CHAPTER 1

INTRODUCTION

The present chapter discusses the basic concepts of multiprocessor scheduling, taxonomy of scheduling, Particle Swarm Optimization (PSO) and a review of the literature on multiprocessor scheduling and PSO.

1.1 GENERAL

Global optimization problems are important in the fields of science, manufacturing industries, engineering and business. The goal of optimization is to find the best value for each variable in order to achieve satisfactory performance. Optimization is an active and fast growing area of research and has a great impact on the real world. Most of the NP-hard (Non-Deterministic Polynomial-Time Hard) problems cannot be solved using exact methods. Hence many heuristic and metaheuristics methodologies are being used to find near optimal solutions for the optimization problems. The objective in such problems is to find the optimal of all possible solutions that minimize or maximize an objective function. A stochastic high quality approximation of a global optimum is more valuable than a deterministic poor quality local minimum provided by a classical method.

Some complex multidimensional problems cannot be solved using classical optimization techniques. This insight has led to an increased interest in a special class of searching algorithms, namely, heuristic algorithms. These algorithms find approximate solutions and suggest some approximations to
the solution of optimization problems with low time complexity. There can be a single or multiple objective functions that evaluate the quality of the generated solution. In recent decades, evolutionary and stochastic algorithms are hybridized with other evolutionary, intelligent and metaheuristic algorithms to solve extremely challenging problems such as single and multi-objective functions.

1.2 MULTIPROCESSOR SCHEDULING AND ITS IMPORTANCE

Though scheduling of multitasks to multiprocessors in a computer system has been a topic of study for decades, it still remains an interesting and challenging problem. Throughout these years of investigation, many algorithms have been proposed, analyzed, and evaluated. Multiprocessors have emerged as a powerful computing resource for running real-time applications, especially in situations where an uniprocessor system would not be sufficient to execute all the tasks. The high performance and reliability of multiprocessors have made them a powerful computing resource, which requires an efficient algorithm to determine when and on which processor a given task should be executed.

Scheduling of tasks on a multiprocessor system is one of the most challenging problem in parallel computing. It consists of finding an optimal distribution of tasks on a set of processors. The goal is to determine the shortest schedule for the given set of tasks, which is known to be NP-Hard. In several real life applications, multiprocessors have been used for improving the speed, performance, reliability and efficiency.

Multiprocessors may be homogeneous or heterogeneous. In homogeneous multiprocessors, all processors have the same functionality, which means, the same units of memory, speed and processing resources. In heterogeneous multiprocessors, each processor has different memory units
and processing resources. Hence, a task will incur different execution costs if it is executed on different processors. In Multiprocessor system, there are three goals to lower the cost (Allen and Tucker 2004)

- Good Processor utilization
- Good Synchronization effectiveness
- Low memory access/communication cost

Figure 1.1 shows the taxonomy of scheduling (Casavant and Kuhl 1998)
Casavant and Kuhl (1988) defined a hierarchical taxonomy for scheduling tasks in distributed systems. It is broadly classified into two, namely, local scheduling and global scheduling. Local scheduling is involved with the assignment of processes to the time slices of a single processor. Global scheduling is the problem of deciding where to execute a process in a multiprocessor environment. Global scheduling is classified into Static and Dynamic Scheduling. In Static Scheduling, the assignment of tasks to processors is done before the start of execution of the program. Information regarding task execution times and processing resources is assumed to be known at compile time. Dynamic Scheduling is based on the redistribution of processes during execution time, by transferring tasks from the heavily loaded processors to the lightly loaded processors. It is called load balancing. The mathematical programming approaches like Integer Programming, Branch and Bound are used to solve multiprocessor scheduling problems.

Dynamic Scheduling is further classified into two types namely, Distributed and Non-distributed. In physically distributed algorithms, the decision making is distributed among the processors. In physically non-distributed or centralized scheduling policies, a single processor makes all the decisions regarding the task transfer and execution. Under the non-cooperative distributed scheduling policies, individual processors make scheduling choices independent of the choices made by other processors. In co-operative scheduling, the processors subordinate local autonomy to the achievement of a common goal. The Static and Cooperative distributed scheduling have been classified into two, as optimal and sub-optimal.

Optimal assignment can be made only if all the information regarding the state of the system is known. Sub-optimal solutions are either approximate or heuristic. Approximate sub-optimal solutions use the same computational model like optimal solutions instead of searching the entire
solution space, according to the algorithm dependent metric. Generating optimal schedules is an NP-complete problem. Heuristic methods rely on rules to guide the scheduling process in the right direction to search a “near” optimal solution. Optimal and approximate schedules employ techniques based on one of the four computational approaches such as enumeration of all possible solutions, graph theory, mathematical programming and queuing theory.

To perform the static analysis on task a static task mapping tool such as Parallax, Hypertool, Prep-P, Oregami and Pyrros (Allen and Tucker 2004) is use in which the task set is represented as a directed acyclic graph and the static task graph is mapped onto a given machine according to an optimizing criterion.

1.3 LITERATURE REVIEW ON MULTIPROCESSOR SCHEDULING

In the past decades, scheduling problems have been subject to intensive research due to their multiple applications in the areas of industry, finance and science. Several research have been carried out to solve multiprocessor job/task scheduling problem using heuristic approaches. This is because, though the traditional methods such as Branch and Bound, Divide and Conquer and Dynamic programming gives global optimum, they are too time consuming and cannot be applied to solve NP-hard problems (Brucker 2006).

Lin and Hsu (1990) proposed a metaheuristic algorithm simulated annealing for static task scheduling problem in distributed computing environment, in which modules of programs have been assigned over a set of interconnected tasks in order to reduce the job turnaround time as well as to obtain the best system performance.
Lo (1988) stated that many of the heuristic algorithms use graphical representation of the task-processor system such that a Max Flow/Min cut algorithms to find assignment of tasks to processors which minimizes total execution and communication costs and conclude that a measure of degree to which an algorithm achieves load balancing can yield fairly unbalanced assignments.

Hou et al (1994) proposed a stochastic search method based on GA for the problem of task scheduling. The search nodes used is represented in the form of a list of computational tasks. He developed a new crossover operator, and guarantees that the new strings generated are legal.

Ahmed and Dhodhi (1996) proposed a Problem-Space Genetic Algorithm (PSGA), which combines the list scheduling heuristic with the Genetic Algorithm for static scheduling, of directed acyclic graphs. A new Genetic Algorithm called Genetic Convex Cluster Algorithm is proposed by Sanchez and Trystram (2005) to solve the task assignment with large communication delays. It uses the convex cluster property and is well suited for parallel systems like cluster of computers with hierarchical communications.

Page and Naughton (2005) proposed an algorithm for Dynamic Scheduling in a heterogeneous environment of heterogeneous tasks, which operates in a batch fashion and total execution time is minimized using GA. Randomly generated task sets have been taken for simulations using uniform normal and Poisson distribution. Efficient results have been achieved.

Chiang et al (2006) proposed a solution to the constrained scheduling problem in display system operation, using the Particle Swarm Optimization. In particle encoding, the authors used a one-dimensional 0-1 array mapping of a three- dimension matrix of a candidate solution for
each particle. It then used sigmoid function to produce probability threshold for velocity updating in each particle. The results show that the proposed approach is capable of obtaining a higher quality solution efficiently in constrained scheduling problems.

Daoud and Kharma (2006) proposed an efficient GA for task scheduling in Heterogeneous distributed Computing Systems known as Genetic Scheduling (GS). The GS has performed well compared to the standard algorithms, namely, Heterogeneous Earliest Finish Time (HEFT) algorithm and Dynamic Level Scheduling (DLS) algorithm in terms of makespan.

Zhang et al (2008) developed a heuristic algorithm based on Particle Swarm Optimization for solving the task scheduling problem in a grid environment. Each particle is represented by a possible solution and the position vector is transformed from the continuous variable. The results of the proposed algorithm show that the Particle Swarm Optimization algorithm is able to get a better schedule than Genetic Algorithm.

Kong et al (2008) developed an alternative PSO algorithm for multiprocessor task scheduling. The framework of the hybrid PSO algorithm for the multiprocessor scheduling is developed according to the permutation-based solution representation to find optimal task-machine pairs. The results show that the proposed algorithm is able to find good quality schedules in reasonable time.

Qing et al (2010) developed a new designed crossover and mutation operators based on the characteristic of the job shop problem and an improved Genetic Algorithm is proposed. The computer simulations are made on a set of benchmark problems and the results indicate the effectiveness of the proposed algorithm.
Iotfii et al (2010) proposed a new coarse grain Genetic Algorithm in which the initial population is divided into multi sub population to reduce the solution search speed and to prevent early convergence by migration between subpopulations. The results proved that the proposed method reduces the makespan and achieves a better scheduling in comparison with the existing approaches.

1.4 PARTICLE SWARM OPTIMIZATION

Particle swarm optimization (PSO) is a population based stochastic optimization technique developed by Kennedy and Eberhart (1995) inspired by social behavior of bird flocking or fish schooling. They developed a concept for the optimization of non-linear functions using particle swarm intelligence. PSO has its roots in artificial and Social Psychology, as well as in Engineering and Computer Science. In addition, PSO uses the swarm intelligence concept, which is the property of a system, where the collective behavior of unsophisticated agents that are interacting locally with their environment create coherent global functional patterns.

PSO is a population based search algorithm and is initialized with a population of random solutions called particles. Each particle in PSO is associated with a velocity. Particles fly through the search space with velocities which are dynamically adjusted according to their historical behaviours. PSO has attracted a lot of attention from the researchers all around the world. PSO is based on two fundamental disciplines, namely, Social Science and Computer Science. The system is initialized with a population of random solutions and searches for optima by updating generations. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles. Therefore, the corner stones of PSO (Eberhart et al 2001) can be described as follows,
Social concepts: It is known that “human intelligence results from social interaction”. Evaluation, comparison and imitation of others as well as learning from experience allow humans to adapt to the environment and determine optimal patterns of behaviour and attitudes. In addition, a second fundamental social concept indicates that “culture and cognition are inseparable consequences of human sociality”. Culture is generated when individuals become more similar due to mutual social learning. The sweep of culture allows individuals to move towards more adaptive patterns of behaviour.

Swarm intelligence principles: Swarm intelligence can be described by considering the following five fundamental principles (Eberhart et al 2001),

- **Proximity**: The population should be able to carry out simple space and time computations.

- **Quality**: The population should be able to respond to quality factors in the environment.

- **Diverse Response**: The population should not commit its activity along excessively narrow channels.

- **Stability**: The population should not change its mode of behaviour every time the environment changes.

- **Adaptability**: The population should be able to change its mode of behaviour when it is worth the computational price.
• Computational characteristics: Swarm intelligence provides a useful paradigm for implementing adaptive systems. It is an extension of evolutionary computation and includes the softening parameterization of logical operators like AND, OR and NOT.

PSO has advantages over Genetic Algorithm (Del Valle et al 2008). They are given below,

• PSO is easier to implement and there are fewer parameters to adjust. In PSO, every particle remembers its own effective memory capability than the GA.

• PSO is more efficient in maintaining the diversity of the swarm (more similar to the ideal social interaction in a community). All the particles use the information related to the most successful particle to improve them. In GA however, the worst solutions are discarded and only the good ones are saved. Therefore, in GA the population evolves around a subset of the best individuals.

In PSO the term “particles” refer to population members which have no mass and volume or have an arbitrarily small mass or volume and subject to velocity and acceleration change towards a better mode of behaviour. In particular, PSO is an extension and a potentially important incarnation of Cellular Automata (CA) (Eberhart et al 2001). The particle swarm can be conceptualized as cells in CA, whose states simultaneously change in many dimensions. The PSO and CA share the following computations attributes, namely,

• Individual particles (cells) are updated in parallel.
Each new value depends only on the previous value of the particle (cell) and its neighbours.

All updates are performed according to the same rules.

1.4.1 Basic PSO Algorithm

The basic operation of PSO is given by,

**Step 1:** Initialize the *swarm* from the solution space

**Step 2:** Evaluate the *fitness* of individual particles

**Step 3:** Modify the *Gbest*, *Pbest* and velocity

**Step 4:** Move each *particle* to a new *position*

**Step 5:** Goto step 2, and repeat until convergence or the stopping condition is satisfied

The basic idea of the PSO is the mathematical modelling and simulation of the food searching activities of a swarm of birds (particles). In the multi-dimensional space where the optimal solution is sought, each particle in the swarm is moved towards the optimal point by adding a velocity with its position. The velocity of a particle is influenced by three components, namely, inertial momentum, cognitive, and social. The inertial component simulates the inertial behaviour of the bird to fly in the previous direction. The cognitive component models the memory of the bird about its previous best position, and the social component models the memory of the bird about the best position among the particles.

The velocity and position updation equation is given by,

\[
V_i = w \times V_i + C_1 \times r_1 \times (P_{best_i} - S_i) + C_2 \times r_2 \times (G_{best_i} - S_i)
\]  

(1.1)
Equation (1.1) describes how the velocity is dynamically updated and Equation (1.2) describes the position update of the flying particles. Equation (1.1) consists of three parts.

- The first component is the initial velocity with which the particle is associated with. The velocity cannot be changed abruptly. It is changed from the current velocity.

- The second component is a linear attraction towards the best position ever found by the given particle $P_{best}$, scaled by a random weight $C_1 \times r_1$. This component is referred to as, “memory”, “self-knowledge”, “remembrance”. It is the learning derived from its own flying experience.

- Third component of the velocity updation equation is a linear attraction towards the best position found by any particle $G_{best}$ scaled by another random weight $C_2 \times r_2$. This component is referred to as, “co-operation”, “social-knowledge”, “group-knowledge” or “shared information”. It is the learning from the group’s flying experience.

where,

\[ S_{i,1} = S_i + V_i \]  \hspace{1cm} (1.2)

$V_i$ : velocity of particle $i$,

$S_i$ : current position of the particle,

$\omega$ : inertia weight,

$C_1$ : cognition acceleration coefficient,

$C_2$ : social acceleration coefficient,
\( P_{\text{best}_i} \): own best position of particle \( i \),

\( G_{\text{best}} \): global best position among the group of particles,

\( r1, r2 \): uniformly distributed random numbers in the range [0 to 1].

\( S_i \): current position,

\( S_{i+1} \): modified position,

\( V_i \): current velocity,

\( V_{i+1} \): modified velocity,

\( V_{p\text{best}} \): velocity based on \( P_{\text{best}} \) and

\( V_{g\text{best}} \): velocity based on \( G_{\text{best}} \).

Figure 1.2 Flow diagram of PSO

Figure 1.2 shows the searching point modification of the particles in PSO. The position of each agent is represented by XY-axis position and the velocity (displacement vector) is expressed by \( v_x \) (the velocity of X-axis) and \( v_y \) (the velocity of Y-axis). Particles can change their searching point from \( S_i \) to \( S_{i+1} \) by adding their updated velocity \( V_i \) with current position \( S_i \). Each particle tries to modify its current position and velocity according to the distance between its current position \( S_i \) and

\[ V_{P\text{best}}, \text{ and the distance between its current position } S_i \text{ and } V_{G\text{best}}. \]
The selection of acceleration constants $C_1$ and $C_2$ control the movement of each particle towards its individual and global best position respectively. Small values limit the movement of the particles, while large numbers may cause the particle to diverge.

1.5 LITERATURE REVIEW ON PARTICLE SWARM OPTIMIZATION

Several research to improve the basic PSO concept are found.

Zhenya et al (1998) developed a four layer Fuzzy Neural Network to realize knowledge acquisition from input and output samples. The network parameters including the necessary membership functions of the input variables and the consequent parameters are tuned using a modified Particle Swarm Algorithm. The experimental results show that similar classification rules can be obtained in comparison to that of other fuzzy neural approaches.

Ozcan and Mohan (1999) developed a method which combines social psychology principles and evolutionary computation. The author has applied the proposed method successfully to non-linear function optimization and Neural Network training. The approach confers generalization to obtain closed form equation for trajectories of particles in a multi-dimensional search space.

Eberhart and Hu (1999) proposed a method for Human Tremor Analyses using Particle Swarm Optimization. The method tends to be extremely fast and accurate. The relatively small size of data sets indicate the need for further testing and development.

Parsopoulos and Vrahatis (2002) presented the recent approaches to global optimization problems through Particle Swarm Optimization techniques for the alleviation of local minima and for detecting multiple
minimizers. Finally, a composite PSO in which the heuristic parameters of
PSO are controlled by a Differential Evolution algorithm during the
optimization is described and the results for many well-known and widely
used test functions are given.

Hu and Eberhart (2002a) proposed a multiobjective optimization
technique using Dynamic Neighborhood Particle Swarm Optimization
(DNPSO). Several benchmark cases are tested and the enhanced performance
of the DNPSO method is proved.

Hu and Eberhart (2002b) proposed an Adaptive Particle Swarm
Optimization technique which automatically tracks various changes in
dynamic system. Various environment detection and response techniques are
tested on the Brock and Parabolic benchmark functions. Performance on the
benchmark functions with various severities is analyzed.

Parsopoulos and Vrahatis (2002) proposed Particle Swarm
Optimization (PSO) method in Multiobjective Optimization (MO) problems.
The author used the ability of PSO to detect Pareto Optimal points and
capture the shape of the Pareto Front studied through experiments on well
known non-trivial test functions. The Weighted Aggregation technique with
fixed or adaptive weights is considered.

Hu et al (2003) proposed a modified dynamic neighbourhood
Particle Swarm Optimization (DNPSO) algorithm for multiobjective
optimization problems. He modified the concept of Particle Swarm
Optimization using Dynamic Neighbourhood strategy. An extended memory
is introduced to store global Pareto optimal solutions to reduce computational
time. Several benchmark cases are tested and the results infer that modified
DNPSO is much more efficient than the original DNPSO and other
multiobjective optimization techniques.
Kennedy et al (2002) investigated the different population structure and particle swarm performance. He says, the ring lattice called as ‘lbest’ is the slowest which is the most indirect communication pattern. Also, the authors recommended the Von Neumann configuration.

Trelea (2003) presented the parameter selection and convergence analysis of PSO algorithm. PSO is analysed using standard results from the dynamic system theory. He explained the trade-off of exploration-exploitation.

Parsopoulos and Vrahatis (2004) proposed various approaches for effectively computing all global minimizers of an objective function. The approaches include transformations of the objective function through deflection and stretching techniques as well as a repulsion source at each detected minimizer.

Gaing (2004) proposed Particle Swarm Optimization approach for optimum design of Proportional-Integral-Derivative controller in Automatic Voltage Regulator (AVG) systems. The proposed method produced better computational efficiency.

Mendes et al (2004) developed the canonical Particle Swarm Optimization algorithm, to optimize drawing inspiration from group behaviour and the establishment of social norms. The results find good solutions in all the benchmark functions.

Shi (2004) has surveyed the research and development of PSO in five categories namely, algorithms, topology, parameters, Hybrid PSO algorithms and Applications. The author suggests that the search process of a PSO algorithm should be a process consisting of both contraction and expansion, so that it can escape from local optima.
Coello et al (2004) developed a method to handle multiple objectives with Particle Swarm Optimization, in which Pareto dominance is incorporated into PSO. The proposed algorithm uses a secondary repository of particles that is later used by other particles to guide their own flight and also incorporates a special mutation operator which enriches the exploratory capabilities. The proposed approach is validated using several test functions and metrics taken from the standard literature. Results indicate that the approach is highly competitive and can be considered a viable alternative to solve multiobjective optimization problems.

Tasgetiren and Liang (2004) proposed a Binary Swarm Optimization Algorithm for Lot Sizing Problem. The problem is to find order quantities to minimize the total ordering and holding costs of ordering decisions. The results prove that the Binary PSO performs better than the other existing approaches.

Yang and Simon (2005) proposed a New Particle Swarm Optimization method (NPSO) for solving the numeric problems, in which, each particle adjusts its position according to its own previous worst solution and its group’s previous worst to find the optimal value. The strategy is used to avoid a particle’s previous worst solution and its group’s previous worst based on similar formulae of the regular PSO. Simulation results shows that the NPSO always finds better solutions than PSO.

Immanuel et al (2007) proposed a new version of the classical PSO called new PSO (NPSO) to solve nonconvex economic dispatch problems. In order to well exploit the promising solution region, a simple local random search (LRS) procedure is integrated with NPSO. The resultant NPSO-LRS algorithm is very effective in solving the nonconvex economic dispatch problems. The proposed method is validated with three test problems having
nonconvex solution spaces and better results are obtained when compared with previous approaches.

Del Valle et al (2008) described the basic concepts of PSO along with its numerous variants that can be employed in different optimization problems. In addition, a review of the applications of PSO in power systems-based optimization problems is presented to give an insight of how the PSO can serve as a solution to some of the most complicated engineering optimization problems.

Shi and Eberhart (2009) discussed and defined several diversity measurements in monitoring Particle Swarm Optimization. A diversity measurement called cognitive diversity is discussed and defined. This can reveal clustering information about the current population of particles and the cognitive diversity together with the convergence/divergence stage, the position of the current population of particles and the stage towards which it moves.

PSO can be applied to several applications like Neural networks, Computer networks, Travelling Salesman Problem, Control systems, Image Processing, Video Analysis Applications, Job shop scheduling, flow shop scheduling problems and Design applications.

1.6 TOPOLOGIES IN PSO

The selection of the neighbourhood topology in PSO plays a major role in reaching the optimal solution faster. The topologies available are Star, Ring, Wheel, Pyramid, Von Neumann and Cluster topology.

The commonly used topologies in PSO are the star topology (global version) and ring topology (local version). In the global version of PSO, each
particle flies through the search space with a velocity that is dynamically adjusted according to the particle’s personal best performance achieved so far and the best performance achieved so far by all the particles. In the global best neighbourhood, the particles are attracted to the best solution found by any member of the swarm. Every particle compares its value with all other particles. It is time consuming in the initial stages as each particle is comparing its value with all others in the entire population. That is the objective of task scheduling with a minimized cost will be achieved in a very few iterations beyond which the cost never decreases. The termination might be in the targeted fitness or the maximum number of iterations. The global best PSO method implements the star topology. This represents a fully connected network in which each particle has access to the information of all other members of the community.

In the local version or ring topology, each particle keeps track of the best vector attained by its local topological neighbourhood of particles which is called as lbest. The ring topology represents a regular graph with a minimum number of edges between its nodes. The graph statistics show that information travels slowly along the graph. This allows for different regions of the search space to be explored at the same time, as information of successful regions takes a long time to travel to the other side of the graph. It is called k best topology in general (each node connected with k nodes). Each particle compares itself with its neighbours. The neighbours of a particle are decided based on the size of the neighbourhood. The neighbourhood can be set as the desired group. Each one results in their own best value known as the local best. All the local bests obtained are then evaluated and the global best is finally obtained. The final global best is the schedule of the desired fitness value. The groups exchange information about local optima. The local best PSO method implements the ring topology. Here, the exploitation of solution space is weakened and exploration becomes stronger as different groups are
involved in the search. The time taken to find the global optimum may be higher in the lbest PSO. There is an increase in time in all the iterations as the global best is obtained by comparing the entire local best Values.

The wheel topology (typically for highly centralized business organization) is a topology in which the individuals are isolated from one another and all the information is communicated to a focal individual. Von Neumann, a two dimensional grid represents a connection with neighbours to the North, East, west and South points. Pyramid represents a three dimensional triangular grid connection. Cluster topology has particles grouped in several cliques.

A lot of researchers have since then worked on improving its performance by designing or implementing different types of neighbourhood structures in PSO. Kennedy and Mendes (2002) claimed that the PSO with smaller neighbourhoods perform better on complex problems while PSO with large neighbourhood perform better for smaller problems. Each neighbourhood topology has its own strengths and weaknesses. It works better on one kind of problem but worse on the other kind of problems. When using the PSO to solve a problem, not only the problem needs to be specified, but the neighborhood structure of the PSO utilized should also be specified clearly. The authors also tested PSO with regular shaped neighbourhoods, such as global version, local version, pyramid version, star structure, Von Neumann and PSO with randomly generated neighbourhoods. They also evaluated all topologies as well as the case of random neighbours. In their investigation with a total number of 20 particles, they found that the best performance occurred in a randomly generated neighbourhood with an average size of five particles. The authors also suggested that the Von Neumann configuration may perform better than other topologies including the Global best version.
1.7 OBJECTIVE OF THE RESEARCH WORK

In the present research a new approach, namely, Improved Particle Swarm Optimization (IPSO) is proposed to solve static and dynamic multiprocessor task scheduling problem. Subsequently, hybrid optimization algorithms are applied to search for an optimal schedule of the problem specified.

The following population based stochastic optimization techniques are proposed in the research to reduce the total finishing time and average waiting time in static task scheduling and improving the schedule, utilization by means of load balancing in dynamic task scheduling.

- Proposed Improved Particle Swarm Optimization (IPSO) algorithm.
- Proposed Hybrid Particle Swarm Optimization (Improved Particle Swarm Optimization with Simulated Annealing)
- Proposed Hybrid Intelligent Approach(Improved Particle Swarm Optimization with Artificial Immune System)
- Proposed Hybrid Heuristic Approach (Improved Particle Swarm Optimization with Ant Colony Optimization)
- Proposed Parallel Improved Particle Swarm Optimization approaches namely, Parallel Synchronous Improved Particle Swarm Optimization (PSIPSO) and Parallel Asynchronous Improved Particle Swarm Optimization (PAIPSO)

The proposed approaches are tested for the following task scheduling problems as,

- Static task scheduling with independent.
• Dynamic task scheduling in which the tasks arrival is dynamic
• Dynamic task scheduling with Load Balancing

Each one of the proposed approaches contributes a distinct methodology for multiprocessor scheduling and is performed in a cooperative manner rather than a competitive manner. The result is a more intelligent schedule compared to the traditional scheduling techniques.

Detailed experiments are conducted to analyze the performance of the proposed approaches. The results obtained are compared with the existing literature to prove their level of accuracy and the implication is highlighted.

1.8 ORGANIZATION OF THE THESIS

The thesis is organized into seven chapters including the Chapter 1. A brief outline of the forthcoming chapters is given below.

Chapter 2 discusses the efficiency of the proposed Improved Particle Swarm Optimization (IPSO) algorithm for Multiprocessor task scheduling problem. The results acquired are compared with Standard PSO, Genetic Algorithm (GA), the LPT and SPT for static scheduling and the previously proposed methods, Particle Swarm Optimization with Fixed and Variable Inertia for dynamic task scheduling.

Chapter 3 presents the proposed hybrid methodology, Improved Particle Swarm Optimization algorithm (IPSO) with Simulated Annealing (SA). The performance of the proposed hybrid approach IPSO-SA is applied to static and dynamic task scheduling with random values and benchmark datasets. The simulation results are also explained.
Chapter 4 summarizes the weakness of the hybrid approach IPSO with SA and proposes another hybrid intelligent algorithm using Improved Particle Swarm Optimization (IPSO) with Artificial Immune System (AIS). The performance of the hybridized technique IPSO-AIS is tested using the benchmark and random datasets and its performance for the defined problem is noted.

Chapter 5 presents the weakness of the hybrid approach IPSO-AIS and the possibilities of avoiding those by combining the strength of IPSO with the power of Ant Colony Optimization (ACO). The proposed hybrid heuristic approach IPSO-ACO is explained in detail. Simulation results are presented to illustrate the usefulness of the proposed hybrid approach.

Chapter 6 presents parallelization approach of Improved Particle Swarm Optimization with two versions, namely, synchronous and asynchronous. The simulation results show that the proposed parallel approach in producing improvements in the schedule is effective when applied to static and dynamic scheduling environments.

Chapter 7 provides the conclusion of the present research as well as suggestions for future work.