Timed Collaborative Systems with Real Time

Max Kanovich, Queen Mary, University of London, UK
Tajana Ban Kirigin, University of Rijeka, HR
Vivek Nigam, Federal University of Paraíba, João Pessoa, Brazil
Andre Scedrov, University of Pennsylvania, Philadelphia, USA

Often one needs to reason about time when setting the rules and goals of a collaboration. For example, agents need to comply with deadlines and react quickly under unexpected events. Although in many situations it is enough to represent time discretely [3], such as in days or hours, sometimes one needs real time.

For instance, in many situations, an agent A needs to know whether another agent B is near, within a radius. A way to determine this is by calculating the round time of a message: A sends a challenge message m to B and remembers the time t_0 when the message was sent. Then once B receives the message m, it computes a response message $f(m)$ and sends it as quickly as possible back to A. When this response message reaches agent A, at some time t_1, A checks whether the round time $t_1 - t_0$ is less than the given threshold. If this is the case, then A can assume that B is within some radius. Otherwise A cannot conclude anything about how distant B is. In fact, this is the basic principle of Distance Bounding Protocols [1].

In order to formally specify and verify collaborative systems involving real time, such as the scenario above, and also to verify whether it is possible for an intruder to appear to be someone else or to appear closer than he actually is, one needs formal models that can mention real time and can *generate fresh values*. Fresh values, also called nonces in protocol security literature [2], are used so that messages sent in previous interaction between agents are not mixed up with current ones. They are used for instance in the authentication of agents.

This paper proposes a rewriting framework that can be used to specify collaborative systems equipped with real time and where agents may create fresh values. It extends our previous work on Timed Local State Transition Systems (TLSTSes) with explicit time [3] where only discrete time was allowed. Modelling real time in TLSTS was left as future work in [3]. Here, we outline the extension of the model and describe the fragment for which the reachability problem is PSPACE-complete.

Rewriting Model TLSTSes are multiset rewriting systems. The state of the system or a system configuration is represented by a multiset of facts. Facts are atomic formulas with a positive real number called *timestamp* associated to each fact. Agents change the state of the system by applying actions. Sequences of actions or *plans* are compliant if, starting from a given initial configuration, they lead to a goal configuration without reaching any configuration that is considered critical. The main problem when studying TLSTSes is the *planning problem*: Given a timed local state transition system $\mathcal{T}$, an initial configuration W and a finite set of goal and critical configurations, is there a compliant plan? In this paper we address the complexity of the planning problem for TLSTSes with real time.

In TLSTSes time is modelled through timestamps attached to facts, through a special fact *Time* representing global time, and through time constrains that can be associated with actions and with configurations. More precisely, a *timestamp* is a positive real number attached to a fact. It can represent time in various ways, for example it can denote the time when the fact was created, or the time time until the fact is valid etc. *Time constraints* are arithmetic comparisons involving exactly two timestamps:

$$T_1 = T_2 \pm a,\ T_1 > T_2 \pm a,\ \text{or}\ T_1 \geq T_2 \pm a, \tag{1}$$

where a is a *natural number* and T_1 and T_2 are time variables, which may be instantiated by the timestamps of any fact including the global time.

Action application can also have time conditions. Time constraints can be attached to actions, to act as a guard of the rule, that is, an action can only be applied if the attached time constraints are all satisfied. Only two types of actions are allowed. The first is the following type of action that increments the global time of a configuration by a *positive real number t*:

$\mathit{Time}@T \mid \{\} \to_{clock} \mathit{Time}@(T+t)$.

This is the only action that can modify the global time of a configuration. Actions of the second type are instantaneous and have necessarily the following form:

$\mathit{Time}@T, W \mid \Upsilon \ \longrightarrow_A \ \exists \vec{x}. \mathit{Time}@T, W'$

where Υ is the guard of the action containing a finite set of constraints. We restrict actions so that all variables appearing in Υ are contained in the set of time variables $\{T_1, \ldots, T_n, T\}$ from the pre-condition. We further impose the following condition on these actions: if $\mathit{Time}@T$ is in the pre-condition W, then all facts created in the post-condition W' are of the form $P@(T+d)$, where d is a *natural number*, possibly zero. That is, all the created facts have timestamps greater or equal to the global time. The existentially quantified variables in a rewrite rule specify the creation of fresh values.

Goals and critical configurations are specified similarly, by allowing one to attach time constraints to them, exactly as in [3].

Complexity When considering the complexity of the planning problem with real time, one has to deal with the unboundedness of time and with the density of time. In our previous work with discrete time [3], we show how to tackle the unboundedness of time, *i.e.*, an internally infinite space of configurations, by using a finite number of δ-representations of configurations. Instead of the actual values of timestamps, δ-representations contain only relative time differences truncated by an upper bound D_{max} deduced from the system specification. It is necessary to assume that the size of facts is bounded and that the timed local state transition system is balanced, that is pre and post-conditions of all actions have the same number of facts.

This approach alone does not work when timestamps are real numbers, as there is an infinite number of possible values for relative time differences. To address the density of time, we introduce the novel equivalence relation among configurations, inspired by [4]. This provides a bounded number of classes called *circle-abstractions*. Similar to [3], we define the δ-configuration of a configuration $\mathcal{S}$ and a natural number D_{max} as follows: it is the list $[F_1, \delta_1, F_2, \ldots, F_{n_1}, \delta_{n-1}, F_n]$ of its facts, $F_1, \ldots, F_n$, ordered according to the values of their timestamps, interleaved by the value δ_i obtained by the truncated time differences w.r.t. D_{max} of the corresponding two neighbouring facts F_i and F_{i+1}.

Definition 1. *Two configurations $\mathcal{S}_1$ and $\mathcal{S}_2$ are equivalent for a given natural number D_{max} if the following two conditions are satisfied: (1 - δ-configurations) Their δ-configurations w.r.t. D_{max} are the same when considering only the integer part of the time differences; and (2 - Circle) when their facts are ordered using only the decimal part of timestamps, one obtains the same list of facts. (If they have the same value, then we indicate this in the list by using the symbol $=$.)*

For instance, the following two configurations are equivalent when $D_{max} = 2$: $\{P_0@0.4, P_1@1.5, \mathit{Time}@5.4, P_2@6.6\}$ and $\{P_0@3.2, P_1@4.5, \mathit{Time}@8.2, P_2@9.6\}$ as they have the same integer parts of truncated relative times, represented by the following δ-configuration $[P_0, 1, P_1, \infty, \mathit{Time}, 1, P_2]$ and the same circle configuration, namely $[\mathit{Time} = P_0, P_1, P_2]$.

We show that circle-abstractions are well-defined with respect to the planning problem. This allows us to represent plans using circle-abstractions only. In particular, we show that all configurations that have the same circle-abstraction satisfy the same time constraints. We extend action application to circle-abstractions. Since abstractions do not contain the information of exact time differences between timestamps and the current time, we cannot apply time incrementing action for a concrete value t to circle-abstractions. In order to model time advancement we add a special action *next* . Application of *next* results in the circle-abstraction in which the time has shifted just enough to change the abstraction, and hasn't shifted too much to jump over some abstractions as time advances.

We show that our formalization of circle-abstractions is sound and complete and, therefore, we conclude that any given planning problem can be conceived as a planning problem over circle-abstractions. For the complexity proofs it is essential to show that for a given planning problem we obtain only a finite number of circle-abstractions with which we are able to represent an infinite space of configurations.

We finally show that in balanced TLSTSes with real time, when the size of facts is bounded and actions are balanced, the planning problem is PSPACE-complete.

Acknowledgments: We thank Catherine Meadows, John Mitchell, and Carolyn Talcott for helpful discussions. This material is based upon work supported by the MURI program under AFOSR Grant No: FA9550-08-1-0352 and upon work supported by the MURI program under AFOSR Grant No. FA9550-11-1-0137. Additional support for Scedrov from NSF Grant CNS-0830949 and from ONR grant N00014-11-1-0555. Nigam was partially supported by the Alexander von Humboldt Foundation and CNPq. Kanovich was partially supported by the EPSRC.

References

[1] S. Brands and D. Chaum. Distance-bounding protocols. In EUROCRYPT, 1993.

[2] N. A. Durgin, P. Lincoln, J. C. Mitchell, and A. Scedrov. Multiset rewriting and the complexity of bounded security protocols. *Journal of Computer Security*, 12(2):247–311, 2004.

[3] M. I. Kanovich, T. B. Kirigin, V. Nigam, A. Scedrov, C. L. Talcott, and R. Perovic. A rewriting framework for activities subject to regulations. In RTA, 2012.

[4] M. I. Kanovich, M. Okada, and A. Scedrov. Specifying real-time finite-state systems in linear logic. *Electr. Notes Theor. Comput. Sci.*, 16(1):42–59, 1998.