean Finish Time Preemptive Schedules, Technical Report 220, Computer Science Department, Pennsylvania State University 1977.

[22] Renan A. S., Romulo S. de O., "A Heterogeneous Preemptive and Non-preemptive Scheduling Approach for Real-Rime Systems on Multiprocessors,", 2012 Second Brazilian Conference on Critical Embedded Systems, 2012, pp.70-75.

[23] Desrochers M., Lenstra J.K., and Savelsbergh M.W.P., "A Classification Scheme for Vehicle Routing and Scheduling Problems", European Journal of Operational Research ,1990, pp. 320–331.

[24] Burns A.. "Preemptive Priority Based Scheduling: An Appropriate Engineering Approach". Technical Report YCS 214, University of York, 1993,pp.12-18.

[25] Jeffay, K.; Stanat, D.F.; Martel, C.U.; "On Non-Preemptive Scheduling of Period and Sporadic Tasks," Real-Time Systems Symposium, Proceedings., Twelfth , vol., no., 1991, pp.129-139.

[26] Zamfirache, F.;,Zaharie, D., Craciun, C. ,"Evolutionary Task Scheduling in Static and Dynamic Environments" Computational Cybernetics and Technical Informatics , International Joint Conference on , vol., no., 2010,pp.619-625.

[27] Ahmad, I.; Dhodhi, M.K.; Ul-Mustafa, R.; , "DPS: Dynamic Priority Scheduling Heuristic for Heterogeneous Computing Systems," Computers and Digital Techniques, IEE Proceedings - , vol.145, no.6, , Nov 1998,pp.411-418.

[28] Norman M. G. and Thanisch P.. "Models of Machines and Computation for Mapping in Multicomputers". ACM Comput. Surv.,1993. pp.263–302.

[29] Sih, G.C.; Lee, E.A.; , "A compile-time scheduling heuristic for interconnection-constrained heterogeneous processor architectures," Parallel and Distributed Systems, IEEE Transactions on , vol.4, no.2, , Feb 1993, pp.175-187.

[30] Iverson M., F. Ozgumer, and Follen G.."Parallelizing Existing Applications in a Distributed Heterogeneous Environment" In Proceedings of the 4th Heterogeneous Computing Workshop (HCW'95), , 1995, pp. 91–99.

[31] Wu, M.Y. and Gajski, D. D., "Hypertool: A Programming Aid for Message-Passing Systems". IEEE Transaction Parallel Distributed. Systems. 1990 pp. 331–344.

[32] Hwang, J.-J., Chow, Y.-C., Anger, F. D., and Lee, C.Y. " Scheduling Precedence Graphs in Systems with Inter-processor Communication Times. SIAM J. Computer, 1989, pp. 245–256.

[33] Sarkar, V. "Partitioning and Scheduling Parallel Programs for Multiprocessors". MIT Press, Cambridge, MA. 1989.

[34] Papadimitriou, C. H. and Yannakakis, M. ,"Towards an Architecture-Independent Analysis of Parallel Algorithms". SIAM J. Computer, 1990,pp. 324–327.

[35] Chen, H., Shirazi, B., and Marquis, J. "Performance Evaluation of a Novel Scheduling Method: Linear Clustering with Task Duplication". In Proceedings of the 2nd International Conference on Parallel and Distributed Systems, 1993,pp. 271–276.

AUTHOR'S PROFILE

Dr. Rafiqul Zaman Khan, is presently working as a Associate Professor in the Department of Computer Science at Aligarh Muslim University, Aligarh, India. He received his B.Sc Degree from M.J.P Rohilkhand University, Bareilly, M.Sc and M.C.A from A.M.U. and Ph.D (Computer Science) from Jamia Hamdard University. He has 18 years of Teaching Experience of various reputed International and National Universities viz King Fahad University of Petroleum & Minerals (**KFUPM**), K.S.A, Ittihad University, U.A.E, Pune University, Jamia Hamdard University and AMU, Aligarh. He worked as a **Head** of the Department of Computer Science at Poona College, University of Pune. He also worked as a **Chairman** of the Department of Computer Science, AMU, Aligarh.

His Research Interest includes Parallel & Distributed Computing, Gesture Recognition, Expert Systems and Artificial Intelligence. Presently **04** students are doing PhD under his supervision.

He has published about **25** research papers in International Journals/Conferences. Names of some Journals of repute in which recently his articles have been published are International Journal of Computer Applications (ISSN: 0975-8887), **U.S.A**, Journal of Computer and Information Science (*ISSN: 1913-8989*), **Canada**, International Journal of Human Computer Interaction (*ISSN: 2180-1347*), **Malaysia,** and Malaysian Journal of Computer Science(*ISSN: 0127-9084*), **Malaysia**. He is the Member of Advisory Board of International Journal of Emerging Technology and Advanced Engineering (IJETAE), Editorial Board of International Journal of Advances in **Engineering** & Technology (IJAET), International Journal of Computer Science Engineering and Technology (IJCSET), International Journal in Foundations of Computer Science & technology (IJFCST) and Journal of Information Technology, and Organizations (JITO).

Javed Ali is a research scholar in the Department of Computer Science, Aligarh Muslim University, Aligarh. He published 5 papers in international repute journals..He received State Scientist Award by making refrigerator without electricity. His research interest include parallel computing in distributed systems.He did Bsc(Hons) in mathematics and MCA from Aligrah Muslim University,Aligarh.