Embedded Systems

Exercise 2:
Scheduling Real-Time Aperiodic Tasks

6. April 2016

Stefan Draskovic, ETZ G81
stefan.draskovic@tik.ee.ethz.ch
Definitions: RT Tasks

Arrival Time $a$

Deadline $d$

Execution Time $C$

Scheduling

Response Time $R$

Lateness $L$
Overview

Scheduling aperiodic tasks with real-time constraints

Some known algorithms:

<table>
<thead>
<tr>
<th></th>
<th>Equal arrival times</th>
<th>Arbitrary arrival times</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>non preemptive</td>
<td>preemptive</td>
</tr>
<tr>
<td>Independent tasks</td>
<td>EDD (Jackson)</td>
<td>EDF (Horn)</td>
</tr>
<tr>
<td>Dependent tasks</td>
<td>LDF (Lawler)</td>
<td>EDF* (Chetto)</td>
</tr>
</tbody>
</table>
Earliest Due Deadline (EDD) (1/2)

Optimization goal:
- minimize the maximum lateness

Assumptions on the task set:
- independent tasks
- synchronous arrival times

EDD is non-preemptive
Earliest Due Deadline (EDD) (2/2)

Jackson’s rule:
execute tasks in order of non decreasing deadlines
Earliest Due Deadline (EDD) (Solution)
Latest Deadline First (LDF) (1/3)

Optimization goal:
– minimize the maximum lateness

Assumptions on the task set:
– tasks with precedence relations
– synchronous arrival times

LDF is non preemptive

Note: Not covered in lecture!
Main Idea [Lawler]:
Build a scheduling queue. At run time, execute tasks from that queue

- Building the scheduling queue:
  - proceed from tail to head
  - among the tasks without successors or whose successors have been all scheduled, select the task with the latest deadline to be scheduled last
  - repeat the procedure until all tasks in the set are selected
Latest Deadline First (LDF) (3/3)

Building the scheduling queue (example):

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>$d_i$</td>
<td>4</td>
<td>4</td>
<td>8</td>
<td>12</td>
<td>15</td>
</tr>
</tbody>
</table>

DAG

Scheduling Queue:

run-time scheduler
Latest Deadline First (LDF) (Solution)

Scheduling Queue: J8 J7

run-time scheduler
Latest Deadline First (LDF) (Solution)

Scheduling Queue: J8 J7 J6

run-time scheduler
Latest Deadline First (LDF) (Solution)

Scheduling Queue: J8 J7 J6 J4 J5

run-time scheduler
Latest Deadline First (LDF) (Solution)

Scheduling Queue: J8, J7, J6, J4, J5, J3

run-time scheduler
Latest Deadline First (LDF) (Solution)

Scheduling Queue: J8 J7 J6 J4 J5 J3 J2

run-time scheduler
Latest Deadline First (LDF) (Solution)

Scheduling Queue: J8 J7 J6 J4 J5 J3 J2 J1

run-time scheduler
Latest Deadline First (LDF) (Solution)

Scheduling Queue: J8 J7 J6 J4 J5 J3 J2 J1
Earliest Deadline First (EDF) (1/3)

Optimization goal:
- minimize the maximum lateness

Assumptions on the task set:
- independent tasks
- arbitrary arrival times (dynamic arrivals)

EDF is preemptive
Earliest Deadline First (EDF) (2/3)

Horn’s rule:
- at any instant execute the task with the earliest absolute deadline
Earliest Deadline First (EDF) (3/3)

- **Guarantee:**
  - worst case finishing time of task $i$: $f_i = t + \sum_{k=1}^{i} c_k(t)$
  - EDF guarantee condition: $\forall i = 1, \ldots, n \quad t + \sum_{k=1}^{i} c_k(t) \leq d_i$

- **algorithm:**

```plaintext
Algorithm: EDF_guarantee (J, J_{new})
{    J′ = J \cup \{J_{new}\}; /* ordered by deadline */
    t = current_time();
    f_{0} = t;
    for (each $J_i \in J'$) {
        f_i = f_{i-1} + c_i(t);
        if (f_i > d_i) return(INFEASIBLE);
    }
    return(FEASIBLE);
}
```
Earliest Deadline First (EDF) (Solution)
Earliest Deadline First (EDF) (Solution)

At time t=3, existing tasks: J1, J2, J3
  • J2 has finished, J1 / J3 are feasible

Schedulability test after arrival of Jx:

\[
\begin{align*}
\text{Put } f_0 &= t = 3 \\
\text{Task } J_3: & f_1 = f_0 + c_3(3) = 3 + 4 = 7 \leq 8 \text{ (ok)} \\
\text{Task } J_x: & f_2 = f_1 + c_x(3) = 7 + 2 = 9 \leq 10 \text{ (ok)} \\
\text{Task } J_1: & f_3 = f_2 + c_1(3) = 9 + 3 = 12 \leq 16 \text{ (ok)}
\end{align*}
\]

Repeat for t= 8, 13
Earliest Deadline First (EDF*) (1/3)

Optimization goal:
- minimize the maximum lateness

Assumptions on the task set:
- tasks with precedence relations
- arbitrary arrival times

EDF* is preemptive
**Earliest Deadline First (EDF*) (2/3)**

**Main idea:**
modify deadlines and arrival times to satisfy the precedence constraints and then use EDF

Modifying arrival times:
✓ Task must start the execution not earlier than its release time
✓ Task must not start the execution earlier than the minimum finishing time of its predecessor

\[ r_j^* = \max(r_j, \max(r_i^* + C_i : T_i \rightarrow T_j)) \]

\[ \text{T1} \quad \text{T2} \quad \text{T3} \]
Main idea:
modify deadlines and arrival times to satisfy the precedence constraints and then use EDF

Modifying deadlines:
✓ Task must finish the execution within its deadline
✓ Task must not finish the execution later than the maximum start time of its successor

\[ d_i^* = \min\left(d_i, \min\left(d_j^* - C_j : T_i \rightarrow T_j\right)\right) \]
Earliest Deadline First (EDF*) (Solution 4)

<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>$r_i^*$</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>$d_i^*$</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>
Earliest Deadline First (EDF*) (Solution 4)
Earliest Deadline First (EDF*) (Solution 5(1))
Earliest Deadline First (EDF*) (Solution 5(2))
Earliest Deadline First (EDF*) (Solution 5(3))

\[ \begin{array}{cccccc}
  \text{A} & \text{B} & \text{C} & \text{D} & \text{E} & \text{F} \\
  r_i^* & 0 & 2 & 2 & 4 & 7 \\
  d_i^* & 2 & 4 & 8 & 6 & 10 & 12 \\
\end{array} \]
Earliest Deadline First (EDF*) (Solution 5(3))
Recap

Feasibility of a schedule:
All tasks complete before their prescribed deadlines