Advanced Real Time Systems

T.R. Gopalakrishnan Nair
Prince Mohammad bin Fahd University (PMU), Saudi Arabia & Dayananda Sagar Institutions (DSI), India

INTRODUCTION

Real-time system has emerged to be one of the major application domains in computer science and engineering, in today's scenario. A real-time system is a hardware cum software unit that must process information and produce a response within a specified time, else risk severe consequences, including failure. Lui Sha (2004) defined it as a system with explicit deterministic or probabilistic timing requirements. For example in avionics, flight control software must complete its execution in a specified time in order to accurately control the flight. In health monitoring, the software to monitor the patient status in ICU must get fully executed within its specified interval. Various other examples can be identified in automated digital control, nuclear power plant, automobile, ABS and telecommunications systems.

In these systems, the response must be produced by computing unit at the specified time limit and it is called a deadline. Based on their deadline commitments, real time systems are classified into hard and soft real time systems. Hard real time system is a computer system in which the response occurs within a specified time. Failure to meet such a timing requirement can have catastrophic consequences. A system in which the missed deadline does not cause any catastrophe but only degrades performance is termed as soft real time system. The classification of these hard and soft real time systems depends on the application and not on the computer system. The correctness of the response produced depends much on the logical design of the system and also the time at which the response was produced. The real time systems may enter into an unstable state when the expected response is not produced at the right time or when there is a sudden change in the schedule of tasks because of external catastrophe. The advances in real-time system are needed to detect the catastrophe and reschedule the tasks to maintain the stability of the system.

DOI: 10.4018/978-1-4666-5888-2.ch689

Before discussing the advances in real-time system, it is better to understand the core concepts of a real-time system. Here, a Job is a unit of work which is properly scheduled and executed by the system. A task is a set of selected jobs. Each task is characterized through several parameters. They are temporal parameters, functional parameters, resource parameters and interconnection parameters. The temporal parameter deals with the timing constraints and behavior of a task. The functional parameter deals with the intrinsic properties of the task. Resource parameter deals with the resource requirements like processor, and memory. Finally, the interconnection parameter deals with the dependency among the tasks. The primary properties of any real time system can be listed as time constraints involved, embedded nature, safety-critical issues, concurrency of tasks, distributed and feedback structure, task criticality, custom hardware, reactive nature, stability of operation, and the ways of exception handling. A real time task is generated when some event occurs. So the number of task emanating in system is more or less directly proportional to the event occurrence. When more number of events occur, the more number of tasks get generated. For example,temperature sensing task occurs periodically in a hard RTS, which is expected to monitor and control a heating furnace.

An adaptive or intelligent real time system needs to have additional parameters to adjust or adapt to the changing environment. The conventional real time system is not capable of handling the multi criticality of tasks in unusual circumstances. However, the general schemes like earliest deadline first scheduling algorithm, and some other priority based algorithms can precisely schedule all the tasks and dispatch them to the processors. The online scheduling algorithms can use the schedulability test to check whether a newly arrived different criticality task can be allowed into the system. The basic properties of any real time task can be depicted through its parameters.

Each task occurring in the system can have the following parameters.

1. Release time (ri) is the instant of time at which the task becomes available for execution.
2. Execution time (ei) is the amount of time required by task to complete its execution.
3. Response time is the length of time from the release time of the job to the time instant when it completes.
4. Deadline (di) is the instant of time at which a task must complete its execution.
5. The deadlines of each task must satisfy the condition ri+ei<=di.
   a. Absolute deadline is the instant of time by which a job is required to be completed
   b. and relative deadline is the maximum allowable response time.
6. Priority (pi) determines the order in which tasks are executed.

A task is said to precede another task, if the first task finishes its execution before the second task starts its execution. When a task Ti precedes another task Tj, then each instance of Ti precedes the corresponding instance of Tj. Tasks often share their results among each other when one task needs to share the results produced by another task; clearly, the second task must precede the first task.

Periodicity: Tasks can be periodic, aperiodic and sporadic. A periodic task Ti is repeated at every fixed time interval and can be represented by a 4 tuple $(\varphi_i, p_i, e_i, d_i)$ where φ_i is the phase of the task, p_i is the period of the task, e_i is the worst case execution time of the task, and d_i is the relative deadline of the task.

A sporadic task Ti occurs at random instants and can be represented by a three tuple: $T_i = (e_i, g_i, d_i)$ where e_i is the worst case execution time of an instance of the task, g_i denotes the minimum separation between two consecutive instances of the task; d_i is the relative deadline. Aperiodic task also occurs at random instant similar to sporadic tasks, but they often have soft deadlines.

Real-Time systems are classified by Kopetz (1997) from different perspectives. The classifications are hard real-time versus soft real-time, fail-safe versus fail-operational, guaranteed-timeliness versus best-effort, resource-adequate versus resource-inadequate, and event-triggered versus time-triggered. Among these, the first two classifications depend on the characteristics of the application, i.e., it depends on factors outside the computer system. The next three classifications depend on the design and implementation, i.e., it depends on factors inside the computer system.

BACKGROUND

Real Time System Scheduling

Earlier real time systems were scheduled by cyclic executives as cited by Lui Sha (2004) in an ad hoc manner. As the complexity was growing up, it became difficult to handle the situation and the generation of results became infeasible. Many important developments were initiated here. The transition from cyclic executive to fixed priority scheduling was initiated by Liu and Sha. This paved the way for Liu and Layland to bring up the successful structure of the fixed priority scheduling algorithms. A paper published by Liu, Chung Laung, and James W. Layland (1973) on scheduling of periodic tasks had certain assumptions. In this model, periodic tasks are time triggered with regular response timing. Basically, the scheduling of tasks can be deterministic clock driven or event driven. Feasibility analysis is used to predict the temporal behavior of tasks. This analysis determines whether the task constraints can be met in runtime or not. They also addressed the scheduling algorithms in a multiprogramming hard real time environment. The two main algorithms are deadline driven scheduling algorithm and rate monotonic scheduling algorithm. The execution of a single algorithm can have only 60% processor utilization. So, the combination of both the algorithms gives higher results. But the analysis and comparison were done on a single processor. Mok, (1978) and Chenk, (1987) state that the optimal scheduling without a prior knowledge is impossible in a multiprocessor case even with no restriction on pre-emption.

A schedule can be pre-computed (offline) or obtained dynamically (online). A task schedule is said to be feasible when all tasks completes its execution before their deadlines. Based on the relative priorities of tasks, preemptions are allowed. The assignment and scheduling of tasks in a multiprocessor environment is an NP-complete problem. The rationale of scheduling real-time tasks is to have a precise schedule for all tasks such that all real-time tasks can meet its deadline. Lui Sha and team (2004) have summarized the key results in real-time scheduling theory and recorded