2 LITERATURE STUDY AND RELATED WORK

In Soft Real-Time Single Processor System, typically, scheduling algorithms are divided into two major categories like static and dynamic scheduling algorithms, which depend on the priority they follow, as explained in section 1.4. The static algorithm uses a unique priority to each process throw out the scheduling, whereas the dynamic algorithm priority changes during the scheduling process [10][24]. Detailed studies of existing static and dynamic scheduling algorithms have been done. Also, the different Hybrid scheduling algorithms which use the characteristic of the static and dynamic scheduling algorithm have been proposed, and their study has been done. In the recent past, genetic algorithms for scheduling have been submitted. ACO-based scheduling algorithm for Soft Real-Time System has changed the complete direction of solving the scheduling problem [21].

2.1 The Dynamic Scheduling Algorithms

Dynamic scheduling algorithms make decisions during the runtime of the system. It allows not only to design a more flexible system but also to associate calculation overhead with it. The dynamic scheduling algorithms decide what task to execute depending on the importance of the task, called priority. The task priority may change during the runtime[25][26]. The dynamic scheduling algorithms widely used with the Soft Real-Time single processor system are the earliest deadline first (EDF) and Least Slack Time First (LST). These algorithms are described as follows.

1) EDF - The earliest deadline first is a dynamic scheduling algorithm, which gives the highest priority to the task, which has the nearest absolute deadline. Priorities of tasks are allocated dynamically and are inversely proportional to the absolute deadlines of the active processes [27][28]. The algorithm executes when the current process completes; or the new process arrives.
2) LST - The LST is a dynamic scheduling algorithm that prioritizes the process with the shortest slack time. The slack time (l) can be calculated at time t with the deadline interval d and the remaining execution time c [29]. The algorithm executes when the current process completes; or the new process arrives.

2.2 The Static Scheduling Algorithms

The static scheduling algorithm can calculate the order of execution before runtime as well. The static scheduling algorithm also decides the sequence of tasks based on priority, but the priority value will not change during runtime [30]. The static scheduling algorithms widely used with the Soft Real-Time Single processor are Rate Monotonic (RM) and Shortest Job First (SJF). These algorithms are described as follows.

1) RM - The rate monotonic is a static scheduling algorithm that gives maximum priority to the process, which has the smallest period or lowest rate [27][31]. The rate of a process is already known in RTOS and is defined as the task occurs again in a given duration. The algorithm executes when the current process completes; or the new process arrives.

2) SJF - The shortest job first algorithm is a static scheduling algorithm that prioritizes the process with the shortest execution time [31]. The execution time is already known in RTOS and is defined as the process requiring CPU time to complete the given task. The algorithm executes when the current process completes; or the new process arrives.

2.3 The Hybrid Scheduling Algorithms

The Dynamic algorithms perform well in the underload scenario and schedule all processes in a given process set. The static algorithms perform well in overload scenarios and try to schedule the maximum task in a given task set. The ideal algorithm can be designed, which uses the features of dynamic and static algorithms, and it performs well in underload as well as overload scenarios. Based on this concept, many Hybrid scheduling algorithms have been introduced. A few of them have been explained below.

1) O_EDF and R_EDF - EDF algorithm performs well when the system is not overloaded, but its performance exponentially decreases when it becomes slightly overloaded. The research has applied certain modifications in the conventional EDF algorithm to overcome this limitation and proposed two algorithms named O_EDF and R_EDF [32].
2) D_EDF - This scheduling algorithm overcomes the limitations of the dynamic algorithm during overloaded conditions. The proposed algorithm D_EDF, simulated and tested for independent, preemptive, periodic tasks on tightly coupled real-time multiprocessor systems under global scheduling. The experiments and result analysis conclude that the

proposed algorithm is efficient in both underloaded and overloaded conditions. Moreover, it always performs better than the conventional EDF algorithm [24].

2.4 The Swarm Intelligence based Scheduling Algorithms

Any problem solving is a search for the optimal solution from a vast solution space. Artificial Intelligence (AI) solves the problem efficiently by heuristic search, embedding the domain knowledge, which guides the search intelligently. AI has successfully demonstrated its capabilities in almost every field of engineering. Swarm Intelligence (SI) is a computational and behavioral metaphor for problem-solving that originally took its inspiration from nature's examples of collective behaviors like Social Insects: Nest building, Foraging, Assembly, Sorting, and Vertebrates: Swarming, Flocking, Herding, Schooling. SI provides a basis, making it possible to explore collective (or distributed) problem-solving methodology without centralized control [33][34]. Based on SI following scheduling algorithm has been proposed for different systems. They all are not specifically for Soft Real-Time Systems, but these algorithms are targeting scheduling problems.

1) Integer PSO for task scheduling in cloud computing systems - Task scheduling is a challenging task in cloud computing systems to meet the requirements of both ends. In this research, a discrete version of the Particle Swarm Optimization (PSO) algorithm, namely Integer-PSO, has been proposed for task scheduling in the cloud computing environment which can be used for optimizing a single objective function and multiple

objective functions. Experimental studies on different types of the task set characterizing normal traffic, and burst traffic in the cloud computing environment show that this approach is better and has good convergence and load balancing [35].

2) A two-level particle swarm optimization algorithm for flexible job-shop scheduling - This algorithm targets the flexible job-shop scheduling problem. The upper level handles the operations-to-machines mapping, while the lower level handles the ordering of operations on machines. In addition, a lower bound-checking strategy on the optimal objective function value is used to reduce the number of visited solutions and objective function evaluations. The algorithm is benchmarked against existing state-of-the-art algorithms for the flexible job-shop scheduling problem [36].

3) Scheduling Workflow in Cloud Computing Environments using PSO - Cloud computing environments facilitate applications by providing virtualized resources that can be provisioned dynamically. In this research, PSO-based heuristic to schedule applications to cloud resources considers both computation cost and data transmission cost. It has been observed that PSO-based algorithm can achieve more cost-saving and good distribution of workload onto resources compare to Best Resource Selection (BRS) algorithm [37].

4) An Adaptive PSO-Based Real-Time Workflow Scheduling Algorithm – This algorithm has been proposed for the Cloud Computing environment. Developing workflow scheduling algorithms can efficiently reduce the cost of executing tasks in cloud systems.

For real-time workflows, reducing execution time and reducing execution cost are two conflicting objectives. This research has targeted this issue and proposes an improved real-time workflow scheduling algorithm based on particle swarm optimization (PSO) [38].

5) GA-PSO Algorithm – This research focuses on simultaneously accepting orders and scheduling decisions in real-time, as is required for the operation of an MTO (make-to-order) flow shop production system, a topic that has received little attention in academia due to the complexity of the problem. This research presents a hybrid genetic algorithm and particle swarm optimization algorithm (GA-PSO) to solve the problem [39].

6) ACO-Based Scheduling Algorithm - The Ant Colony Optimization (ACO) algorithm is a mathematical model enlivened by the system searching conduct of ants. By taking a gander at the qualities of ACO, it is most suitable for scheduling tasks in soft real-time systems. ACO-based scheduling algorithm has been compared with the Earliest Deadline First (EDF) algorithm. It is noted that the new algorithm is equally efficient under loaded conditions when CPU load is less than one. ACO-based scheduling algorithm performs superior during the overloaded conditions when CPU load is more than one, whereas the EDF algorithm performance is degraded in overload conditions [21].

Although the PSO is integrated into scheduling in all concerned fields, there is still sufficient scope for exploration. This thesis is addressing a few of such gaps in this area. Swarm Intelligence algorithms strategies are considered adaptive strategies and are typically applied

to search and optimization domains. This research selects PSO because it is the right approach when the problem size is between 20 to 40 [40][41]. Second, this research has considered scheduling task problems in a soft real-time system for periodic task sets. Static and Dynamic scheduling algorithms have their advantages and disadvantages, and both cannot perform well in overload and underload scenarios. This research proposes a PSO-based scheduling algorithm that will overcome the disadvantages of the Static and Dynamic scheduling algorithm by retaining its advantages and introducing a scheduling algorithm whose performance is better than an ACO-based scheduling algorithm.