CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION

The scheduling problem has been addressed in several applications such as information processing, database systems, parallel computing, weather forecasting, image processing, fluid flow, process control, economics, operation research and real time high-speed simulations of dynamical systems, etc. Parallel computing is the most promising approach for meeting the increased computational requirements that have introduced a number of problems including multiprocessor task scheduling. The problem of Task graph scheduling of a parallel program onto a parallel and distributed computing system is an NP-hard problem that has received much interest and it is considered the most difficult problems in parallel computing (Mohamed and Awadalla; 2011). Usually a large program is divided into smaller tasks having some dependencies representing the precedence constraints such that a task cannot be started until all its predecessors have finished. The goal of a task scheduling algorithm is to schedule all the tasks on the given number of available processors without violating precedence constraints so as to optimize the different performance measures in order to maximize the efficiency, throughput and utilization of the system. The parallel tasks must be allocated into the processors such that the total completion time must be as less as possible.

Since the beginning of the research in this field, many approaches have been developed for the solution of task scheduling on multiprocessor systems. Some are heuristic based approach; some rely on metaheuristics including evolutionary and neighbourhood search approaches and some follows the hybrid methods. Most of the metaheuristics outperformed traditional heuristic based algorithms at the cost of extra time and computing effort. Therefore, metaheuristics such as Genetic Algorithm, Simulated annealing, Tabu search, Particle swarm optimization, etc. have been adopted by researchers for obtaining the better quality schedules. Hybridization of metaheuristics with heuristics is the next choice for improving the solution quality.
Hence, in this chapter, some of the important literature in the field of multiprocessor task scheduling has been presented in respect of techniques ranging from the traditional heuristic methods to modern metaheuristic methods.

2.2 HEURISTIC ALGORITHMS

Traditional scheduling research focused on the heuristic based algorithms which are divided into three categories: list, clustering and duplication based scheduling algorithms. List scheduling algorithms are the most significant and based on assigning priority to each task according to commonly used attributes such as t-level (top-level), b-level (bottom level), static level (sl) and ALAP (As Late As Possible) start time (Kwok and Ahmed; 1999). The highest priority task is allocated to the processor, which gives the earliest start time. Task selection can be random or based on some rules for the task having the same priority. There are many heuristic based methods for solving multiprocessor task scheduling approach and the best heuristic approaches are based on task list technique (Shirazi et al.; 1990). Clustering based scheduling is also the most widely studied and exploited heuristics for multiprocessor task scheduling. Commonly used clustering based heuristics are LC (Kim and Brown; 1988) and EZ (Sarkar; 1989). Duplication based heuristics are more efficient for the fine grain task graphs and networks with increased communicational latencies. DSH (Kruatrachue and Lewis; 1987) and BTDH (Chung and Ranka; 1992) are commonly used duplicate based heuristics.

Adam et al. (1974) proposed Highest Level First with Estimated Time (HLFET) algorithm which is the simplest list scheduling algorithm that uses static b-level as node priority. It assigns the task to the processor according to minimum start time. HLFET uses no-insertion approach, i.e. an idle time slot is not utilized, which affects the performance. The Insertion Scheduling Heuristic (ISH) algorithm, proposed by Kruatrachue and Lewis (1987), improves the HLFET algorithm by utilizing the idle time slots in the scheduling. Initially, it uses the same approach as HLFET to make a ready list based on the static b-level and schedule the first node in the ready list using the non-insertion approach. The difference is that, once the scheduling of this node creates an idle slot, ISH checks if any task in the ready list can be inserted into the idle slot but cannot be scheduled earlier on the other processors. The algorithm schedules such tasks as much as possible into the idle slots (Jin et al.; 2008).
Hwang et al. (1989) proposed the ETF (Earliest Time First) algorithm and computes the earliest start times at each step for all ready nodes and chooses the one with the smallest start-time. The earliest start time of a node can be computed by examining the start time of the node on all processors exhaustively. When two nodes have the same value at the earliest start-times, the ETF algorithm breaks the tie by scheduling the one with the higher static priority.

Wu and Gajski (1990) developed Modified Critical Path (MCP) algorithm that uses the ALAP of a node as the scheduling priority. The MCP algorithm first computes the ALAPs of all the nodes and then constructs a list of nodes in an ascending order of ALAP times. Ties are broken by considering the ALAP times of the children of a node. The MCP algorithm schedules the nodes on the list (one by one) such that a node is scheduled to a processor that allows the earliest start time using the insertion approach.

Sih and Lee (1993) proposed the DLS algorithm that used an attribute called dynamic level (DL), which is the difference between the static b-level of a node and its earliest start-time on a processor. At each scheduling step, the algorithm computes the DL for every node in the ready pool on all the processors. The node-processor pair which provides the largest value of DL is selected for the scheduling.

Ahmad et al. (1995) compared the performance of arbitrary processor network (APN) scheduling algorithms using edge weighted directed acyclic graph (DAG) with a set of homogeneous processors for minimization of the completion time. The algorithms were tested on a large number of test cases and global and pair wise comparisons were performed for ranking the algorithms according to the performance. The drawback was that they have not considered the effects of topology and routing strategy on the performance of these algorithms.

Kwok and Ahmad (1999) presented an extensive survey of algorithms for the static scheduling problem in parallel processing represented by a directed task graph on a multiprocessor system for minimizing the program completion time. A lot of heuristics such as branch-and-bound, integer-programming, searching, graph theory, randomization, genetic algorithms, and evolutionary methods have been developed. The objective of all the heuristic was to describe various scheduling algorithms and their functionalities in a contrasting fashion as well as examine their relative merits in terms of performance and time-complexity. They considered 27 algorithms for this survey and proposed a taxonomy that classifies these algorithms into four different categories, the UNC (unbounded number of clusters) scheduling, the BNP (bounded number of processors) scheduling, the
TDB (task duplication based) scheduling, and APN (arbitrary processor network) scheduling. They showed a lot of examples as well as analytical results to illustrate the functionality of surveyed algorithms. They compared different scheduling algorithms for a nine-task problem.

Amoura et al. (1999) proposed a heuristic called Divide Uniprocessor Tasks (DUT) for scheduling a set of ‘n’ independent dedicated multiprocessor tasks on three processors with the objective of minimization of makespan. Proposed heuristic was compared with previously known heuristics from literature and proved that the proposed heuristic was effective from others available in the literature.

Guan et al. (2002) considered a scheduling problem in which processors are arranged along a straight line and each job requires simultaneous processing by multiple consecutive processors. The problem was motivated by the operation of berth allocation. They considered the problem with the objective of minimizing the total weighted completion time of the jobs. Heuristic was also proposed for the problem and shows that the solution obtained by heuristic was almost 100% for the general case.

Davidovic and Crainic (2006) proposed two new sets of test problem instances for multiprocessor scheduling problem with communication delays (MSPCD). Task graph generators were also defined in a way appropriate to ensure that the corresponding problem instances obey the theoretical principles. The first set includes completely random instances and may be used for any multiprocessor architecture. The second set includes problem instances with known optimal solutions for specific processor interconnection networks. Several representative heuristics have been used for evaluating the sets of problem instances Task graph density, communication delay, number of processors and connections between them are the different parameters used for analyzing the dependency of the deviation of the heuristic solution from the best known or the optimal one. It was shown that the set of proposed problem instances provide the necessary variations and provides a theoretically sound and experimentally comprehensive framework for the fair evaluation of heuristics for the MSPCD.

Davidovic et al. (2007) focused on the comparison of list scheduling approaches and proposed a single pass deterministic algorithm, chaining, based on list scheduling techniques. McCreary et al. (1994) compared various heuristics for scheduling DAGs on multiprocessors. Five heuristics were considered for comparisons, including graph based method, a list scheduling technique and three critical path methods. They applied these heuristics to ten specific DAGs which represent program dependence graphs of important
applications. They concluded that critical path heuristics perform well for the dependence graph which considers the uniform edge weights, but ineffective when the edge weights are non uniform. They mainly considered LAST algorithm among the category of list scheduling and concluded that LAST was ineffective for most of the DAGs used for the comparison except for FFT. They further showed that priority based schemes like LAST were difficult to generalize and need to be considered on the merit of the priority parameter. Further graph decomposition method produces most uniform results and almost for every graph, its efficiency was best or second best.

Johannes (2006) minimized makespan for offline and online scheduling parallel jobs with release dates on identical parallel machines using greedy like algorithms. He showed that no list-scheduling algorithm can achieve a better performance guarantee than 2 for the non pre-emptive problem with parallel jobs and release dates, no matter which priority list is chosen. He also stated that the list scheduling algorithms work well in the online setting where jobs arrive over time and the length of a job becomes known only. He claimed that no list scheduling algorithm has a constant competitive ratio and proposed algorithm was the first online algorithm for scheduling parallel jobs with a constant competitive ratio.

Benoit et al. (2009) proposed a fault tolerant scheduling heuristic for mapping precedence task graphs on heterogeneous systems based on active replication scheme. They designed three algorithms FTSA, MC-FTSA & RFTSA. They showed that the FTSA is superior to FTBAR both in terms of computational complexity and quality of the resulting schedule. The MC-FTSA generates better schedules than FTBAR when there are small numbers of failures and RFTSA produces schedules with a better latency compared to FTBAR, with the extra property of being more reliable schedules.

Rajak and Katti (2013) proposed an algorithm based on fork join method for task scheduling (TSFJ) in a multiprocessor system. The performance of the TSFJ algorithm was measured on 9 tasks DAG and comparison was based on scheduling length, speedup and efficiency with heuristic algorithms. It was proved that the proposed TSFJ algorithm gives minimum scheduling length as compared to HLFET, MCP, ETF and DLS algorithms. The normalized scheduling length of TSFJ is almost same as ETF and DLS algorithms. The load balancing is achieved higher in TSFJ algorithm and lower in ETF and DLS algorithms.
The heuristic algorithms follow the procedure that narrow the search for a very small portion of the solution space. However, due to the greedy nature, heuristic-based approaches are not likely to produce consistent results for a wide range of problems, especially when the complexity of the scheduling problem increases. Therefore metaheuristics are the appealing choice for the solution of multiprocessor task scheduling with increased complexity and can be used for a wider range of the problems.

2.3 METAHEURISTICS

Metaheuristics are the methods that guide the search space, using surrogate algorithms. Starting from an initial solution, metaheuristics improve it iteratively until a stopping criterion is met. The stopping criterion can be elapsed time, number of iterations and the number of evaluations of the objective function and so on. Osman and Laporte (1996) provide an extensive bibliography on metaheuristics and its applications in combinatorial optimization problems. Voss et al. (1999) described a metaheuristic as “Iterative master processes that guides and modifies the operations of subordinate heuristics to efficiently produce high quality solutions”. Olafsson (2006) presented the survey regarding the use of metaheuristics for simulation optimization. To illustrate the key points, three metaheuristics namely genetic algorithms, tabu search, and the nested partitions methods were discussed. The main focus was on bridging the current gap between the practical use of such methods and research with some promising directions for research. The emphasis was on the two issues which was accounting for the simulation noise in guiding the search and deriving convergence results. For addressing the first issue, ideas for ordinal optimization and statistical selection can be incorporated which helps in determining whether a move should be made or not. Also the local search can be embedded into a global guidance system that allows the search to recover from incorrect moves. For the second issue, the ability to make convergence statements was of value, but the focus should be shifted away from the traditional asymptotic convergence results to finite time behavior and goal softening.

Some of the widely researched metaheuristic algorithms include adaptive search techniques like genetic algorithm (GA), artificial immune system (AIS) and neural networks, neighborhood search techniques like simulated annealing (SA), and tabu search with their hybrid versions.
2.3.1 Genetic Algorithm

Genetic Algorithm (GA) based methods, inspired by Darwin’s theory of natural selection and survival of the fittest, were invented by Holland (1975) and first proposed for scheduling problems by Goldberg (1989). In the Simple GA based approach, the various stages like evaluation, selection, crossover and mutation are repeatedly executed after initialization until a stopping criterion is met. The algorithm works on multiple solutions simultaneously.

Hou et al. (1994) proposed the first and most important work that has used GA for multiprocessor task scheduling which uses the height of tasks in input DAG with the objective of minimizing the schedule length. Schedules in the form of list of computational tasks were presented which eliminates the need for considering the precedence relation between tasks in different processors. They further considered the reproduction operators in such a way that always produce the valid chromosomes (schedule). They tested the proposed method on random task graphs and Newton Euler inverse dynamic equation task graphs for Stanford manipulator & elbow manipulator. For comparison purposes, they run list scheduling algorithm from literature on random task graphs and it was also proved that proposed GA approach performs better than list scheduling algorithm and was within the 10% of optimal schedule.

Bohler et al. (1999) presented the genetic algorithm for minimizing the schedule length for a general task graph on a multiprocessor system. The proposed genetic algorithm was easily adaptable to a variety of task graphs. Different experiments have been performed on task graphs on 20, 40 & 50 tasks and it is concluded that the proposed algorithm found an optimal solution in 20 generations only for 20 tasks graph problem. The task graphs of 40 tasks, the proposed algorithm found a good solution in 100 generations, but for 50 tasks, the solution was found in 500 generations.

Golub et al. (2002) presented efficient genetic algorithms for scheduling precedence constrained task graphs with negligible inter task communication onto multiprocessors without taking communication channels into consideration. Experiments have been conducted on the different NP hard problems from the literature and proved that the proposed genetic algorithm provides optimum or near optimum solutions at a very reasonable time. They also suggested that the best results can be obtained by optimizing the parameters of the proposed algorithm.
Rinehart et al. (2003) proposed a genetic algorithm (BCGA) for scheduling task graphs onto multiprocessors. The BCGA was based on a bi-chromosomal representation which is capable of being incorporated into a cluster/merging optimization framework. The proposed algorithm outperforms the genetic algorithm for scheduling task graphs with the similar complexity.

Wu et al. (2004) developed a genetic algorithm (GA) for the multiprocessor task scheduling problem using partial solutions to direct the GA search. The proposed approach eliminates the need for special operators or repair mechanisms to ensure formation of valid chromosomes. The computational experiments were performed on the nine problems and the performance comparison has been done with three traditional scheduling techniques: ISH, DSH, and CPFD. It was observed that the proposed GA outperforms the two out of nine problems and also provides comparable results in the rest of the problems. They further stated that GA ‘s performance can still be maintained by providing additional resources such as a larger population or more generations at the cost of longer execution time for large size problems. They also verified the performance of the GA with heterogeneous processors and observed that GA achieves the top performance on seven out of nine problems and is second-best on remaining two problems. It was concluded that the proposed GA was simple to use, requires minimal problem specific information and effectively adapt to dynamically changing environments at the cost of longer executing time than traditional scheduling algorithms.

Rahmani and Vahedi (2008) presented a genetic algorithm based on a new fast technique, named elitism stepping technique for the task scheduling problem in multiprocessor systems. The objective was to reduce the schedule length within an acceptable computation time. Experiments were performed on various task graphs and compared the results with the previous genetic algorithm, basic genetic algorithm proposed by Hou et al. (1994). It was concluded that the proposed algorithm outperforms the basic genetic algorithm in terms of schedule length and computation time with the same conditions,

Oguj and Ehlan (2005) proposed a genetic algorithm for hybrid flow shop scheduling problem with multiprocessor tasks. They developed a new crossover operator (NXO) and compared with PMX crossover. Some preliminary tests were performed for tuning of different parameters of GA such as population size, crossover rate and mutation rate. Different computational experiments were also performed for predicting the performance of GA with different genetic operators and found that GA with NXO crossover and
insertion mutation performs better in terms of the percentage deviation of the solution from the lower bound value. They also compared the results with tabu search algorithm and found that the genetic algorithm was a better approach for the considered hybrid flow shop multiprocessor scheduling problem.

Arumaikkannu et al. (2005) described a Genetic Algorithm (GA) with Design of Experiments (DoE) approach for prediction of the optimized surface roughness and porosity characteristics of the parts produced by FDM. They developed a mathematical model (MM) using Response Surface Methodology (RSM) for the prediction of the selected parameters. They used the MM in GA as fitness function for finding the optimal sets of process parameters and prediction of the corresponding surface quality characteristics. They concluded that the developed GA could be used for prediction of the optimal process (input) parameters as well as the output responses accurately.

Jelodar et al. (2006) proposed a novel representation for solving the multiprocessor task scheduling problem using genetic algorithm (GA) with the objective of minimizing the total execution time. Duplication of tasks was allowed in the method and therefore no need of adding repair mechanisms for valid chromosomes. They stated that the novel representation was good in terms of length of schedule because of its adaptability for the given problem. They proved that order of search space has been reduced and consequently outperforms some recently studied GA based scheduling methods over 120 times with respect to the number of fitness evaluations.

Shenassa et al. (2006) proposed a novel intelligent solution for multiprocessor systems with the objective of minimizing the entire run time and average response time. The proposed method was based on genetic algorithm and chromosome background tree. The proposed method considered the time of transferring data between processes for minimization of the entire run time and average response time. They claimed that the proposed method was an optimal method for solving the problem and was more efficient than other related methods from the literature.

Jin et al. (2008) compared nine scheduling algorithms for multiprocessor task scheduling with communication delays on two well known problems of linear algebra: LU decomposition and Gauss–Jordan elimination. The algorithms considered for comparison are min–min, chaining, A*, genetic algorithms, simulated annealing, tabu search, HLFET, ISH, and DSH with task duplication. Based on experimental results, they concluded that Duplication Scheduling Heuristic (DSH) which is one-shot heuristic algorithm can provide short scheduling time and schedules with the minimum makespan,
but at the cost of the extra overhead of duplication of tasks on multiple processors. So, One-shot heuristic algorithms without task duplication can provide adequate performance and fast scheduling time and Insertion Scheduling Heuristic (ISH) is the best of this group.

Next the iterative group of algorithms such as genetic algorithms, simulated annealing, tabu search, and A* except simulated annealing require an order of magnitude longer computation time, but provide better solutions with a shorter makespan than the one-shot heuristic algorithms without task duplication. They concluded that in this group GA and tabu search provide the best solutions. They further concluded that use of these algorithms can be justified in the situations where scheduling can be done off-line, the need for repeated execution of the schedules or the makespan of the application is significantly longer than the scheduled time.

Hwang et al. (2008) addressed the problem of multiprocessor task scheduling with communication cost. The problem was represented as a directed acyclic graph (DAG). For solving this problem, a priority based GA was proposed that follow a new encoding mechanism with a multi-functional chromosome and so called priority-based multi-chromosome (PMC). New crossover method weight mapping crossover (WMX) was designed for the efficient working of PMC. Experiments were performed on several task graphs and compared with various list scheduling heuristic algorithms and previous GA approaches. From the results, it was concluded that proposed priority based GA has effective performance in various parallel environments of scheduling methods.

Kahraman et al. (2008) considered the makespan minimization for hybrid flow shop scheduling using an effective genetic algorithm. They concluded that the proposed GA algorithm obtained better with an average deviation of the GA Algorithm is 1.50 % when compared to AIS and B&B on the standard benchmarks. It has also been concluded that CPU times of the GA are much smaller than AIS and B&B and hence the proposed GA is a good problem solving technique for a scheduling problem.

Bonyadi and Moghaddam (2009) proposed Bipartite Genetic Algorithm (BGA) for minimizing the makespan for the multiprocessor task scheduling problem. The proposed method considers two parts and each part were based on different genetic schemes, such as genome presentation and genetic operators. The first part applied a genetic method for finding an adequate sequence of tasks and the second one was used to get the best match processors. Preliminary test was performed in parameters setting of GA for better performance. The results were compared with GA based & heuristic based
algorithms from literature in terms of STD, average makespan, best obtained makespan and iterations and showed that the proposed method had a better makespan with at least 10% less iterations in most of the cases.

Alvarez et al. (2009) showed that how Genetic Algorithms can be used coupled with RSM for the optimization. Response Surface Methodology (RSM) is a combination of experimental designs and statistical techniques for empirical model building and optimization which has been applied to a wide range of fields. Genetic algorithms have been very useful in the optimization of the response variable and also in multi-response cases. The Response Surface can be optimized using a genetic algorithm in order to find the optimum values of the independent variables. They stated the advantages of GAs such as flexibility, variable independent etc. They addressed a major drawback of GA is influence of genetic operators and parameters in the convergence of the algorithm and in the final solution. Despite this, they stated that genetic algorithms have opened up many possibilities in RSM.

Goh et al. (2009) proposed a hybrid evolutionary algorithm (HEA) for the solution of the heterogeneous multiprocessor scheduling problem with precedence constraint and makespan minimization. The proposed method considers the two local search heuristics partial list scheduling and duplication scheduling approach for exploiting the intrinsic structure of the solution and specialized genetic operators for promoting the exploration of the search space. Experiments were performed on a set of benchmark problems and from the results, it was concluded that the proposed genetic operators, when coupled with the local search operators performed better where any one of the operators are omitted. They also showed that the use of local search operators, particularly duplication local search into the overall proposed algorithm, reduces the convergence time.

Sedaghat et al. (2009) proposed a multi-objective genetic algorithm (MOGA) for scheduling, static soft real time tasks on a heterogeneous multi-processor system with the objective of maximizing the system utilization and minimizing the total tardiness. They considered the problem with some real world constraints such as the precedence relationship between tasks, different arrival time for each task as well as communication delays between the processors. They used an adaptive weight approach (AWA) for fitness function that utilizes some useful information from the current population to readjust weights. For comparison purposes, they also proposed two greedy algorithms in which each algorithm aims to optimize a single objective either idle time or communication delay. They performed the experiments on two types of graphs, i.e. sparse and non-sparse.
From the results, they concluded that the proposed MOGA performs better and sometimes same as other two greedy algorithms, but the computation time for MOGA was more than another two greedy algorithm.

Kahraman et al. (2010) proposed Parallel Greedy Algorithm (PGA) for Hybrid Flow Shop Scheduling with Multiprocessor Task (HFSMT) which is applied by two phases iteratively, called destruction and construction. Four constructive heuristic methods were also proposed for solving the HFSMT problems and different preliminary test for optimal values of control parameters, such as population size, subgroup number, and iteration number was performed by a full factorial experimental design. From the results, it was concluded that the proposed parallel greedy algorithm approach was very effective in terms of reduced total completion time or makespan.

Engin et al. (2011) proposed an efficient genetic algorithm (GA) approach for hybrid flow shop scheduling with multiprocessor task (HFSMT) problem with the objective of minimizing the makespan. The proposed GA was more efficient than simple GA because it uses a new mutation operator called Neighborhood Based Mutation (NBM) and examines the performances of various reproductions, crossover and mutation operators, and ratings for HFSMT problems by using the full factorial experimental design. Experiments have been performed on 240 instances from the literature and it was concluded that the proposed approach was effective in terms of a reduced makespan for the attempted problems.

Chitra et al. (2011) considered the multi-objective task scheduling problem in heterogeneous distributed computing systems (HDCS). They stated that for reducing the impact of failures on an application running on HDCS, scheduling algorithms must be devised which not only minimized the schedule length (makespan) but also the failure probability of the application (reliability). Therefore, task scheduling problem with two objectives of makespan and reliability index was considered and developed two Multi-Objective Evolutionary Algorithms such as Multi-objective Genetic Algorithm (MOGA) and Multi-objective Evolutionary Programming (MOEP) with non-dominated sorting. Different experiments were performed on different random task graphs and a real time numerical application graph. From the results it was proved that, MOEA algorithms were well suited for obtaining good Pareto-optimal solutions in a single run for task scheduling problem by considering the above two objectives and MOGA results into reduced computation time compared with the weighted sum method in the literature. They also
stated that the difficulty in selecting appropriate weights for the objectives to solve the multi-objective problem can be overcome by the proposed MOEAs.

Heidari et al. (2012) proposed a genetic algorithm with node duplication (NGA) for the multiprocessor task scheduling problem. Efficiency of NGA based technique was compared against some of the existing deterministic scheduling techniques for minimizing inter-processor traffic communication. A comparative study of the results obtained from simulation shows that a genetic algorithm can be used for scheduling the tasks to meet deadlines, in turn to obtain high processor utilization. Performance of NGA, GA, List Scheduling and FCFS was found to be almost 64.1%, 55.55%, 52.08, and 41.033% respectively.

2.3.2 Simulated Annealing

Simulated Annealing (SA) algorithms, based on the analogy of the annealing process of metals, were proposed by Metropolis et al. (1953) and were first applied to combinatorial optimization problems by Kirkpatrick et al. (1983). SA is considered to be an improvement heuristic where a given initial solution iteratively improves upon. A number of researchers have applied SA to scheduling problems. Some of them include Osman and Potts (1989) and Ogbu and Smith (1990, 1992). The advantage of SA based methods is their ability to avoid getting trapped in local minimums by accepting solutions that worsen the objective function. The number of iterations required to converge to the optimal solution depends on the parameters selected for the heuristic. The parameters may be classified into two categories viz. Problem-specific like initial solution, neighborhood generation and generic like temperature, stopping criterion among others. Zegordi et al. (1995) showed that a good set of parameters can result in much quicker convergence to the optimal solution. Thus, in SA based methods, it is imperative to obtain the most desirable set of parameters in order to achieve faster convergence within a reasonable computation time.

Davidovic et al. (2007) considered the problem of multiprocessor task scheduling with communication delay which depends upon the amount of exchanging data between a pair of dependent tasks & distance between processors in the architecture of homogeneous processors. They developed two Mixed-Integer Bilinear Programming formulations for the solution. They used two methods to linearize and tested them with CPLEX. A set of instances with known optima is taken as a benchmarking tool for the
new heuristic algorithms. For further improvements they modified the original model by using Variable Neighborhood Search heuristic.

Kalashnikov and Kostenko (2008) proposed a sequential and parallel algorithm based on simulated annealing for multiprocessor scheduling problem and compared with classical sequential, sequential & parallel algorithms. They showed that the sequential algorithm reduces the time of problem solution up to 3 times as compared to the classical algorithm of simulated annealing. It retains and improves the quality of resulting solutions in several cases. Also the parallel algorithm can be efficiently used on a local network as no intensive traffic of data exchange is required between the network nodes.

Nadathur et.al (2008) presented two statistical optimization approaches for scheduling the task dependence graphs with variations in execution time onto the heterogeneous multiprocessor system. They proposed a statistical analysis based on accurate Monte-Carlo simulations compared to static approaches based on worst-case estimates for two scheduling algorithms. One is a heuristic based on a critical path analysis of the task dependence graph and 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.

Dhingra and Chandna (2010) developed the different heuristic based simulated annealing algorithms for efficient scheduling of jobs in a multi-objective flow shop. The initial seed sequence was developed from the constructive heuristics and combined with the seed sequence in the simulated annealing. It was proved that the heuristics based hybrid simulated annealing provides better results, especially for large sized problems when compared to others in a reasonable time.

Wang et al. (2011) proposed and developed a simulated annealing (SA) algorithm for minimization of makespan in the hybrid flow shop scheduling problem with multiprocessor tasks in which three decode methods (List Scheduling (LS), permutation scheduling (PS), and first-fit method (FF)) are used to obtain the objective function value for the problem. They improved the proposed SA by adding a new neighborhood mechanism for generating neighbor solutions and performed the tests on two benchmarks from literature and observed that the three decode methods would not be dominated by each other. They further observed that proposed SA outperforms the other algorithms in terms of solution quality and computation time, especially for large-scale problems.

Hasani et al. (2014) considered the problem of scheduling a set of jobs on two parallel machines for makespan minimization with the constraint that setup required by
each job must be done by a single server. Two approximate algorithms, simulated annealing and genetic algorithm were proposed for the solution of the problem. Experiments were performed on the instances up to 1000. They compared the results with existing algorithms from the literature and concluded that the proposed algorithms outperformed all heuristics available in the literature in terms of the deviation from the known lower bounds. They also stated that the proposed algorithms performed better because of composite neighborhood (mutation), generating several neighbors from sub-neighborhoods with different probabilities and taking the best solution as generated neighbor.

### 2.3.3 Hybrid Metaheuristics

Another work that presents a new technique based on problem-space genetic algorithms (PSGAs) (Hou et al.; 1994) utilized the search power of GAs in combination with list scheduling heuristics, such as ISH and DSH (Kruatrachue and Lewis; 1987) The results showed that the GA when combined with heuristics could work efficiently and schedule the tasks on several processors. In literature, different methods have attempted to combine the heuristic and genetic approaches for the solution of the problem.

Chang and Hsu (1995) developed two algorithms based on an A* algorithm to efficiently solve the task scheduling problem on uniprocessor and multiprocessor system. The objective function considered was total completion time. They showed that the proposed approaches can achieve the optimal schedule by reducing search tree sizes.

Sutar et al. (2006) developed the memetic algorithm based on the combination of GA and SA for solving the multiprocessor task scheduling problem with the objective of makespan minimization. They considered the division of tasks into subtasks and represented as DAG. They stated that GA has been widely accepted for this kind of optimization but there are some shortcoming of GA in terms of complexity and potential of convergence to optimal solution.

Yoo and Gen (2007) 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 has been used for multiple objectives. The convergence of the GA is improved by introducing the probability of SA as the criterion for acceptance of the new trial solution. Various
experiments have been performed to check the effectiveness of proposed algorithm & from results, it is concluded that the results of the proposed MOHGA are better than that of other algorithms. Further, they want to include the communication time in their scheduling algorithm.

Sivanandam et al. (2007) 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 have been performed on the benchmark problems and shows that the proposed hybrid method was effective and efficient in finding near optimal solutions.

Azghadi et al. (2008) developed an immune genetic approach and combine it with the proposed heuristic resulting in a hybrid scheme for the multiprocessor task scheduling problem. This hybrid scheme uses initial population based on heuristic approach which results in faster convergence & best solution in some cases. They showed that the proposed hybrid approach was very much effective in multiprocessor scheduling systems.

Dahal et al. (2008) 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. The comparative performance of using heuristics EDF and SCTF with GA has been presented through a set of experiments. From comparative analysis, it is concluded the SCTF when hybrid with GA provides better performance as compared to the EDF. They also stated that the proposed algorithm can obtain feasible solutions with high utilization of processors for a task set of 20.

Ercan (2008) proposed hybrid particle swarm optimization (PSO) algorithm for minimizing the maximum completion time of all jobs for the multiprocessor task scheduling in the hybrid flow shop environment. It was found that the proposed algorithm approximates the performance of the GA and ACS algorithms. It has also been concluded that PSO hybridized with SA provides the best results at the cost of increased complexity.

Jouglet et al. (2009) proposed a memetic algorithm with a combination of genetic algorithm (GA) and constraint programming based on branch & bound algorithm (CP) for the hybrid flow shop scheduling with multiprocessor tasks. They first implemented GA, then CP & lastly combined both in memetic algorithm (MA). Performance comparison of the three algorithms has been done by executing them on several sets of instances from
the literature & proved that MA performs better than GA & CP in terms of solution quality & efficiency. However GA provides comparable results with MA in terms of quality produced, but still in term of efficiency MA was better. Finally, they concluded that the memetic algorithm proves to be an effective approach for hybrid flow shop scheduling problems with multiprocessor tasks.

Jahanshahi et al. (2009) proposed Learning Algorithm which uses the learning automata as local search in the memetic algorithm for overcoming the weakness of the GA based method. The problem of scheduling tasks in a distributed system with the objective of minimizing Makespan and communication cost while maximizing CPU utilization was considered. They proved that the proposed method outperforms the existent GA based method in terms of communication cost, CPU utilization and Makespan.

Dhingra and Chandna (2010) proposed hybrid GASA for the solution of the multi-criteria flow shop scheduling problems. From the computational analysis, it was found that that the proposed hybrid GASA provides better results when compared to those obtained with simple GA and SA alone in terms of solution quality especially for larger sized problems.

Mohamed and Awadalla (2011) developed a modified list scheduling heuristic (MLSH) and a hybrid approach composed of Genetic Algorithm (GA) and MLSH for task scheduling in the multiprocessor system. Three different representations for the Chromosomes of genetic algorithm was proposed: task list (TL), processor list (PL) and a combination of both (TLPLC). Computational experiments were performed on different random and real-world application graphs such as Gauss-Jordan, LU decomposition, Gaussian elimination and Laplace equation solver problems and compared with the most related algorithms like: list scheduling heuristic algorithm (LSHs, Bipartite GA (BGA) and Priority based Multi-Chromosome (PMC)). From the results, they concluded that among the three representations (TL, PL & TLPLC) GA with TLPLC performs better. They also compared the TLPLC GA with other GA based algorithms and found that proposed approach outperforms the others in terms of best makespan, average makespan and processor efficiency.

Wen et al. (2011) proposed a heuristic-based hybrid genetic-variable neighborhood search algorithm which incorporates GA with both Variable Neighborhood Search (VNS) and a heuristic extracted from traditional list scheduling algorithms for the minimization of makespan in the heterogeneous multiprocessor scheduling problem.
Experiments were performed on benchmark problems extracted from three well-known parallel applications and based on results; the performance of the proposed approach is compared with four related algorithms, HEFT, AIS, VNS and IGA. It was concluded that the proposed algorithm consistently outperforms the other four algorithms in terms of schedule quality, with the exception that the task duplication-based IGA algorithm achieves the highest quality in several test cases with a high CCR value.

Roy et al. (2012) proposed a heuristic based Genetic Algorithm for task scheduling in multiprocessor systems by choosing the eligible processor on educated guesses. Variation of HLFET algorithm with genetic algorithm was proposed and experiments were performed on Standard Task Graphs (STG). From results it has been found that the algorithm has better average makespan than Elitism stepping method in lesser number of evaluations.

Chen et al. (2012) combined the idea of multi-objective evolutionary algorithm and heuristic algorithm for optimizing the multiple objectives simultaneously for task scheduling in the heterogeneous distributed systems with minimization of the last task’s finish time (makespan) and maximization of the system reliability probability. HEFT algorithm was combined with non-domination sort genetic algorithm (NSGA) and experiments were performed on randomly generated graphs and the graphs of some real applications such as Gaussian elimination problems and fast Fourier transformation. The results were compared with two well known methods from literature, i.e. heterogeneous earliest finish time (HEFT) algorithm and Critical Path Genetic Algorithm. From the comparison, it was concluded that the proposed method outperform the other two in terms of makespan, scheduling length ratio (SLR) & system reliability.

Xu et al. (2012) formulated the tugboat scheduling problem as a multiprocessor task scheduling problem (MTSP). They proposed hybrid simulated annealing-based ant colony algorithm for minimization of the total operation times for all tugboats in a port. They proved that the proposed hybrid algorithm was very much effective as compared to others.

Yamazaki et al. (2013) proposed a multistep scheduling algorithm for task scheduling problem in a heterogeneous environment. They also proposed a clustering algorithm based on SCAN which is an algorithm for finding clusters in a network. The clustering algorithm based on SCAN can find task parallelism in a task graph. They stated that the original SCAN was an algorithm for a non directed graph and however, the clustering algorithm based on SCAN can be applied to a DAG and showed that the
proposed multistep scheduling algorithm is superior to others in terms of computational time.

2.3.4 Miscellaneous

Miller (1982) addressed the problem of scheduling different categories of programs, called task groups on the dissimilar processors. Multiprocessor architecture was proposed in which processors were chosen to have different implementations of a single system wide instruction set, i.e. the Single Architecture Heterogeneous Multiprocessor (SAHM) system with the goal of finding an effective scheduling algorithm for any SAHM. The linear programming technique was used for the static case of such multiprocessor system where task group loads & task/processor suitability values are known in advance. So, there were two main results with first one concern the determination of which task group to be executed by each processor within SAHM system & second one with the practical task scheduling application of this result.

Wang (1992) stated the advantages of a dynamic round robin (RR) task scheduling which was the extension of the traditional RR method. The RR method does not issue load balance message which reduces the burden on communication network of a multiprocessor system. Further, the RR method does not use task migration mechanism which reduces the network load. For evaluating the speedup behavior & efficiency of RR method, he programmed a simulator, which can simulate the multiprocessor system with 4, 8, 16, 32 PEs. From the simulation results, it was concluded that speedup was increased proportionally to the number of PEs which indicate that the RR method provides good result when applied to large scale parallel system. He further stated that the RR method can be applied to a wide range of applications because of its efficiency & flexibility.

Hoogeveen et al. (1994) investigated the computational complexity of scheduling multiprocessor tasks with prespecified processor allocations. They considered two criteria: minimizing the schedule length and minimizing the sum of the task completion times. Also the complexity of the problems is investigated when precedence constraints or release dates are involved.

Blazewicz and Liu (1996) considered a deterministic multiprocessor scheduling problem in which some of the tasks may require more than one processor at a time. The problem was considered with the assumptions of non-pre-emption & precedence constraints for minimization of the makespan. They proved that the problem of minimizing the schedule length was NP hard for three processors, even if all the tasks
have unit processing times & precedence constraints form a set of chains. They also proved that if processor requirements of the tasks in the chain were uniform or monotonically decreasing/increasing for ‘m’ processor case, it can be solved in polynomial time. Also, pre-emptable tasks & uniform processors can be included for further work.

Drozdowski (1996) considered the scheduling of multiprocessor tasks which require multiple processors simultaneously. This concept of scheduling emerged with the advent of parallel computing. Scheduling multiprocessor tasks problem for both parallel & dedicated processors was considered & also presented the results for the both.

Bianco and Blazewicz (1997) considered the pre-emptive deterministic multiprocessor task scheduling in which tasks require more than one processor at a time and then scheduling of such tasks in time windows on particular processors. They proved that low order polynomial time algorithms can be considered for solving the makespan (Cmax) criterion with two and three processors. Even when the numbers of processors are fixed, the said problem for both Cmax and Lmax criterion can be solved in polynomial time by use of the linear programming procedure.

Chen and Lee (1999) considered a multiprocessor task scheduling problem and proposed a pseudo polynomial algorithm to solve optimally the two-machine problem and a combination of a fully polynomial scheme and a heuristic to solve the three-machine problem. Then, they extended it to a general m-machine problem. They provided an effective lower bounding scheme which lays the foundation for optimal solution of the general m-machine problem. They also provided a fully polynomial time approximation algorithm combined with a heuristic algorithm for the ‘m’ machine problem. They proved that the algorithms can also be applied for solving the special case of the three-machine problem in pseudo polynomial time. They demonstrated that the both pseudo polynomial algorithms (for a two - machine and three-machine problems) are much more efficient than those in the literature.

Liou and Palis (1997) showed the effectiveness of two phase method of scheduling, in which task clustering & load balancing (LB) was used for cluster merging. Experiments were performed on a number of task graphs and from the computational results, it was concluded that the two-phase method consisting of task clustering and load balancing is a simple & an effective strategy for scheduling task graphs on distributed memory parallel architectures.
Drozdowski and Dell’Olmo (2000) considered the problem of scheduling unit execution time and preemptable multiprocessor tasks on parallel identical processors for minimizing the flow time and mean weighted flow time. Complexity status of the problem was analyzed and concluded that when the tasks have unit execution time & the number of processors is arbitrary then the problem is NP hard, but if the number of processors are fixed or a power of 2 the problem can be solved in polynomial time.

Baruah (2001) considered the scheduling of periodic tasks on the uniform multiprocessor system. The uniform multiprocessor system consists of a set of processors in which each processor is characterized by its computing capacity rather than identical processors where all processors have the same computing power. It has been found that the study of scheduling periodic real time tasks on identical processors cannot be extended for the uniform multiprocessor system. It is also concluded that in the absence of Integer Boundary Constraint (IBC), the feasibility-analysis on uniform multiprocessors cannot be utilization based and has a runtime complexity that was not quite linear while in both the uniprocessor and identical multiprocessor case feasibility analysis test can be implemented with a runtime complexity that is linear in the number of tasks. It was also proved that in uniprocessor & identical multiprocessor case a system of periodic tasks that was feasible in the absence of the IBC remains feasible if the IBC is also imposed while in uniform multiprocessor systems, feasibility analysis in the absence of the IBC can be done in an efficient polynomial time, but determining feasibility is NP hard if the IBC is imposed.

Andersson et al. (2001) proposed a static priority scheduling algorithm for periodic tasks on a platform comprised of several identical processors. The proposed algorithm is an extension of the uni-processor rate monotonic scheduling algorithm. Performance of proposed algorithm was evaluated using theoretical & experimental results, with respect to other static priority multiprocessor scheduling algorithm. It was concluded that the proposed algorithm has successfully scheduled any periodic task system with a worst-case utilization no more than a third the capacity of the multiprocessor platform. It was also confirmed that no static-priority multiprocessor scheduling algorithm (partitioned or global) can guarantee schedulability for a periodic task set with utilization higher than one half the capacity of the multiprocessor platform.

Chen et al. (2002) proposed a semi normal scheduling algorithm for the special case of multiprocessor parallel job scheduling problem which is represented as P\textsubscript{3}|fix|C\textsubscript{max}. They generalized the concept of normal scheduling by semi-normal
scheduling and compared the results with classical Graham List scheduling and proved that proposed algorithm was an approximation algorithm which improves the approximation ratio of 7/6 to 9/8 of the considered problem.

Baptiste and Schieber (2003) studied the problem of scheduling ‘n’ tasks to ‘m’ parallel processors with the constraints of release date & due date for minimizing the maximum tardiness using the two algorithms. First one was based on a linear programming model of the problem & second was combinatorial algorithm. They concluded that the minimum maximal tardiness can be computed in a polynomial time.

Blazewicz et al. (2003) considered the problem of scheduling multiprocessor tasks in the windows of processor availability. They considered the preemptive version of scheduling and proposed three algorithms for three different cases: one is the case of maximum lateness criterion and a fixed number of processors, next is the case of schedule length criterion when tasks have various ready times and require either one or all processors, and last is the case of schedule length criterion when the sizes of the tasks are powers of 2. They proved that the proposed algorithms were able to provide the solution in a polynomial time.

Kang and Aggrawal (2003) presented a task duplication-based, scalable scheduling algorithm (S3MP) to schedule the tasks of a DAG onto a bus-based Symmetric Multiprocessor (SMP) environment. Experiments have been performed on various practical DAGs & compared the schedule length & ratio of communication to computation cost. From the simulation results, it has been shown that the proposed algorithm achieves performance improvement over the modified STDS algorithm of literature for high CCR values. It is also proved that the proposed algorithm has a worst case time complexity of $O(V^2)$, where $V$ is the number of nodes of the DAG. Further, it was planned to use the proposed scheduler in a network cluster environment.

Blazewicz et al. (2004) considered the problem of scheduling malleable tasks on $m \times n$ parallel processors with the objective of minimizing makespan. A malleable task is one which can be executed by several processors simultaneously. They also stated that processing speeds of the tasks depend nonlinearly on the number of processors granted. They presented an $O(n)$ algorithm to solve the problem when processing speed functions are convex. They further said that if processing speed functions are concave and the number of tasks is a constant, the problem can be solved in polynomial time. They further relaxed the problem with the condition that the number of processors allocated to each task is not required to be integer & proved that such problems can be solved in $O(n$
max\{m, n \log_2 m\}) time. They proved that for n = 2 or n = 3, an optimal solution for the relaxed problem can be converted into an optimal solution for the original problem in a constant time.

Oguz and Ercan (2005) 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 processor requirements. Three well known local search heuristic algorithms, Simulated Annealing, Tabu Search, and Genetic Algorithms were employed. Experiments were also performed in evaluating the reduction achieved in completion time by each heuristic and compare the results with simple list based heuristics. The results show that local search heuristics significantly outperforms the list-based heuristics, but due to their large computation times, SA, TS, or GA can be used in deterministic cases.

Kwok and Ahmad (2005) proposed optimal algorithms for static scheduling of task graphs with arbitrary parameters to multiple homogeneous processors. The proposed algorithms are based on the A* search technique with a computationally efficient, cost function and a number of state-space pruning techniques. A parallel algorithm based on reducing interprocessor communication as well as with static and dynamic load-balancing schemes to evenly distribute the search states to the processors. & approximate algorithms that guarantee a bounded deviation from the optimal solution, but executes in a considerably shorter time are proposed. From extensive experimental evaluation of the algorithms, it is concluded that the parallel algorithm with pruning techniques is an efficient scheme for generating optimal solutions of reasonably large problems while the approximate algorithm is effective if slightly degraded solutions are acceptable.

Wojtyla et al. (2006) proposed artificial immune system (AIS) for scheduling parallel programs given by DAG onto processors. The proposed algorithm has the ability to extract the knowledge from previous schedules when scheduling new, potentially similar, programs. It is concluded that the proposed algorithm provides the comparable results when compared with genetic algorithm (GA) but at the cost of fraction of function evaluations.

Baruah (2006) determined the feasibility conditions for non-preemptive multiprocessor scheduling upon processing platform comprised of several identical processors. He presented sufficient rather than exact feasibility conditions for determining whether a given periodic task system will meet all deadlines if scheduled non-
preemptively upon a multiprocessor platform using the earliest-deadline first (EDF) scheduling algorithm. He stated that the used EDF-feasibility analysis algorithm can be used for systems comprised of sporadic tasks & non-concrete periodic/sporadic task systems. He also proved that sufficient feasibility tests presented was indeed robust, i.e. if a periodic task system, it passes these tests, and then it is guaranteed that EDF (t) will meet all deadlines provided all jobs have execution requirements no more than the worst-case execution requirements of the periodic tasks that generate them. Also the proposed feasibility test is particularly suited for use upon systems that can be modeled as liquid task systems.

Fumin et al. (2007) presented a novel real-time fault-tolerant Rate-Monotonic Best-Fit algorithm (ERMBF) for multiprocessor systems. The proposed algorithm was based on pre-allocation of a fixed number of processors before task assignment, which enlarge searching spaces for task copies, thus making as many backup copies to be executed as passive backup copies as possible. Also, when a new processor is added, task copies reallocation strategy is used to obtain better assignment configuration. As a result, fewer processors are needed and schedulability of system is improved. Through their experimental results, they proved that ERMBF significantly outperform existing comparable algorithms in terms of schedulability.

Angelelli et al. (2007) considered a semi-online version of the multiprocessor scheduling in which the total size of tasks was known in advance. They proposed a simple algorithm with competitive ratio of 1.5 and improved the performance of the algorithm by a preprocessing strategy.

Nadathur et al. (2007) used a decomposition strategy to speed up constraint optimization for a statically scheduling, task graph with communication delays to multiprocessors. They stated that although list scheduling heuristic is good for multiprocessor scheduling problem, but difficult to extend with specialized implementation and resource constraints. Though constraint optimization methods such as MILP can be used, but these methods are limited to medium sized problems (less than 30 tasks). They stated that their decomposition strategy can be easily used to problem size of 100 tasks and overcomes the limitations of common heuristic methods for scheduling. They further stated that the decomposition strategy with constraint optimization is a good tool for resource constrained static scheduling problems.

Kuszner et al. (2007) addressed the problem of preemptive scheduling of multiprocessor tasks on dedicated processors for minimizing the sum of completion
times. They accepted from the literature that the problem is strongly NP-hard and there are only a few polynomial time algorithms for restricted cases: two processors, four processors and bi-processor tasks and one- or all-processor tasks. They generalized all the cases using the idea of reduced conflicting graph and concluded by giving the complexity results for multiprocessor task scheduling.

Dummler et al. (2007) proposed an extension methodology for scheduling algorithms for independent multiprocessor tasks to handle precedence constraints, which is accomplished by a layer based approach. Experiments were performed on large task graphs up to 1000 nodes and 256 processors for calculating the runtime of the scheduling algorithms as well as the quality of the generated schedules. From experimental results, comparison of the extended algorithms with existing layer-based scheduling algorithms were made and a guideline derived for the suitability of the algorithm depending on the parallel application and target platform.

Hamzeh et al. (2007) proposed a fuzzy scheduling approach for real time periodic and non periodic tasks in multiprocessor system which balances task loads of the processors successfully while considering starvation prevention and fairness. Simulation analysis was carried out for performance analysis. From the experimental results, it was concluded that the proposed fuzzy scheduler creates feasible schedules for homogeneous and heterogeneous tasks.

Block et al. (2008) focused on schemes for enacting task share changes i.e. reweighting on real time multiprocessor platform. They presented two new multiprocessor reweighting schemes, changeable – earliest – deadline - first (CNG-EDF) and non -preemptive-changeable - earliest - deadline - first (NP- CNG - EDF) which reduce migration costs and preemptions at the expense of allowing deadline misses. They compared the two schemes with a more accurate, but more migration-prone scheme, PD$^2$-OI, and two less accurate partitioning schemes partitioned-adaptive scheduling (PAS) & non-preemptive-partitioned-adaptive-scheduling (NP - PAS) that have lower tardiness. From the experimental results, they concluded that each of the algorithms was the best choice in certain application scenarios. Also the proposed schemes offered a mix of accuracy & average case performance for soft real time systems.

Detti (2008) considered a high multiplicity multiprocessor scheduling problem in which jobs can be assigned only to machines not greater than a given index. The architecture consists of heterogeneous processors. Strongly polynomial algorithms were
proposed for two job classes for the feasibility version of the problem and for maximizing the number of scheduled jobs.

Chaudhuri and Elcock (2008) proposed a task scheduling algorithm which uses duplication for optimal scheduling any application represented in the form of a directed acyclic graph (DAG). The proposed algorithm does not consider communication or computation costs of tasks. The results were compared from literature and proved that the proposed algorithm produces an optimal schedule with time complexity directly proportional to number of tasks.

Cong and Gururaj (2009) proposed a novel energy aware scheduling algorithm based on mathematical programming formulation and was able to exploit variation in workloads of tasks in applications to provide a low energy scheduling solution. The proposed algorithm considers various constraints such as latency, resource, etc. for producing a scheduling solution such that the average energy consumption is minimized. Randomly generated task graphs have been considered in the experimental analysis and proved that proposed algorithm runs in polynomial time and saves up to 30% energy over existing techniques.

Jung et al. (2009) presented a global scheduling algorithm for scheduling simply periodic task systems upon uniform multiprocessor platforms. The proposed algorithm first transforms a given simply periodic task system into another using a technique and then each transformed simply periodic task system is guaranteed to be successfully scheduled upon any uniform multiprocessor using a partitioned scheduling algorithm. They used rate-monotonic (RM) algorithm on each processor and proved that the proposed algorithm is able to achieve the maximum utilization for simply periodic task system on uniform multiprocessors. They further stated that if due to semantic reasons, a task cannot split into subtasks, even then the proposed task splitting approach will work sub optimal, if each processor is allowed to have small slack.

Almeida et al. (2009) used metaheuristics for minimizing execution time of parallel programs. They stated that there is a need of heuristic approaches from which metaheuristic solutions can be easily developed. In the work, two case studies were considered. Firstly, homogeneous set of processes assigned to processors in a heterogeneous environment. Secondly, set of tasks assigned by a master processor to a set of slave processors in a heterogeneous system. They showed that the systematic application of metaheuristics provides the satisfactory results in mapping and modelled time.
Giaro et al. (2009) addressed a classical 1- and 2-processor unit execution time scheduling problem on dedicated machines where machines have non-simultaneous availability times and tasks have arbitrary release times and due dates for generalization. The problem is modeled as a cost list edge-coloring (CLEC) problem. It is proved that in sparse instances, the problem can be solved in polynomial time, but for dense instances the problem becomes NP-hard, even if there are only two machines.

Jingui and Rongheng (2009) proposed some approximation algorithms for the offline version of the Multiprocessor task scheduling problem based on Split Scheduling, the First Fit and the Large Width First techniques. They considered the general case of multiprocessor task scheduling problem with ‘m’ processors for not only the unit process time tasks but also the arbitrary processing time tasks. From results, they concluded that proposed approximation algorithms gave approximation ratios which were better than the results available in the literature.

Solano et al. (2009) proposed an ant colony optimization procedure for job scheduling in a hybrid flow shop configuration with the objective of optimizing the makespan and the total completion time of jobs simultaneously. Computational experiments were performed with randomly generated data and compared with two well known mono-criterion heuristic algorithms, as benchmarks, that are known to give very good results. They showed that the proposed algorithm outperforms the other two algorithms and claimed that the proposed work is a first step in literature for solving such a complex two-stage hybrid flow shop problem.

Switalski and Seredynski (2010) proposed a novel metaheuristic algorithm, named as Generalized External Optimization (GEO) based on Bak–Sneppen model and compared with GA and PSO for multiprocessor scheduling problems. They proved that all the algorithms found the optimal solution with GEO having better average performance for simple and complex graphs as compared to others.

Shahul and Sinnen (2010) proposed the best-first search algorithm A* for solving tasks scheduling problem and showed that the A* scheduling algorithm can produce optimal schedules in reasonable time for small to medium sized task graphs with up to 40 nodes. They compared the proposed algorithm with the previous approach & concluded that the proposed algorithm offers significant improvements in cost function $f(s)$ by dramatically reducing the search space of the algorithm. Further reduction in the search space can be achieved significantly by several pruning techniques.
Kim and Cho (2011) presented a new laxity based scheduling algorithm, named PL (pseudo-laxity) for scheduling a synchronous periodic task set on multiprocessors. From experimental results, it was concluded that the number of scheduling decisions and task preemptions in a PL schedule is mostly smaller than the Pfair (a notion called proportionate fairness (Pfair) for multiprocessor systems; a Pfair schedule is optimal in terms of schedulability. Three Pfair algorithms in literature are: PF, PD, and PD\(^2\)) algorithms.

Srikanth et al. (2012) reported the different approaches for real time task scheduling in multiprocessors. The problem of optimally allocating a set of real time tasks that can be assigned to processors, without violating the deadlines, is an NP-Complete problem. They presented the characteristics of the underlying architecture, tasks considered and the parameters used by the different scheduling algorithms. The makespan, utilization of the processor, cache performance, power consumption, workload balance, scheduling penalties and switch cost are the basic parameters normally used to compare the various scheduling algorithms. They stated that this survey of existing algorithms is just an attempt to reveal the research challenges existing in the field of real time task scheduling.

Yumin and Shufen (2014) proposed quantum random walk optimization method and compared with GA (genetic algorithm), ACO (ant colony optimization), and SAA (simulated annealing algorithm) for server traffic control and task scheduling problems. The model and algorithm prove to increase in the throughput and efficiency of the system.

### 2.4 LIMITATIONS/GAPS OF EXISTING WORK

After a comprehensive study made in the existing literature, a lot of limitations/gaps have been found in the area of multiprocessor task scheduling:

i. The majority of the work reported for multiprocessor task scheduling problems has been dealt with the minimization of makespan only. For the requirement of maximum efficiency, effective utilization of processors, maximum throughput and proper load balancing, etc., the makespan and total completion time of the schedule should be minimum. Therefore, total completion time criteria is also important and has to be given due attention along with the makespan.
ii. Different metaheuristics for multiprocessor task scheduling problems have been developed, but again for the single criteria i.e. makespan. Also, simulated annealing has been given less attention as very few have considered the simulated annealing for multiprocessor task scheduling problems. Different researchers also proposed the hybrid scheme of the metaheuristics for improved results. Apart from that, the performance of the algorithm is still the major issue for the judgment of the optimal solution.

iii. The performance of most of the genetic algorithm and simulated annealing is dependent on the selection of the operating parameters and very few have considered the optimal parameter selection for multiprocessor task scheduling and that too limited to genetic algorithm. Operating parameter selection is the main issue of any metaheuristic which decides effectiveness and hence has to be fixed for achieving the optimal results accurately.

*It is concluded from literature that population, neighborhood and heuristic based metaheuristics (especially GA and SA) has become interesting preference for researchers for most of the multiprocessor task scheduling problems. Development of heuristics and metaheuristics are still the major issues related to multiprocessor task scheduling problems.

Therefore, in the present work, multiprocessor task scheduling problems with bi-criteria performance measures, including minimization of weighted sum of makespan and total completion time have been considered. An attempt has been made to develop Hybrid Genetic Algorithm (HGA), Hybrid Simulated Annealing (HSA) and Hybrid Genetic Algorithm and Simulated Annealing (GASA) for the solution of the multiprocessor task scheduling problems. The next chapter describes the formulation of the multiprocessor task scheduling problem with the different assumptions.