Real Time Systems

Liu

According to their timing attributes, real time applications are of 4 types

Purely Cyclic

every task executes periodically, I/O operations are polled, resources do not vary from period to period. eg: digital controllers

Mostly cyclic

most tasks execute periodically, must respond to external events asynchronously. eg: modern avionics

Asynchronous and somewhat predictable

tasks not periodic, resources vary from period to period. eg: radar signal processing

Asynchronous and unpredictable

tasks have high run time complexities

Important Terms

- Release time: instant of time at which job becomes available for execution
- Deadline: instant of time by which execution is required to be completed
- Response time: length of time from the release time of the job to the instant when it completes
- Relative deadline: maximum allowable response time
- Deadline=release time+relative deadline
- Tardiness: measure of how late a job completes respective to deadline.

Important Terms

- Release time jitter: Range between earliest release time and latest release time
- Execution time, ei: amount of time required to complete the execution
- Laxity type: indicated whether the job is soft or hard
- Usefulness function: gives usefulness of result as a function of tardiness.

Hard vs Soft

Hard Deadline

failure to meet deadline is fatal

Soft Deadline

failure to meet deadline is not fatal

Performance degrades in case of missing deadlines

Periodic Task

- task is executed repeatedly at regular or semi regular time intervals
- Its period (pi) is the minimum length of all time intervals between release times of consecutive jobs
- Its execution time (ei) is the maximum execution time of all jobs in it
- Accuracy of the periodic task model decreases with increase in jitter in release times and variations in execution times

Periodic Task

- Release time of first job in each task is called the phase of that task (φi).
- Hyperperiod (Hi) is the LCM of periods of all tasks
- Utilization of a task, ui=ei/pi
- Total Utilization = sum of all individual utilizations

Sporadic

Tasks containing jobs that are released at random time instants and have hard deadlines

Aperiodic

Tasks containing jobs that are released at random time instants and have soft deadlines or no deadlines

Preemptable Jobs

its execution can be stopped any time and can be resumed from the point of suspension

Nonpreemptable Jobs

execution should not be interrupted

Conditions for a valid schedule

- Every processor is assigned to at most one job at any time
- Every job is assigned at most one processor at any time
- No job is scheduled before its release time
- The total amount of processor time assigned to every job is equal its maximum or actual execution time
- All the precedence and resource usage constrains are satisfied.

Clock Driven Approach

- Decisions on what jobs execute at what time are made at specific time instances. These instances are chosen before the system begins execution.
- All parameters of a hard real time job are fixed and known
- Schedule of the jobs is computed offline and is stored for use at runtime. Scheduler schedules according to this schedule at each scheduling decision time
- Minimal runtime overhead

Weighted Round-Robin Approach

- Every job joins a FIFO queue when it becomes ready for execution
- If the job is not completed by the end of the time slice, it is preempted and places at the end of the queue
- Processor sharing algorithm
- Each job gets 1/nth share of the processor
- More the weight of the job, more the time fraction allocated to the job

Priority Driven Approach

- resource idles only when no job requiring the resource is ready for execution
- it tries to make locally optimal decisions
- Processor allocated to processes according to priority

Earliest Deadline First (EDF)

- process with earliest deadline given highest priority
- processes do not share resources
- Can be preempted

Least Slack Time First (LSF)

- slack = relative deadline – execution left
- Optimal whenever EDF is optimal

Rate Monotonic Scheduling (RM)

- For periodic tasks only
- Task priority inversely proportional to its period