Embedded Systems FS 2016

Solution to Exercise 2: Aperiodic Scheduling - RT Systems

Discussion Date: 23.03.2016 / 06.04.2016

Aufgabe 1: EDD

Check whether the Earliest Deadline Due (EDD) algorithm produces a feasible schedule for the following task set, given that all tasks are synchronous and arrive at time $t = 0$.

<table>
<thead>
<tr>
<th>$J_1$</th>
<th>$J_2$</th>
<th>$J_3$</th>
<th>$J_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_i$</td>
<td>3</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>$D_i$</td>
<td>8</td>
<td>15</td>
<td>3</td>
</tr>
</tbody>
</table>

Solution - Task 1

EDD (Earliest Deadline Due) is a scheduling algorithm that minimizes the maximum lateness. The Jackson’s rule says that: given a set of $n$ independent tasks, any algorithm that executes the tasks in order of non-decreasing deadlines is optimal with respect to minimizing the maximum lateness.

Assumptions about the task set for applying EDD:
- tasks have same arrival times (synchronous arrivals)
- tasks are independent

EDD is non-preemptive.

The EDD produces a feasible schedule given in the figure below.

The tasks are scheduled in the order of non-decreasing deadlines as follows: $J_3, J_1, J_4, J_2$. 
Given the precedence graph in Figure 1 and the following table of task execution times and deadlines, determine the Latest Deadline First (LDF) schedule. Is the schedule feasible?

<table>
<thead>
<tr>
<th></th>
<th>J1</th>
<th>J2</th>
<th>J3</th>
<th>J4</th>
<th>J5</th>
<th>J6</th>
<th>J7</th>
<th>J8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ci</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Di</td>
<td>5</td>
<td>8</td>
<td>11</td>
<td>15</td>
<td>12</td>
<td>18</td>
<td>19</td>
<td>20</td>
</tr>
</tbody>
</table>

Figure 1: Precedence graph.

Solution - Task 2

LDF (Latest Deadline First) is a scheduling algorithm that minimizes the maximum lateness. It assumes synchronous task activations and is non preemptive.

The algorithm to produce an LDF schedule proceeds in two stages: firstly, a precedence graph is constructed. Going from tail to head: among the tasks without successors or whose successors have been all selected, LDF selects the tasks with the latest deadline to be scheduled last. At runtime, tasks are extracted from the head of the queue: the first task inserted in the queue will be executed last.

The queue of tasks scheduled with LDF is as follows: J1 - J2 - J3 - J5 - J4 - J6 - J7 - J8 (on the next page, you can find a sketch of how the LDF algorithm proceeds). The LDF schedule is presented below:

Figure 2: LDF schedule.
The LDF algorithm proceeds as depicted above (follow the figures left to right).

**Aufgabe 3: EDF**

Given are five tasks with arrival times, execution times and deadlines according to the following table. (1) Determine the Earliest Deadline First (EDF) schedule. Is the schedule feasible?

<table>
<thead>
<tr>
<th></th>
<th>$J_1$</th>
<th>$J_2$</th>
<th>$J_3$</th>
<th>$J_4$</th>
<th>$J_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_i$</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>8</td>
<td>13</td>
</tr>
<tr>
<td>$C_i$</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>$d_i$</td>
<td>16</td>
<td>7</td>
<td>8</td>
<td>11</td>
<td>18</td>
</tr>
</tbody>
</table>
(2) At time $t = 3$, a new task $J_x$ arrives with execution time $C_x = 2$ and deadline $d_x = 10$. Can you guarantee the schedulability of the task set with this new task or do you have to reject it?

Solution - task 3

EDF (Earliest Deadline First) is optimal in the sense of feasibility (it minimizes the maximum lateness under following assumptions: scheduling algorithm is preemptive, the tasks are independent, and may have arbitrary arrival times). The Horn’s rule says that given a set of $n$ independent tasks with arbitrary arrival times any algorithm that at any instant executes the task with earliest absolute deadlines among the ready tasks is optimal with respect to the maximum lateness.

(1) The EDF schedule is feasible, and the respective schedule is shown in the Figure 3.

![Figure 3: EDF schedule.](image)

(2) The arrival of a new task $J_x$ at time point $t = 3$ still maintains the schedule feasible. We can check this by computing ahead the finishing times of the tasks (with respect to the interesting time points). In the analysis, we consider the tasks in order of increasing deadlines.

We assume an on-line scenario, i.e. in the schedulability test we consider only tasks currently present in the system. The test is performed every time a new task arrives.

From the schedule in the preceding sub-problem, we know that before task $J_x$ arrives at time $t = 3$ we have:

- tasks $J_1$, $J_2$ and $J_3$ are feasible
- task $J_2$ finishes before its deadline at $t = 3$

That is, up to $t = 3$ all active tasks in the system are feasible.

At $t = 3$ we have three tasks in the system: $J_1$, $J_3$ and the new task, $J_x$. For them we perform the schedulability test:

Put $f_0 = t = 3$ (We have absolute deadlines in the problem specification.)

Task $J_3$: $f_1 = f_0 + c_3(3) = 3 + 4 = 7 \leq 8 \text{ (ok)}$

Task $J_x$: $f_2 = f_1 + c_x(3) = 7 + 2 = 9 \leq 10 \text{ (ok)}$

Task $J_1$: $f_3 = f_2 + c_1(3) = 9 + 3 = 12 \leq 16 \text{ (ok)}$

Thus, at $t = 3$ all tasks in the system are feasible.

The next task to arrive is $J_4$. It arrives at $t = 8$. At $t = 8$, we have thee active tasks: $J_x$, $J_4$ and $J_1$. The schedulability test at $t = 8$ proceeds as follows:
Put $f_0 = t = 8$

Task $J_x$: $f_1 = f_0 + c_x(8) = 8 + 1 = 9 \leq 10$ (ok)

Task $J_4$: $f_2 = f_1 + c_4(8) = 9 + 2 = 11 \leq 11$ (ok)

Task $J_1$: $f_3 = f_2 + c_1(8) = 11 + 3 = 14 \leq 16$ (ok)

Thus, at $t = 8$ all tasks in the system are feasible.

Finally, task $J_5$ arrives at $t = 13$. At $t = 13$, we have two active tasks: $J_1$ and $J_5$. The schedulability test at $t = 13$ proceeds as follows:

Put $f_0 = t = 13$

Task $J_1$: $f_1 = f_0 + c_1(13) = 13 + 1 = 14 \leq 16$ (ok)

Task $J_5$: $f_2 = f_1 + c_5(13) = 14 + 3 = 17 \leq 18$ (ok)

Hence, the whole schedule is feasible. It is given in the figure below:

Aufgabe 4: EDF$^*$

Given are seven tasks $A, B, C, D, E, F, G$ with following precedence constraints:

\[ A \rightarrow C, \quad B \rightarrow C, \quad C \rightarrow E, \quad D \rightarrow F, \quad B \rightarrow D, \quad C \rightarrow F, \quad D \rightarrow G \]

All tasks arrive at time $t_0 = 0$, have a common deadline $d = 20$ and the following execution times:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_i$</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>1</td>
</tr>
</tbody>
</table>

1. Construct the precedence graph for this task set. Then, modify the release times and deadlines so that EDF$^*$ can be used for its scheduling.

2. Draw the resulting EDF$^*$ schedule and compute the average response time of the tasks.

3. Assume the additional precedence constraint $E \rightarrow A$. Is there still a feasible schedule for the above task set? Justify your answer.
Solution - task 4

EDF* schedules tasks with precedence constraints. Release time and deadline of individual tasks are modified such that all the precedence constraints are satisfied. By doing this, the scheduling problem is transformed into a problem without precedence constraints, which can then be handled by a "normal" EDF scheduler.

There are several points to take into consideration while modifying tasks’ release times and deadlines:

- **Release time modification:**
  - task must start the execution not earlier than its release time
  - task must start the execution not earlier than the minimum finishing time of its predecessors

- **Deadline modification:**
  - task must finish within its deadline
  - task must finish not later than the maximum start time of its successors

(1) The precedence graph for the given task set is:

The modified release times and deadlines are:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r^*_r$</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>7</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>$d^*_r$</td>
<td>11</td>
<td>11</td>
<td>15</td>
<td>15</td>
<td>20</td>
<td>20</td>
<td>20</td>
</tr>
</tbody>
</table>

In particular, the release time modification proceeds as follows:

- for initial nodes $\{A, B\}$ set $r^*_A = r_A, r^*_B = r_B$
- $r^*_C = \max\{r_c, \max\{r^*_A + C_A, r^*_B + C_B\} = \max\{0, \max\{3, 2\}\} = 3$
- $r^*_D = \max\{r_D, r^*_B + C_B\} = \max\{0, 0 + 2\} = 2$
- $r^*_E = \max\{r_F, r^*_C + C_C, r^*_D + C_D\} = \max\{0, 3 + 4, 2 + 3\} = 7$
- $r^*_F = \max\{r_E, r^*_C + C_C\} = \max 0, 3 + 4 = 7$
- $r^*_G = \max\{r_G, r^*_D + C_D\} = \max 0, 2 + 3 = 5$

The deadline modification proceeds as follows:

- for the terminal nodes $\{E, F, G\}$ set $d^*_E = d^*_F = d^*_G = 20$
- $d^*_C = \min\{d_C, d^*_E - C_E, d^*_F - C_F\} = \min\{20, 20 - 2, 20 - 5\} = 15$
- $d^*_D = 15$
- $d^*_A = 11$
- $d^*_B = 11$
(2) The resulting EDF* schedule (based on the modified release times and deadlines) is presented below.

The response time of a task is defined as the difference between its finishing time and release time. Therefore, the average response time for the above schedule is computed as follows:

\[
\bar{r} = \frac{1}{7} \sum_{i=1}^{7} (f_i - r_i) = \frac{5 + 2 + 12 + 8 + 20 + 18 + 13}{7} = 11.14
\]

Note that for the computation of a task’s response time we consider its original (not modified) release time.

(3) No, the task set is no longer schedulable. Under the new conditions, the constraints among tasks A, C and E introduce a cycle in the precedence graph. As a result, none of the three tasks can be executed as first and therefore, no feasible schedule exists.

**Aufgabe 5: EDF with Precedence Constraints**

Given is a set of 6 aperiodic tasks named A, B, C, D, E and F with the following precedence relations:

\[A \rightarrow B, \quad A \rightarrow C, \quad B \rightarrow D, \quad B \rightarrow E, \quad C \rightarrow F\]

The execution times \(C\), arrival times \(r\) and absolute deadlines \(d\) are as follows:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>(C)</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>(d)</td>
<td>4</td>
<td>10</td>
<td>8</td>
<td>6</td>
<td>10</td>
<td>12</td>
</tr>
<tr>
<td>(r)</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>7</td>
<td>9</td>
</tr>
</tbody>
</table>

(1) Draw the schedule for the case when the EDF policy ignores any precedence constraints.
(2) Draw the schedule for the case when the EDF policy releases a task after all its predecessors (defined by the precedence constraints above) have finished execution.

(3) Determine the modified release times and deadlines for the EDF* policy and draw the corresponding schedule.

Which of the previous schedules would you accept?

Solution - task 5

(1) The EDF schedule is feasible (all deadlines are met), but the given precedence constraints are not respected. Therefore, it cannot be accepted.

![Figure 6: EDF schedule (disregarding precedence relationships)](image)

(2) The EDF schedule that respects the precedence constraints is not feasible, since the deadline of task $D$ ($d_D = 6$) cannot be met.

![Figure 7: EDF schedule (considering precedence relationships)](image)

(3) The precedence graph and the modified release times and deadlines are:

<table>
<thead>
<tr>
<th></th>
<th>$r_i^*$</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r_i^*$</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>7</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>$d_i^*$</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>6</td>
<td>10</td>
<td>12</td>
<td></td>
</tr>
</tbody>
</table>
The resulting EDF* schedule is feasible and all constraints are met.