Part 2 : Timed systems

Exercise 1 – Modeling and verification

Romeo wakes up between 6 and 7 o'clock, learns Timed automata during 6 to 8 hours, then
swims during 2 to 3 hours, makes a jogging during 2 to 4 hours (altogether he makes sport for
less than 6 hours), and goes to “Chez Uppaal” bar for the rest of the day. He goes to bed at 22
hours.

Juliet is awake from 7 to 21, and during all that time she alternates 3 hours of learning Hybrid
systems and 1 hour of hanging at “Chez Uppaal”.

The specification says that they should never meet “Chez Uppaal”.

1. Model Romeo and Juliet behaviours by timed automata.
2. Represent the verification of the specification as emptiness checking for an intersection of
timed regular languages. Hint : you will need a special event for their rendez-vous.

Exercise 2 – About timed languages

Consider the following timed languages on the alphabet \{a, b\}

a) \(L_{10}\) : timed words with integer duration multiple of 10.

b) \(L_{\square}\) : timed words with integer exact square duration.

Can they be recognized by timed automata

A) with \(\epsilon\)-transitions ;

B) without \(\epsilon\)-transitions ?

Hint : you have 4 almost independent questions. To solve question a)B) consider a long word
with only one event one event \(a\) at the very end.

Exercise 3 – Decidable or not?

The result is due to Patricia Bouyer

Consider the class of updatable automata UA. They are timed automata with two more features :

— a clock can be incremented by 1 at a transition ;

— guards of the form \(x - y < c\) (or \(>\) or \(=\)) are authorized

See example below :

\[
\begin{align*}
& a, x < 2/y := y + 1 \\
& b, x - y < 3 \\
& a, x = 5/x := x + 1
\end{align*}
\]

1. Do all UA admit a finite bisimulation? Justify.

2. Is reachability semi-decidable for the class of UA? Justify.

3. Prove that reachability is undecidable for the class of UA.

Hint :

— represent a counter (in a direct way by a clock \(x = n\))

— build gadgets that increment (easy), decrement (hard) and test (easy) a counter.

— adapt everything to two counters

— explain how to simulate a 2-counter machine

— conclude