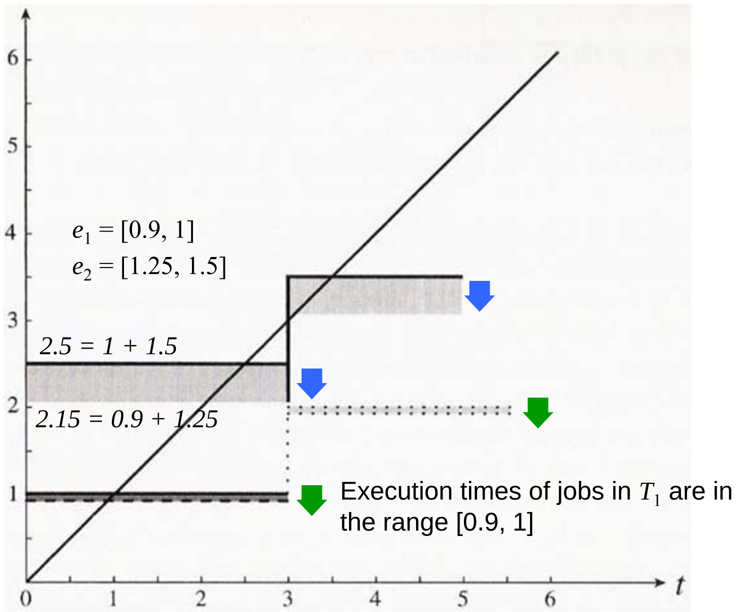
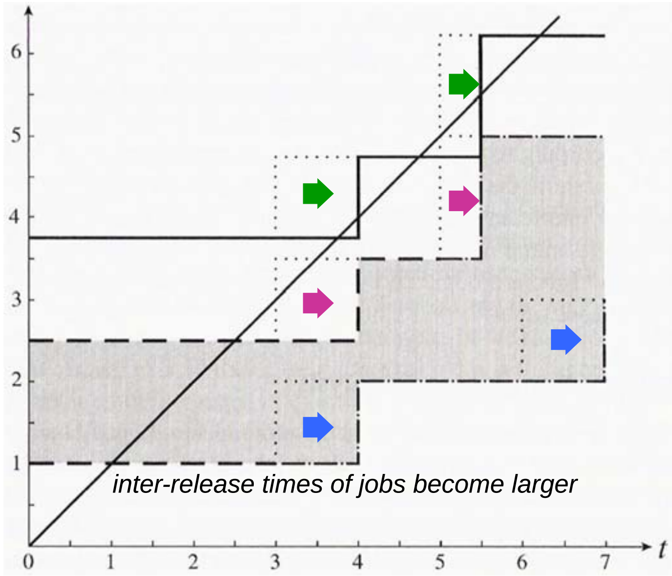




Outline

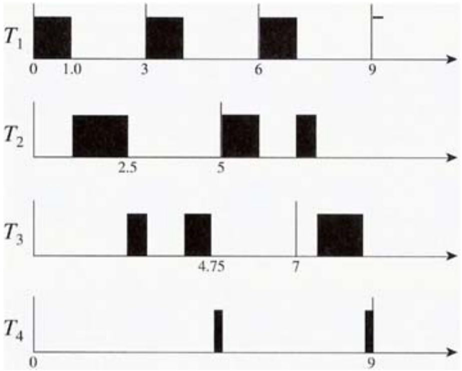
- A Schedulability Test for Fixed-Priority Tasks with Short Response Times:
  1. Critical Instants
  2. Time-Demand Analysis
  3. ***Alternatives to Time-Demand Analysis***

Alternative of Time-Demand Analysis

- We can determine whether a system of independent preemptable tasks is schedulable by simply *simulating the condition and observing whether the system is then schedulable*.
  - A way to test the schedulability of such a system is to construct a schedule of it according to the given scheduling algorithm
  - As long as Theorem 5 holds, it suffices for us to construct only the initial segment of length equal to the largest period of the tasks.

Worst-Case Simulation Method

- Assumptions:
  - The tasks are in phase
  - The actual execution times and inter-release times of jobs in each task T_i are equal to e_i and p_i, respectively.



A worst-case schedule of
- $T_1 = (3, 1)$
- $T_2 = (5, 1.5)$
- $T_3 = (7, 1.25)$
- $T_4 = (9, 0.5)$

The time complexity: $O(np_n/p_1)$.

TDAM vs. WCSM

- We can easily extend the time-demand analysis method to deal with other factors, such as nonpreemptivity and self-suspension, that affect the schedulability of a system, but we cannot easily extend the simulation method.
- When these factors must be taken into account, either we no longer know the worst-case condition for each task or the worst-case conditions for different tasks are different.

Outline

- Assumptions
- Fixed-Priority vs. Dynamic-Priority Algorithms
- Maximum Schedulable Utilization
- Optimality of the RM and DM Algorithms
- A Schedulability Test for Fixed-Priority Tasks with Short Response Times
- *Schedulability Test for Fixed-Priority Tasks with Arbitrary Response Times*
- Sufficient Schedulability Conditions for the RM and DM Algorithms

Introduction

- This section describes a general time-demand analysis method to determine the schedulability of tasks whose *relative deadlines are larger than their respective periods*.

  ➡ Since the response time of a task may be larger than its period, it may have *more than one job ready for execution at any time*.

- **References**
  - Lehoczky, J. P., “Fixed Priority Scheduling of Periodic Task Sets with Arbitrary Deadlines,” Proceedings of the IEEE Real-Time Systems Symposium, December 1990.

Busy Intervals (1/2)

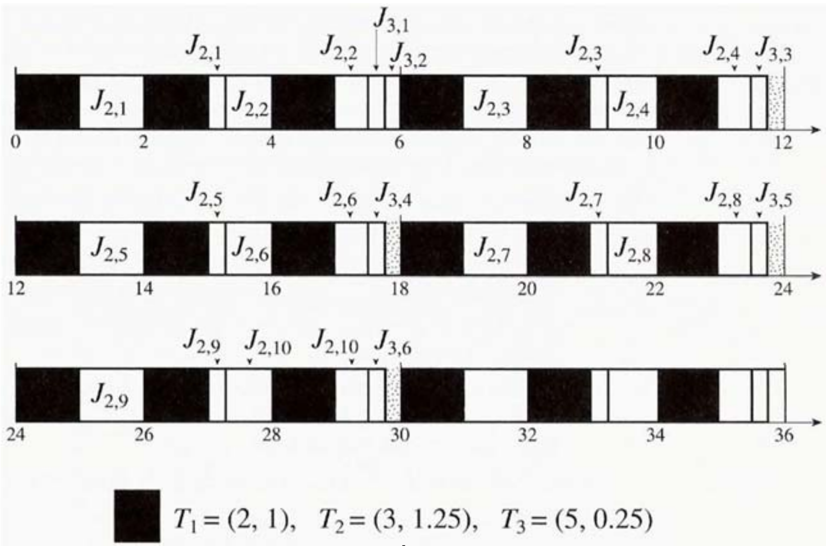
- A *level-π_i busy interval* $(t_0, t]$ begins at an instant t_0 when
  1. All jobs in $\mathbf{T}_i$ released before the instant have completed.
  2. A job in $\mathbf{T}_i$ is released.

  The interval ends at the first instant t after t_0 when all the jobs in $\mathbf{T}_i$ released since t_0 are complete.

Busy Intervals (2/2)

- In the interval $(t_0, t]$, the processor is busy all the time executing jobs with priorities π_i or higher, all the jobs executed in the busy interval are released in the interval, and at the end of the interval there is no backlog of jobs to be executed afterwards.
- **Definition.** We say that a level-π_i busy interval is in phase if the first jobs of all tasks that have priorities equal to or higher than priority π_i are executed in this interval and have the same release time.

Example Illustrating Busy Intervals (1/3)



$\pi_1 = 1, \pi_2 = 2,$ and $\pi_3 = 3$

Example Illustrating Busy Intervals (2/3)

- Every level-1 busy interval always ends 1 unit time after it begins.
- For this system, all the level-2 busy intervals are in phase.
  - They begin at times 0, 6, and so on which are the least common multiples of the periods of tasks T_1 and T_2.
  - Why? How about 3, 9, and so on?
  - The length of these intervals are all equal to 5.5.
- The second level-3 busy interval begins at time 6. (not in phase!)

Example Illustrating Busy Intervals (3/3)

- Given a task set: $T_1 = (1, 4, 2)$, $T_2 = (2, 5, 2)$, and $T_3 = (10, 1)$. Please identify the range of the 2nd level-2 busy interval and the range of the 2nd level-3 busy interval.

General Schedulability Test

- It still suffices to confine our attention to the special case where the tasks are in phase.
- However, the first job $J_{i,1}$ may no longer have the largest response time among all jobs in T_i.
  - Consider T_1=(70,26) and T_2=(100,62): seven jobs of T_2 execute in the first level-2 busy interval with response times = 114, 102, 116, 104, 118, 106, and 94.
- We must examine **all the jobs** of T_i that are executed in the first level-π_i busy interval.
- If the response times of all these jobs are no greater than the relative deadline of T_i, T_i is schedulable; otherwise, T_i may not be schedulable.

General Time-Demand Analysis Method (1/3)

- Test one task at a time starting from the highest priority task T_1 in order of decreasing priority.
- For the purpose of determining whether a task T_i is schedulable, assume that all the tasks are in phase and the first level-π_i busy interval begins at time 0.
- While testing whether all the jobs in T_i can meet their deadlines (i.e., whether T_i is schedulable), consider the subset $\mathbf{T}_i$ of tasks with priorities π_i or higher.

General Time-Demand Analysis Method (2/3)

1) If the first job of every task in $\mathbf{T}_i$ *completes by the end of the first period of the task*, check whether the first job $J_{i,1}$ in T_i meets its deadline. T_i is schedulable if $J_{i,1}$ completes in time. Otherwise, T_i is not schedulable.
2) If the first job of some task in $\mathbf{T}_i$ ***DOES NOT*** *complete by the end of the first period of the task*, do the following:

General Time-Demand Analysis Method (3/3)

a) Compute the length of the in phase level-π_i busy interval by solving the equation $t = \sum_{k=1}^{i} \lceil t / p_k \rceil \cdot e_k$ iteratively, starting from $t^{(1)} = \sum_{k=1}^{i} e_k$ until $t^{(l+1)} = t^{(l)}$ for some $l \geq 1$. The solution $t^{(l)}$ is the length of the level-π_i busy interval.

b) Compute the maximum response times of all $\lceil t^{(l)} / p_i \rceil$ jobs of T_i in the in-phase level-π_i busy interval in the manner described below and determine whether they complete in time.

T_i is schedulable if all these jobs complete in time; otherwise T_i is not schedulable.

Time-Demand Function $w_{i,1}(t)$ (1/2)

- The response time of the first job $J_{i,1}$ of T_i in the first in-phase level-π_i busy interval is similar to the time-demand function in Page 58.
- An important difference is that the expression remains valid for all $t > 0$ before the end of the busy interval.

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

- The maximum possible response time $W_{i,1}$ of $J_{i,1}$ is equal to the smallest value of t that satisfies the equation $t = w_{i,1}(t)$.

Time-Demand Function $w_{i,1}(t)$ (2/2)

- To obtain the maximum possible response time $W_{i,1}$ of $J_{i,1}$, we solve the equation iteratively and terminate the iteration only when we find $t^{(l+1)}$ equal to $t^{(l)}$.
- Because U_i is no greater than 1, this equation always has a finite solution, and the solution can be found after a finite number of iterations.

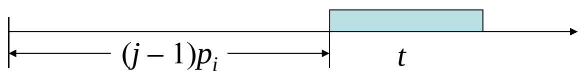
Time-Demand Function $w_{i,j}(t)$

- **Lemma 6.** The maximum response time $W_{i,j}$ of the j-th job of T_i in an in-phase level-π_i busy interval is equal to the smallest value of t that satisfies the equation

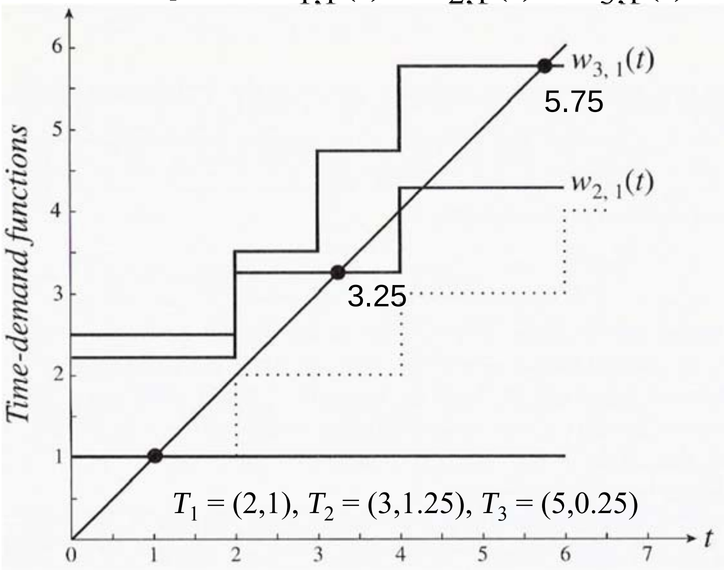
$$t = w_{i,j}(t + (j-1)p_i) - (j-1)p_i$$

where $w_{i,j}(\,)$ is given by

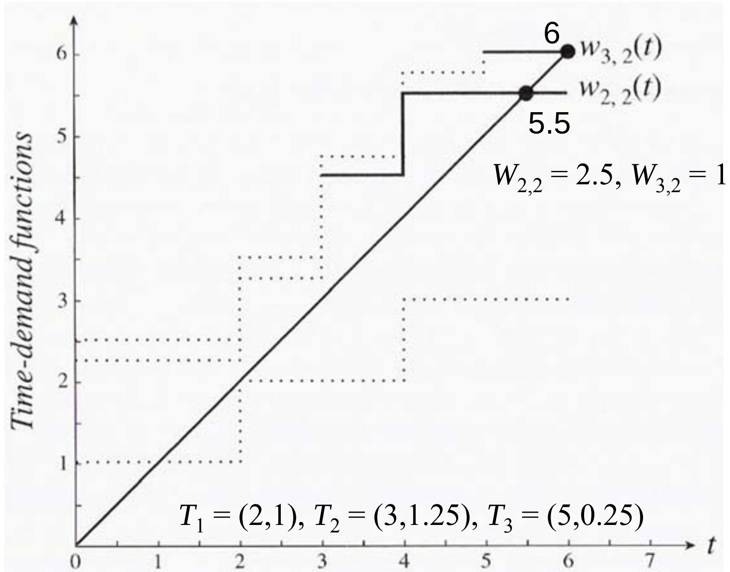
$$w_{i,j}(t) = je_i + \sum_{k=1}^{i-1} \left\lceil \frac{t}{p_k} \right\rceil \cdot e_k \qquad \text{for } (j-1)p_i < t \leq w_{i,j}(t)$$



Example: $w_{1,1}(t)$, $w_{2,1}(t)$, $w_{3,1}(t)$



Example: $w_{2,2}(t)$, $w_{3,2}(t)$



Example: Find $W_{2,2}$

$T_1 = (2,1)$, $T_2 = (3,1.25)$, $T_3 = (5,0.25)$

- $t = w_{i,j}(t + (j-1)p_i) - (j-1)p_i$
- $w_{i,j}(t) = je_i + \sum_{k=1}^{i-1} \lceil t / p_k \rceil \cdot e_k$ for $(j-1)p_i < t \le w_{i,j}(t)$

$$t = 2 \times 1.25 + \lceil (t+3)/2 \rceil - 3$$

Substitute $t^{(1)} = 1.25$ on the right hand side of the equation.

We obtain: $W_{2,2} = 2.5$

How about $W_{3,2}$?

Outline

- Assumptions
- Fixed-Priority vs. Dynamic-Priority Algorithms
- Maximum Schedulable Utilization
- Optimality of the RM and DM Algorithms
- A Schedulability Test for Fixed-Priority Tasks with Short Response Times
- Schedulability Test for Fixed-Priority Tasks with Arbitrary Response Times
- *Sufficient Schedulability Conditions for the RM and DM Algorithms*