TASK ASSIGNMENT AND SCHEDULING

Introduction

Real-time systems are systems that carry real-time tasks. These tasks need to be performed immediately with a certain degree of urgency. In particular, these tasks are related to control of certain events (or) reacting to them. Real-time tasks can be classified as hard real-time tasks and soft real-time tasks.

A hard real-time task must be performed at a specified time which could otherwise lead to huge losses. In soft real-time tasks, a specified deadline can be missed. This is because the task can be rescheduled (or) can be completed after the specified time.

Task Assignment

Definition:

Task assignment is the process of allocating specific tasks or responsibilities to individuals or teams within an organization. It involves determining who is responsible for completing a task, by providing them with the necessary information and resources, and setting clear expectations for the desired outcome.

Task assignment is important for the following reasons:

(i) It ensures that work is distributed efficiently and effectively among team members.

(ii) It allows for better utilization of individual skills and expertise.

(iii) It helps in balancing workload and avoiding bottlenecks or overload.

(iv) It promotes accountability and clarity regarding responsibilities.

(v) It improves productivity and task completion within the desired timelines.

When assigning tasks, we will consider the following factors:

(i) Individual skills, knowledge, and expertise required for the task.

(ii) Availability and workload of team members.

(iii) Deadlines and priority of tasks.

(iv) Communication and collaboration requirements.

(v) Dependencies and relationships between tasks.

(vi) Consideration of individual development or growth opportunities.

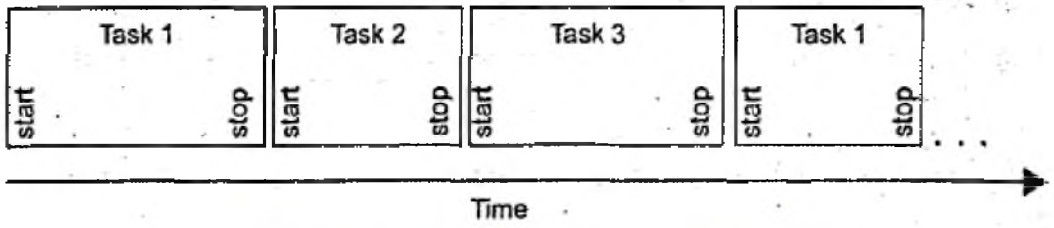
(vii)Balancing workload and avoiding overburdening or underutilization.

Scheduling

In real-time systems, the scheduler is considered as the most important component which is typically focusing to reduce the response time associated with each of the associated processes when handling the deadline.

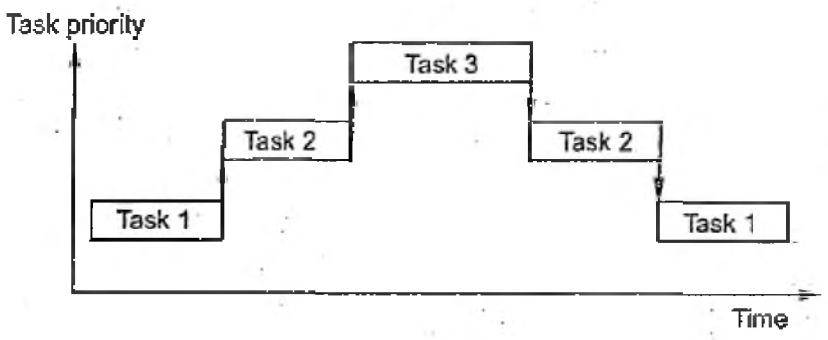
Definition:

The way that time is allocated between tasks is termed as “scheduling". The scheduler is the software that determines which task should run next. The logic of the scheduler and the mechanism that determines when it should be run is the scheduling algorithm.



Preemptive scheduling is used in real-time systems where the tasks are usually configured with different priorities and time critical tasks are given higher priorities. A higher priority task can stop a lower priority one at any time, gram and use the CPU until it releases it.

Preemptive scheduling is the most commonly used scheduling algorithm in realtime systems. Here, the tasks are prioritized and the task with the highest priority among all other tasks gets the CPU time.



In the case of a non-preemptive scheduling, even if the highest priority is allocated to the task, it needs to wait until the completion of the current task. This task can be either slow (or) lower priority, which can lead to a longer wait.

A better approach is designed by combining both preemptive and non-preemptive scheduling. This can be done by introducing time-based interrupts in priority based systems which means the currently running process is interrupted on a time-based interval and if a higher priority process is present in a ready queue, it is executed by preempting the current process.

Classifications of Task Scheduling Algorithms:

In static priority algorithms, the task priority does not change with a mode. In dynamic priority algorithms, the priority can change with time.

Based on schedulability, implementation (static or dynamic), and the result (self or dependent) of analysis, the scheduling algorithm are further classified as follows:

(i) Static Table-Driven approaches:

These algorithms usually perform a static analysis associated with scheduling and capture the schedules that are advantageous. This helps in providing a schedule that can point out a task with which the execution must be started at run time.

(ii) Static Priority-Driven Preemptive approaches:

These types of algorithms also use static analysis of scheduling. The difference is that instead of selecting a particular schedule, it provides a useful way of assigning priorities among various tasks in preemptive scheduling.

(iii) Dynamic Planning-based approaches:

Here, the feasible schedules are identified dynamically (at run time). It carries a certain fixed time interval and a process is executed if and only if satisfies the time constraint.

(iv) Dynamic Best Effort approaches:

These types of approaches consider deadlines instead of feasible schedules Therefore, the task is aborted if its deadline is reached. This approach is used widely in most of the real-time systems.

Advantages

The advantages of scheduling in real-time systems are,

(i) Meeting Timing Constraints:

Scheduling ensures that real-time tasks are executed within their specified timing constraints. It guarantees that critical tasks are completed on time, preventing potential system failures or losses.

(ii) Resource Optimization:

Scheduling algorithms allocate system resources effectively which ensures efficient utilization of processor time, memory,

and other resources. This helps maximize system throughput and performance.

(iii) Priority-Based Execution:

Scheduling allows for priority-based execution, where higher-priority tasks are given precedence over lower-priority tasks. This ensures that time-critical tasks are promptly executed, leading to improved system responsiveness and reliability.

(iv) Predictability and Determinism:

Real-time scheduling provides predictability and determinism in task execution. It enables developers to analyze and guarantee the worst-case execution time and response time of tasks, ensuring that critical deadlines are met.