TD Real-Time Systems (version 7.1)

1 RM, EDF and LLF

Let be the synchronous periodic tasks (ready at t = 0) with implicit deadlines (D = T).

T1: (C = 1, T = 3)
T2: (C = 1, T = 4)
T3: (C = 2, T = 6)

For each of the RM, EDF and LLF scheduling policies:

• Calculate U, the processor utilization factor, what can we conclude from it?
• Give the chronogram of the tasks. What comment can we make?

Reminder : $n(2^{(1/n)}-1) = 0{,}78$ pour n = 3.

2 Response Time

Let be the synchronous periodic tasks (ready at t = 0) with implicit deadlines (D = T).

T1: (C = 25, T = 100)
T2: (C = 50, T = 200)
T3: (C = 100, T = 300)

Calculate their response time to determine if they are schedulable with RMS.

Reminder : $R_i(t) = \Sigma_{j \text{ in } hp(i)} C_j * \lceil t/T_j \rceil$ with hp (i) containing the tasks with higher priority than task i

3 Aperiodic Task Servers

3.1 T1 being a periodic task

Let be the synchronous periodic tasks (ready at t = 0) with implicit deadlines (D = T).

T1: (C = 2, T = 10)
T2: (C = 4, T = 14)
T3: (C = 5, T = 14)

Demonstrate without doing the complete chronogram that the system is schedulable.

3.2 T1 being a deferred server

We now assumed that T1 is a deferred server. We introduce in the previous system A, an aperiodic task characterized by an release time 28, a budget 6

Give the chronogram illustrating the scheduling of T1, T2, T3 and A from t = 25. Explain the issue et the reasons.

3.3 T1 being a polling server

It is now assumed that T1 is a polling server. We introduce the same aperiodic task. Give the chronogram and the response time of A.

3.4 T1 serveur sporadique

It is now assumed that T1 is a sporadic server. We introduce the same aperiodic task. Give the chronogram and the response time of A.

4 Integrated Modular Avionic

Let be the synchronous periodic tasks (ready at t = 0) with implicit deadlines (D = T). They run on a partitioned ARINC 653 mono-processor platform.

Name	Budget	Period / Deadline
T1	1	3
T2	1	6
T3	2	6
T4	2	12

We want to apply preemptive Rate Monotonic scheduling. In case of equality of priority, the lexicographical order is used.

4.1 Question 1

Explain why the schedulability test cannot conclude on the schedulability of this set of tasks. Calculatee the chronogram over the hyper-period.

4.2 Question 2

Tasks T1 and T3 have the same level of criticality A and T2 and T4 have the same level of criticality B. We have to group the tasks by level of criticality to form partitions. We keep the previous scheduling as it is. Define the Partition Windows to enforce the criticality requirements and calculatee the number of partition switches.

4.3 Question 3

We want to reduce this number of switches. Suggest another configuration of partition windows so as to reduce the number of partition switches.

5 EDF and LLF in multicores systems (m cores)

Let be the synchronous periodic tasks (ready at t = 0) with implicit deadlines (D = T).

T_1 : (C=2*epsilon, T)

…

T_m : (C=2*epsilon, T)

T_{m+1} : (C=T, T+epsilon)

5.1 Utilisation factor

Calculate the utilisation factor of the system when epsilon is small and T is large.

5.2 Global EDF

Apply Global EDF to this set of tasks. What do we see?

5.3 Global LLF

Apply Global LLF to this set of tasks. What do we see?

5.4 Conclusions

What can we conclude from these examples?

6 Multicore partitioned scheduling (m cores)

Find a scheduling of **identical** synchronous periodic tasks (ready at t = 0) with implicit deadlines (D = T) so that they are not schedulable in partitioned but schedulable in global

7 Global and partitioned schedulings on 2 cores

Let be the synchronous periodic tasks (ready at t = 0) with implicit deadlines (D = T).

Name	Budget	Period/Deadline
T1	4	6
T2	7	12
T3	4	12
T4	10	24

7.1 Pfair

Is the task system schedulable using PFair ? Produce the chronogram (split into subtasks).

7.2 Global-RM

Is the task system schedulable using Global-RM ? Produce the chronogram.

7.3 Partitioned-RM

Is the task system schedulable using a partitioned architecture, each partition scheduling its tasks with RMS ? Produce the chronogram.

8 MC/DC code coverage

We have the following code

F3(x, y, z) : if (((x==1) && (y==2)) ll (z==3)) {F1(x, y, z) ;} ; F2(x, y, z) ;

8.1 2 tests

According to MC / DC, are the F3 (1, 2, 3) and F (2, 2, 3) tests redundant or complementary? Justify your anwser.

8.2 Logical table

Build the logical table.

9 CAN Bus

Recall the arbitration principles of the CAN bus. You detail the result of the simultaneous transmission of messages on the bus by tasks with identifiers 4, 5, 6 and 7.

10 Scheduling with blocking time

We want to execute on a mono-processor a set of tasks sharing 2 locks S1 and S2.

Name	C	T	Execution code
T1	2	8	L(S1) ; Exec(1) ; U(S1) ; Exec(1) ;
T2	3	10	L(S1) ; Exec(1) ; U(S1) ; Exec(1) ; L(S2) ; Exec(1) ; U(S2)
T3	3	20	Exec(1) ; L(S2) ; Exec(1) ; U(S2) ; Exec(1) ;
T4	7	40	L(S1) ; Exec(3) ; U(S1) ; Exec(1) ; L(S2) ; Exec(3) ; U(S2)

Let be the synchronous periodic tasks (ready at t = 0) with implicit deadlines (D = T). They run on a partitioned ARINC 653 mono-processor platform. We want to apply a pre-emptive

Earliest Deadline First scheduling. We are studying the scheduling in the case of PIP-EDF and SRP-EDF synchronization protocols. Determine the blocking times. Apply system scheduling tests in the presence of blockages for EDF.

SOLUTION

1 RM, EDF and LLF

U=1/3+1/4+2/6=11/12

The system can be scheduled by LLF and EDF because it checks the necessary and sufficient condition $U \leq 1$.

Nothing can be said for RMS because the sufficient condition is not verified because $U \geq n(2^{(1/n)}-1) = 0,78$ pour n = 3. On the other hand, the chronogram over hyper-period 12 shows that the system is schedulable with RMS.

2 Response Time

We calculate by iteration the response time given by $R_i(n+1)\ \Sigma_{j\leq i} C_j * \lceil R_i(n)/T_j \rceil$

for all tasks j of higher priority than task i. Knowing that $R_i(0) = C_i$. The final response time must be less than the deadline.

$R_1(0) = 25$, $R_1(1) = R_1(0) = 25$ or $R_1(1) < D_1$: T1 respects its deadline with RMS.
$R_2(0) = 50$, $R_2(1) = 25*\lceil 50/100 \rceil + 50=75$, $R_2(2) = R_2(1) = 75$, $R_2(1) < D_2$: T2 respects its deadline.
$R_3(0) = 100$, $R_3(1) = 25*\lceil 100/100 \rceil + 50*\lceil 100/200 \rceil + 100 = 175$, $R_3(1) = 25*\lceil 175/100 \rceil + 50*\lceil 175/200 \rceil + 100 = 200$, $R_3(1) = 25*\lceil 200/100 \rceil + 50*\lceil 200/200 \rceil + 100 = 200$ $R_3(3) = R_3(2) = 200$, $R_3(3) < D_3$: T3 respects its deadline.

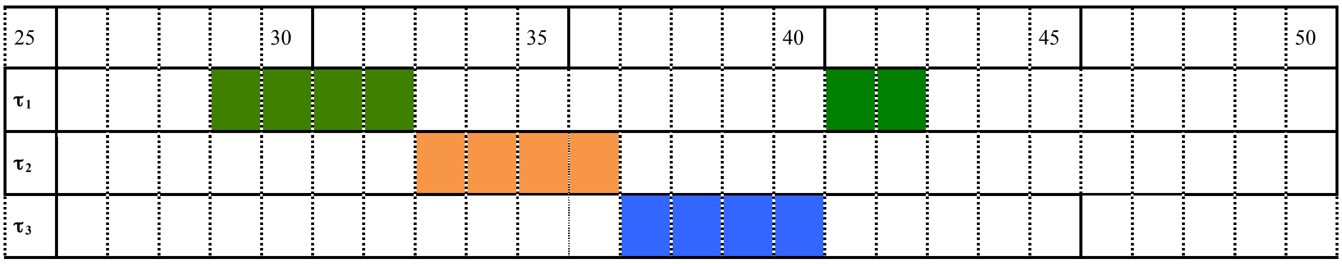
3 Aperiodic Task Servers

3.1 T1 being a periodic task

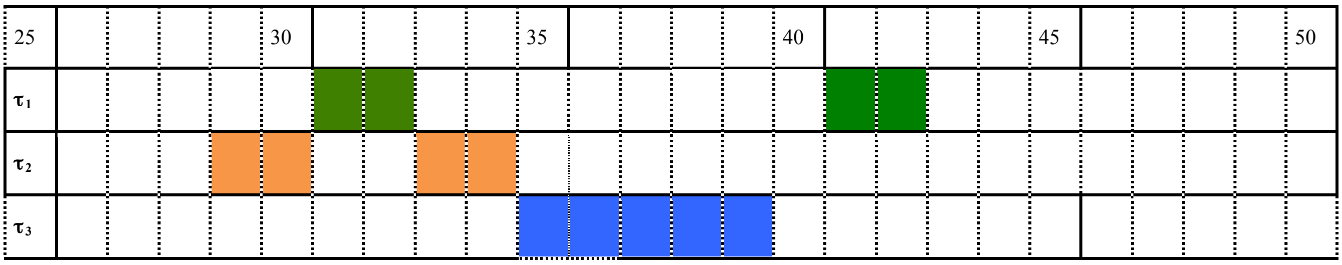
We calculate by iteration the response times to show that the system is schedulable. Note that we can regroup T2 and T3 since they have the same period.

3.2 T1 being a deferred server

Task T3 misses its deadline at T = 42.



3.3 T1 being a polling server



3.4 T1 being sporadic server

25					30					35					40					45					50
τ_1				🟩	🟩									🟩	🟩									🟩	🟩
τ_2						🟧	🟧	🟧	🟧																
τ_3										🟦	🟦	🟦	🟦			🟦									

4 Integrated Modular Avionic

4.1 Question 1

U = 1 therefore with RMS we cannot conclude.

1	2	3	4	5	6	7	8	9	10	11	12
T1	**T2**	**T3**	**T1**	**T3**	**T4**	**T1**	**T2**	**T3**	**T1**	**T3**	**T4**

4.2 Question 2

There are 8 partition switches (including the one occurring a the hyper period)

P1	**P2**	**P1**	**P1**	**P1**	**P2**	**P1**	**P2**	**P1**	**P1**	**P1**	**P2**
T1	**T2**	**T3**	**T1**	**T3**	**T4**	**T1**	**T2**	**T3**	**T1**	**T3**	**T4**

4.3 Question 3

We group tasks T1 and T3 then T2 and T4. There are 3 partition switches.

P1	**P1**	**P1**	**P1**	**P2**	**P2**	**P2**	**P2**	**P1**	**P1**	**P1**	**P1**
T1	**T3**	**T3**	**T1**	**T2**	**T4**	**T4**	**T2**	**T1**	**T1**	**T3**	**T3**

5 EDF and LLF in multi-core systems (m cores)

5.1 Utilisation factor

U = (2m/P)*epsilon + P/P+epsilon so U is approximately 1. Much less than m.

5.2 Global-EDF

With Global-EDF, T1 to Tm start on C1 to Cm cores. At t = 2 * epsilon, T1 to Tm end and Tm + 1 can run, ends at 2 * epsilon + P and miss its deadline.

5.3 Global-LLF

With Global-LLF, Tm + 1 starts on C1 and T1 to Tm-1 starts on the other cores then when T1 ends on C2, Tm starts on C2 and all the tasks respect their deadline.

5.4 Conclusions

On the one hand, even if U = 1 Global-EDF can fail in multi-core systems (U=m) ! On the other hand, Global-LLF seems to dominate Global-EDF (proved in the littérature).

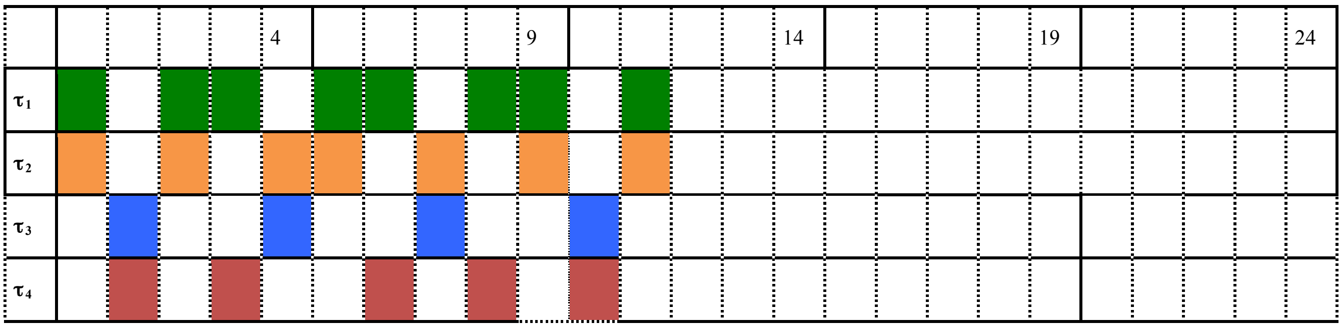
6 Multicore partitioned scheduling (m cores)

The idea is to ensure that the use of two (identical) tasks does not fit on one core (U> 1) and that the sum of the utilization factor fits on m cores. To do this, we choose m + 1 tasks of C = (T + epsilon) / 2. 2U = (T + epsilon) / T> 1. But (m + 1) (T + epsilon) / 2T <= m
if (T + epsilon) <= m (T-epsilon), so as soon as 2 <= m.

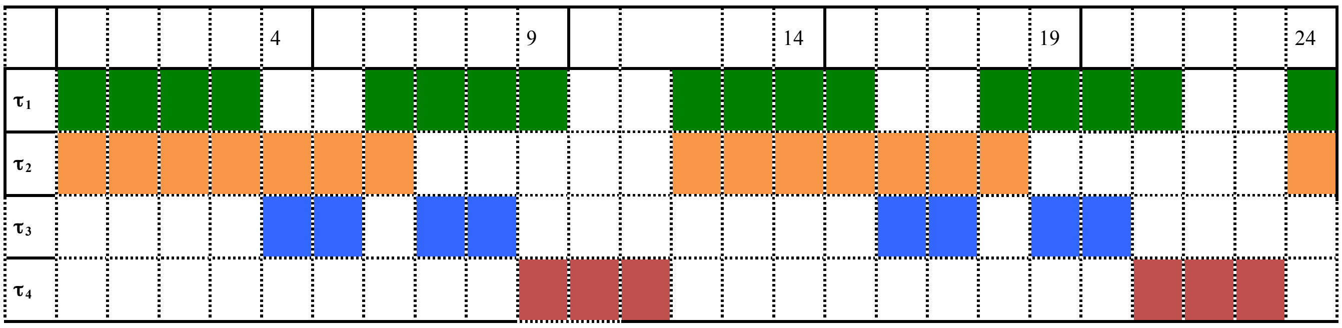
7 Global and partitioned schedulings on 2 cores

7.1 PFair

For each task i, we create Ci sub-tasks (a unit j for each budget unit) such that the activation (resp the deadline) is ⌊ (j-1) / Ui⌋ (resp ⌈j / Ui⌉) .
For T1, intervals are (0 ; 2), (1 ; 3), (3 ; 5), (4 ; 6), (6 ; 8), (7 ; 9), (9 ; 11), (10 ; 12) (U = 4/6)
For T2, intervals are (0 ; 2), (1 ; 4), (3 ; 6), (5 ; 7), (6 ; 9) (8.11), (10.12) (U = 7 / 12)
For T3, intervals are (0 ; 3), (3 ; 6), (6 ; 9), (9 ; 12) (U = 4/12)
For T4, intervals are (0 ; 3), (2 ; 5), (4 ; 8), (7 ; 10), (9 ; 12), (12 ; 15), (14 ; 17), (16 ; 20), (19 ; 22) (21 ; 24). (U = 10/24)



7.2 Global-RM



Deadline miss at t=12.

7.3 Partitioned-RM

The global utilisation factor being 2 for 2 cores available, each core must support the execution of tasks with a utilisation factor of 1. The only solution is to put T2 (7/12) with T4 (10/24) and therefore T1 (4/6) with T3 (4/12). We quickly calculate that the response time of T3 is 12 less than its deadline and that of T4 is 24.

8 MC/DC code coverage

8.1 Two tests

They are redundant because like z == 3 one varies x and y without modifying the decision. The final test is still true.

8.2 Logical table

x==1	y==2	z==3	decision
F	F	V	V
F	F	F	F
F	V	F	F
V	V	F	V

9 CAN Bus

The bus starts transmitting the identifier over the bus, bit by bit. Bit 1 is recessive (the emitter loses in a conflict that would involve at least one node sending a bit 0). In the example, we will detail, bit by bit, the decisions that the nodes take to determine whether they are emitters or receivers (4 = 1000, 5 = 1001, 6 = 1010, 7 = 1011). Ultimately, nodes with weaker identifiers have the highest priority.

10 Scheduling with blocking time

See example in slides.