Lecture 13

Scheduling with precedence constraints:

Lecture note 7 Real time systems 4^{th} year

3.5.2 EDF WITH PRECEDENCE CONSTRAINTS

The problem of scheduling a set of *n* tasks with precedence constraints and dynamic activations can be solved in polynomial time complexity only if tasks are preemptable. In 1990, Chetto, Silly, and Bouchentouf [CSB90] presented an algorithm that solves this problem in elegant fashion. The basic idea of their approach is to transform a set *J* of dependent tasks into a set *J* □ of independent tasks by an adequate modification of timing parameters. Then, tasks are scheduled by the Earliest Deadline First (EDF) algorithm. The transformation algorithm ensures that *J* is schedulable and the precedence constraints are obeyed if and only if *J* □ is schedulable. Basically, all release times and deadlines are modified so that each task cannot start before its predecessors and cannot preempt their successors.

MODIFICATION OF THE RELEASE TIMES

The rule for modifying tasks' release times is based on the following observation.

Given two tasks *Ja* and *Jb*, such that *Ja* → *Jb* (that is, *Ja* is an immediate predecessor of *Jb*), then in any valid schedule that meets precedence constraints the following conditions must be satisfied :

$sb \geq rb$ (that is, *Jb* must start the execution not earlier than its release time);

$sb \geq ra + Ca$ (that is, *Jb* must start the execution not earlier than the minimum finishing time of *Ja*).

Therefore, the release time r_b of J_b can be replaced by the maximum between r_b and $(r_a + C_a)$ without changing the problem. Let r^*_b be the new release time of J_b. Then,

$r^*_b = \max(r_b, \; r_a + C_a)$.

The algorithm that modifies the release times can be implemented in $O(n^2)$ and can be described as follows:

1. For any initial node of the precedence graph, set $r^*_i = r_i$.

2. Select a task J_i such that its release time has not been modified but the release times of all immediate predecessors J_h have been modified. If no such task exists, exit.

3. Set $r^*_i = \max[r_i, \; \max(r^*_h + C_h : J_h \rightarrow J_i)]$.

4. Return to step 2.

MODIFICATION OF THE DEADLINES

The rule for modifying tasks' deadlines is based on the following bservation. Given two tasks J_a and J_b, such that $J_a \rightarrow J_b$ (that is, J_a is an immediate predecessor of J_b), then in any feasible schedule that meets the precedence constraints the following conditions must be satisfied :

$f_a \leq d_a$ (that is, J_a must finish the execution within its deadline);

$f_a \leq d_b - C_b$ (that is, J_a must finish the execution not later than the maximum start time of J_b).

Therefore, the deadline d_a of J_a can be replaced by the minimum between d_a and $(d_b - C_b)$ without changing the problem. Let d^*_a be the new deadline of J_a. Then,

$d^*_a = \min(d_a, \; d_b - C_b)$.

The algorithm that modifies the deadlines can be implemented in $O(n^2)$ and can be described as follows:

1. For any terminal node of the precedence graph, set $d^*_i = d_i$.
2. Select a task J_i such that its deadline has not been modified but the deadlines of all immediate successors J_k have been modified. If no such task exists, exit.
3. Set $d^*_i = \min[d_i, \; \min(d^*_k - C_k : J_i \rightarrow J_k)]$.
4. Return to step 2.

Question 3.5 at page 78 of the book

Given seven tasks, A, B, C, D, E, F, and G, construct the precedence graph from the following precedence relations:

$A \rightarrow C$

$B \rightarrow C$ $\quad$ $B \rightarrow D$

$C \rightarrow E$ $\quad$ $C \rightarrow F$

$D \rightarrow F$ $\quad$ $D \rightarrow G$

Then, assuming that all tasks arrive at time $t = 0$, have deadline D = 25, and computation times 2, 3, 3, 5, 1, 2, 5, respectively, modify their arrival times and deadlines to schedule them by EDF

Solution:

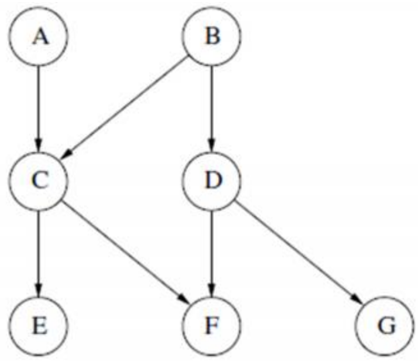

Figure 13.3 Precedence graph for Exercise 3.5.

	C_i	r_i	$r*_i$	d_i	$d*_i$
A	2	0	0	25	20
B	3	0	0	25	15
C	3	0	3	25	23
D	5	0	3	25	20
E	1	0	6	25	25
F	2	0	8	25	25
G	5	0	8	25	25