Second Labs on Real-Time Scheduling

Hamza Mouddene

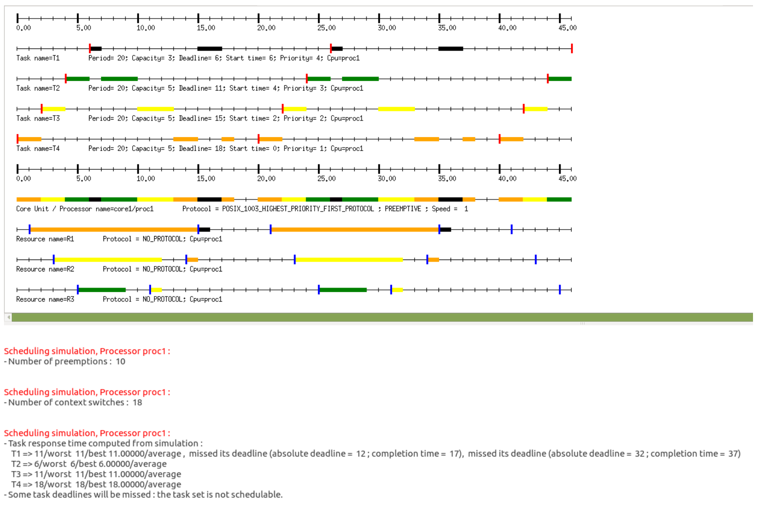
December 3, 2021

Exercise 1

Let's assume the following task sharing resources R_1, R_2 and R_3:

	First release	WCET	D	P	Priority
T_1	6	$3 : \boxed{R_1}\boxed{}$	6	20	4
T_2	4	$5 : \boxed{R_3}\boxed{R_3}\boxed{R_3}\boxed{}$	11	20	3
T_3	2	$5 : \boxed{R_2}\boxed{R_2}\boxed{R_2 R_3}\boxed{}$	15	20	2
T_4	0	$5 : \boxed{R_1}\boxed{R_1}\boxed{R_1 R_2}\boxed{}$	18	20	1

1. The simulation of this configuration without a specific protocol for resource allocation unveils that the task set is not schedulable, because T_1 missed its deadline twice.

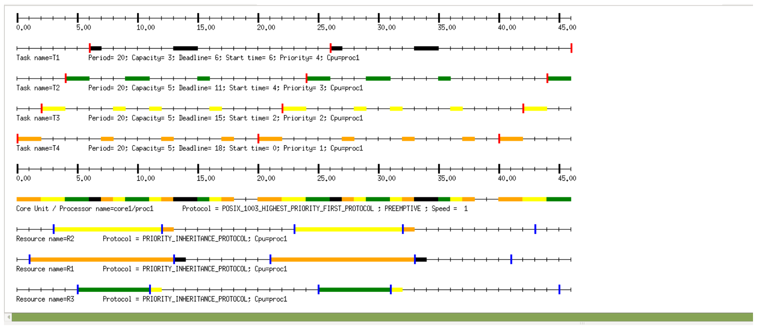


```
Task name=T1        Period= 20; Capacity= 3; Deadline= 6; Start time= 6; Priority= 4; Cpu=proc1
Task name=T2        Period= 20; Capacity= 5; Deadline= 11; Start time= 4; Priority= 3; Cpu=proc1
Task name=T3        Period= 20; Capacity= 5; Deadline= 15; Start time= 2; Priority= 2; Cpu=proc1
Task name=T4        Period= 20; Capacity= 5; Deadline= 18; Start time= 0; Priority= 1; Cpu=proc1
Core Unit / Processor name=core1/proc1        Protocol = POSIX_1003_HIGHEST_PRIORITY_FIRST_PROTOCOL ; PREEMPTIVE ; Speed =  1
Resource name=R1        Protocol = NO_PROTOCOL; Cpu=proc1
Resource name=R2        Protocol = NO_PROTOCOL; Cpu=proc1
Resource name=R3        Protocol = NO_PROTOCOL; Cpu=proc1
```

```
Scheduling simulation, Processor proc1 :
- Number of preemptions :  10

Scheduling simulation, Processor proc1 :
- Number of context switches :  18

Scheduling simulation, Processor proc1 :
- Task response time computed from simulation :
  T1 => 11/worst  11/best 11.00000/average ,  missed its deadline (absolute deadline =  12 ; completion time =  17),  missed its deadline (absolute deadline =  32 ; completion time =  37)
  T2 => 6/worst  6/best 6.00000/average
  T3 => 11/worst  11/best 11.00000/average
  T4 => 18/worst  18/best 18.00000/average
- Some task deadlines will be missed : the task set is not schedulable.
```

Figure 1: Simulation of the task configuration - without a specific protocol

2. The simulation of this configuration with the Priority Inheritance protocol for resource allocation unveils that the task set is not schedulable, because T_1 and T_2 missed its deadlines twice.

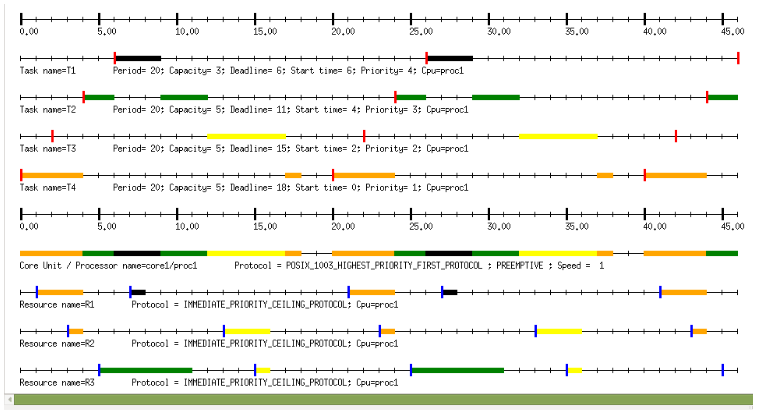


```
Task name=T1        Period= 20; Capacity= 3; Deadline= 6; Start time= 6; Priority= 4; Cpu=proc1
Task name=T2        Period= 20; Capacity= 5; Deadline= 11; Start time= 4; Priority= 3; Cpu=proc1
Task name=T3        Period= 20; Capacity= 5; Deadline= 15; Start time= 2; Priority= 2; Cpu=proc1
Task name=T4        Period= 20; Capacity= 5; Deadline= 18; Start time= 0; Priority= 1; Cpu=proc1
Core Unit / Processor name=core1/proc1        Protocol = POSIX_1003_HIGHEST_PRIORITY_FIRST_PROTOCOL ; PREEMPTIVE ; Speed =  1
Resource name=R2        Protocol = PRIORITY_INHERITANCE_PROTOCOL; Cpu=proc1
Resource name=R1        Protocol = PRIORITY_INHERITANCE_PROTOCOL; Cpu=proc1
Resource name=R3        Protocol = PRIORITY_INHERITANCE_PROTOCOL; Cpu=proc1
```

```
Scheduling simulation, Processor proc1 :
- Number of preemptions :  18

Scheduling simulation, Processor proc1 :
- Number of context switches :  26

Scheduling simulation, Processor proc1 :
- Task response time computed from simulation :
   T1 => 9/worst  9/best 9.00000/average ,  missed its deadline (absolute deadline =  12 ; completion time =  15),  missed its deadline (absolute deadline =  32 ; completion time =  35)
   T2 => 12/worst  12/best 12.00000/average ,  missed its deadline (absolute deadline =  15 ; completion time =  16),  missed its deadline (absolute deadline =  35 ; completion time =  36)
   T3 => 15/worst  15/best 15.00000/average
   T4 => 18/worst  18/best 18.00000/average
- Some task deadlines will be missed : the task set is not schedulable.
```

Figure 2: Simulation of the task configuration - Priority Inheritance protocol

3. The simulation of this configuration with the Stack-based Protocol (Immediate Ceiling Inheritance Protocol) for resource allocation unveils that the task set is schedulable if you computed the scheduling on the feasibility interval.



```
Task name=T1        Period= 20; Capacity= 3; Deadline= 6; Start time= 6; Priority= 4; Cpu=proc1
Task name=T2        Period= 20; Capacity= 5; Deadline= 11; Start time= 4; Priority= 3; Cpu=proc1
Task name=T3        Period= 20; Capacity= 5; Deadline= 15; Start time= 2; Priority= 2; Cpu=proc1
Task name=T4        Period= 20; Capacity= 5; Deadline= 18; Start time= 0; Priority= 1; Cpu=proc1
Core Unit / Processor name=core1/proc1        Protocol = POSIX_1003_HIGHEST_PRIORITY_FIRST_PROTOCOL ; PREEMPTIVE ; Speed =  1
Resource name=R1        Protocol = IMMEDIATE_PRIORITY_CEILING_PROTOCOL; Cpu=proc1
Resource name=R2        Protocol = IMMEDIATE_PRIORITY_CEILING_PROTOCOL; Cpu=proc1
Resource name=R3        Protocol = IMMEDIATE_PRIORITY_CEILING_PROTOCOL; Cpu=proc1
```

```
Scheduling simulation, Processor proc1 :
- Number of preemptions :  4

Scheduling simulation, Processor proc1 :
- Number of context switches :  11

Scheduling simulation, Processor proc1 :
- Task response time computed from simulation :
   T1 => 3/worst  3/best 3.00000/average
   T2 => 8/worst  8/best 8.00000/average
   T3 => 15/worst  15/best 15.00000/average
   T4 => 18/worst  18/best 18.00000/average
- No deadline missed in the computed scheduling : the task set is schedulable if you computed the scheduling on the feasibility interval.
```

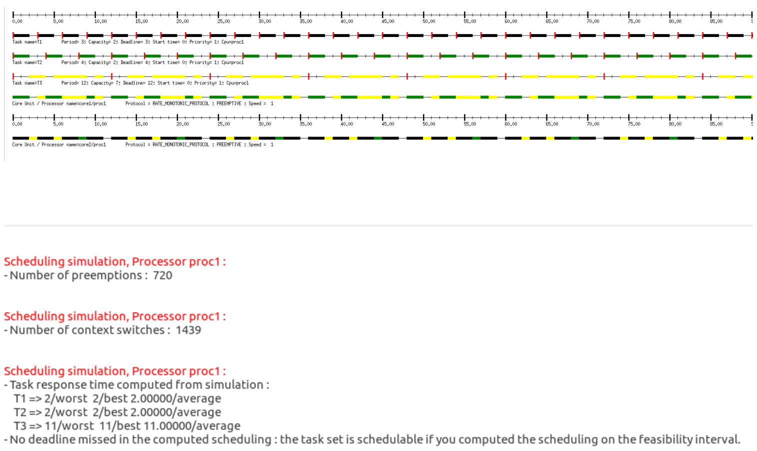
Figure 3: Simulation of the task configuration - Stack-based Protocol

Exercise 3

1. Lets assume that we have the following task configuration on one processor with two cores using a fully global Rate Monotonic scheduler: The simulation of this configuration with a fully global Rate

	First release	WCET	D	P
T_1	0	2	3	3
T_2	0	2	4	4
T_3	0	7	12	12

   Monotonic scheduler unveils that the task set is schedulable if you computed the scheduling on the feasibility interval.


0,00
5,00
10,00
15,00
20,00
25,00
30,00
35,00
40,00
45,00
50,00
55,00
60,00
65,00
70,00
75,00
80,00
85,00
Task name=T1 Period= 3; Capacity= 2; Deadline= 3; Start time= 0; Priority= 1; Cpu=proc1
Task name=T2 Period= 4; Capacity= 2; Deadline= 4; Start time= 0; Priority= 1; Cpu=proc1
Task name=T3 Period= 12; Capacity= 7; Deadline= 12; Start time= 0; Priority= 1; Cpu=proc1
Core Unit / Processor name=core1/proc1 Protocol = RATE_MONOTONIC_PROTOCOL ; PREEMPTIVE ; Speed = 1
Core Unit / Processor name=core2/proc1 Protocol = RATE_MONOTONIC_PROTOCOL ; PREEMPTIVE ; Speed = 1

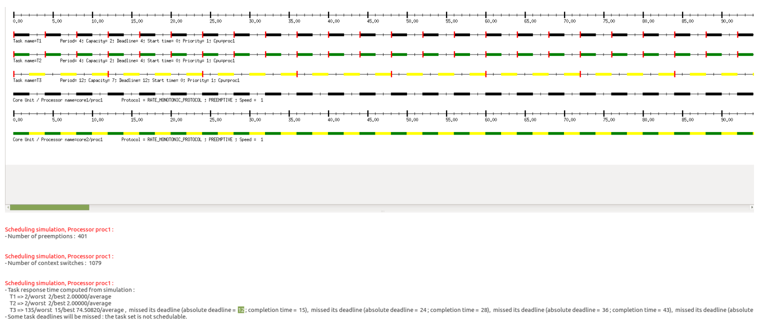
```
Scheduling simulation, Processor proc1 :
- Number of preemptions : 720

Scheduling simulation, Processor proc1 :
- Number of context switches : 1439

Scheduling simulation, Processor proc1 :
- Task response time computed from simulation :
   T1 => 2/worst  2/best 2.00000/average
   T2 => 2/worst  2/best 2.00000/average
   T3 => 11/worst  11/best 11.00000/average
- No deadline missed in the computed scheduling : the task set is schedulable if you computed the scheduling on the feasibility interval.
```

Figure 4: Simulation 1 of the task configuration - fully global Rate Monotonic scheduler

2. Now, we will change only the deadline and the period of task T_1, then we will retry the same experiment, so the configurations becomes like the following: The simulation of this configuration with the Priority

	First release	WCET	D	P
T_1	0	2	4	4
T_2	0	2	4	4
T_3	0	7	12	12

   Inheritance protocol for resource allocation unveils that the task set is not schedulable, because T_3 missed its deadline.



```
Task name=T1        Period= 4; Capacity= 2; Deadline= 4; Start time= 0; Priority= 1; Cpu=proc1
Task name=T2        Period= 4; Capacity= 2; Deadline= 4; Start time= 0; Priority= 1; Cpu=proc1
Task name=T3        Period= 12; Capacity= 7; Deadline= 12; Start time= 0; Priority= 1; Cpu=proc1
Core Unit / Processor name=core1/proc1       Protocol = RATE_MONOTONIC_PROTOCOL ; PREEMPTIVE ; Speed =  1
Core Unit / Processor name=core2/proc1       Protocol = RATE_MONOTONIC_PROTOCOL ; PREEMPTIVE ; Speed =  1

Scheduling simulation, Processor proc1 :
- Number of preemptions :  401

Scheduling simulation, Processor proc1 :
- Number of context switches :  1079

Scheduling simulation, Processor proc1 :
- Task response time computed from simulation :
   T1 => 2/worst  2/best 2.00000/average
   T2 => 2/worst  2/best 2.00000/average
   T3 => 135/worst  15/best 74.50820/average ,  missed its deadline (absolute deadline = 12 ; completion time =  15),  missed its deadline (absolute deadline =  24 ; completion time =  28),  missed its deadline (absolute deadline =  36 ; completion time =  43),  missed its deadline (absolute
- Some task deadlines will be missed : the task set is not schedulable.
```

Figure 5: Simulation 2 of the task configuration - fully global Rate Monotonic scheduler

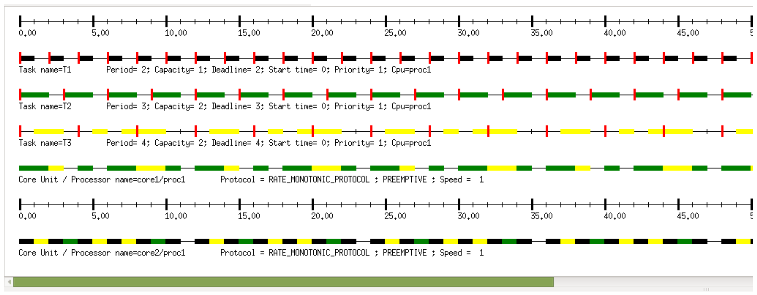
3. We conclude that if 2 tasks have the same periods and are synchronized, the third one won't fit with 2 cores, despite the fact that the payload of T_1 decrease comparing with the simulation 1. Tasks must be different to create an alternation and share T_3 responsibility

Exercise 4

Let's assume the following task sharing resources R_1, R_2, R_3 and R_4: The worst response time (4) is not

	First release	WCET	D	P	Priority
T_1	6	4 : $\boxed{}\boxed{R_4}\boxed{R_4R_3}\boxed{}$	6	20	4
T_2	4	4 : $\boxed{}\boxed{R_3}\boxed{R_3R_4}\boxed{}$	9	20	3
T_3	2	4 : $\boxed{}\boxed{R_2}\boxed{R_2R_1}\boxed{}$	13	20	2
T_4	0	4 : $\boxed{}\boxed{R_1}\boxed{R_1R_2}\boxed{}$	16	20	1

meet at time 0 (3). This means that the worst scenario is not when every task wakes up at 0. This anomaly can be explained the same way as the previous exercise. The synchronisation of free time (e.g $time = 5$) generates a waste of calculus time and therefore worsen the response time. In other words, if 2 tasks finish at the same time, you can not cut the T_3 in two pieces to fill both core free time.



```
Task name=T1        Period= 2; Capacity= 1; Deadline= 2; Start time= 0; Priority= 1; Cpu=proc1
Task name=T2        Period= 3; Capacity= 2; Deadline= 3; Start time= 0; Priority= 1; Cpu=proc1
Task name=T3        Period= 4; Capacity= 2; Deadline= 4; Start time= 0; Priority= 1; Cpu=proc1
Core Unit / Processor name=core1/proc1        Protocol = RATE_MONOTONIC_PROTOCOL ; PREEMPTIVE ; Speed =  1
Core Unit / Processor name=core2/proc1        Protocol = RATE_MONOTONIC_PROTOCOL ; PREEMPTIVE ; Speed =  1
```

```
Scheduling simulation, Processor proc1 :
- Number of preemptions :  54

Scheduling simulation, Processor proc1 :
- Number of context switches :  323

Scheduling simulation, Processor proc1 :
- Task response time computed from simulation :
  T1 => 1/worst  1/best 1.00000/average
  T2 => 2/worst  2/best 2.00000/average
  T3 => 4/worst  2/best 3.00000/average
- No deadline missed in the computed scheduling : the task set is schedulable if you computed the scheduling on the feasibility interval.

Scheduling simulation, Processor proc1 :
- Blocking time from simulation :
  T1 => 0/worst  0/best 0.00000/average
  T2 => 0/worst  0/best 0.00000/average
  T3 => 0/worst  0/best 0.00000/average
```

Figure 6: Simulation of the task configuration

Exercise 5

Let's assume the following configuration of tasks. The simulation of this configuration for the both scheduler

	WCET	D	P
T_1	2	6	6
T_2	4	8	8
T_3	3	10	10
T_4	12	20	20
T_5	1	50	50
T_6	20	50	50
T_7	5	100	100
T_8	1	100	100

unveils that the task set is schedulable if you computed the scheduling on the feasibility interval.