1 INTRODUCTION

This chapter briefly introduces the Real-Time System. It briefs about the Type of Real-Time System, Type of Task, and Type of Scheduling algorithm to schedule the given task set. It also talks about the motive behind this work, the problem statement, and the research contributions.

1.1 The Real-Time System

Real-Time systems have become part of human lives to complete their day-to-day needs. Real-Time System has many applications, such as digital control systems, flight control, vehicle control, healthcare devices, IoT devices, and many more. In the 21st century, usage of Real-Time systems has increased widely. Like a conventional operating system, we also use Real-Time systems in our day-to-day life, but when real-time systems work well, they make us forget their existence. Real-Time System focuses on completing the task before its deadline, whereas the conventional operating system tries to give minimum response time for any given time. There is always a specific deadline associated with Real-Time tasks, whereas a typical task does not have any particular timeframe. Text Editor, Browser, music players are examples of such typical applications, whereas Smart Watch, Aircraft Control, and Missile Control Systems are examples of Real-Time applications [1][2].

The Real-Time system is the system in which the accuracy of system is defined by the logical accuracy and the time it takes to produce the result. Real-Time systems have decisive,

unchanging time restrictions, i.e., the task must be completed within the specified duration; otherwise, the system fails. Since the last few years, Real-Time systems usage has been increasing in time-critical applications. Designing systems that are expected to deliver real-time results involves an equal emphasis on managing the timing constraints of various functionalities of the system [3]. Processes in the real-time system have defined deadlines and need to complete the process within its deadline. Real-time systems need a scheduling algorithm that assigns the tasks to the processor by considering the deadline constraints and supporting other scheduling requirements.

1.2 Type of Real-Time System

Real-Time System is divided into three categories: Hard Real-Time, Soft Real-Time, and Firm Real-Time System based on their timing constraints. Real-Time systems will be considered Hard Real-Time systems if it fails to meet its deadline is deemed a fatal fault. A few examples of Hard Real-Time Systems are Metro Train and its signal system, Missile technology, Flight control system. In Contrast, the Soft Real-Time System with few missed deadlines does not cause serious harm; only the system's overall performance worsens when more tasks miss their deadline. A few examples of Soft Real-Time systems include ATM systems, Mobile applications, and telephone switches. In a Firm Real-Time system, if a task misses its deadline, the result of the given task will be ignored. This type of Real-Time System has specific time constraints which are not strict, and it may cause undesired effects. An example of a firm Real-Time System is automated visual inspection in industrial automation. This system examines and detects the defective parts of the assembly line [4].

In Real-Time Systems literature, the distinction between hard and soft timing constraints is sometimes stated quantitatively in terms of the usefulness of results as functions of the tardiness of task. The tardiness of a task measures how late it completes respective to its deadline. Its tardiness is zero if the task completes at or before its deadline; otherwise, if the task is late, its tardiness is equal to the difference between its completion time and its deadline. If a task must never miss its deadline, then the deadline is hard. On the other hand, if its deadline can be missed occasionally with some acceptably low probability, its timing constraint is soft [5].

1.3 Type of Task in Real-Time System

The real-time system has three kinds of task models called Periodic, Aperiodic, and Sporadic tasks. In the periodic task, each task is generated at regular time intervals. The Real-Time system is invariably required to respond to external events and respond; it executes aperiodic or sporadic tasks whose release times are not known to the system in advance. The interarrival times between consecutive tasks in such a task may vary widely, and, in particular, it can be arbitrarily small. We called the task is aperiodic if the process in it has soft deadlines. Each unit of work is scheduled and executed by the system as a task [6]. This definition of periodic tasks differs from the one often found in Real-Time Systems literature. In many published works, periodic task refers to a truly periodic task; that is, inter-release times of all jobs in a periodic task are equal to its period. This definition has led to the common misconception that scheduling and validation algorithms based on the periodic task model are applicable only when periodic tasks are truly periodic [5][7].

Each task has a different characteristic, like release time, deadline, period, and execution time. The release time of a task is the instant of time at which the task becomes available for execution [8]. The task can be scheduled and executed at any time after its release. The deadline for a task is the instant of time by which its execution needs to be completed. The deadline for a task is sometimes called an absolute deadline, which is equal to its release time plus its relative deadline. The execution time of any task is considered the unit amount of time required for the task to execute it on the processor. If the task is periodic, then the period of the task indicates the occurrence interval of the given task [6]. The task set can also be divided into preemptive and non-preemptive as well.

1.4 Type of Scheduling Algorithm

In Real-Time Systems, selecting the scheduling algorithm is a critical task, and it will be decided based on the characteristics of the Real-Time System and the task type [9]. The scheduling algorithm, sometimes called a scheduler (in Real-Time Systems), decides which task should execute on a processor at any given time. Scheduling algorithms are generally required to be simple and fast because they execute alongside other tasks. There is a broad classification of scheduling algorithms based on whether the schedule is generated before run-time or at run-time. Please refer the Figure 1.1 [10][11]. In offline scheduling, the scheduling is done before run-time. Whereas offline scheduling often requires the exact knowledge of the arrival time of all the tasks that need to be scheduled, or it makes pessimistic assumptions about the arrival times of tasks [12]. In online scheduling, the scheduling activity is carried out at run-time. In online

scheduling, the decision to schedule a task is made after the task arrived. Therefore, it is not necessary to know the exact arrival time of tasks [13][14].

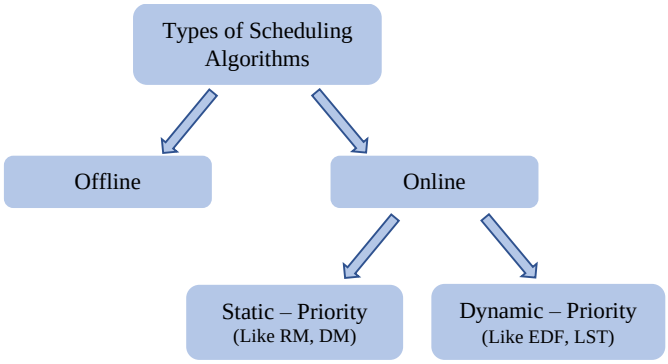

Figure 1.1 - Types of Scheduling Algorithms in Real-Time Systems

The online scheduling methods are widely separated into two categories: Static priority and Dynamic priority scheduling methods. Static scheduling methods allocate all priorities at design time, and it remains steady for the lifespan of a task. Dynamic scheduling methods keep changing the priority at the scheduled time based on the design parameters of any task. Dynamic methods can be endured with static or dynamic priority. Rate Monotonic (RM) and Deadline Monotonic (DM) are examples of the online scheduling algorithm with static priority [15][16]. Example of online scheduling algorithms with dynamic priority include Earliest Deadline First (EDF) and Least Slack Time First (LST). These algorithms are most favorable where jobs are preemptable, consist of a single processor, which in turn is under-loaded [16][17]. However, the constraint of such an algorithm is its performance, which diminishes exponentially if the system becomes somewhat overloaded [18]. The scheduling is treated as online if the scheduling algorithm forges scheduling outcomes and doesn't know about the task to be released in the future. The goodness of the algorithm can be measured by its competitive factor: an online algorithm of competitive factor r $(0 < r < 1)$ is guaranteed to achieve a cumulative value at

least r times the cumulative value achievable by any clairvoyant algorithm on any sequence of tasks. It is stated that, in an overloaded situation, no other online scheduling algorithm can attain a competitive factor prominent than 0.25. Certainly, many research papers have identified that for any system whose loading factor is nearly equal to 1, the competitive factor of an online scheduling algorithm is nearly equivalent to 0.385 [19].

1.5 Motivation for This Work

In Real-Time Operating System, as explained in section 1.4, the scheduling algorithms can be divided into static and dynamic, depending on the priority they follow in selecting the task for execution. The static algorithm uses a unique priority to each task throughout the scheduling, and the dynamic algorithm priority changes during the scheduling process. Many researchers have proposed different static and dynamic scheduling algorithms. It has been observed that static scheduling algorithms perform well during overload scenarios, and dynamic algorithms perform better than static scheduling algorithms in the underload scenario [20]. The Ant Colony Optimization (ACO) based scheduling algorithm was proposed for Soft Real-Time System in 2009 [21].

Many Research papers have proposed the entirely new methods and direction for scheduling tasks using Artificial Intelligence and Swarm Intelligence Techniques. Swarm Intelligence is the study of computational systems inspired by collective intelligence. Collective Intelligence emerges through the cooperation of large numbers of homogeneous agents in the environment. Examples include schools of fish, flocks of birds, and colonies of ants. Such intelligence is

decentralized, self-organizing, and distributed throughout an environment. Using Swarm Intelligence, it has been proposed to find the optimal solutions for problems like scheduling the task [22][23]. The research paper has proposed ACO based scheduling algorithm, and it has been shown that the Swarm Intelligence-based scheduling algorithm performs equally well as a Dynamic scheduling algorithm in an underload scenario. It gives better performance in an overload scenario as well. The ACO-based scheduling method for the soft real-time operating system (RTOS) has been profound with practical proof [21].

The motivation of thesis work is to study the different swarm intelligence techniques, identify the best suitable strategies for scheduling purposes and develop the new scheduling techniques for the Soft Real-Time System. Many swarm intelligence techniques like the Bees Algorithm, Particle Swarm Optimization, Intelligent water drop, Gravitational Search Algorithm, and many more. They are still not explored with Soft Real-Time System, specifically in scheduling task problems. The detailed study of different swarm intelligence techniques have been done and research has been focused on the Particle Swarm Optimization technique to develop the new scheduling algorithm for Soft Real-Time systems.

1.6 Problem Statement, Objectives, and Research Contribution

1.6.1 Problem Statement

To design and develop the efficient scheduling algorithm for Soft Real-Time Systems in a Single Processor environment using Swarm Intelligence Techniques, which are suitable in underload and overload condition.

1.6.2 Objectives

The Objectives of the thesis are:

1) Analyze the existing Static and Dynamic scheduling algorithms and observe their performance in overload and underload scenarios.
2) Design of a hybrid scheduling algorithm using the characteristics of Static and Dynamic scheduling algorithms.
3) Give Theoretical proof of existing ACO-based scheduling algorithm.
4) Explore different Swarm Intelligence Techniques for Scheduling in Soft Real-Time System.
5) Identify the Swarm Intelligence Technique and develop a scheduling algorithm for Soft Real-Time Single Processor System.

1.6.3 Research Contribution

For designing and developing of Swarm Intelligence Based Scheduling algorithm, the following are research contributions through this work

1) Critical analysis has been done for two Static (RM and SJF) and two Dynamic (EDF and LST) scheduling algorithms for the Soft Real-Time Single processor system. These algorithms have been compared with respect to SR (success ratio) and ECU (effective CPU utilization) parameters.
2) Based on critical analysis, the Hybrid scheduling algorithm (S_LST) for Soft Real-Time systems has been proposed. S_LST algorithm has been compared with LST and SJF scheduling algorithm and its performance has been observed.
3) Detail study of the existing ACO-based scheduling algorithm has been done and the Theoretical proof has been derived. As a part of theoretical proofs three propositions and two theorems have been proposed.
4) During research a comprehensive study on Swarm Intelligence has been done to identify suitable swarm techniques for scheduling purposes in the Soft Real-Time Single Processor system.
5) The Optimized Scheduling Algorithm for Soft Real-Time Single Processor System using Particle Swarm Optimization Technique has been designed and developed. Its effectiveness has been compared with EDF and ACO based scheduling algorithms.

Above all research contributions have been published in six different papers and the details of all papers have been given in the PUBLICATION section.

1.6.4 Scope of Problem Statement

The thesis has targeted the scheduling optimization problem with Soft Real-Time System in a single processor environment. All experiments during the research are carried out on a periodic task set (Explained in Section 1.4). It has been considered that when the task is released, system already know about the task deadline and imperative data to figure out the required time to complete the task. Therefore, the task set is considered preemptive and pretending that the system doesn't have any constraint of resource clash. Furthermore, it also considered that scheduling and preemption do not have any other overhead. In Soft Real-Time System, each task possesses a positive value. The simple ideology is to yield as much benefit as possible. If a particular task succeeds, then the system contemplates its benefits; otherwise, the system attains less advantage from the task.

1.7 The Overall Organization of Thesis

The overall thesis has been divided into eight chapters with a brief introduction of the chapter at the beginning and a conclusion at the end of each chapter. The first chapter, as discussed above, was about the Real-Time System. The evolution of the Real-Time System in our day-to-day life has motivated this work, and a brief introduction has been given in this chapter. The introduction chapter gives the motive behind this work, a brief introduction to work, which has been carried

out regarding the problem statement, objectives, and research contributions. The rest of the chapters has been organized as follow:

Chapter 2 – This section gives a detailed analysis of the work carried out for scheduling techniques in a Real-Time System. It describes different scheduling approaches like Static Scheduling, Dynamic Scheduling, and Hybrid Scheduling. Finally, it represents an entirely new approach to scheduling using swarm intelligence.

Chapter 3 – This section gives a brief detail about the performance parameters, dataset, and simulator used in this research. All algorithms in this research have been evaluated based on ECU (Effective CPU Utilization) and SR (Success Ratio) parameters. The large dataset has been considered for scheduling purposes which vary in terms of the number of tasks and utilization factor for each task set. Finally, it describes the simulator which has been developed and implemented in the C programming language.

Chapter 4 – This section gives a critical analysis of existing static and dynamic scheduling algorithms. This section has implemented two static (RM and SJF) and two dynamic (EDF and LST) scheduling algorithms on the simulator and analyzed these algorithms concerning ECU and SR parameters, and observation has been discussed.

Chapter 5 – This section has proposed the hybrid scheduling algorithm for scheduling of Soft Real-Time System. It uses the characteristics of LST and SJF algorithms. Hybrid scheduling

algorithm has also been compared with LST and SJF concerning ECU and SR parameters, and observation has been discussed.

Chapter 6 – This section gives theoretical proof for ACO based scheduling algorithm. The ACO-based scheduling algorithm for the soft real-time system was proposed in 2009. This section has analyzed the ACO-based scheduling algorithm, and the mathematical proof for this algorithm has been given.

Chapter 7 – This section gives an entirely new approach for scheduling in a soft real-time system. It has proposed the PSO-based scheduling algorithm. The algorithm has been compared with EDF and ACO-based scheduling algorithms concerning ECU and SR parameters, and time complexity analysis has been given. This section also provides the mathematical proof for the proposed approach.

Chapter 8 – This section concludes the work and gives a roadmap to the future work. It has discussed all results which has been observed during this research work. It has also given possible research work using swarm intelligence for different task type and in multiprocessor environment.

PUBLICATION – This section lists out the publications that emerged through this research work.

REFERENCES – This section lists the references which have been used to carry out this research work.