The Compositional Specification of Timed Systems — A Tutorial

Joseph Sifakis

Verimag, 2 rue Vignate, 38610 Gières, France
Joseph.Sifakis@imag.fr

Motivation

The analysis of reactive systems requires models representing the system, its interaction with the environment and taking into account features of the underlying execution structure. It is important that such models are timed if analysis concerns performance, action scheduling or in general, dynamic aspects of the behavior. In practice, timed models of systems are obtained by adding timing constraints to untimed descriptions. For instance, given the functional description of a circuit, the corresponding timed model can be obtained by adding timing constraints about propagation delays of the components; to build a timed model of a real-time software, quantitative timing information concerning execution times of the statements and significant changes of the environment must be added.

The construction of timed models of reactive systems raises some important questions concerning their composition and in particular, the way some well-understood constructs for untimed systems can be extended to timed systems.

In this tutorial, we present an overview of existing executable timed formalisms with a global notion of time, by putting emphasis on problems of compositional description. The results on compositionality have been developed in collaboration with S. Bornot, at Verimag.

Timed Formalisms

Timed formalisms are extensions of untimed ones by adding *clocks*, real-valued variables that can be tested and modified at transitions. Clocks measure the time elapsed at states. Timed automata [AD94,ACH+95], timed process algebras [NS91] and timed Petri nets can be considered as timed formalisms.

The semantics of timed formalisms can be defined by means of transition systems that can perform time steps or (timeless) transitions. A state is a pair (s, v), consisting of a control state s (of the untimed system) and a valuation of the clocks. As a rule, transitions are specified by a guard (predicate) on clocks and an assignment of new values to clocks. They correspond to *actions* of the considered system. *Time progress conditions* are predicates on clocks associated with control states s that specify how time can progress: a time step of duration d can be performed from s only if all the intermediate states satisfy the time progress condition.

N. Halbwachs and D. Peled (Eds.): CAV'99, LNCS 1633, pp. 2–7, 1999.

An important feature of timed models is the possibility to express *urgency* of an action (transition). An action enabled at a state (s, v), becomes urgent if time cannot progress at v. As time cannot advance, the urgent action can be executed. Expressing urgency is essential in modeling the real-time behavior of systems. However, stopping time progress to simulate urgency, can be a source of problems, especially when composing timed models. The independent description of transitions and of time progress conditions may induce undesirable deadlock situations where time cannot progress and no action is enabled.

To avoid timelocks, a class of timed formalisms has been studied where time progress conditions are associated with the transitions in the form of *deadlines* [SY96,BS98,BST97]. The deadline of a transition is a predicate on clocks which implies the associated guard and represents the set of the clock valuations at which the transition becomes urgent. Inclusion of deadlines in the corresponding guards implies *time reactivity* that is, whenever time progress stops, there exists at least one enabled transition. The use of deadlines has another interesting consequence. Each transition with the associated guard, deadline and assignment, corresponds to an elementary timed system, called *timed action*.

We show how a timed transition system can be obtained as the composition of timed actions.

Composition of Timed Systems

As usual, the behavior of a timed system is obtained by composing the behavior of its components. Most of the work on the composition of timed systems, concerns timed process algebras. Very often it adopts a principle of independence between timed and untimed behavior: transitions and time steps of the system are obtained by composing independently the transitions and time steps of the components. Furthermore, a strong synchrony assumption is adopted for time progress. Time can progress in the system by some amount d only if all the components agree to let time advance by d. This leads to elegant *urgency preserving* semantics in the sense that component deadlines are respected. However, this orthogonality between time progress and transitions may easily introduce timelocks, especially when an untimed description with communication allowing waiting, e.g. rendez-vous, is extended into a timed description. In such cases, it is questionable whether the application of a strong synchronization rule for time progress is always appropriate. For instance, if two systems are in states from which they will never synchronize, it may be desirable not to further constrain time progress by the strong synchronization rule.

As an alternative to urgency preserving semantics, *flexible* composition semantics have been studied [BST97,BS98]. This semantics preserve time reactivity. To avoid timelocks, urgency constraints are relaxed in some manner that is shown to be optimal. The main idea behind flexible semantics, is to adjust waiting times of the components so as to achieve a desirable global behavior satisfying by construction, the following two *sanity* properties.

One property is time reactivity which can be guaranteed by construction and is related to absence of timelock. Contrary to other stronger well-timedness properties, time reactivity is very easy to satisfy by construction.

The second property is activity preservation and is related to absence of (local) deadlock. It requires that if some action can be executed after waiting by some time in a component, then some (not necessarily the same) action of the system can be executed, after waiting by some (not necessarily the same) time.

The Compositional Framework

We show how timed systems can be built from timed actions by preserving both time reactivity and activity of components.

The set of the timed actions on given set of clocks, set of control states and vocabulary of action names, consists of a transition on control states labeled by a tuple (a, g, d, f) where a is an action name, g is a guard, d is a deadline and f is a function on clocks. The guard g and the deadline d are predicates on clocks such that d implies g, representing respectively the set of enabling and the set of the urgent states of the timed action. The function f represents the effect of the execution on clock states.

A timed system is a set of timed actions. Following a standard process algebra approach, it can be described in an algebra of terms generated from some constant, representing the idle system, by using timed action prefixing, non deterministic choice and recursion. Equality of terms is the congruence obtained by assuming associativity, commutativity and idempotence of non deterministic choice, that is, the labeled transition structures of the terms are bisimilar, where equality of two labels means identity of their action names and equivalence of the corresponding guards, deadlines and functions.

We define two kinds of operators on timed systems: priority choice operators and parallel composition operators. The operators are timed extensions of untimed operators. We give sufficient conditions for preserving both time reactivity and activity of components.

Priority choice operators

Priority is a very useful concept for modeling interrupts or preemption in real-time systems. A well-known difficulty with introducing priorities, is that they are badly compatible with compositionality and incrementality of specification [BBK86,CH90,BGL97].

We define priority choice operators, that is choice operators depending on a relation between actions. This relation is an order on action names parameterized by non negative reals representing *degrees of priority*. Roughly speaking, if action a_2 has priority over action a_1 of degree d, then in the priority choice of two timed actions with labels a_2 and a_1, action a_1 will be disabled if action a_2 will be enabled within d time units. The main results concerning priority choice are the following:

– Priority choice operators can be expressed in terms of non deterministic choice operators, by restricting appropriately the guard and the deadline of

actions of lower priority. The restricted guards and deadlines can be specified in a simple modal language. However, modalities are just a macronotation, as they represent quantification over time which can be eliminated.
- We provide sufficient conditions on the priority order, for the priority operators to be associative, commutative and idempotent. This result allows to consider priority choice operators as basic operators, generalizations of non deterministic choice. The latter can be considered as the choice operator for the empty priority order.
- We show that under these conditions, priority order operators preserve activity in the following sense: for every state, if an action a is enabled under the non deterministic choice then either a or a higher priority action will be enabled under the priority choice.

Parallel composition operators

Parallel composition operators for timed systems are considered as extensions of parallel composition operators for untimed systems. We suppose, as usual, that the latter are defined in terms of choice operators and some associative and commutative synchronization operator on actions, by means of an expansion rule [Mil83,Mil89]. Synchronization operators associate with pairs of actions the action resulting from their synchronization. The main results concerning parallel composition operators are the following:

- Parallel composition operators can be expressed in terms of choice operators, by appropriately extending the synchronization operators on timed actions. Synchronization operators are associative and commutative and compose componentwise the guards and the deadlines of the synchronizing actions.
- For the composition of guards, different *synchronization modes* of practical interest are studied. Apart from the usual *and*-synchronization, where the synchronization guard is the conjunction of the guards of the synchronizing actions, are considered *max*-synchronization allowing waiting, and *min*-synchronization allowing interruption by the fasted component.
- Parallel composition operators are associative and commutative if they are extensions of untimed operators satisfying the same properties.
- We show that maximal progress can be achieved in synchronization by using priority choice in the expansion rules. Furthermore, we provide sufficient conditions for activity preservation.

The algebraic framework is completed by studying a simple algebra with synchronization operators for timed actions. We deduce laws for timed systems that take into account the structure of the actions and there properties.

Typed Actions - A Simplified Framework

A practically interesting simplification of the theoretical framework comes from the (trivial) remark that any timed action can be expressed as the non deterministic choice between a *lazy* action and an *eager* action. A lazy action is an

action whose set of urgent states is empty and an eager action has its deadline equal to its guard. This allows to consider only these two types of actions in specifications and simplifies the rules for synchronization.

Sometimes it is useful in practice, to consider a third type of urgency, *delayable* actions. An action is delayable if its deadline is exactly the falling edge of the guard. That is, it cannot be disabled without becoming urgent. We show that parallel composition of systems with delayable actions yields systems with delayable actions.

Discussion

The distinction between urgency preserving and flexible approach seems to be an important one and is related to the ultimate purpose of the specification. When a complete specification is sought, in view of analysis and verification, it is reasonable to consider that the violation of component deadlines is an error. On the contrary, if the purpose of the specification is to derive a system which is correct with respect to given criteria, knowing the behavior of its components, the flexible approach is appropriate. This approach provides a basis for constructing timed systems that satisfy the two sanity properties, time reactivity and activity preservation. It is very close to synthesis and can be combined with automatic synthesis techniques.

An important outcome of this work is that composition operators for untimed systems admit different timed extensions due to the possibility of controlling waiting times and "predicting" the future. The use of modalities in guards drastically increases succinctness in modeling and is crucial for compositionality. It does not imply extra expressive power for simple classes of timed systems, where quantification over time in guards can be eliminated.

The definition of different synchronization modes has been motivated by the study of high level specification languages for timed systems, such as Timed Petri nets and their various extensions[SDdSS94,SDLdSS96,JLSIR97]. We have shown that the proposed framework is a basis for the study of the underlying semantics and composition techniques; if they are bounded, then they can be represented as timed systems with finite control.

An outstanding fact is that the combined use of the different synchronization modes, drastically helps keeping the complexity of the discrete state space of the descriptions low [BST97]. Both *max*-synchronization and *min*-synchronization can be expressed in terms of *and*-synchronization but this requires additional states and transitions. Furthermore, this destroys compositionality, in the sense that timed specifications cannot be obtained from untimed specifications by preserving the control structure.

We believe that *max*-synchronization and *min*-synchronization are very powerful primitives for the specification of asynchronously cooperating timed systems. The use of *and*-synchronization is appropriate when a tight synchronization between the components is sought. The other two synchronization modes allow avoiding "clashes" in cooperation, for systems of loosely coupled components. For instance, *max*-synchronization corresponds to timed rendez-vous and

can be used to obtain in a straightforward manner, timed extensions of asynchronously communicating untimed systems.

The presented framework requires further validation by examples and practice. We are currently applying the flexible approach to the compositional generation of timed models of real-time applications and in particular, to scheduling.

References

[ACH+95] R. Alur, C. Courcoubetis, N. Halbwachs, T. Henzinger, P. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138:3–34, 1995.

[AD94] R. Alur and D.L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126:183–235, 1994.

[BBK86] J.C.M. Baeten, J.A. Bergstra, and J.W. Klop. Syntax and defining equations for an interrupt mechanism in process algebra. *Fundamenta Informaticae IX (2)*, pages 127–168, 1986.

[BGL97] P. Bremond-Gregoire and I. Lee. A process algebra of communicating shared resources with dense time and priorities. *Theoretical Computer Science, 189*, 1997.

[BS98] S. Bornot and J. Sifakis. On the composition of hybrid systems. In *First International Workshop Hybrid Systems : Computation and Control HSCC'98*, pages 49–63, Berkeley, March 1998. Lecture Notes in Computer Science 1386, Spinger-Verlag.

[BST97] S. Bornot, J. Sifakis, and S. Tripakis. Modeling urgency in timed systems. In *International Symposium: Compositionality - The Significant Difference*, Malente (Holstein, Germany), September 1997. Lecture Notes in Computer Science 1536, Springer Verlag.

[CH90] R. Cleaveland and M. Hennessy. Priorities in process algebra. *Information and Computation, 87(1/2)*, pages 58–77, 1990.

[JLSIR97] M. Jourdan, N. Layaida, L. Sabry-Ismail, and C. Roisin. An integrated authoring and presentation environment for interactive multimedia documents. In *4th Conference on Multimedia Modeling*, Singapore, November 1997. World Scientific Publishing.

[Mil83] R. Milner. Calculi for synchrony and asynchrony. *Theoretical Computer Science*, 25:267–310, 1983.

[Mil89] R. Milner. *Communication and Concurrency.* Prentice Hall, 1989.

[NS91] X. Nicollin and J. Sifakis. An Overview and Synthesis on Timed Process Algebras. In *Proceedings of CAV'91.* Aalborg, Denmark. LNCS 575, Springer Verlag, July 1991.

[SDdSS94] P. Sénac, M. Diaz, and P. de Saqui-Sannes. Toward a formal specification of multimedia scenarios. *Annals of telecomunications*, 49(5-6):297–314, 1994.

[SDLdSS96] P. Sènac, M. Diaz, A. Léger, and P. de Saqui-Sannes. Modeling logical and temporal synchronization in hypermedia systems. In *Journal on Selected Areas in Communications*, volume 14. IEEE, jan. 1996.

[SY96] J. Sifakis and S. Yovine. Compositional specification of timed systems. In *13th Annual Symposium on Theoretical Aspects of Computer Science, STACS'96*, pages 347–359, Grenoble, France, February 1996. Lecture Notes in Computer Science 1046, Spinger-Verlag.