The Concurrency Column

BY

Luca Aceto

BRICS, Department of Computer Science
Aalborg University, 9220 Aalborg Ø, Denmark
`luca@cs.auc.dk`, `http://www.cs.auc.dk/~luca/BEATCS`

It has become very common for conferences in theoretical computer science to have a large number of satellite workshops, and CONCUR is no exception. This trend has both negative and positive implications. On the positive side, the number of participants to these events has grown considerably, and one has a chance to meet many colleagues, and to listen to a varied and stimulating programme of talks. On the negative side, there is often so much going on in parallel that one ends up missing some very interesting talks — including invited addresses.

One of the talks I missed during CONCUR 2003 in Marseille was Eugene Asarin's invited address at the First International Workshop on Formal Modeling and Analysis of Timed Systems (FORMATS 2003). I am therefore very happy to make up for having missed that talk by devoting this issue of the Concurrency Column to a piece by Eugene Asarin that is based upon that invited address of his. This contribution reports on the author's stimulating opinions on the status of development of the theory of timed regular languages vis-a-vis that of the time honoured theory of regular languages — whose classic status and robustness are witnessed by the plethora of alternative characterizations of the notion of "regular language". There is much food for thought in this column, and I trust that the list of open problems it contains will stimulate some research on this fascinating topic.

I take this opportunity for reminding the readers that CONCUR 2004, the 15th International Conference on Concurrency Theory, will be held in London in the period 31 August–3 September 2004. The event has a wealth of satellite workshops, and an interesting list of invited speakers. I have no doubt that the list of contributed talks will be one of the best yet. I hope that many of you will attend that event. See you in London!

CHALLENGES IN TIMED LANGUAGES:
FROM APPLIED THEORY TO BASIC THEORY[*]

Eugene Asarin
LIAFA – Université Paris 7, case 7014, 2 pl. Jussieu, 75251, Paris, France
asarin@liafa.jussieu.fr; www.liafa.jussieu.fr/~asarin

Abstract

Current state and perspectives of development of the theory of timed languages are analyzed. A large list of open problems is suggested.

1 Why study timed systems?

In most cases computer science deals with sequences of events or actions (we will call them *behaviors*). Powerful formalisms have been developed to express sets or trees of such sequences (various automata, grammars, expressions, process algebras, Petri nets, logics). Some of them are more adapted for a human user, others are excellent for machine treatment[1]; all of them are interesting research objects and are tightly related. Formal languages and related techniques find numerous applications in semantics, specification, verification – in all the situations where we are interested in exploring sets of *behaviors*. Another important class of applications (and source of research problems) concerns *texts* – both in artificial and natural languages. A text is rather a sequence of symbols and not a sequence of events, but I don't see a mathematical difference between the two of them.

However, one aspect is forgotten (or abstracted) in all of the abovementioned approaches, namely *time*. But in reality even for some texts timing is important (a text transmitted via a communication channel can be considered as a timed sequence of characters, a spoken text as a timed sequence of phonemes, and a piece of music can be also seen as a sequence of notes each of which has a duration and starts at some moment of time). As for behaviors, in our physical world, they always happen in time. Sometimes the timing does not matter and it is often a good idea to prove correctness of an algorithm without thinking how much time each

[*]Partially based on the invited talk at FORMATS'03 workshop. Supported by the European community project IST-2001-35304 AMETIST

[1]I first learned this idea from a talk by Amir Pnueli.

operation takes; or to prove that each request is eventually granted, regardless of when the granting actually takes place. However, there are many other situations when one has to make sure that a reactive program always finishes processing a task before the next one arrives, and that each request is granted within 10ms. Also, many programs or protocols work because they use timing (e.g. using time-outs) or because their inputs are subject to some timing constraints.

These practical issues, and especially a need for tools for specification and verification of protocols relying on timing led in the early 90s to the creation of a timed systems theory, which takes into account not only the order of events but also their position in time. The creation of this domain (at least if we consider only automata-based approach) was related with the invention of timed automata by Alur and Dill together with basic decidability results for language emptiness and model-checking problems, and with implementation of these results in verification tools such as Kronos and Uppaal. Nowadays timed systems theory and applications are a well-established area with several thesis defenses every year, a bunch of papers in each CAV, ICALP and CONCUR conference, and a couple of specialized workshops.

So we arrive at our first result.

Result 1. *Sequences of events "embedded" in the time are a ubiquitous reality. It is practically useful and theoretically challenging to study them using approaches of computer science*[2]*. This is done by the timed systems community.*

In the sequel I will try to analyze what has been done in the domain (without being exhaustive) and, more importantly, what has **not** been done and what **should** be done.

2 Historical strength and historical weakness

Let me start with a comparison. The classical theory of finite automata and regular languages (together with technologies based upon this theory) can be seen as a chef d'œuvre of computer science. This beautiful castle (see Figure 1) is based on a solid basement of finite automata theory. A deeper understanding is obtained thanks to the algebraic theory of automata based on monoids. The next floor of the castle is a well-established theory of regular languages (of finite words) and their alternative logical characterizations (Monadic second order logic etc.). A theory of languages of infinite words relies on the theory of finitary languages and allows to express properties of infinite behaviors. For all these formalisms

[2]The author is aware of one well-established theory describing such sequences in a probabilistic setting: the theory of continuous time Markov chains (and its generalizations to point processes).

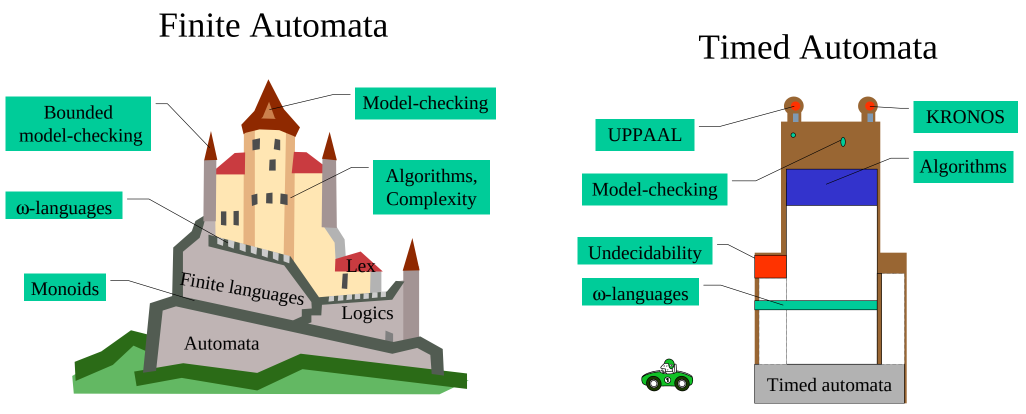

Figure 1: Two theories

various algorithmic problems have been studied, complexity bounds and effective algorithms are known and implemented in tools. All this impressive collection of theories finds numerous applications, some of them are related to analysis of texts (LEX), others to algorithmic methods of verification (and testing). This castle was built without plan but certainly bottom-up, and each subsequent floor was constructed after the previous one had been more or less achieved. Also, its construction certainly moved from theory to practice, from basic theory giving a deep understanding of things to algorithms, from algorithms to concepts of applications, from concepts to more and more efficient tools.

Look now at the next picture in Figure 1 representing the actual (or maybe the recent) state of the timed theory. The aim of constructing this strange skyscraper was certainly to build verification tools for timed systems (such as the two constructions on its roof). All the rest was subordinated to this task, and only the necessary parts – a solid and nice basement of timed automata, some theory of infinite words, useful algorithms, undecidability results (to avoid useless efforts) – have been constructed. Such an approach allowed the research community to construct the building (at least its roof) incredibly fast. However, if we want to have a full understanding of timed languages and systems, we should either add missing parts to the skyscraper (both to fill the empty space and to build a new class of applications), or to rebuild it completely.

We can hence formulate the following two challenges. The first one is closely related to and partly inspired by B. Trakhtenbrot's ideas on pushing the "trinity" Logic-Nets-Automata from finite-state systems to timed and hybrid ones [25].

Challenge 1. *To complete or to rebuild a theory of timed systems and timed languages.*

The second challenge concerns the forgotten class of applications

Challenge 2. *Apply the theory to problems of timed text analysis, e.g. speech or music.*

In the sequel I will discuss some approaches, results and perspectives related only to the first challenge.

3 Re-considering foundations

Let me focus first on the foundations and address the notion of a timed regular language (and of a timed automaton). I will adopt the algebraic approach, and will try to follow the classical path, that is first define "timed" monoids and next consider recognizable and rational subsets of these monoids. The result will be rather instructive.

The reader is referred to [23] for another original approach to building a theory of timed languages and automata from first principles in the framework of [25].

3.1 Timed Monoids

We have introduced timed monoids in [5]. The definitions are quite natural, but differently to the discrete case (where variants of construction, like Moore or Mealy automata, lead to the same free monoids) there are several inequivalent opportunities.

Time-event sequences. Let Σ be an alphabet of actions. A *time-event sequence* (or TES) over Σ has a form

$$t_0 a_1 t_1 a_2 t_2 \ldots a_n t_n$$

where the $t_i \in \mathbb{R}^+$ represent time lapses between events and the $a_i \in \Sigma$ represent events. The concatenation of such sequences can be defined in a natural way, for example:

$$(1.5\,a\,2\,b\,4.2) \cdot (3\,b\,3.141\,b\,0.8) = 1.5\,a\,2\,b\,7.2\,b\,3.141\,b\,0.8$$

TES with concatenation form a monoid $\mathcal{T}_\Sigma$ (the subscript will be omitted in most cases), which can be seen as a direct sum (co-product) of the free monoid Σ^* and the additive monoid $\mathbb{R}^+$. This monoid truthfully represents sequences of events "immersed" in the time axes.

Signals. Another kind of timed monoid is needed to describe a system switching from time to time among several discrete states. A typical example is a logical gate with voltage switching from logical 0 to logical 1. A *signal* over Σ has the form:

$$a_1^{t_1} a_2^{t_2} \dots a_n^{t_n}$$

where $a_i \in \Sigma$ represent signal values and $t_i \in \mathbb{R}^+$ represent their durations. Notice, that if the same value is taken during two adjacent time intervals, than these intervals can be merged, e.g. $a^5 a^{3.2} = a^{8.2}$. Concatenation of signals is done in the natural way, e.g.

$$(a^2 b^{3.1}) \cdot (b^4 a^{1.8}) = a^2 b^{7.1} a^{1.8}$$

Algebraically this monoid $\mathcal{S}_\Sigma$ is the co-product of $|\Sigma|$ copies of $\mathbb{R}^+$ (each copy corresponds to an element of Σ).

Notice, that similarly one can build a monoid of "sigacts"[3] allowing signals mixed with discrete actions.

Monoids $\mathcal{S}$ and $\mathcal{T}$ are probably not isomorphic, but some encodings of signals by TES are known (e.g. representation of a boolean signal by a sequence of raising and falling edges).

Open question 1. *Explore algebraic relations (natural morphisms etc.) between $\mathcal{S}$ and $\mathcal{T}$ and maybe the monoid of "sigacts".*

This exercise should neither be very hard nor very interesting, but it should be done for completeness. It would be a typical chapter 2 of a Ph.D. thesis.

A more general, more important, and less concrete question is

Open question 2. *Explore algebraic properties of timed monoids.*

The bad news is that timed monoids do not have most of the good properties described e.g. in [17]. In spite of that, it is still possible to define rational and recognizable subsets (and to understand the meaning of these classes). Let me proceed with this exercise.

Applying classical recipes. Let us focus on the TES monoid $\mathcal{T}$.

We recall that a subset (language) L of a monoid M is called recognizable if the set of distinct left quotients $x\backslash L$ is finite (such a quotient is, by definition, $\{y \mid xy \in L\}$).

Consider for example the singleton language $\{5\}$ in the monoid $\mathbb{R}^+$ (or, if the reader prefers, in the monoid $\mathcal{T}$). For any $x \in [0; 5]$ the quotient is the singleton set $x\backslash\{5\} = \{5 - x\}$, all these singleton sets are different, hence even such a simple language is not recognizable.

[3]Term of Cătălin Dima.

The following result [14] characterizes recognizable languages in the monoids $\mathbb{R}^+$ and $\mathcal{T}$.

Result 2. *1. There are four recognizable subsets in* $\mathbb{R}^+$*, namely* $\emptyset$, $\{0\}$, $(0;\infty)$ *and the whole* $\mathbb{R}^+$.

2. Recognizable subsets in $\mathcal{T}$ *are those that can be represented by regular expressions with the following atoms:* a *for all* $a \in \Sigma$, 0, *and* $(0;\infty)$.

3. Recognizable subsets in $\mathcal{T}$ *are those that can be recognized by "qualitatively timed" automata described below.*

The qualitatively timed automata cannot measure time, but they have three categories of states:

transient states the automaton can spend 0 time there, that is after entering a transient state it should leave it immediately;

non-transient states each time the automaton enters such a state, it should stay and may stay any positive amount of time;

liberal states no restrictions apply.

In [23] one can find a logical characterization of similar classes of languages.

The good news is that we have a complete and simple description of recognizable subsets of $\mathcal{T}$. The very bad news is that they are only qualitatively timed, which is certainly insufficient for most practical aims.

Another alternative could be to define *rational* subsets of $\mathcal{T}$. One possible definition could be as follows: a subset is rational if it can be described by a regular expression with singletons (elements of $\mathcal{T}$) as atoms. The big disadvantage of this definition is that such languages are countable, hence nothing like "wait between 1 and 5 minutes and then make a" can be expressed (this description corresponds to an uncountable language $\{ta \mid t \in [1;5]\}$). On the other hand a singleton can include uncomputable lapses of time, so there is no hope to insure decidability of basic questions.

For this reason let us consider another Kleene algebra, namely all the sets that can be described by rational expressions using elements of Σ and integer-bounded intervals of reals as atoms. A typical expression would be

$$(a[1;2)b)^* + 3^*a$$

This class of languages already allows for some quantitative timing (specification of delays between consecutive events). In [15] a class of automata equivalent to this class of expression was described. They are mainly finite automata, each state of which is labeled by an interval bounding the staying time in this state. As shown in [15], this class of languages has nice and simple theoretical properties, but restricting to possibility to specify delay only between consecutive events limits drastically their expressive power. In particular, no parallel composition is possible.

Let me summarize:

Result 3. *It is possible and not difficult to define natural classes of languages (recognizable and rational) in timed monoids. It is possible to find simple characterizations of these languages. Unfortunately, these classes happen to be rather restrictive and insufficient to express non-trivial timing properties.*

Natural constructions lead to unsatisfactory language classes (from the practical standpoint). Maybe this is a reason why classes of languages defined ad hoc ways are so popular. We recall that the most known class of timed languages corresponds to those accepted by timed automata [2] that are automata augmented with several special continuous variables (clocks) used to measure time intervals between (not necessarily adjacent) events. Several subclasses of such automata have already been considered, in particular deterministic timed automata, event-clock and event-recording automata [3], hierarchical event-clock automata [19]. Automata of each of these classes have their own theoretical and practical advantages and disadvantages. However I believe there is no strong evidence that one of these classes is **the** natural class of timed "regular" languages. It would be extremely interesting to give an answer to one of the following open questions.

Open question 3. *Give a simple and natural algebraic characterization of a known class of timed languages – confirming that it is the correct class of languages.*

For me the most probable candidates to such a "proof of excellence" are languages recognized by timed automata (because of the elegance of the automata) or by event-recording automata (because of their logical properties and in spite of the heaviness of their definition).

Open question 4. *Define a new natural class of timed languages with good theoretical properties and rich enough to describe non-trivial timed behaviors.*

Adding topology and handling imprecision. The topological approach can help solve the last two open problems, it has also a more practical motivation.

But let me start the discussion with the following reasoning. When defining timed monoids we took into account (in Σ^*) the sequential discrete character of events. As for time, we represented its additive, "shiftable" nature in the additive monoid $\mathbb{R}^+$. What is missing from our models is the continuous nature of time. There is no possibility to say that TES $a\,0.99\,b$ and $a\,1\,b$ are close to each other.

Moreover, in most approaches the continuous nature of time creates strange artifacts. For example many mathematical properties of timed automata, related, for example, to discretization, drastically depend on the type of inequalities used in the transition guards of the automata. Notice that practically there is not much difference between a lapse of time in the interval (5,7) or in [5,7), while theoretically automata with these guards behave very differently.

I am aware of several attempts to deal with this problem. A. Puri [22] considered an alternative semantics of timed automata allowing infinitesimal deviations from the precise behavior. This approach leading to a larger language is justified when the automaton models the uncontrolled environment, and hence we should admit (non-small) consequences of small deviations from the ideal behavior. In another cycle of papers T. Henzinger et al. (see [18, 20]) are considering another semantics of timed automata, leading to a smaller language via taking into account only timed words accepted by the automaton together with their neighborhood. This is reasonable when we want to implement a controller. Recently this implementability issue has been explored with respect to discretely clocked controllers [13].

In spite of all these results, the general understanding of topological aspects is quite incomplete. According to the bottom up approach suggested in this article, I propose to explore the following open questions.

Open question 5. *Define natural topologies on the timed monoids.*

An interesting approach to this problem (in a more general setting) based on Skorohod topology has been proposed in [12]. I suppose that the best solution to this problem would not be a unique topology, but a class of these and a methodology to choose a suitable one. In particular, the topology can depend on commutation (independence) between some events (like in Mazurkiewicz traces). This is important because if a and b commute, then the TES $1a\,0.01\,b\,1$ is close to $1\,b\,0.01\,a\,1$. If a and b are dependent, then it is very important which of them happens before the other, and hence the same two TES differ a lot.

I believe that the previous open problem by itself is not so hard, and in order not to be sterile it should find one or two of the following applications.

Open question 6. *Define classes of "topological regular languages" over monoids with topology. Characterize these classes for timed monoids.*

If by some miracle these classes are nice enough and rich enough, this would give a solution to the Open problem 4. Otherwise, there is still some hope to solve the next problem

Open question 7. *Use topologies on timed monoids to clarify abovementionned aspects of languages of timed automata (tolerance to noise, implementability etc.)*

The topological issues are highly relevant not only for timed automata, but also for hybrid automata, in a smaller measure to pushdown automata, and in general to automata augmented by (infinite-state) variables. An interesting idea suggested in [10] consists in developing a general theory of automata with data. This important problem goes beyond the subject of my article.

4 Lifting the classical theory

In the previous part of this paper we discussed how to improve on existing, or find new foundations to the theory of timed languages. In this part we discuss some problems related to the properties of these languages. In order to proceed we need to choose a class of such languages, and in a slight contradiction to what preceded, I will speak about the languages defined by Alur-Dill timed automata, which admit a nice definition, have nice properties, a rather elaborated algorithmics, and numerous applications. And even if, as explained above, I am not fully satisfied with this definition, it is close to the perfect solution, and it is interesting to work in this paradigm until a better one is found. In the rest of the paper I will call such languages timed regular languages.

I will adopt the same approach as above: to try to "lift" the classical theory of regular languages to timed regular ones.

4.1 What is known

My picture of an empty skyscraper is of course a caricature. Many things are known about timed languages since Alur's thesis [1] and other works of the same period. Let me recall some of the basic facts (an accessible reference is [2]).

- Timed automata and timed regular languages are rich enough to represent faithfully many real-life examples (real-time programs, scheduling problems, circuits . . .)
- Membership and Empty Language problems are decidable (using a finite bisimulation).
- Inclusion and Universal Language problems are undecidable.
- Timed regular languages closed under $\cup, \cap$ and morphism (renaming)
- Timed regular languages are not closed under complementation.

- The determinization of timed automata is in general impossible.
- Model-checking algorithms for timed temporal logics, such as TCTL, exist and are implemented in verification tools like KRONOS and UPPAAL

4.2 Some well-posed questions

Most of the classic problems that admit a simple "yes or no" answer have already been solved (as mentioned above). I am aware of several such questions that are still open:

Shuffle A *shuffle* $(x \sqcup\!\sqcup y)$ of two elements x and y of a monoid M consists of all the products $x_1 y_1 \ldots x_n y_n$, such that $x_1 \ldots x_n = x$ and $y_1 \ldots y_n = y$. This operation can be naturally lifted to subsets of M. A shuffle of two regular languages (in Σ^*) is also regular.

Open question 8 (Shuffle). *Is the shuffle of timed regular languages always regular?*

A positive answer to this problem, interesting in itself, would also simplify known algebraic characterizations of the class of timed regular languages

The straightforward approach of switching between two timed automata does not work because the flow of time in the inactive automaton should be stopped, which is impossible since the clocks in Alur-Dill model never stop. However, in some cases the shuffle is regular : $5 \sqcup\!\sqcup 3$ is merely 8, and

$$5a \sqcup\!\sqcup 3b = \{satb \mid 5 \leq s \wedge s + t = 8\} \cup \{sbta \mid 3 \leq s \wedge s + t = 8\},$$

which is timed regular.

Decidability problems Several decidability problems have been stated and partially solved in [27]. A typical one is

Open question 9 (Decidability à la [27]). *Is it possible, given a timed automaton $\mathcal{A}$, to decide whether it is equivalent to a deterministic one?*

This problem is trivially decidable in the non-timed case: every automaton is equivalent to a deterministic one. In the timed case S. Tripakis has proved in [27] that there is no algorithm that for any automaton **decides** whether it is determinizable, **and** if this is the case **computes** an equivalent deterministic automaton. The decidability of the determinizability question alone is still open.

4.3 Some ill-posed problems

In this section we consider more interesting and stimulating problems that consist in lifting to timed languages important results of the theory of finite automata and regular languages that cannot be lifted directly.

Does Kleene's theorem hold? As mentioned above, the class of rational timed languages (=described by rational expressions) is much smaller then the class of timed regular languages (=accepted by timed automata). Hence, no direct analog of Kleene's theorem can hold. Nonetheless, I was involved in two and am aware of four or five versions of Kleene's theorem for timed languages [4, 5, 8, 9, 6].

In order to explain how it can be possible I will recall the main result of [5]. A formalism introduced in that paper is similar to rational expressions. It contains atoms for all actions $a \in \Sigma$, a special symbol τ for an arbitrary time lapse, all the usual operations of Kleene algebra $(+, \cdot, ^*)$, the time restriction operator $(\langle L \rangle_I)$ that selects only those elements of a language L that have a duration in the integer-bounded interval I. For example, the expression

$$\langle (\langle a\tau \rangle_{[1,2]})^* a \rangle_{[2000,2005]}$$

denotes the timed language consisting of all the sequences of a spaced by 1 to 2 time units and with a total duration between 2000 and 2005. In order to match the expressive power of timed automata two more operations are needed: $\wedge$ for language intersection and $a \to b$ for renaming actions.

The analog of Kleene's theorem stated in [4, 5] is the following:

Result 4. *The formalism described above has the same expressive power as timed automata.*

The reader is referred to articles [5, 8, 9, 6] for proof details, alternative formalisms and their comparison.

Just in case, I formulate

Open question 10 (Kleene's theorem). *Find better regular expressions-like formalisms equivalent to timed automata.*

Are timed regular languages captured by a logic? A well-known result of Büchi-Elgot-Trakhtenbrot (see e.g. [11]) states that a language is regular if and only if it can be defined by a formula of the monadic second order logic. No such characterization of timed regular languages is possible just because these languages are not closed under complementation. Nonetheless, several nice results on equivalence of certain logics and certain classes of timed automata have been obtained. Ten years ago Wilke [29] introduced a logic $\mathcal{L}d$ and described its fragment equivalent to timed automata.

A more recent result [19] establishes an equivalence between a so-called State-clock Logic and hierarchical event-clock automata (their languages **are** closed under complementation). Along the same lines [16] gives a logical characterization of event-clock automata.

Open question 11 (Logical characterization). *Find better (simpler) logical formalisms equivalent to timed automata.*

Does Myhill-Nerode theorem hold? How to minimize timed automata? As explained above (Result 2), recognizable timed languages are only languages with qualitative timing. This class is much smaller than the class of timed regular languages, hence the direct analog of Myhill-Nerode theorem fails.

In a recent work [21] Myhill-Nerode theorem is reformulated in terms of rewriting, and in this form it is ported to languages of deterministic timed automata. The approach suggested in that paper can potentially clarify foundational questions of our Section 3. Probably this work gives a partial answer to:

Open question 12 (Myhill-Nerode theorem). *Obtain a characterization of timed regular languages (or deterministic timed regular languages) in terms of existence of a "finite representation" of a congruence relation.*

In the untimed case Myhill-Nerode theorem has its practical aspect, namely the minimization procedure. Much interesting work has been done on simplification of timed automata, but little of it can be seen as minimization, that is construction of a canonical small automaton related to a given (deterministic) timed automaton. This is done for a specially designed class of automata in [24]. In [28] an interesting minimal object is constructed for a timed automaton by taking a time-abstracting bisimulation.

I formulate the minimization problem in a very generic way:

Open question 13 (Minimization). *Define a canonical object representing a (deterministic) timed automaton.*

A related question is how to determinize timed automata? The simple answer is well-known: In general determinization is impossible. However the main method used in [26] to build observers consists in constructing for a given timed automaton an equivalent deterministic timed transition system. I fail to formulate a general open problem here, but I think that this kind of construction can be important in the context of canonical representation of timed automata.

Pumping Lemma & Co I refer the reader to [7] where it is stated that direct analogs to the Pumping Lemma fail for timed regular languages, and some alternative versions of this lemma holding in the timed case are proved. Again, the open problem is quite generic:

Open question 14. *Develop simple techniques allowing to prove that a given timed language **is not** regular.*

5 Some final remarks

The standpoint chosen in this paper, namely a bottom-up (from foundation to theory, from theory to applications) language-theoretic approach inspired by the theory of regular languages, permitted to identify a series of important questions. I believe that getting answers to them would substantially improve our understanding of the area, and would be useful for applications.

Nonetheless, other approaches to this research area are also possible, in particular the top-down application-oriented one, and the view of timed systems as a source of amazing mathematical puzzles. These popular approaches are completely justified (and sometimes used by my own self) and lead to other open problems not addressed here.

And the last remark. Here I have only considered timed languages and automata, and never other timed models (such as timed Petri nets or timed process algebras). I believe that some of the problems presented are also valid for such models, but it is beyond the scope of this work.

6 Acknowledgements

Many people contributed to this work. It would never appear without long discussions and a ten years long collaboration with Oded Maler. I am also thankful to him for valuable comments on the manuscript. I was certainly influenced and enlightened by Paul Caspi, Stavros Tripakis, Sergio Yovine, Joseph Sifakis, Cătălin Dima and Peter Niebert. During the preparation of the FORMATS'2003 invited talk Rajeev Alur, Paul Caspi, Thomas A. Henzinger, Oded Maler, Joël Ouaknine, P.S. Thiagarajan, and Stavros Tripakis answered my call for open problems, and

I have used their ideas sometimes directly, sometimes as a source of inspiration. Last but not least, a kind invitation to this Concurrency Column and an efficient and patient editor's work of Luca Aceto are thankfully acknowledged.

References

[1] R. Alur. *Techniques for Automatic Verification of Real-Time Systems*. PhD thesis, Stanford University, 1991.

[2] R. Alur and D. Dill. A theory of timed automata. *Theoretical Computer Science*, 126:183–235, 1994.

[3] R. Alur, L. Fix, and T. A. Henzinger. Event-clock automata: a determinizable class of timed automata. *Theoretical Computer Science*, 211:253–273, 1999.

[4] E. Asarin, P. Caspi, and O. Maler. A Kleene theorem for timed automata. In *Proceedings of LICS'97*, pages 160–171, 1997.

[5] E. Asarin, P. Caspi, and O. Maler. Timed regular expressions. *Journal of the ACM*, 49(2):172–206, 2002.

[6] E. Asarin and C. Dima. Balanced timed regular expressions. In *Proceedings of MTCS'2002*, volume 68 of *ENTCS*, 2002. issue 5.

[7] D. Beauquier. Pumping lemmas for timed automata. In *Proceedings of FoSSaCS'98*, volume 1378 of *LNCS*, pages 81–94, 1998.

[8] P. Bouyer and A. Petit. Decomposition and composition of timed automata. In *Proceedings of ICALP'99*, volume 1644 of *LNCS*, pages 210–219, 1999.

[9] P. Bouyer and A. Petit. A Kleene/Büchi-like theorem for clock languages. *Journal of Automata, Languages and Combinatorics*, 7(2):167–186, 2002.

[10] P. Bouyer, A. Petit, and D. Thérien. An algebraic approach to data languages and timed languages. *Information and Computation*, 182(2):137–162, 2003.

[11] J. Büchi. Weak second order logic and finite automata. *Z. Math. Logik, Grundlag. Math.*, 5:66–62, 1960.

[12] P. Caspi and A. Benveniste. Toward an approximation theory for computerised control. In *Proceedings of EMSOFT'2002*, volume 2491 of *LNCS*, pages 294–304, 2002.

[13] F. Cassez, T. A. Henzinger, and J.-F. Raskin. A comparison of control problems for timed and hybrid systems. In *Proceedings of HSCC'2002*, volume 2289 of *LNCS*, pages 134–148, 2002.

[14] C. Dima. *An algebraic theory of real-time formal languages*. PhD thesis, Université Joseph Fourier, Grenoble, France, 2001.

[15] C. Dima. Real-time automata. *Journal of Automata, Languages and Combinatorics*, 6:3–23, 2001.

[16] D. D'Souza. A logical characterisation of event clock automata. *Intl. Journal of Foundations of Computer Science*, 14:625–639, 2003.

[17] S. Eilenberg. *Automata, Languages, and Machines*, volume A. Academic Press, 1974.

[18] V. Gupta, T. A. Henzinger, and R. Jagadeesan. Robust timed automata. In *Proceedings of HART Workshop*, volume 1201 of *LNCS*, pages 331–345, 1997.

[19] T. Henzinger, J.-F. Raskin, and P.-Y. Schobbens. The regular real-time languages. In *Proceedings of ICALP'98*, volume 1443 of *LNCS*, pages 580–591, 1998.

[20] T. A. Henzinger and J.-F. Raskin. Robust undecidability of timed and hybrid systems. In *Proceedings of HSCC'2000*, volume 1790 of *LNCS*, pages 145–159, 2000.

[21] O. Maler and A. Pnueli. On recognizable timed languages. In *Proceedings of FoSSaCS'2004*, volume 2987 of *LNCS*, pages 348–362, 2004.

[22] A. Puri. Dynamical properties of timed automata. *Discrete Event Dynamic Systems*, 10(1/2):87–113, 2000.

[23] A. Rabinovich. Automata over continuous time. *Theoretical Computer Science*, 300:331–363, 2003.

[24] J. G. Springintveld and F. W. Vaandrager. Minimizable timed automata. In *Proceedings of FTRTFT'96*, volume 1135 of *LNCS*, pages 130–147. Springer-Verlag, 1996.

[25] B. Trakhtenbrot. Origins and metamorphoses of the trinity: Logics, nets, automata. In *Proceedings of LICS'95*, pages 506–507, San Diego, 1995. IEEE Computer Society.

[26] S. Tripakis. Fault diagnosis for timed automata. In *Proceedings of FTRTFT'2002*, volume 2469 of *LNCS*, pages 205–224, 2002.

[27] S. Tripakis. Folk theorems on the determinization and minimization of timed automata. In *Proceedings of FORMATS'2003*, volume 2791 of *LNCS*, pages 182–188, 2004.

[28] S. Tripakis and S. Yovine. Analysis of timed systems based on time-abstracting bisimulations. *Formal Methods in System Design*, 18:25–68, 2001.

[29] T. Wilke. Specifying timed state sequences in powerful decidable logics and timed automata. In *Proceedings of FTRTFT'94*, volume 863 of *LNCS*, pages 694–715, 1994.