徽标

中度可信度描述已自动生成**First Labs on Real-Time Scheduling**

**Guohao DAI**

November 4th, 2022

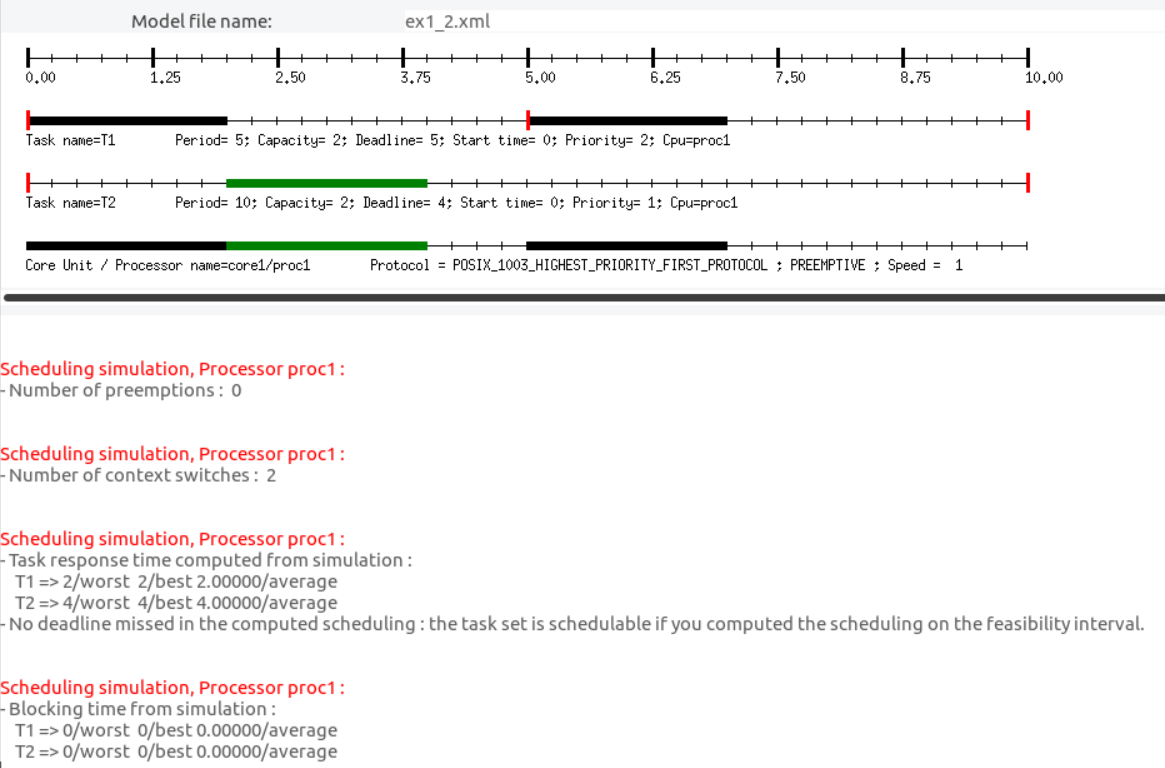
**Exercise 1**

1. 表格

   描述已自动生成Simulate the following task configuration on one core using *deadline monotonic*.
   1. No deadline missed in the computed scheduling: the task set is schedulable if we computed the scheduling on the feasibility interval.
   2. 表格

      描述已自动生成The worst-case response times of the is 4 and is 2.
2. When we set as the first priority and as the second priority, the results of the simulated scheduling are as follows:

* “No deadline missed in the computed scheduling: the task set is schedulable if we computed the scheduling on the feasibility interval.”



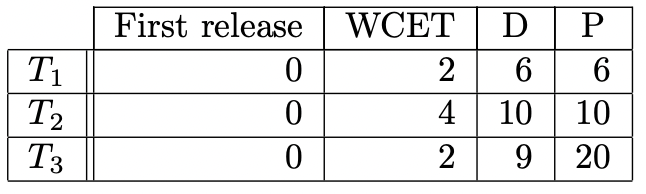
**Exercise 2**

1. 表格

   描述已自动生成Let’s assume the following independant task configuration.

* 图形用户界面, 应用程序, Word

  描述已自动生成It is schedulable by a fixed priority.

1. Same question with the following independant task configuration.

* 图形用户界面, 应用程序, Word

  描述已自动生成It is not schedulable by a fixed priority. However, it is schedulable by a Earliest Deadline First: “No deadline missed in the computed scheduling: the task set is schedulable if we computed the scheduling on the feasibility interval.”

1. 表格

   描述已自动生成Same question with the following independant task configuration.

* It is not schedulable by a fixed priority. However, it is schedulable by a Earliest Deadline First: “No deadline missed in the computed scheduling: the task set is schedulable if we computed the scheduling on the feasibility interval.”

图形用户界面, 应用程序, Word

描述已自动生成

**Exercise 3**

In this exercise we investigate another dynamic scheduling policy: LLF (Least Laxity First). This policy selects the task to run among the ready tasks according to a dynamic priority called ‘laxity’: the smaller the laxity, the higher the priority. , the laxity of a task at time t can be computed by

where is the remaining capacity of the task at time .

表格

描述已自动生成Let’s assume the following independant task configuration.

* 图形用户界面, 应用程序, Word

  描述已自动生成Result of scheduling simulation in ***Earliest Deadline First***
* Result of scheduling simulation in ***Least Laxity First***

图形用户界面, 应用程序, Word

描述已自动生成

* Through the two scheduling simulation results, the ***Earliest Deadline First*** algorithm has fewer context switches and preemptions under this task configuration. So it will consume less extra resources and have better performance.

**Exercise 4**

In this exercise we investigate the effect of non preemption on scheduling.

1. 表格

   描述已自动生成Let’s assume the following independant task configuration.

* Result of scheduling simulation in ***Preemptive Rate Monotonic***

图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成

* Result of scheduling simulation in ***Non Preemptive Rate Monotonic***

图形用户界面, 文本, 应用程序

描述已自动生成

* In preemptive scheduling, once a task needs to be scheduled (preempted), the CPU will context switch. Therefore, context switching will be more frequent in this scheduling mode.
* In non-preemptive scheduling, the CPU does not respond to preemption (interrupts). Correspondingly, this mode will have fewer context switches

1. 表格

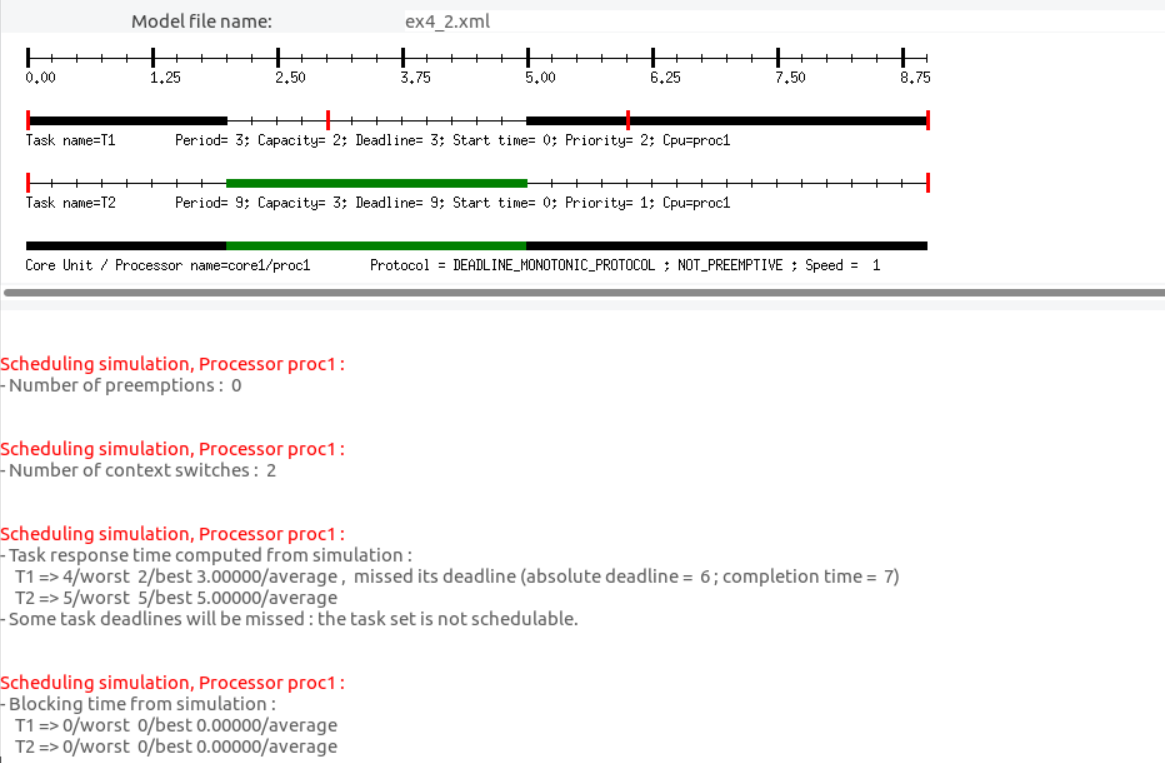
   描述已自动生成Same question with the following independant task configuration.

* Result of scheduling simulation in ***Preemptive Rate Monotonic***

图形用户界面, 文本, 应用程序, 电子邮件

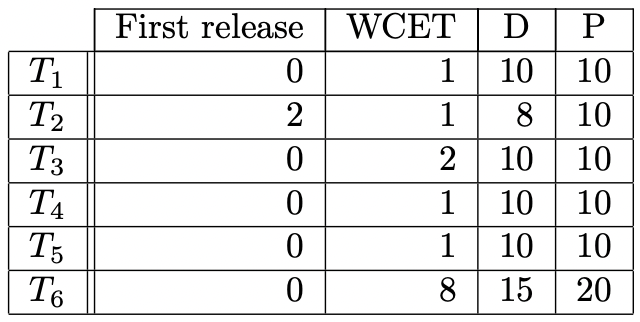
描述已自动生成

* Result of scheduling simulation in ***Non Preemptive Rate Monotonic***



* In this task configuration, the task set is only schedulable in preemptive mode.
* In non-preemptive mode, missed its deadline(absolute deadline = 6; completion time = 7).

Exercise 5

Let’s assume the following dependant task configuration:

We have the following precedence constraints:

* and have to complete execution before starts,
* has to complete execution before and .

1. It is not schedulable using the rate monotonic approach presented in the lectures.

表格

描述已自动生成

1. It is schedulable using the Earliest Deadline First approach presented in the lectures.