On the shuffle of regular timed languages

Olivier Finkel
Equipe de Logique Mathématique,
U.F.R. de Mathématiques, Université Paris 7
2 Place Jussieu 75251 Paris cedex 05, France.
finkel@logique.jussieu.fr

Abstract

We show that the class of regular timed languages is not closed under shuffle. This gives an answer to a question which was raised by E. Asarin in [2].

Keywords: Timed automata; timed regular languages; closure properties; shuffle operation.

1 Introduction

We assume the reader to be familiar with the basic theory of timed languages and timed automata (TA) [1].
The set of positive reals will be denoted $\mathcal{R}$. A (finite length) timed word over a finite alphabet Σ is in the form $t_1 \cdot a_1 \cdot t_2 \cdot a_2 \cdots t_n \cdot a_n$, where, for all integers $i \in [1, n]$, $t_i \in \mathcal{R}$ and $a_i \in \Sigma$. It may be seen as a *time-event sequence*, where the $t_i \in \mathcal{R}$ represent time lapses between events and the letters $a_i \in \Sigma$ represent events. The concatenation of such time-event sequences may be defined in a natural manner. For instance,

$$(0.5 \cdot a \cdot 3.43 \cdot b \cdot 7.2) \cdot (2.4 \cdot b \cdot 7 \cdot a) = 0.5 \cdot a \cdot 3.43 \cdot b \cdot 9.6 \cdot b \cdot 7 \cdot a$$

Then timed words with concatenation form a monoid $\mathcal{T}_\Sigma$ which can be seen as a direct sum of the free monoid $\Sigma^\star$ and of the additive monoid $\mathcal{R}$. The set of all (finite length) timed words over a finite alphabet Σ is the set $(\mathcal{R} \times \Sigma)^\star$. A timed language is a subset of $(\mathcal{R} \times \Sigma)^\star$.

We consider a basic model of timed automaton, as introduced in [1]. A timed automaton $\mathcal{A}$ has a finite set of states and a finite set of transitions. Each transition is labelled with a letter of a finite input alphabet Σ. We assume that each transition of $\mathcal{A}$ has a set of clocks to reset to zero and only *diagonal-free* clock guard [1]. As usual, we denote $L(\mathcal{A})$ the timed language accepted (by final states) by the timed automaton $\mathcal{A}$. A timed language L is said to be timed regular iff there is a timed automaton $\mathcal{A}$ such that $L = L(\mathcal{A})$.

It is well known that the class of timed regular languages is closed under union, intersection, but not under complementation. Another usual operation is the shuffle operation. Recall that the shuffle $x \bowtie y$ of two elements x and y of a monoid M is the set of all products in the form $x_1 \cdot y_1 \cdot x_2 \cdot y_2 \cdots x_n \cdot y_n$ where $x = x_1 \cdot x_2 \cdots x_n$ and $y = y_1 \cdot y_2 \cdots y_n$. This operation can naturally be extended to subsets of M by setting, for $R_1, R_2 \subseteq M$, $R_1 \bowtie R_2 = \{x \bowtie y \mid x \in R_1 \text{ and } y \in R_2\}$.
We know that the class of regular (untimed) languages is closed under shuffle, but the question of the closure of the class of timed regular languages under shuffle was still open and is raised by E. Asarin in [2]. We show here that the answer is negative, giving a simple example of two timed regular languages whose shuffle is not timed regular.

2 Non closure under shuffle

Proposition 2.1. *The shuffle of timed regular languages is not always timed regular.*

Proof. Let a, b be two different letters and $\Sigma = \{a, b\}$.
Let R_1 be the language of timed words over Σ in the form

$$t_1 \cdot a \cdot 1 \cdot a \cdot t_2 \cdot a$$

for some positive reals t_1 and t_2 such that $t_1 + 1 + t_2 = 2$, i.e. $t_1 + t_2 = 1$.
It is clear that R_1 is a timed regular language of finite timed words.

Remark. As remarked in [1, page 217], a timed automaton can compare delays with constants, but it cannot remember delays. If we would like a timed automaton to be able to compare delays, we should add clock constraints in the form $x + y \leq x' + y'$ for some clock values x, y, x', y'. But this would greatly increase the expressive power of automata: the languages accepted by such automata are not always timed regular, and if we allow the addition primitive in the syntax of clock constraints, then the emptiness problem for timed automata would be undecidable [1, page 217].

Notice that the above language R_1 is timed regular because a timed automaton $\mathcal{B}$ reading a word in the form $t_1 \cdot a \cdot 1 \cdot a \cdot t_2 \cdot a$, for some positive reals t_1 and t_2, can compare the delays t_1 and t_2 in order to check that $t_1 + t_2 = 1$. This is due to the fact that the delay between the two first occurrences of the event a is *constant* equal to 1.
Using the shuffle operation we shall construct a language $R_1 \bowtie R_2$, for a regular timed language R_2. Informally speaking, this will "insert a variable delay" between the two first occurrences of the event a and the resulting language $R_1 \bowtie R_2$ will not be timed regular.
We give now the details of this construction.

Let R_2 be the language of timed words over Σ in the form

$$1 \cdot b \cdot s \cdot b$$

for some positive real s.
The language R_2 is of course also a timed regular language.

We are going to prove that $R_1 \bowtie R_2$ is not timed regular.

Towards a contradiction, assume that $R_1 \bowtie R_2$ is timed regular. Let R_3 be the set of timed words over Σ in the form

$$t_1 \cdot a \cdot 1 \cdot b \cdot s \cdot b \cdot 1 \cdot a \cdot t_2 \cdot a$$

for some positive reals t_1, s, t_2. It is clear that R_3 is timed regular. On the other hand the class of timed regular languages is closed under intersection thus the timed language $(R_1 \bowtie R_2) \cap R_3$ would be also timed regular. But this language is simply the set of timed words in the form $t_1 \cdot a \cdot 1 \cdot b \cdot s \cdot b \cdot 1 \cdot a \cdot t_2 \cdot a$, for some positive reals t_1, s, t_2 such that $t_1 + t_2 = 1$.

Assume that this timed language is accepted by a timed automaton $\mathcal{A}$.

Consider now the reading by $\mathcal{A}$ of a word in the form $t_1 \cdot a \cdot 1 \cdot b \cdot s \cdot b \cdot 1 \cdot a \cdot t_2 \cdot a$, for some positive reals t_1, s, t_2.
After reading the initial segment $t_1 \cdot a \cdot 1 \cdot b \cdot s \cdot b \cdot 1 \cdot a$ the value of any clock of $\mathcal{A}$ can only be $t_1 + s + 2$, $2 + s$, $1 + s$, or 1.
If the clock value of a clock C has been at some time reset to zero, its value may be $2 + s$, $1 + s$, or 1. So the value t_1 is not stored in the clock value and this clock can not be used to compare t_1 and t_2 in order to check that $t_1 + t_2 = 1$.
On the other hand if the clock value of a clock C has not been at some time reset to zero, then, after reading $t_1 \cdot a \cdot 1 \cdot b \cdot s \cdot b \cdot 1 \cdot a$, its value will be $t_1 + s + 2$. This must hold for uncountably many values of the real s, and again the value $t_1 + s + 2$ can not be used to accept, from the global state of $\mathcal{A}$ after reading the initial segment $t_1 \cdot a \cdot 1 \cdot b \cdot s \cdot b \cdot 1 \cdot a$, only the word $t_2 \cdot a$ for $t_2 = 1 - t_1$.

This implies that $(R_1 \bowtie R_2) \cap R_3$ hence also $(R_1 \bowtie R_2)$ are not timed regular. $\square$

References

[1] R. Alur and D. Dill, A Theory of Timed Automata, Theoretical Computer Science, Volume 126, p. 183-235, 1994.

[2] E. Asarin, Challenges in Timed Languages, From Applied Theory to Basic Theory, Bulletin of the European Association for Theoretical Computer Science, Volume 83, p. 106-120, June 2004.