Time4sys2imi: A tool to formalize real-time system models under uncertainty

Étienne André[2] **Jawher Jerray**[1] Sahar Mhiri[1]

[1] Université Paris 13, LIPN, CNRS, UMR 7030, F-93430, Villetaneuse, France

[2] Université de Lorraine, Nancy, France

2 novembre 2019

Context : Verifying real-time systems

- Real-time systems :
  - Strong constraints on time. (e. g., a response passed a deadline is invalid even if its content appears to be correct.)
  - Real-time systems are everywhere
- Critical real-time systems :
  - Failures (in correctness or timing) may result in dramatic consequences

$\Rightarrow$ Existing scheduling techniques do not apply well in the presence of uncertainty.

Context : Verifying real-time systems

- Real-time systems :
  - Strong constraints on time. (e. g., a response passed a deadline is invalid even if its content appears to be correct.)
  - Real-time systems are everywhere
- Critical real-time systems :
  - Failures (in correctness or timing) may result in dramatic consequences

$\Rightarrow$ Existing scheduling techniques do not apply well in the presence of uncertainty.

Deepwater Horizon

Amagasaki Railway Accident

Flight 214 crash of Asiana Airlines

Real-time system

A real-time system is made of a set of tasks to execute on a processor

Real-time system

A real-time system is made of a set of tasks to execute on a processor

A task is characterized by :

- B : its best-case execution time
- W : its worst-case execution time
- D : its relative deadline

Real-time system

A real-time system is made of a set of tasks to execute on a processor

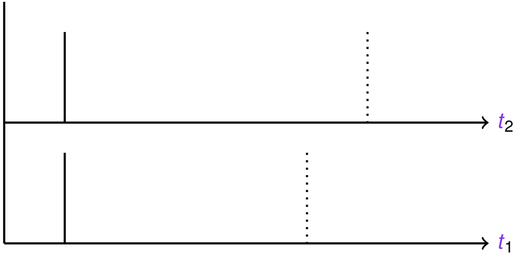
A task is characterized by :

- B : its best-case execution time
- W : its worst-case execution time
- D : its relative deadline

Tasks have instances that are activated (usually periodically)

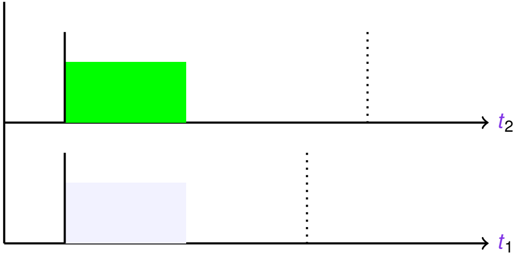
Example : shortest job first (SJF)

Task	B	W	D
t_1	3	3	4
t_2	2	2	5



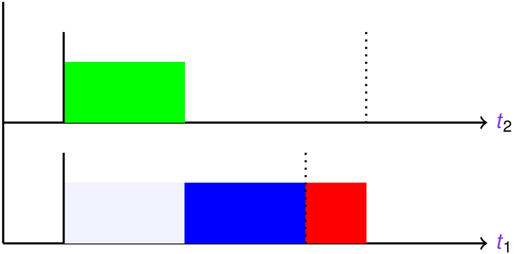
Example : shortest job first (SJF)

Task	B	W	D
t_1	3	3	4
t_2	2	2	5



Example : shortest job first (SJF)

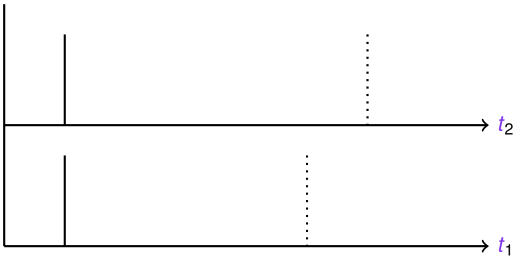
Task	B	W	D
t_1	3	3	4
t_2	2	2	5



Task t_1 misses its deadline

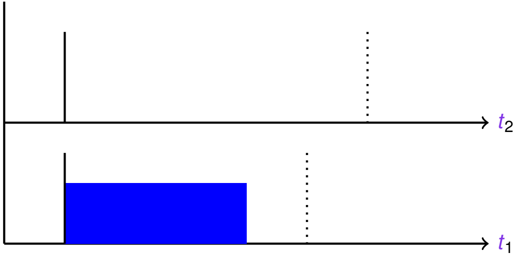
Example : preemptive fixed priority scheduler (FPS)

Task	B	W	D	priority
t_1	3	3	4	high
t_2	2	2	5	low



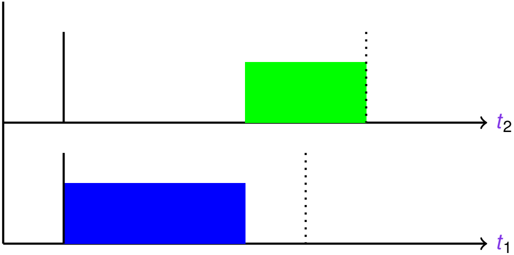
Example : preemptive fixed priority scheduler (FPS)

Task	B	W	D	priority
t_1	3	3	4	high
t_2	2	2	5	low



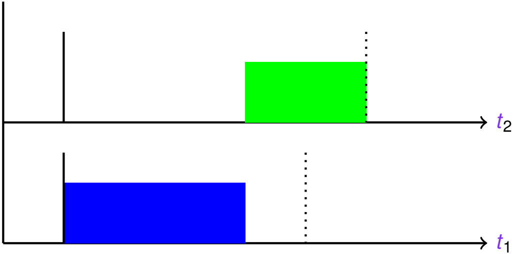
Example : preemptive fixed priority scheduler (FPS)

Task	B	W	D	priority
t_1	3	3	4	high
t_2	2	2	5	low



Example : preemptive fixed priority scheduler (FPS)

Task	B	W	D	priority
t_1	3	3	4	high
t_2	2	2	5	low



The system is schedulable

Scheduling

Scheduling

- Decide which task the processor runs at each moment.

- **Timing constraint :** priority, deadline, reactivity, preemption, . . .
- **Two main contexts :**
  - Centralized system [LL73]
  - Distributed system [TS06]

. [LL73] C. L. Liu et J. W. Layland, “Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment”, **Journal of the ACM**, t. 20, n^{o} 1, p. 46-61, 1973, ISSN : 0004-5411. DOI : 10.1145/321738.321743.

. [TS06] A. S. Tanenbaum et M. v. Steen, **Distributed Systems : Principles and Paradigms (2Nd Edition)**. Upper Saddle River, NJ, USA : Prentice-Hall, Inc., 2006, ISBN : 0132392275.

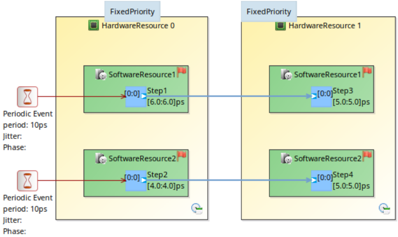
Schedulability analysis

Definition

- A system is schedulable if all tasks meet their deadline for all possible behaviors (according to the periods, interarrival rates, dependencies between tasks. . .).

Time4sys

- Time4sys is a formalism developed by Thales that provides an environment to prepare the design phase of a system through the graphical visualization developed.
- Time4sys Design tool allows users to define the following elements :
  - Hardware Resource is a processor, and it contains a set of tasks ; it is also assigned a scheduling policy.
  - Software Resource is a task, and it features a (relative) deadline.
  - Execution Step is a subtask. It is characterized by a BCET, a WCET, and a priority.
  - Event can be seen as an activation policy for tasks. There are two main types of Events : PeriodicEvent and SporadicEvent.



Problem

Problems

- Time4sys lacks for a formalization : it does not perform any verification nor simulation, nor can it assess the schedulability of the depicted systems.

Problem

Problems

- Time4sys lacks for a formalization : it does not perform any verification nor simulation, nor can it assess the schedulability of the depicted systems.

Objectives

- Verify formally Time4sys.
- Allow uncertainty (deadlines, ...) in the models, thus allowing uncertainty in our formalization.
- Develop a tool which allows to translate Time4sys into parametric timed automata.
- synthesize some timing constants guaranteeing schedulability.

ASTREI : Analysis of Real-Time Systems Modeled by Time4sys In the presence of Uncertainty

Time4sys logo

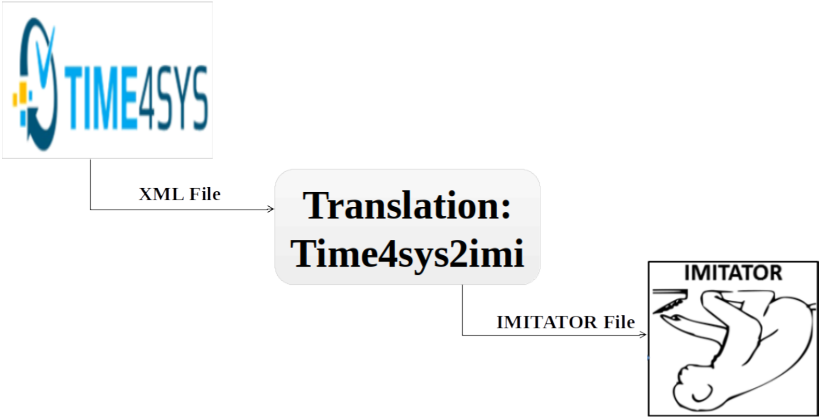
IMITATOR logo

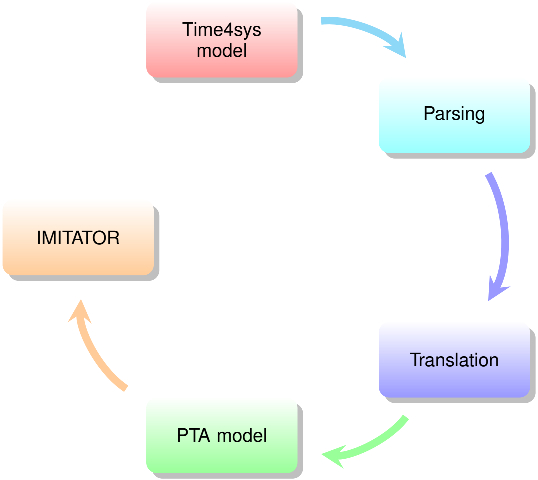
ASTREI : Analysis of Real-Time Systems Modeled by Time4sys In the presence of Uncertainty

Time4sys logo

IMITATOR logo

⇒ Develop an automatic translation of the Time4sys formalism into the IMITATOR input language.


TIME4SYS
XML File
Translation:
Time4sys2imi
IMITATOR File
IMITATOR

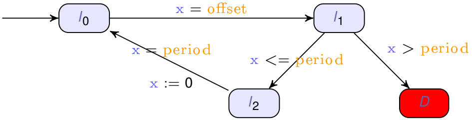


Workflow of Time4sys2imi

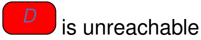
Time4sys2imi

- Time4sys2imi is a tool which allows to translate Time4sys into parametric timed automata (PTAs) described in the input language of IMITATOR.
- The aim of Time4sys2imi is for a given real-time system with some unknown timing constants (period, jitter, deadlines...), analyse the timing constants for which the system is schedulable.

Parametric timed model checking

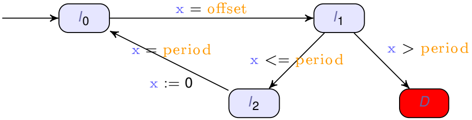


A model of the system



A property to be satisfied

Parametric timed model checking



A model of the system

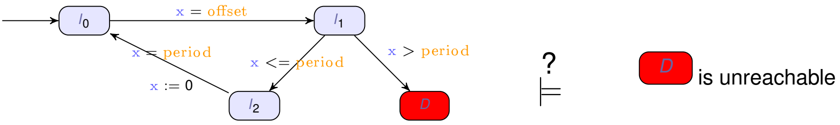
$\overset{?}{\models}$

D is unreachable

A property to be satisfied

- Question : for what values of the parameters does the model of the system satisfy the property ?

Parametric timed model checking



A model of the system

A property to be satisfied

- Question : for what values of the parameters does the model of the system satisfy the property ?

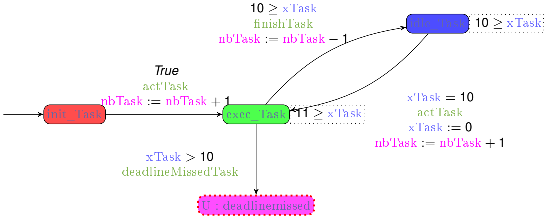
Yes if...

$$\text{offset} < \text{period}$$
$$\wedge \text{period} < 17.54$$

Why parametric timed automata ?

Parametric timed automata [AHV93]

- Formal semantics : automated formal analyzes possible.
- Allow very high **expressivity** : encoding inter-task dependencies, different scheduling policies [FLMS12], sporadic or periodic tasks, etc.
- Can be extended with stopwatches, to model preemption.
- Existence of the model checker IMITATOR.



Example of PTA

. [AHV93] R. Alur, T. A. Henzinger et M. Y. Vardi, "Parametric real-time reasoning", in **STOC**, San Diego, California, United States : ACM, 1993, p. 592-601, ISBN : 0-89791-591-7.

. [FLMS12] L. Fribourg et al., "Robustness Analysis for Scheduling Problems using the Inverse Method", in

IMITATOR

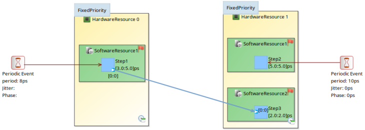
- Translate the network of PTA to the IMITATOR input language [AFKS12].
- IMITATOR is a tool for modeling and verifying real-time systems with unknown constants modeled with parametric timed automata[AHV93]. This parametric model checker takes as input networks of PTA extended with useful features such as synchronization actions and discrete variables.

. [AFKS12] É. André et al., "IMITATOR 2.5 : A Tool for Analyzing Robustness in Scheduling Problems",, t. 7436, Paris, France, 2012. DOI : 10.1007/978-3-642-32759-9_6.

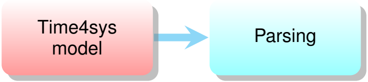
. [AHV93] R. Alur, T. A. Henzinger et M. Y. Vardi, "Parametric real-time reasoning", in **STOC**, San Diego, California, United States : ACM, 1993, p. 592-601, ISBN : 0-89791-591-7.

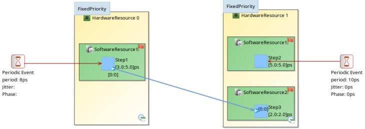
Translation procedures

Time4sys
model



Translation procedures





⇒

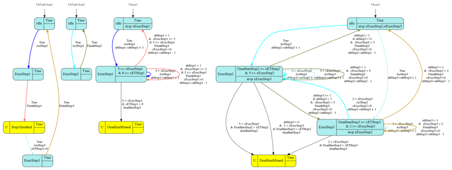

```
– <time4sys:Project xmi:version="2.0" xmi:id="_wfCWYE5UEemWNKmXRKnq_g" name="Test5">
 – <design xmi:id="_wfC9cE5UEemWNKmXRKnq_g">
  – <workloadBehavior xmi:id="_wfC9cU5UEemWNKmXRKnq_g">
   – <behavior xmi:id="_OW-2gE5VEemWNKmXRKnq_g" name="Behavior Scenario">
    – <steps xsi:type="gqam:ExecutionStep" xmi:id="_OW_dkE5VEemWNKmXRKnq_g" name="Step3" cause="_IHi4wE5VEemWNKmXRKnq_g"
     bestCET="4ps" worstCET="4ps" priority="2" concurRes="_EYbisE5VEemWNKmXRKnq_g">
      <outputPin xmi:id="_-OrSAE5bEemWNKmXRKnq_g" successors="_-Or5EE5bEemWNKmXRKnq_g"/>
     </steps>
    – <steps xsi:type="gqam:ExecutionStep" xmi:id="_P6vvUE5VEemWNKmXRKnq_g" name="Step2" cause="_HuSEME5VEemWNKmXRKnq_g"
     respTime="0ps" bestCET="2ps" worstCET="2ps" blockingTime="0ps" priority="2" concurRes="__S5MQE5UEemWNKmXRKnq_g">
      <outputPin xmi:id="__Z0PcE5bEemWNKmXRKnq_g" successors="__Z0PcU5bEemWNKmXRKnq_g"/>
     </steps>
    – <steps xsi:type="gqam:ExecutionStep" xmi:id="_Q4Ln8E5VEemWNKmXRKnq_g" name="Step4" bestCET="5ps" worstCET="5ps" priority="1"
     concurRes="_FlVasE5VEemWNKmXRKnq_g">
      <inputPin xmi:id="__Z0PcU5bEemWNKmXRKnq_g" predecessors="__Z0PcE5bEemWNKmXRKnq_g"/>
     </steps>
    – <steps xsi:type="gqam:ExecutionStep" xmi:id="_SPdj8E5VEemWNKmXRKnq_g" name="Step1" bestCET="3ps" worstCET="3ps" priority="1"
     concurRes="_9AtuQE5UEemWNKmXRKnq_g">
      <inputPin xmi:id="_-Or5EE5bEemWNKmXRKnq_g" predecessors="_-OrSAE5bEemWNKmXRKnq_g"/>
     </steps>
    </behavior>
   – <demand xmi:id="_HuSEME5VEemWNKmXRKnq_g" effect="_P6vvUE5VEemWNKmXRKnq_g">
     <pattern xsi:type="gqam:PeriodicPattern" xmi:id="_HuSrQE5VEemWNKmXRKnq_g" jitter="0ps" phase="0ps" period="20ps"/>
    </demand>
   – <demand xmi:id="_IHi4wE5VEemWNKmXRKnq_g" effect="_OW_dkE5VEemWNKmXRKnq_g">
     <pattern xsi:type="gqam:PeriodicPattern" xmi:id="_IHi4wU5VEemWNKmXRKnq_g" jitter="0ps" phase="0ps" period="15ps"/>
    </demand>
   </workloadBehavior>
  – <resourcePackage xmi:id="_wfC9ck5UEemWNKmXRKnq_g">
   – <ownedElement xsi:type="hrm:HardwareComputingResource" xmi:id="_5srxsE5UEemWNKmXRKnq_g" name="HardwareResource 0"
    mainScheduler="_5s0UkE5UEemWNKmXRKnq_g">
```

Translation procedures

Time4sys model → Parsing → PTA model

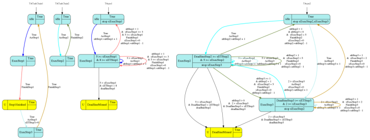

```xml
– <time4sys:Project xmi:version="2.0" xmi:id="_wfCWYE5UEemWNKmXRKnq_g" name="Test5">
  – <design xmi:id="_wfC9cE5UEemWNKmXRKnq_g">
    – <workloadBehavior xmi:id="_wfC9cU5UEemWNKmXRKnq_g">
      – <behavior xmi:id="_OW-2gE5VEemWNKmXRKnq_g" name="Behavior Scenario">
        – <steps xsi:type="gqam:ExecutionStep" xmi:id="_OW_dkE5VEemWNKmXRKnq_g" name="Step3" cause="_IHi4wE5VEemWNKmXRKnq_g"
          bestCET="4ps" worstCET="4ps" priority="2" concurRes="_EYbisE5VEemWNKmXRKnq_g">
            <outputPin xmi:id="_-OrSAE5bEemWNKmXRKnq_g" successors="_-Or5EE5bEemWNKmXRKnq_g"/>
          </steps>
        – <steps xsi:type="gqam:ExecutionStep" xmi:id="_P6vvUE5VEemWNKmXRKnq_g" name="Step2" cause="_HuSEME5VEemWNKmXRKnq_g"
          respTime="0ps" bestCET="2ps" worstCET="2ps" blockingTime="0ps" priority="2" concurRes="__S5MQE5UEemWNKmXRKnq_g">
            <outputPin xmi:id="__Z0PcE5bEemWNKmXRKnq_g" successors="__Z0PcU5bEemWNKmXRKnq_g"/>
          </steps>
        – <steps xsi:type="gqam:ExecutionStep" xmi:id="_Q4Ln8E5VEemWNKmXRKnq_g" name="Step4" bestCET="5ps" worstCET="5ps" priority="1"
          concurRes="_FlVasE5VEemWNKmXRKnq_g">
            <inputPin xmi:id="__Z0PcU5bEemWNKmXRKnq_g" predecessors="__Z0PcE5bEemWNKmXRKnq_g"/>
          </steps>
        – <steps xsi:type="gqam:ExecutionStep" xmi:id="_SPdj8E5VEemWNKmXRKnq_g" name="Step1" bestCET="3ps" worstCET="3ps" priority="1"
          concurRes="_9AtuQE5UEemWNKmXRKnq_g">
            <inputPin xmi:id="_-Or5EE5bEemWNKmXRKnq_g" predecessors="_-OrSAE5bEemWNKmXRKnq_g"/>
          </steps>
        </behavior>
      – <demand xmi:id="_HuSEME5VEemWNKmXRKnq_g" effect="_P6vvUE5VEemWNKmXRKnq_g">
          <pattern xsi:type="gqam:PeriodicPattern" xmi:id="_HuSrQE5VEemWNKmXRKnq_g" jitter="0ps" phase="0ps" period="20ps"/>
        </demand>
      – <demand xmi:id="_IHi4wE5VEemWNKmXRKnq_g" effect="_OW_dkE5VEemWNKmXRKnq_g">
          <pattern xsi:type="gqam:PeriodicPattern" xmi:id="_IHi4wU5VEemWNKmXRKnq_g" jitter="0ps" phase="0ps" period="15ps"/>
        </demand>
      </workloadBehavior>
    – <resourcePackage xmi:id="_wfC9ck5UEemWNKmXRKnq_g">
      – <ownedElement xsi:type="hrm:HardwareComputingResource" xmi:id="_5srxsE5UEemWNKmXRKnq_g" name="HardwareResource 0"
        mainScheduler="_5s0UkE5UEemWNKmXRKnq_g">
```


⇒



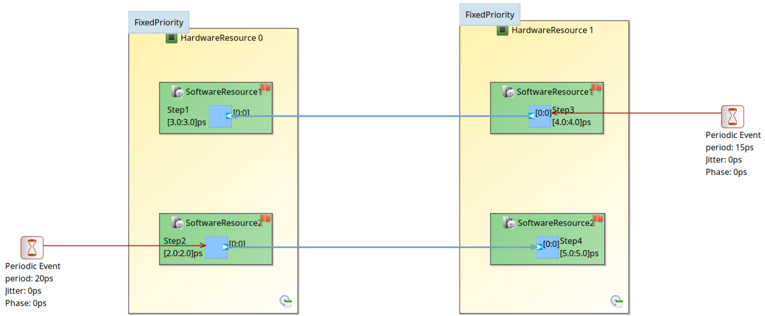
Translation procedures

Time4sys model → Parsing → PTA model → IMITATOR



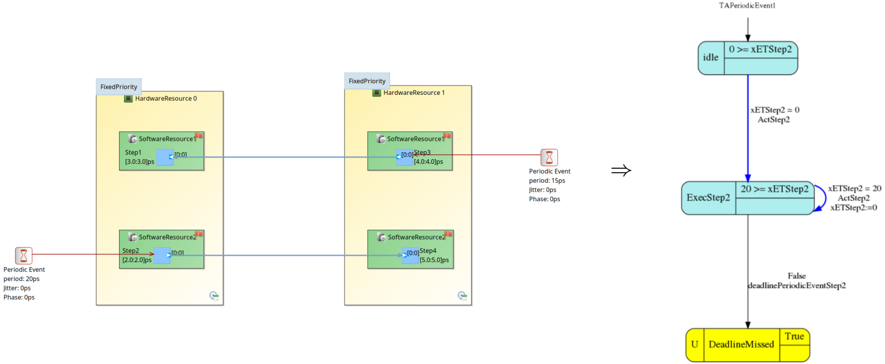
⇒

```
(*************************************************************)
  automaton TAcpu1
(*************************************************************)
synclabs : ActStep1,ActStep2,FinishStep1,FinishStep2,deadlineStep1,deadlineStep2;
loc ExecStep1: invariant xETStep1<=DeadlineStep1 & xExecStep1<=WCETStep1 stop {xExecStep2}
        when xExecStep1<WCETStep1 sync ActStep1 do { nbStep1:=nbStep1 + 1 } goto ExecStep1;
        when xExecStep1<WCETStep1 sync ActStep2 do { nbStep2:=nbStep2 + 1 } goto ExecStep1;
        when xExecStep1<WCETStep1 & xETStep1=DeadlineStep1 sync deadlineStep1  goto DeadlineMissed;
        when xExecStep2<WCETStep2 & xETStep2=DeadlineStep2 & nbStep2>0 sync deadlineStep2  goto DeadlineMissed;
        when xExecStep1<=WCETStep1 & xExecStep1>=BCETStep1 & nbStep1>1 sync FinishStep1 do {nbStep1:=nbStep1-1 , xExecStep1:=0} goto ExecStep1;
        when xExecStep1<=WCETStep1 & xExecStep1>=BCETStep1 & nbStep1=1 & nbStep2=0 sync FinishStep1 do {nbStep1:=nbStep1-1 , xExecStep1:=0} goto idle;
        when xExecStep1<=WCETStep1 & xExecStep1>=BCETStep1 & nbStep1=1 & nbStep2>=1 sync FinishStep1 do {nbStep1:=nbStep1-1 , xExecStep1:=0} goto ExecStep2;


loc ExecStep2: invariant xETStep2<=DeadlineStep2 & xExecStep2<=WCETStep2 stop {xExecStep1}
        when xExecStep2<WCETStep2 sync ActStep1 do {nbStep1:=nbStep1 + 1 , xExecStep1:=0} goto ExecStep1;
        when xExecStep2<WCETStep2 sync ActStep2 do { nbStep2:=nbStep2 + 1 } goto ExecStep2;
        when xExecStep2<WCETStep2 & xETStep2=DeadlineStep2 sync deadlineStep2  goto DeadlineMissed;
        when xExecStep2<=WCETStep2 & xExecStep2>=BCETStep2 & nbStep2>1 sync FinishStep2 do {nbStep2:=nbStep2-1 , xExecStep2:=0} goto ExecStep2;
        when xExecStep2<=WCETStep2 & xExecStep2>=BCETStep2 & nbStep2=1 sync FinishStep2 do {nbStep2:=nbStep2-1 , xExecStep2:=0} goto idle;


loc idle: invariant  True stop {xExecStep1, xExecStep2}
        when  True sync ActStep1 do { nbStep1:=nbStep1 + 1 } goto ExecStep1;
        when  True sync ActStep2 do { nbStep2:=nbStep2 + 1 } goto ExecStep2;


urgent loc DeadlineMissed: invariant  True
```

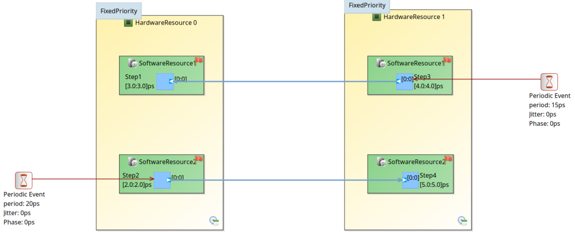


This example features :

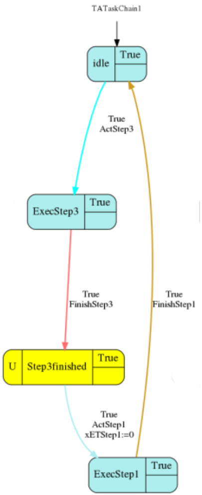
- Two CPUs
- 4 tasks (two in each CPU) :
  - Step1 : WCET = BCET = 3 ps
  - Step2 : WCET = BCET = 2 ps
  - Step3 : WCET = BCET = 4 ps
  - Step4 : WCET = BCET = 5 ps
- Two periodic tasks (Step2 characterized by a 20 ps period and Step3 characterized by a 15 ps period)



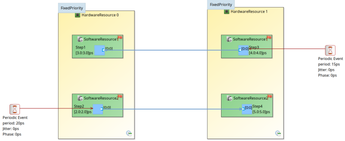
- The translation of the periodic event Step2 to a timed automata (one TA for each periodic event).



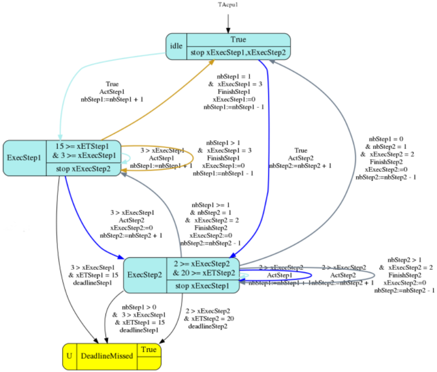
⇒



- The translation of the task chain (Step3, Step1) to a timed automata (one TA for each task chain).



⇒



- The translation of the CPU1 to a timed automata using the scheduling policy FPS with preemption (one TA for each CPU).

Results of the example

results of analysis with IMITATOR and their execution times		
Parameters	**Results**	**E.T**(seconds)
No parameter	True	0.07
DeadlineStep4	$DeadlineStep4 >= 9$	1.8
WCETStep4 and BCETStep4	$BCETStep4 >= 0$ $\wedge\ 18 >= WCETStep4$ $\wedge\ WCETStep4 >= BCETStep4$	3.267
DeadlineStep4, WCETStep4 and BCETStep4	$DeadlineStep4 >= 4 + WCETStep4$ $\wedge\ DeadlineStep4 > 13$ $\wedge\ BCETStep4 >= 0$ $\wedge\ WCETStep4 >= BCETStep4$ $\wedge\ 17 >= DeadlineStep4$ $\wedge\ 11 >= WCETStep4$ $\ldots$	10.6

Perspectives

Perspectives

- Optimize the translation : optimize the size of the automata or reduce the clocks.
- For non-parametric models, we plan to develop a translator to the input language of UPPAAL too, because we believe that using the UPPAAL model checker instead of IMITATOR may be more efficient.
- Some optimizations (on the IMITATOR side) should help to make the analysis faster.

É. ANDRÉ, L. FRIBOURG, U. KÜHNE et R. SOULAT, "IMITATOR 2.5 : A Tool for Analyzing Robustness in Scheduling Problems",, t. 7436, Paris, France, 2012. DOI : 10.1007/978-3-642-32759-9_6.

R. ALUR, T. A. HENZINGER et M. Y. VARDI, "Parametric real-time reasoning", in **STOC**, San Diego, California, United States : ACM, 1993, p. 592-601, ISBN : 0-89791-591-7.

L. FRIBOURG, D. LESENS, P. MORO et R. SOULAT, "Robustness Analysis for Scheduling Problems using the Inverse Method", in **TIME**, Leicester, UK : IEEE Computer Society Press, 2012, p. 73-80. DOI : 10.1109/TIME.2012.10.

C. L. LIU et J. W. LAYLAND, "Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment", **Journal of the ACM**, t. 20, n° 1, p. 46-61, 1973, ISSN : 0004-5411. DOI : 10.1145/321738.321743.

A. S. TANENBAUM et M. v. STEEN, **Distributed Systems : Principles and Paradigms (2Nd Edition)**. Upper Saddle River, NJ, USA : Prentice-Hall, Inc., 2006, ISBN : 0132392275.

Source of the graphics used I

Title : Deepwater Horizon offshore drilling unit on fire
Author : Unknown
Source : https://commons.wikimedia.org/wiki/File:Deepwater_Horizon_offshore_drilling_unit_on_fire_2010.jpg
License : Public domain

Title : The Amagasaki rail crash
Author : To Sawa
Source : https://commons.wikimedia.org/wiki/File:Fukuchiyama_joko20051.jpg
License : CC BY-SA 3.0

Title : Asiana Airlines Plane Crash
Author : Alexander Novarro
Source : https://commons.wikimedia.org/wiki/File:Asiana_Airlines_Plane_Crash.png
License : CC BY-SA 3.0