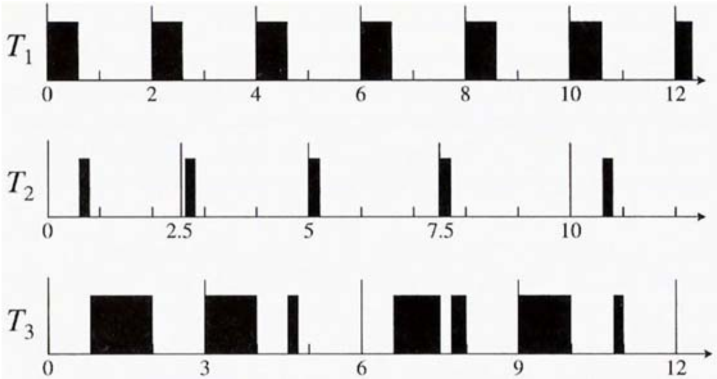
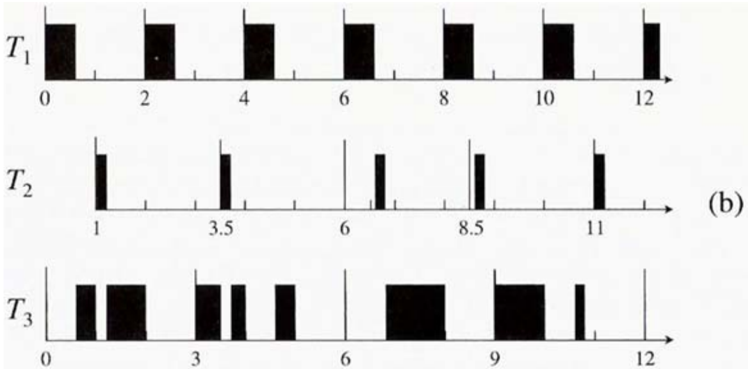
Example of Critical Instants (1/2)

- $T_1 = (2, 0.6)$, $T_2 = (2.5, 0.2)$, $T_3 = (3, 1.2)$
  - The response times of the jobs in T_2: **0.8**, 0.3, 0.2, 0.2, 0.8, ….
  - The response times of the jobs in T_3: **2**, 1.8, 2, 2, ….



Example of Critical Instants (2/2)

- $T_1 = (2, 0.6)$, $T_2 = (1, 2.5, 0.2)$, $T_3 = (3, 1.2)$
  - The response times of the jobs in T_2: 0.2, 0.2, **0.8**, 0.3, 0.2, ….
  - The response times of the jobs in T_3: 2, 2, **2**, 1.8, ….



Another Reason for Critical Instants

- Whenever release-time jitters are not negligible, the information on release times cannot be used to determine whether any algorithm can feasibly schedule the given system of tasks.
- Under this circumstance, we have no choice but to judge a fixed-priority algorithm according to its performance for tasks that are in phase because all fixed-priority algorithms have their worst-case performance for this combination of phases.

Outline

- A Schedulability Test for Fixed-Priority Tasks with Short Response Times:
  1. Critical Instants
  2. ***Time-Demand Analysis***
  3. Alternatives to Time-Demand Analysis

Time-Demand Analysis

- To determine whether a task can meet all its deadlines:
  - Compute the total demand for processor time by a job released at a critical instant of the task and by all the higher-priority tasks as a function of time from the critical instant.
  - Check whether this demand can be met before the deadline of the job.
- We consider one task at a time, starting from the task T_1 with the highest priority in order of decreasing priority.

Step by Step (1/2)

Assume all the tasks with higher priorities than T_i are schedulable, we want to determine whether the task T_i is schedulable.

1. Suppose the release time t_0 of the job is a critical instant of T_i.
2. At time $t_0 + t$ for $t \geq 0$, the total (processor) time demand $w_i(t)$ of this job and all the higher-priority jobs released in $[t_0, t]$ is given by

$$w_i(t) = e_i + \sum_{k=1}^{i-1} \lceil t / p_k \rceil \cdot e_k \quad \text{for } 0 < t \leq p_i$$

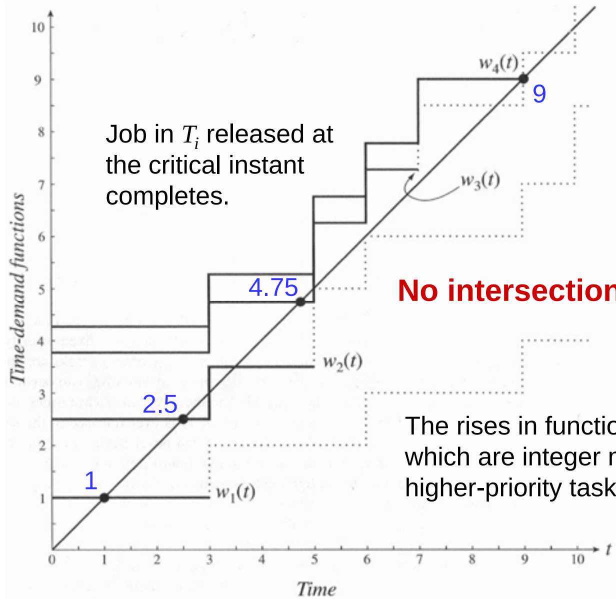
Supply of processor time: t
Demand of processor time in the interval: $w_i(t)$

Step by Step (2/2)

3. If $w_i(t) \leq t$ for some $t \leq D_i$, where D_i is equal to or less than p_i, all jobs in T_i can complete by their deadline.
4. If $w_i(t) > t$ for all $0 < t \leq D_i$, this job cannot complete by its deadline; T_i, and hence the given system of tasks, cannot be feasibly scheduled by the given fixed-priority algorithm.

- *Note that if the given tasks have known phases and periods and the jitters in release times are negligibly small, T_i may nevertheless be schedulable even though the time-demand analysis test indicates that it is not.*

Illustration of Time Demand Analysis



$T_1 = (\phi_1, 3, 1)$
$T_2 = (\phi_2, 5, 1.5)$
$T_3 = (\phi_3, 7, 1.25)$
$T_4 = (\phi_4, 9, 0.5)$

The rises in function occur at time instants which are integer multiples of periods of higher-priority tasks.

Time Demand Analysis Method (1/3)

- If we are not interested in the values of its maximum possible response time, only whether a task is schedulable, it suffices for us to check whether the time-demand function of the task is equal to or less than the supply at these instants.

Time Demand Analysis Method (2/3)

- Check whether the inequality

$$w_i(t) = e_i + \sum_{k=1}^{i-1} \lceil t / p_k \rceil \cdot e_k \le t$$

is satisfied for values of t that are equal to

$$t = jp_k;\ k = 1, 2, ..., i;\ j = 1, 2, ..., \lfloor \min(p_i, D_i) / p_k \rfloor$$

If this inequality is satisfied at any of these instants, T_i is schedulable.

- The time complexity of the time-demand analysis for each task is $O(np_n/p_1)$.

Time Demand Analysis Method (3/3)

- If we want to know the maximum possible response time W_i of each task T_i …
  - This can be done in an iterative manner, starting from an initial guess $t^{(1)}$ of W_i.

$$t^{(l+1)} = e_i + \sum_{k=1}^{i-1} \left\lceil \frac{t^{(l)}}{p_k} \right\rceil \times e_k$$

  - We terminate the iteration either when $t^{(l+1)}$ is equal to $t^{(l)}$ and $t^{(l)} \le p_i$ for some l or when $t^{(l+1)}$ becomes larger than p_i, whichever occurs sooner.
- How about W_3 in the previous example?

Robustness of the Time-Demand Analysis Method

- The conclusion that a task T_i is schedulable remains correct when
  - *the execution times of jobs may be less than their maximum execution times* and
  - *inter-release times of jobs may be larger than their respective periods*.