

Aperiodic Scheduling

Earliest Deadline Due, Latest Deadline First, Earliest Deadline First

Exercise class 4

Presenter:
Jürgen Mattheis

In cooperation with:
Pascal Walter

Based on the lecture of:
Marco Zimmerling

November 28, 2022

University of Freiburg, Chair for Embedded Systems



Gliederung

Organisation

Overview Aperiodic Task Scheduling

Task 1

Task 2

Task 3



Task 4

Organisation






Organisation I

► feedback for me: <https://forms.gle/f3YN8EFrZ1vsfPoC6>

Feedback for Introduction to ESE Tutor Jürgen  

Questions **Responses** Settings

0 responses  

Accepting responses 

Waiting for responses

Overview Aperiodic Task Scheduling



Overview Aperiodic Task Scheduling

	Equal arrival times non preemptive	Arbitrary arrival times preemptive
Independent tasks	EDD (Jackson)	EDF (Horn)
Dependent tasks	LDF (Lawler)	EDF* (Chetto)

Task 1

Task 1

Earliest Deadline Due

Task 1.1:

	J_1	J_2	J_3	J_4
C_i	3	6	2	4
D_i	8	15	3	11

$(\forall J_i : r_i = 0)$

Requirements 1.1:

- ▶ *non-preemptive*
- ▶ tasks have *same arrival times* (synchronous arrivals)
- ▶ $\min(D_i)$ for all remaining J_i
- ▶ *minimizing the maximum lateness*
- ▶ tasks are *independent*

Task 1

Earliest Deadline Due

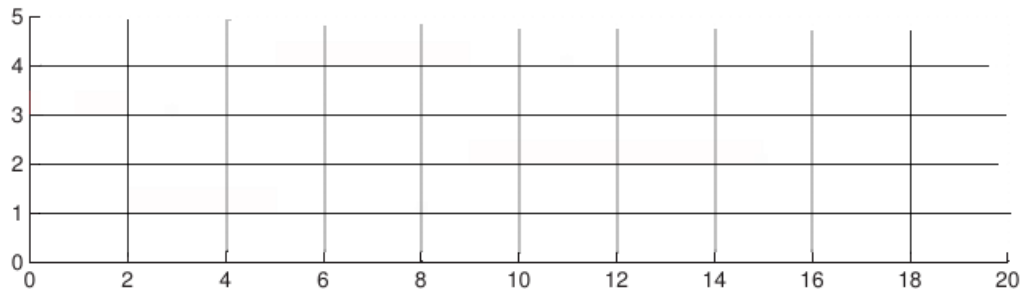


Figure 1: EDD schedule.

Task 1

Earliest Deadline Due

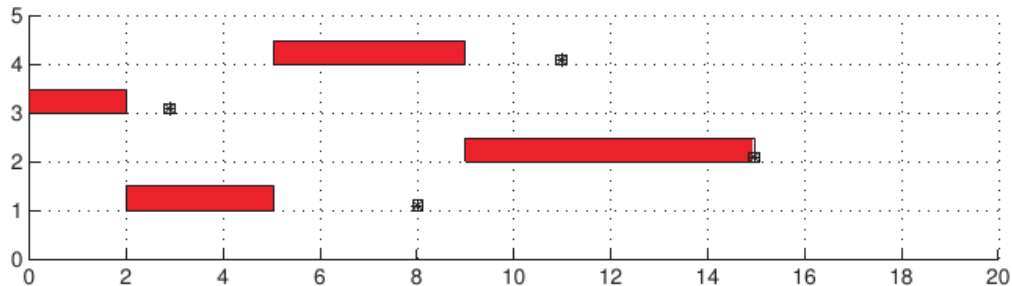


Figure 1: EDD schedule.

Task 2



Task 2

Latest Deadline First

Task 2.1:

	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8
C_i	3	4	2	3	3	2	2	1
D_i	5	8	11	15	12	18	19	20

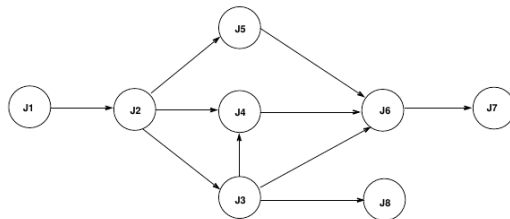


Figure 2: Precedence graph.

Task 2

Latest Deadline First

Requirements 2.1:



- ▶ *is non-preemptive*
- ▶ *synchronous task activations*
- ▶ *$\min(D_i)$ for all tasks J_i without successors or whose successors have been all selected in the precedence graph*
- ▶ *minimizes the maximum lateness*

Task 2

Latest Deadline First

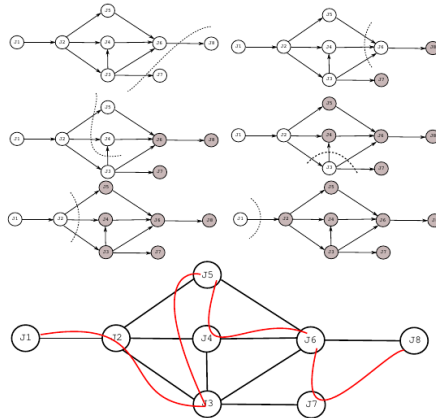


Figure 4: The LDF algorithm proceeds as depicted (figures left to right)

Task 2

Latest Deadline First

► queue of tasks: (, , , , , , ,)

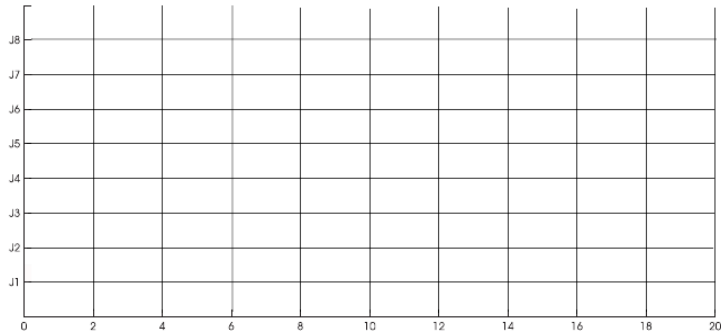


Figure 3: LDF schedule.

Task 2

Latest Deadline First

- queue of tasks: $(J_1, J_2, J_3, J_5, J_4, J_6, J_7, J_8)$

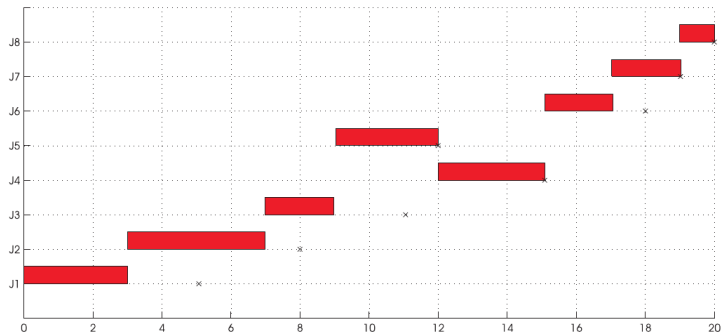


Figure 3: LDF schedule.

Task 3



Task 3

Earliest Deadline First

Task 3.1:

	J_1	J_2	J_3	J_4	J_5
a_i	0	2	0	8	13
C_i	3	1	6	2	3
d_i	16	7	8	11	18

Requirements 3.1:

- ▶ *is preemptive*
- ▶ *arbitrary arrival times*
- ▶ *the tasks are independent*
- ▶ *minimizes the maximum lateness*
- ▶ *$\min(D_i)$ for all remaining tasks J_i that have already arrived (are ready) and not finished*

Task 3

Earliest Deadline First

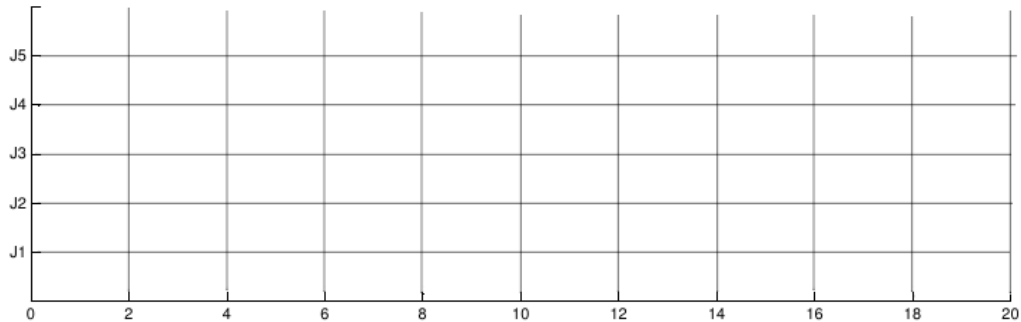


Figure 5: EDF schedule

Task 3

Earliest Deadline First

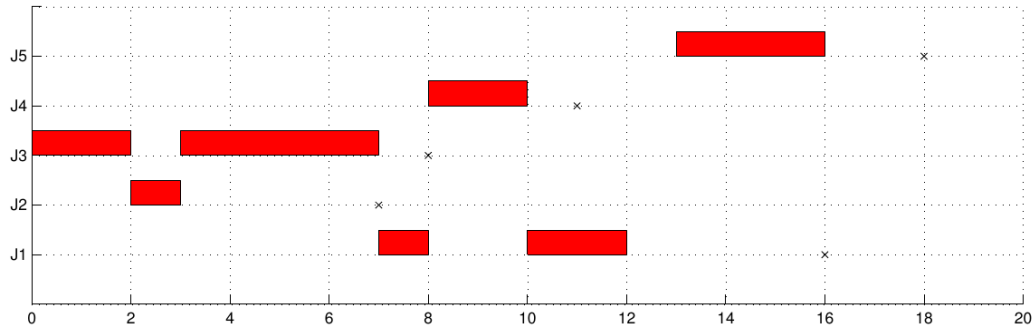


Figure 5: EDF schedule

Task 3

Earliest Deadline First

Task 3.2:

- ▶ at time $t = 3$, a new task J_x arrives with execution time $C_x = 2$ and deadline $d_x = 10$.
- ▶ still **guarantee the schedulability** of task set?

Requirements 3.2:

- ▶ $\forall i = 1, \dots, n \quad t + \sum_{k=1}^i c_k(t) \leq d_i$

Task 4



Task 4

EDF*

- ▶ EDF* is an alternative version of the EDF Algorithmus
- ▶ EDF* helps us to schedule Tasks with arbitrary arrival times and precedences. (contrary to EDF)
- ▶ EDF* manages this in polynomial time!

Task 4 I

EDF* - Transformation

- ▶ EDF* transforms the arrival time and deadline of every task in the following way::

Deadline:

1. Task must finish the execution time within its deadline: $f_i \leq d_i$
2. Task must not finish the execution time later than the maximum start time of its successor(s): $f_i \leq d_j - C_j$

$$\rightarrow d_i^* = \min(d_i, \min(d_j^* - C_j : J_i \rightarrow J_j))$$

Task 4 II

EDF* - Transformation

Arrival time

1. Task must start the execution not earlier than its release time: $s_j \geq r_j$
2. Task must not start execution earlier than the minimum finishing time of its predecessor(s): $s_j \geq r_i + C_i$

$$\rightarrow r_j^* = \max(r_j, \max(r_i^* + C_i : J_i \rightarrow J_j))$$

Task 4 I

EDF* - Example

Given tasks A, B, C, D, E, F, G with precedences $A \rightarrow C, B \rightarrow C, C \rightarrow E, D \rightarrow F, B \rightarrow D, C \rightarrow F, D \rightarrow G$.

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

	A	B	C	D	E	F	G
	3	2	4	3	2	5	1

We will now prepare the tasks for EDF*

Task 4

EDF* - Precedence graph example

Given the precedences $A \rightarrow C$, $B \rightarrow C$, $C \rightarrow E$, $D \rightarrow F$, $B \rightarrow D$, $C \rightarrow F$, $D \rightarrow G$ we first draw the precedence graph:

Task 4

EDF* - Precedence graph example

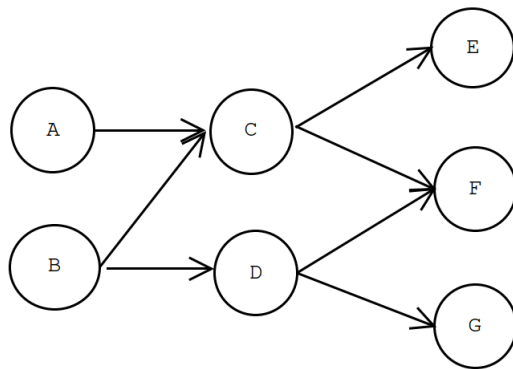


Figure 1: Task 4: precedence graph

Task 4 I

EDF* - Transformation example

- ▶ $r_A^* = r_A, r_B^* = r_B$
- ▶ $r_C^* = \max\{r_C, \max\{r_A^* + C_A, r_B^* + C_B\}\} = \max\{0, \max\{3, 2\}\} = 3$
- ▶ $r_D^* = \max\{r_D, r_B^* + C_B\} = \max\{0, 2\} = 2$
- ▶ $r_F^* = \max\{r_F, \max\{r_C^* + C_C, r_D^* + C_D\}\} = \max\{0, \max\{7, 5\}\} = 7$
- ▶ $r_E^* = \max\{r_E, r_C^* + C_C\} = \max\{0, 7\} = 7$
- ▶ $r_G^* = \max\{r_G, r_D^* + C_D\} = \max\{0, 5\} = 5$

Task 4 II

EDF* - Transformation example

- ▶ $d_E^* = d_F^* = d_G^* = 20$
- ▶ $d_C^* = \min\{d_C, \min\{d_E^* - C_E, d_F^* - C_F\}\} = \min\{20, \min\{18, 15\}\} = 15$
- ▶ $d_D^* = 15$
- ▶ $d_A^* = 11$
- ▶ $d_B^* = 11$

We now successfully have transformed the problem into one without precedence and can simply use EDF!

Task 4

EDF* - Schedule

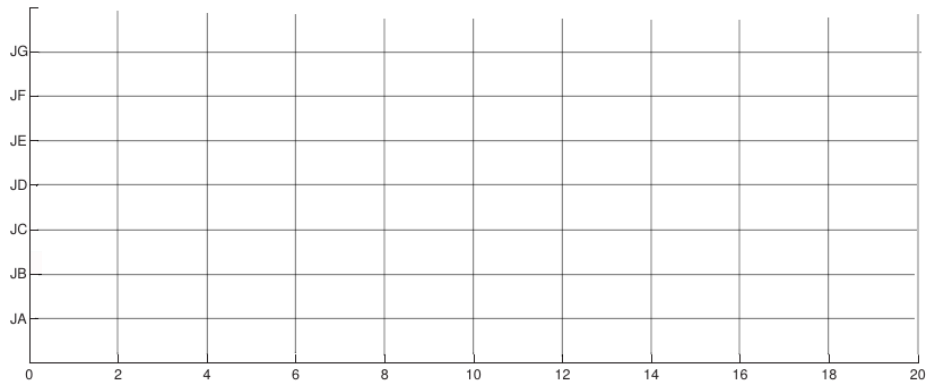


Figure 7: EDF* schedule

Task 4

EDF* - Schedule

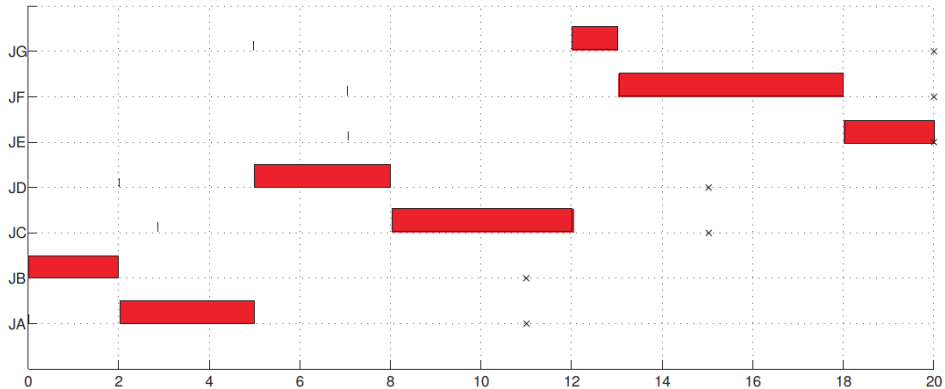


Figure 7: EDF* schedule