HYBRID GASA FOR BI-CRITERIA MULTIPROCESSOR TASK SCHEDULING WITH PRECEDENCE CONSTRAINTS

Sunita Dhingra[1], Satinder Bal Gupta[2] and Ranjit Biswas[3]

[1]Department of Computer Science & Engineering, University Institute of Engineering & Technology,
Maharshi Dayanand University Rohtak-124001 Haryana INDIA
[2]Department of Computer Science, Vaish College of Engineering Rohtak-124001 Haryana INDIA
[3]Department of Computer Science & Engineering, Jamia Hamdard University New Delhi-110062 INDIA

ABSTRACT

Task scheduling on different processors with precedence constraints is NP hard and finds a prominent place in the field of parallel computing and combinatorial optimization. However, it is quite difficult to achieve an optimal solution to this problem with traditional optimization approaches owing to the high computational complexity. Amongst the metaheuristics, Simulated Annealing (SA) and Genetic Algorithm (GA) represent the powerful combinatorial optimization methods with corresponding strengths and weaknesses. Borrowing the respective advantages of the two paradigms, an effective combination of GA and SA called hybrid GASA has been proposed for multiprocessor task scheduling problems with precedence constraints. The bi-criteria objective function, including the weighted sum of makespan and total completion has been considered for the analysis. Comparative analysis with the help of defined performance index on the standard problems shows that the proposed hybrid GASA provides better results when compared to simple GA and SA alone in terms of solution quality.

KEYWORDS

Multiprocessor Task Scheduling, Genetic Algorithm, Simulated Annealing, Metaheuristics, Makespan, Total CompletionTtime.

1. INTRODUCTION

Scheduling is a decision making process and concerned with the allocation of limited resources to tasks overtime with goal of optimizing one or more objectives. Multiprocessor scheduling is two dimensional in which the scheduler has to decide the order of tasks execution on different processors. This extra dimension greatly complicates scheduling on multiprocessors. Another complicating factor is that all the tasks are unrelated in some systems, whereas in others all tasks are related. Tasks which are related to each other i.e. there exists dependency between some of the tasks. Dependent Tasks can be executed only after the completion of tasks on which they are dependent. So, in the situation of dependency, communication cost is the major factor to be considered by a scheduler. The multiprocessor task scheduling problem with dependent tasks can be easily represented by directed acyclic graphs (DAG). In a DAG G= (V, E), V the set of vertices represents the tasks and E, set of directed edges show the dependency between tasks. The computation weight of each vertex shows the number of CPU cycles required by a task and the computated weight on each directed edge shows the communication cost.

Based on different characteristics of the tasks to be scheduled and the multiprocessor system, as well as the availability of a priori information regarding the processing time, the multiprocessor scheduling problem can be categorized into many different classes. In the present work, only the static scheduling problem has been considered, the information such as task processing times, data dependencies, and communication costs between dependent tasks are known.

Majority of the research in field of multiprocessor task scheduling is concerned with the minimization of the single criteria i.e. makespan. However, in practice, many fields have tradeoffs in their scheduling problems where different objectives need to be considered for optimizing the overall performance of the system. Obviously, scheduling problems with more than one objective are more complex and it is hard to find a compromise solution as the objectives are often inconsistent, conflicting or even contradictory. The multiprocessor task scheduling problem with precedence constraints with multi-objectives is NP hard with greater complexity. The importance of multiprocessor task scheduling led to several comparative studies starting from various heuristics leading to metaheuristics and hybrid methods. Genetic algorithms and simulated annealing are the, fast and robust methods for solving NP hard problems and a lot of research of scheduling surrounds theses methods and their hybrid versions. The use of Holland's genetic algorithms (GAs) [1] in scheduling, which apply evolutionary strategies to allow for the fast exploration of the search space of schedules, allows good solutions to be found quickly. Hwang et al. [2], addresses the challenge of multiprocessor task scheduling parallel programs, represented as directed acyclic task graph (DAG), for execution on multiprocessors with communication costs. Genetic algorithm was used for solving the problem along with design of the new encoding mechanism with a multi-functional chromosome—the so-called priority-based multi-chromosome (PMC). Hou et al. [3] developed efficient method based on genetic algorithm for multiprocessor scheduling. They developed crossover operator which was based on task graphs with dependencies, but without communication delays. They showed that the results of GA were within 10% of the optimal schedules when compared with others. Wu et al. [4] proposed a novel GA which allows both valid and invalid individuals in the population. This GA used an incremental fitness function and gradually increases the difficulty of fitness values until a satisfactory solution found. This approach is not scalable to large problems since much time is spent evaluating invalid individuals that may never become valid ones. Bonyadi ·and Moghaddam [5] proposed Bipartite Genetic Algorithm (BGA) for minimizing the makespan for a multiprocessor task scheduling problem. They proved that BGA provides the best results with GA-based and heuristic based algorithms from literature in terms of STD, average makespan, best obtained makespan and iterations.

Simulated annealing (SA) belongs to the meta-heuristic algorithm which was introduced by Kirkpatrick et al. [6] for solving combinatorial optimization problems. M. Fikret Ercan et al. [7] evaluated the solution quality of heuristic algorithms developed for scheduling multiprocessor tasks for a class of multiprocessor architectures designed to exploit temporal and spatial parallelism simultaneously. The main focus was on the scheduling a number of pipelined multiprocessor tasks with arbitrary processing times and arbitrary processor requirements. Three well-known local search heuristic algorithms, Simulated Annealing, Tabu Search, and Genetic Algorithms are employed. Experiments were performed to evaluate the reduction achieved in completion time by each heuristic and to compare the results with simple list-based heuristics. The results show that local search heuristics significantly outperform the list based heuristics but due to their large computation times, SA, TS, or GA can be used in deterministic cases. Nadathur Satish et al. [8] presented two statistical optimization approaches for scheduling task dependence graphs with variations in execution time onto heterogeneous multiprocessor system. They proposed a statistical analysis based on accurate Monte-Carlo simulations compared to static approaches based on worst-case estimates. They used this analysis in two scheduling algorithms. One is a heuristic based on a critical path analysis of the task dependence graph & the other is a simulated annealing algorithm using incremental timing analysis. Both algorithms show a 25-30% improvement in makespan over methods based on static worst-case analysis. They further planned to integrate their solution methods into a practical design space exploration tool for multiprocessor systems. Kalashnikov et al. [9] proposed sequential and parallel algorithm of simulated annealing for multiprocessor scheduling problem. They compared classical sequential, sequential, parallel algorithms and showed that the sequential algorithm reduces the time of problem solution up to 3 times as compared to the classical algorithm of simulated annealing.

Sivanandam et al. [10] proposed a particle swarm optimization/simulated annealing (PSO/SA) hybrid algorithm for static allocation of tasks in a heterogeneous distributed computing system for minimizing the cost. Various versions of PSO algorithm like the simple PSO, the global PSO and Hybrid PSO with Simulated Annealing have been implemented. Different experiments performed on the benchmark problems and showed that the proposed hybrid method was effective and efficient for finding near optimal solutions. Yoo et al. [11] proposed a new scheduling algorithm for real-time tasks using multi-objective hybrid genetic algorithm (MOHGA) on heterogeneous multiprocessor environment with the objective of minimizing the total tardiness and completion time simultaneously. The adaptive weight approach was used for the multiple objectives. The convergence of GA was improved by introducing the probability of SA as the criterion for acceptance of the new trial solution. Various experiments performed to check the effectiveness of proposed algorithm. It was concluded that the results of the proposed MOHGA provide the better results than other algorithms without communication cost. Dahal et al. [12] hybridized the Genetic Algorithm (GA) with well known heuristics such as ‘Earliest Deadline First (EDF)’ and ‘Shortest Computation Time First (SCTF)’ for dynamic scheduling of real-time tasks in a multiprocessor system. It was concluded the SCTF when hybrid with GA provides better performance as compared to the EDF. Azghadi et al. [13] developed an immune genetic approach and combined with the proposed heuristic resulting into a hybrid scheme for multiprocessor task scheduling problem. The hybrid scheme uses initial population based on heuristic approach which results in faster convergence and best solution in some cases. They showed that the proposed hybrid approach was very much effective for the multiprocessor scheduling systems. Jouglet et al. [14] proposed a memetic algorithm (MA) with combination of genetic algorithm (GA) and constraint programming based on branch and bound algorithm (CP) for the hybrid flow shop scheduling with multiprocessor tasks. They proved that MA performs better then GA and CP in terms of solution quality and efficiency. Mohamed et al. [15] developed a modified list scheduling heuristic (MLSH) and a hybrid approach composed of GA and MLSH for task scheduling in multiprocessor system. They proposed three different representations for the chromosomes of genetic algorithm: task list (TL), processor list (PL) and combination of both (TLPLC) and found that proposed approach outperforms the others in terms of best makespan, average makespan and processor efficiency

Chitra et al. [16] considered the multi-objective task scheduling problem in heterogeneous distributed computing systems (HDCS) with two objectives of makespan and reliability index. They developed two Multi-Objective Evolutionary Algorithms and experiments were performed on various random task graphs and a real-time numerical application graph. They showed that, MOEA algorithms are well-suited for obtaining good pareto optimal solutions in a single run for task scheduling problem.

Different advantages and importance of GA and SA for combinatorial optimization provides the motivation for hybridization of these methods resulting into a hybrid GASA approach for multiprocessor task scheduling with precedence constraints.

2. PROBLEM STATEMENT

The multiprocessor task scheduling problem along with precedence constraints on homogeneous processors has been considered with the objective of minimizing the makespan and total completion time simultaneously. Various assumptions along with bi-criteria objective function is described below:

2.1. Assumptions

a) A static version of multiprocessor task scheduling is considered, i.e. processing time, data dependencies, communication cost between dependent tasks are known in advance.
b) The dependencies along with execution time and communication cost are represented by a DAG.
c) Two tasks scheduled on same processor have no communication cost and any two tasks scheduled on different processor have the communication cost specified by the edge weight in DAG.
d) The multiprocessor system consists of a set of homogeneous processors i.e all processors have same execution time to run a task individually.
e) Pre-emption of tasks is not allowed.
f) Task duplication is not allowed.
g) All processors & tasks are available at time $t = 0$.

2.2. Bi-criteria Objective Function

The present work considers the two objectives to be minimized simultaneously, which are makespan and total completion time.

Makespan of a schedule is the time at which the last task completes for a particular schedule i.e. C_{max}. Total completion time of a schedule is calculated as $\sum_{i=1}^{n} Ci$ where C_i is the completion time of i_{th} task of a schedule.

Makespan and total completion time are weighted according to the relative importance and the two weighted functions are added together for the bi-criteria objective function as:
$Min\ f = \alpha\ C_{max} + (1 - \alpha) \sum_{i=1}^{n} Ci$,

Where α is the weight coefficient in the range 0 and 1. Since the objectives have varying range, they are normalized in the range 0 to 1. When $\alpha = 1$, only the makespan objective is considered and when $\alpha = 0$ only the total completion time objective is considered. By varying the values of α the trade-off between the makespan and total completion time can be determined.

3. METAHEURISTICS BASED HYBRID APPROACH

Meta-heuristics are common methods that direct the search through the solution space, using some form of heuristics and local search. Metaheuristics improve the initial solution obtained by some heuristic iteratively, until the stopping criterion is met. Simulated annealing and genetic algorithm are among the widely used metaheuristics for scheduling problems. Metaheuristic based hybrid approach (GASA) has been proposed for solving NP hard problems and compared with the results obtained with Simple Genetic Algorithm (GA) and Simple Simulated Annealing (SA).

3.1. Outline of Hybrid GASA

It has been commonly accepted that finding optimality to NP hard problems is not a viable option since large amount of computational time is needed for judgment of such solutions. In reality, a good initial solution can be obtained by a heuristic/meta-heuristics in a reasonable computational time. Simple GA and SA generate initial seed sequence randomly and drawbacks is that, the choice of the initial seed sequence has an important influence on the quality of solution and a better initial sequence might provide better results. Due to the large search space in multiprocessor task scheduling, it is expected that random generation of initial solutions provides relatively weak results. For this, initial solution is obtained by application of Simple GA for finding near to optimal results in a very reasonable time and final optimal solution by Simple SA procedure and called as hybrid GASA. Outline of GASA described as:

i. Encoding: Encoding give the representation of a chromosome. In the present work, chromosome is represented as (T, P) pair where T' is task sequence t_1, t_2,.......,t_n & P is allocated processor sequence p_1,p_2,....,p_n.Each task sequence is a permutation of task numbers & each processor sequence is a permutation of processor numbers (1, 2... m) with length equal to number of tasks.
ii. Initialization for GA: Each task sequence is a permutation of task numbers, so each task will be processed according to its appearance. As dependency exists between tasks, so each task in the task sequence should appear before all of its children and after all of its parents. Therefore, some permutations of the tasks may not be valid and some mechanism would be needed to

validate the invalid sequences. The initial population in the present work is generated using the following steps:

a) Generate the valid task sequences (TS) of population size (ps) using the algorithm as stated by Bonyadi and Moghaddam [5].
b) Generate the processor sequences (PS) of ps randomly.

iii. Map the each task sequence (T) from TS to randomly selected processor sequence (P) from PS giving each chromosome in the form (T, P) i.e. task sequence followed by a mapping processor sequence

iv. Reproduction: Different reproduction operators are used for task and processor sequences due to different nature. The sequences are firstly separated from a chromosome and then used individually for performing crossover and mutation. Generated task sequences after reproduction may not be valid in terms of dependency, so a mechanism is used for validating the task sequences as stated by Bonyadi and Moghaddam [5]. Then valid task sequences after reproduction (TS') are mapped to processor sequences after reproduction (Ps') based on minimum fitness value.

The algorithm uses the following steps for generating the new offsprings:

a) Scores each member of the current population by computing fitness (i.e. weighted sum of makespan and total completion time).
b) Best Individuals as per fitness in the current population act as elite and concede in the next population.
c) Selects parents based on the fitness value as per selection function for reproduction.

v. Stopping criteria of GA: The algorithm generates a sequence called as {S} when the maximum number of generations reaches 50 which is the seed sequence in SA procedure.

vi. Initialization for SA: Sample schedule is assigned sequence {s} along with initializations of initial temperature and reanneal interval.

vii. Neighborhood generation (move function) and acceptance: It is a method of generating new schedules. The basic mechanism of neighborhood generation contains the following steps:

a) Separate the task sequence (T) from a processor sequence (P) in a schedule. Apply the neighbourhood generation method on both sequences individually for providing new task sequence (T') and processor sequence (P').
b) As new task sequence (T') may not be valid in terms of dependency, so a mechanism is used for validating the task sequence(T') using the algorithm of validation as stated by Bonyadi and Moghaddam [5] resulting in a validated task sequence(T'').
c) The validated task sequence (T'') is combined with processor sequence (P').

Now if the new schedule is better than the current in terms of the objective function, then it becomes the next schedule, otherwise the probability of acceptance is less than 50%.

viii. Cooling strategy (Temperature Function): It is related to systematically lowering the temperature, which is proportional to the cooling rate and store the best point found so far.

ix. Re-annealing: Re-anneal Interval is a parameter of simulated annealing which raises the temperature after the algorithm accepts a certain number of new points and starts the search again at the higher temperature.

x. Stopping limit: The algorithm stops when the limit of maximum number of iterations reached.

4. RESULTS AND DISCUSSIONS

Hybrid genetic algorithm and simulated annealing has been proposed in which initial feasible sequence has been obtained by the GA procedure which is the seed sequence for SA for the considered multiprocessor task scheduling problems in MATLAB environment. As the metaheuristic algorithms are approximate methods, so for each instance, all the algorithms have run five times for taking final average. Parameters fixed for hybrid GASA are given in table 1.

Table 1. Parameters fixed for GASA.

Parameter	Value
Population size	75
Elite Count	2
Crossover fraction	0.5
Selection function	Tournament
Stopping Criteria (maximum number of iterations)	100(50 for GA and 50 For SA)
Crossover function(Task)	Position based
Mutation function(Task)	Swap
Crossover function (processor)	One point crossover
Mutation function (Processor)	Uniform mutation
Initial Temperature	240
Annealing Function	Swap
Temperature Function	Fast
Reanneal Interval	150

The minimization of weighted sum of makespan and total completion time has been considered for the multiprocessor task scheduling problem. Some of the standard problems [5, 17] along with their computed weight coefficient (α) have been used for the comparative analysis between GASA, SA and GA as shown in table 2.

The comparative analysis has been done by computing the performance Index (PI) as:-

$$\text{Performance Index (PI)}\ (\%) = \left[1 - \frac{Algorithm_{solution} - Best_{solution}}{Best_{solution}} \right] \times 100$$

Algorithm$_{solution}$ is the average solution obtained for a given task problem by the different algorithms and Best$_{solution}$ is the best known solution or best solution among different algorithm for the problem in all the five runs. PI nearer to 100% provides the best results.

Table 2. Standard Multiprocessor Task Scheduling Problems[17] along with weight coefficient.

Problems	**No. of Tasks**	**No. of processors considered**	**Communication cost**	**Reference**	**Remarks**	**Weight Coefficient**
T9	9	2,3,4	Variable	Bonyadi and Moghaddam [5]	------	0.82
T14_1	14	2,3,4	Fixed(20)	Tsuchiya et al. [19]	LU decomposition	0.89
T14_2	14	2,3,4	Fixed(80)	Tsuchiya et al. [19]	LU decomposition	0.89
T16_1	16	2,3,4	Fixed(40)	Wu and Gajski [18]	Laplace	0.90
T16_2	16	2,3,4	Fixed(160)	Wu and Gajski [18]	Laplace	0.90
T18	18	2,3,4	Variable	Bonyadi and Moghaddam[5]	Gaussian Elimination	0.91

Comparative analysis for makespan and total completion time criteria has been shown in figure 1 and 2 respectively. All the algorithms have been compared for the different problems on two, three and four processors on the basis of makespan and total completion time of the schedule. For fair comparison, the three algorithms have been run for maximum of 100 iterations.

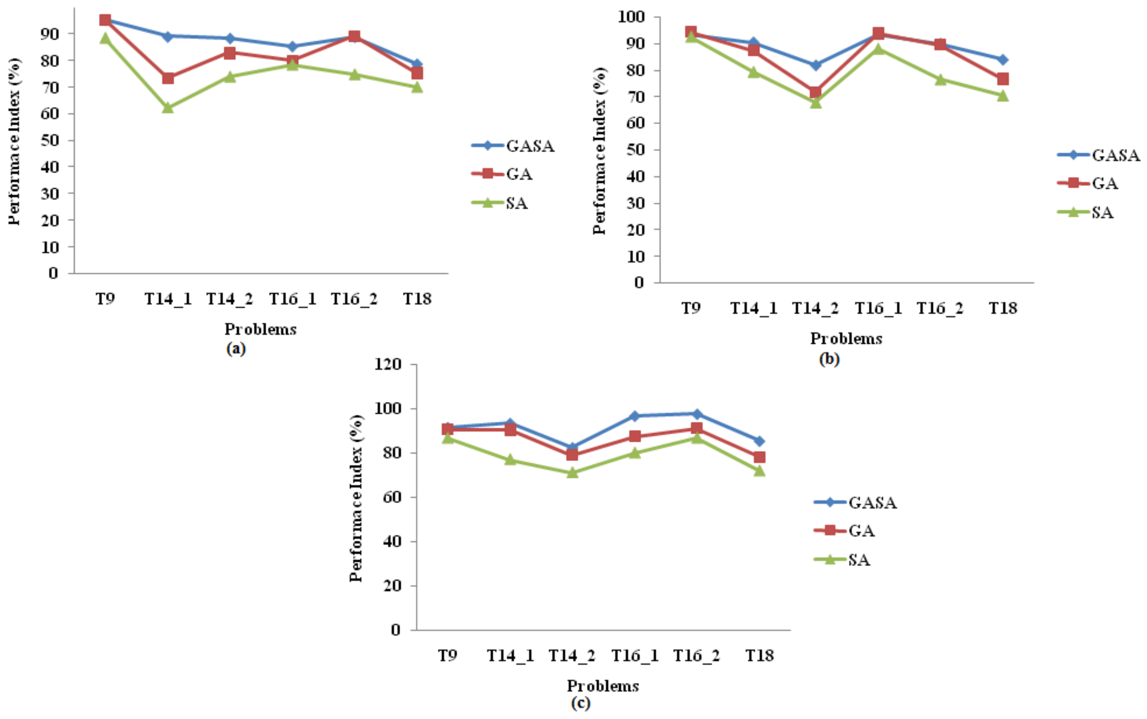

Figure 1. Performance Index (%) of GASA, GA and SA for Makespan (a) Two Processor (b) Three Processor (c) Four Processor

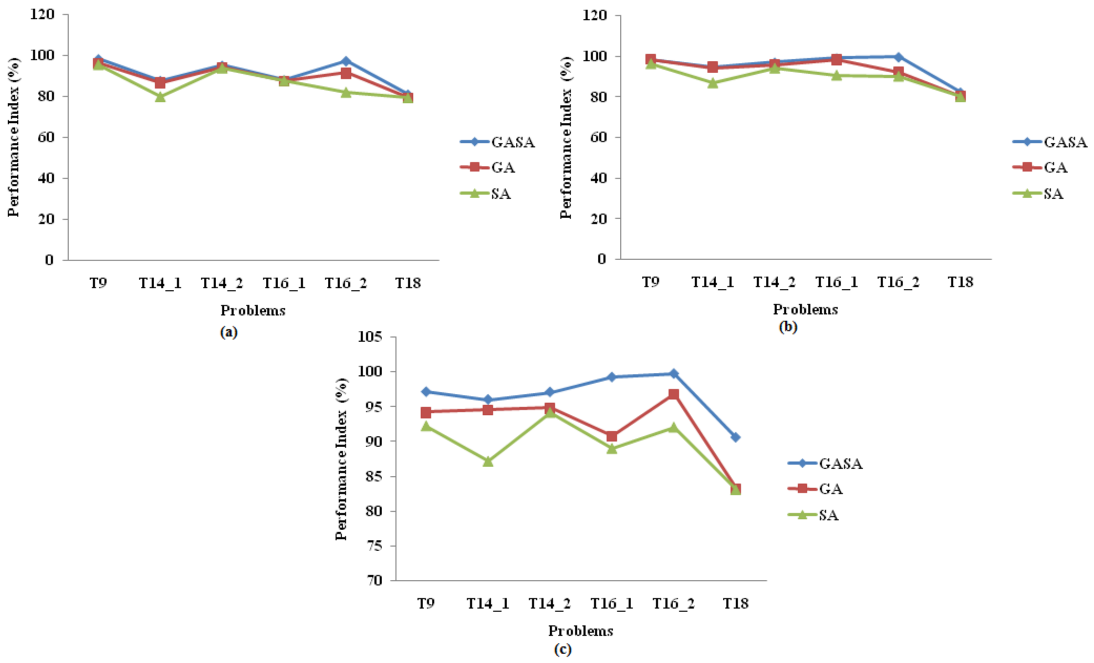

Figure 2. Performance Index (%) of GASA, SA and GA for total completion time (a) Two Processor (b) Three Processor (c) Four Processor

It can be seen that hybrid GASA shows superiority for the multiprocessor task scheduling problems for makespan and total for 2, 3 and 4 machines has been calculated for the all the entire three algorithms which have been completion time especially for larger size problems when compared to GA and SA alone. The average PI for 2, 3 and 4 machines has been calculated for the all the three algorithms and it has been found GASA provides the best PI for all the problems considered as shown in table 3.

Table 3. Average PI (%) for GASA, SA and GA on different processors

No. of (→) Processors	**2**		**3**		**4**	
HSAs(↓)	**MS**	**TCT**	**MS**	**TCT**	**MS**	**TCT**
GASA	**87.51**	**91.04**	**88.75**	**94.85**	**91.08**	**96.56**
SA	74.64	86.06	79.04	89.58	78.91	89.56
GA	82.62	89.15	85.54	92.88	86.04	92.35

MS: Makespan & TCT: Total Completion Time

5. CONCLUSIONS

In the present work, multiprocessor task scheduling problems have considered with minimizing the weighted sum of makespan and total completion time. Hybrid GASA has been proposed and compared to Simple GA and Simple SA alone and tested on the standard problems upto 18 tasks and 4 processors with communication cost and precedence constraints. To ensure fair grounds of comparison among Hybrid GASA, Simple GA and Simple SA, the stopping limit of all the metaheuristics is set to maximum number of generations/iterations. From the analysis, it has been observed that hybridization of genetic algorithm with simulated annealing (GASA) proved to be an effective approach for minimizing the makespan and total completion time criteria. Also the performance of the meta-heuristics discovered the absolute superiority of proposed GASA when compared to Simple GA and SA alone.

REFERENCES

[1] Holland, J. H. (1992) Adaptation in Natural and Artificial Systems. Cambridge, MA, USA: MIT Press.

[2] Hwang, R., Gen, M. and Katayam, H., (2008) "A comparison of multiprocessor task scheduling algorithms with communication costs", The Journal of Computers & Operations Research, Vol. 35, pp. 976 – 993.

[3] Hou, E.S.H., Ansari, N. and Hong, R., (1994) "A Genetic Algorithm for Multiprocessor Scheduling", IEEE Transactions on Parallel and Distributed Systems. Vol. 5, No. 2, pp. 113 – 120.

[4] Wu, A.S., Yu, H. , Jin, S. , Lin, K. C. and Schiavone, G., (2004)"An incremental genetic algorithm approach to multiprocessor scheduling", IEEE Transactions on Parallel and Distributed Systems, Vol. 15, No. 9, pp. 824–834

[5] Bonyadi, M. R. and Moghaddam, M. E. , (2009) "A bipartite genetic algorithm for multi-processor task scheduling", International Journal of Parallel Programming, Vol. 37, No. 5, pp. 462- 487.

[6] Kirkpatrick S, Gelatt CD, Vecchi M.P., (1983) "Optimization by simulated annealing, Science", Vol. 220, pp. 671–680.
[7] M. Fikret Ercan, Ceyda Oguz, (2007) " Performance of local search heuristics on scheduling a class of pipelined multiprocessor tasks", Computers and Electrical Engineering, Vol. 31 , pp. 537–555.
[8] Nadathur Satish, Kaushik Ravindran, Kurt Keutzer, (2008) "Scheduling Task Dependence Graphs with Variable Task Execution Times onto Heterogeneous Multiprocessors", EMSOFT'08, October 19–24, Atlanta, Georgia, USA.
[9] Kalashnikov, A.V. and Kostenko, V., A., (2008)"A Parallel Algorithm of Simulated Annealing for Multiprocessor Scheduling", Journal of Computer and Systems Sciences International, Vol. 47, No. 3, pp. 455–463.
[10] Sivanandam, S.N.., Visalakshi, ,P., and Bhuvaneswari, A., (2007) "Multiprocessor Scheduling Using Hybrid Particle Swarm Optimization with Dynamically Varying Inertia" , International Journal of Computer Science & Applications, Vol. 4, No.3, pp 95-106,.
[11] Yoo,M., and Gen, M. , (2007) "Scheduling algorithm for real-time tasks using multiobjective hybrid genetic algorithm in heterogeneous multiprocessors system", Computers and Operations Research, Vol. 34, pp. 3084 – 3098, 2007.
[12] Dahal, K., Hossain, A. , Varghese, B. , and Abraham, A., (2008) " Scheduling in Multiprocessor System Using Genetic Algorithms", Proceedings Of 7th Conference on Computer Information Systems and Industrial Management Applications(CISIM-2008), pp. 281 – 286.
[13] Azghadi M. R., Bonyadi, M.R., Hashemi, S. and Moghadam, M.E., (2008) "A Hybrid Multiprocessor Task Scheduling Method Based on Immune Genetic Algorithm", Proceedings of 5th International ICST Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness Hong Kong, Hong Kong , July 28 - 31.
[14] Jouglet, C. O˘guz, and Sevaux, M., (2008) "Hybrid Flow-Shop: a Memetic Algorithm Using Constraint-Based Scheduling for Efficient Search", Journal of mathematical modeling and algorithms, Vol. 8, pp. 271 -292.
[15] Mohamed, M.R., Awadalla, M.H.A., (2011) "Hybrid Algorithm for Multiprocessor Task Scheduling", IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 3, No. 2, 2011, pp. 79-89.
[16] Chitra,P., Venkatesh, P. and Rajaram, R., (2011) "Comparison of evolutionary computation algorithms for solving bi-objective task scheduling problem on heterogeneous distributed computing systems", Sadhana Vol. 36, Part 2, 2011, pp. 167–180
[17] Dhingra, S., Gupta, S.B. and Biswas, R., (2014) "Comparative analysis of heuristics for multiprocessor task scheduling problem with homogeneous processors", Advances in Applied Science Research, Vol. 5, No. 3, pp. 280 – 285.
[18] Wu, M.Y., Gajski, D.D., (1990) "Hypertool: A programming aid for message-passing systems", IEEE Transactions on Parallel and Distributing System, Vol. 1, No. 3, pp. 330–343.
[19] Tsuchiya, T., Osada, T., and Kikuno, T., (1998) "Genetics-based multiprocessor scheduling using task duplication", Journal of Microprocessors and Microsystems, Vol. 22, Vol. 3–4, pp. 197–207.