Stochastic and Real Time in Process Algebra: A Conceptual Overview

Mario Bravetti

Dipartimento di Scienze dell'Informazione, Università di Bologna, Mura Anteo Zamboni 7, I-40127 Bologna, Italy. E-mail: bravetti@cs.unibo.it

Abstract

It is widely recognized that dealing with time related aspects in process algebra is often crucial for the specification and analysis of complex real systems. Research work in this field has led to a rather huge literature, where several kinds of time have been taken into account: time may be either based on a discrete or continuous domain, time elapsing may be either probabilistically (so-called stochastic-time) or deterministically (so-called real-time) bounded. In this paper we perform a conceptual dissertation about the treatment of the various kinds of time in transition systems where notions of composition are defined (as e.g. by defining a process algebra). We discuss general problems which are independent from the kind of time considered (concerning, e.g., the usual assumption of maximal progress of actions over time). Moreover, we show the conceptual relationship between the notion of time considered and the kind of semantics (in the sense of classical process algebra literature) which must be adopted for representing such a notion of time in the composition operators.

1 Introduction

In the last years, the necessity of extending the expressiveness of classical process algebras, so to make them more suitable for the specification and analysis of real case studies, has led to the definition of several timed calculi (see [7,2,3,1]. Such calculi differ, first of all for the "kind" of time they express. They may represent so-called real-time (see [9,7] and the references therein), where exact time bounds are specified and analysis typically consists of verification of exact time properties (e.g. via model checking). Alternatively, they may represent stochastic-time (see [3,2,1] and the references therein), where time is specified probabilistically via duration distributions and analysis is typically conducted via performance evaluation techniques. Stochastic-time approaches are further distinguished by the kind of probability distributions they can express: in most cases the limitation to exponential distribution is assumed, and we deal with so-called Markovian stochastic-time (see [2,1]), in other, more complex cases, it is possible to deal with any kind of dis-

This paper is electronically published in Electronic Notes in Theoretical Computer Science URL: www.elsevier.nl/locate/entcs

tribution (see [3] and the references therein). In the latter scenario, it is easy to see that the process algebra becomes so expressive that is capable to express real-time constraints as well. The "kind" of time considered may also differ in the domain of time values: discrete time or continuous time domains may be adopted for both the real-time and stochastic-time approaches. Secondly, timed extensions (e.g. referring to the same "kind" of time) may differ just for the technical solution adopted for introducing time passage (independently of the particular kind of time considered) in process algebra. For example a quite common technical design choice is to impose the so-called *maximal progress assumption* [9]: the possibility of executing internal transitions prevents the execution of timed transitions, thus expressing that the system cannot wait if it has something internal to do. Another example is the use of *clocks* to provide a symbolic finite representation of the system behavior even when the time domain is continuous. In general the choice of good technical solutions is critical to obtain a clean equational theory for the resulting algebra.

In this paper we perform a conceptual dissertation about the treatment of the various kinds of time in transition systems where notions of composition are defined (as e.g. by defining a process algebra). While doing this, we present a unifying process algebraic theory for most of the time extensions cited above. In particular, we present an approach which is based on standard weak bisimulation, which yields congruence for all the operators, and which makes it possible to obtain complete axiomatizations over finite-state processes. The idea is that, by providing a smooth modification of the standard machinery for observational congruence in classical process algebra, we obtain, for each kind of timed extension, the same kind of properties (e.g., congruence) and results (e.g., axiomatizations) as for classical process algebra.

2 Developing Timed Calculi

2.1 The Basic Calculus

We start from a *basic calculus*: the algebra of finite-state agents (made up of choice, prefix and recursion only) used by Milner in for axiomatizing observational congruence in presence of recursion, extended with δ prefixing, where δ actions have lower *priority* than internal τ actions. Such a calculus can be interpreted in this way [8]: δ actions represent "generic" time delays, classical actions of CCS are executed in zero time, and the priority of τ actions over δ actions derives from the maximal progress assumption. The presence of a priority mechanism makes the standard Milner's complete proof system for observational congruence no longer sound. In particular this happens for the axiom $recX.(\tau.X+P) = recX.\tau.P$ (a δ action performable by P is pre-empted in the left-hand term but not in the right-hand term) which makes it possible to equate τ divergent expressions to non-divergent ones so to remove unguarded recursion. In [4] we showed that it was possible to solve the long time open problem of axiomatizing priority using standard observational congruence by introducing an auxiliary operator $pri(P)$, by suitably modifying the axiom above and by introducing some new axioms. Our technique provides a complete axiomatization for Milner's observational congruence over finite-state terms of a process algebra with this simple kind of priority and recursion.

2.2 Full Calculi

In the following we will make use of the CSP parallel composition operator "$P \parallel_S Q$", where standard actions with type in S are required to synchronize, while the other standard actions are executed independently from P and Q. Such an operator will be used in combination with a hiding operator "P/L", which turns all the standard actions of P whose type is in L into τ and does not affect the other standard actions. The choice of using CSP parallel, instead of, e.g., CCS parallel, allows us to discuss separately the problems related to introducing maximal progress in: (i) parallel execution/synchronization of processes and (ii) dynamic generation of τ actions. This is done just for clarity reasons: we claim our discussions and results to be independent of the particular kind of (untimed) parallel operator considered.

Due to the maximal progress assumption, the generation of τ actions must cause all alternative δ actions to be pre-empted. In terms of the hiding operator, we have the following semantics for δ moves [1] :

$$\frac{P \xrightarrow{\delta} P' \qquad \not\exists a \in L. P \xrightarrow{a}}{P/L \xrightarrow{\delta} P'/L}$$

Note that, by using terms of our prioritized basic calculus as *normal forms* for extended processes, it is easy to axiomatize a full calculus with such a dynamic generation of prioritized τ actions: since priority is already captured inside the "+" operator, the set of axioms for the hiding operator is just the standard one that yields normal forms.

As far as the parallel operator is concerned, if we consider a purely *prioritized approach*, the basic calculus does not allow for an easy extension with parallel. In classical prioritized calculi the parallel operator is usually managed in two ways: either by implementing *local* pre-emption or *global* pre-emption (see [6]). For instance in $\tau.P \parallel_\emptyset \delta.Q$ the action δ of the right-hand process is not pre-empted by the action τ of the left-hand process, as instead happens if we assume global pre-emption. Even if assuming local pre-emption preserves congruence w.r.t. Milner's observational congruence, it causes the introduction of location information in the semantics and makes it really problematic to produce an axiomatization. If global pre-emption is, instead, assumed, then standard Milner's notion of observational congruence is not a congruence for the parallel operator (see [6]). More precisely, in $P \parallel_S Q$, the following rule is considered for δ moves of P [2] :

$$\frac{P \xrightarrow{\delta} P' \qquad Q \not\xrightarrow{\tau}}{P \parallel_S Q \xrightarrow{\delta} P' \parallel_S Q}$$

[1] If CCS would have been considered, than this pre-emption mechanism via a negative premise should have been captured in the rule for parallel (see [7]).

[2] A symmetric rule is considered for δ moves of Q.

which says that P may perform a δ action only if Q cannot execute any τ action. Observational congruence is not a congruence because, e.g., $\tau.\underline{0}$ is observationally congruent to $recX.\tau.X$, but $\tau.\underline{0}\,\|_{\emptyset}\,\delta.P$, whose semantics is that of $\tau.\delta.P$, is not observationally congruent to $recX.\tau.X\,\|_{\emptyset}\,\delta.P$, whose semantics, due to global pre-emption, is that of $recX.\tau.X$. In general note that the problem with congruence is related to the behavior of parallel for *processes Q which may initially execute neither "τ" prefixes, nor "δ" prefixes*, among which is $\underline{0}$ (for any such Q, the use of $\tau.Q \simeq recX.(\tau.X+Q)$ with the context "$\_\,\|_{\emptyset}\,\delta.P$" provides a counterexample to congruence). In this case a possibility is to resort to a finer notion of observational congruence (which is divergent sensitive in certain cases) similar to that presented in [8].

On the contrary, when *priority derives from time (maximal progress)*, i.e. when δ actions represent time delays and classical actions of CCS are executed in zero time, it is possible to develop a natural extension of the basic calculus which is compatible with standard weak bisimulation. In this context, adopting the operational rule above implementing global pre-emption (as done in [8]) does not seem to be the most natural choice. Conceptually, the problem with congruence derives from the fact that the parallel operator deals with the terminated process $\underline{0}$ (and in general with processes which may initially execute neither τ actions nor δ actions) as if it let time pass. For example $\underline{0}\,\|_{\emptyset}\,\delta$ may execute δ and become $\underline{0}\,\|_{\emptyset}\,\underline{0}$. This is obviously in contrast with the fact that $\underline{0}$ is weakly bisimilar to $recX.\tau.X$, which is clearly a process that does not let time pass (in the context of time it represents a Zeno process which executes infinite τ actions at the same time point): it produces a so-called *time deadlock*.

As a consequence, a very clean solution is to consider, as processes which can let time pass, only processes which can actually execute δ actions. In this way $\underline{0}$ is interpreted not as a terminated process which may let time pass, but as a time deadlock. Hence, the definition of the parallel operator changes: it must be defined, similarly as in [7], in such a way that the *absence* of δ actions within the actions executable by a process (which means that the process cannot let time pass) pre-empts the other process from executing a timed action δ. More precisely, in $P\,\|_S\,Q$, a rule like the following is considered for δ moves of P [3]:

$$\frac{P \xrightarrow{\delta} P' \qquad Q \xrightarrow{\delta} Q'}{P\,\|_S\,Q \xrightarrow{\delta} P'\,\|_S\,R}$$

(where the definition of R depends, as we will see, on the particular "kind" of time delays adopted) which says that P may perform a δ action only if Q may execute a δ action as well.

Pre-emption caused by the *absence* of δ actions differs from pre-emption caused by the *presence* of τ actions (see the global pre-emption rule) exactly for the class of processes that were misinterpreted, i.e. processes which cannot execute neither τ actions nor δ actions. The new interpretation of such processes (as in [7]) is that, consistently with weak bisimilarity, either they immediately execute a visible

[3] A symmetric rule is considered for δ moves of Q.

action or they cause a time deadlock. For instance "$\underline{0}$" is now correctly treated as representing a time deadlock from the parallel operator as well.

Therefore a solution to the problem of developing a natural extension of the basic calculus which is compatible with standard weak bisimulation is obtained by adopting the particular form of priority used in [7] which is neither local nor really global, but is *specialized for time*.

3 Kinds of Time Expressible via Atomic Transitions

The discrete real-time approach (see [7] and the references therein) and the Markovian
Stochastic-Time approach (see [2,1] and the references therein) are simple time models where delay prefixes are executed "atomically" in just one transition. In the discrete real-time approach elapsing of time (prefix δ) is typically represented by a special prefix "$\surd$", called "tick". Ticks take a fixed (unspecified) amount of time to be executed which is the same for all processes and are assumed to synchronize over all system processes ($R \equiv Q'$ in the rule above), e.g. the semantics of $\surd.\underline{0} \,\|_{\emptyset}\, \surd.\underline{0}$ is that of $\surd.\underline{0}$. In the Markovian Stochastic-Time approach elapsing of time (prefix δ) is represented by special prefixes "λ" $\in \mathbb{R}^+$ denoting time delays with a *probabilistic duration*.In particular the duration of a delay "λ" is assumed to have probability distribution given by a (continuous) exponential distribution with parameter "λ" (intuitively *the speed* of the delay). The restriction to exponential distributions (w.r.t. considering any kind of distribution) has the great advantage that parallel of delays can be correctly represented as their interleaving ($R \equiv Q$ in the rule above) due to the memoryless property of such a distribution, e.g. the semantics of $\lambda.\underline{0} \,\|_{\emptyset}\, \lambda'.\underline{0}$ is that of $\lambda.\lambda'.\underline{0} + \lambda'.\lambda.\underline{0}$.

4 Kinds of Time Expressible via Non-atomic Transitions

In order to represent more complex time models it is necessary to consider transition systems where delays are *not executed atomically* in a single transition, but *start* in a given state, evolve through several states, and *terminate* in another state. This is needed (see [3]), e.g., for expressing continuous real-time (see [9] and the references therein) and general stochastic-time (see [3] and the references therein). As far as continuous real-time is concerned, elapsing of time (prefix δ) can be represented by delay prefixes "D", where $D \subseteq \mathbb{R}^+$ is a *set of non-negative real numbers*: the possible durations for the delay. D can, e.g., be an interval or a set of intervals obtained via a set of constraints on the amount of time delay. In this context, since we are in a continuous time domain, it is important for system analysis to obtain semantic models based on *clocks* (like *timed automata*) where *time elapsing is not explicitly expressed via numerical time values*, but it is instead represented symbolically via start and termination of clocks. The same holds true for general stochastic-time. In this context elapsing of time (prefix δ) is represented by delay prefixes "f", where $f \in PDist(\mathbb{R}^+ \cup 0)$ is a *general probability distribution* over non-negative real numbers: it expresses the probabilistic duration of the delay. Since we consider general probability distributions on a continuous time domain, models based on *clocks* are necessary to obtain finite behavioral representations: they are like the standard Generalized Semi-Markov Processes where clocks are called "elements".

When dealing with non-atomic timing, it is necessary to represent the execution of time delays "δ" as the combination of the two events of *delay start* and *delay termination* in such a way that the termination of a given delay is *uniquely related* to its start. As we observed in [3], this corresponds to giving so-called ST semantics to time delay prefixes. Note that, as for standard action prefixes to which ST semantics is classically applied, even delay prefixes may be of different "observable" types (different sets D in the context of continuous real-time or different time probability distribution f in the context of general stochastic-time).

Technically, ST semantics can be expressed in several different ways in transition systems: no matter which technical solution is adopted ST bisimilarity (i.e. the pairs of processes which are bisimilar when ST semantics is considered) is always the same. In particular in [5] we have introduced three techniques for expressing ST semantics: a technique based on static names, a technique based on dynamic names and a "stack" technique. The first two techniques are based on the idea of expressing the relationship between start and terminations by assigning (statically or dynamically) unique names to prefixes of the same type, the third technique is, instead, based on pointers. Name techniques are particularly adequate in the context of time in that the name produced by giving ST semantics to time delays are like clock names in a *timed automaton*. More precisely, we get a transition system labeled with: standard action transitions, "δ_n^+" delay start transitions (where "n" is the name generated for the delay by the semantics) and "δ_n^-" delay termination transitions. Supposed that we are in a continuous real-time context, and δ is some $D \subseteq \mathbb{R}^+ \cup 0$, D is the type of the delay which is started/terminated by D_n^+/D_n^- and the *entire* "D_n" is the unique clock name for the delay (similarly for the general stochastic-time context). From a theoretical viewpoint, choosing the dynamic name technique of [5] to express the semantics of non-atomic time delays is particularly elegant. This because, as opposed to a technique based on static names, the dynamic name technique generates a *canonical name* "n" for every starting delay according to the order of execution of delays: the smallest natural number not currently in use by delays of the same type. This makes it possible to establish (weak) ST bisimulation simply by applying standard (weak) bisimulation over the "specialized" transition systems obtained. Thanks to this fact and to the compositionality of the technique (so-called *level-wise renaming* is exploited which recomputes canonical names at each structural level) a complete axiomatization for finite-state processes is obtained via a smooth extension of the standard machinery for axiomatizing observational congruence (see [5]). In particular every finite-state process is turned into a normal form: a process of the basic calculus which uses "δ_n^+" and "δ_n^-" prefixes instead of just "δ" prefixes.

References

[1] M. Bravetti, *"Revisiting Interactive Markov Chains"*, in Proc. of the *3rd Int. Workshop on Models for Time-Critical Systems (MTCS 2002)*, ENTCS 68(5), Brno (Czech Republic), August 2002

[2] M. Bravetti, M. Bernardo, *"Compositional Asymmetric Cooperations for Process Algebras with Probabilities, Priorities, and Time"*, in Proc. of the *1st Int. Workshop on Models for Time-Critical Systems (MTCS 2000)*, ENTCS 39(3), State College (PA), 2000

[3] M. Bravetti, R. Gorrieri, *"The Theory of Interactive Generalized Semi-Markov Processes"*, in Theoretical Computer Science, 282:5-32, 2002

[4] M. Bravetti, R. Gorrieri, *"A Complete Axiomatization for Observational Congruence of rioritized Finite-State Behaviors"*, in Proc. of the *27th Int. Colloquium on Automata, Languages and Programming (ICALP 2000)*, U. Montanari, J.D.P. Rolim and E. Welzl ed., LNCS 1853:744-755, Geneva (Switzerland), 2000

[5] M. Bravetti, R. Gorrieri, *"Deciding and Axiomatizing Weak ST Bisimulation for a Process Algebra with Recursion and Action Refinement"*, in ACM Transactions on Computational Logic 3(4):465-520, 2002

[6] R. Cleaveland, G. Luttgen, V. Natarajan, *"Priority in Process Algebras"*, in Handbook of Process Algebra, Chapter 12, pp. 711-765, Elsevier, 2001

[7] M. Hennessy, T. Regan, *"A Process Algebra for Timed Systems"*, in Information and Computation, 117(2):221-239, 1995

[8] H. Hermanns, M. Lohrey, *"Priority and Maximal Progress Are Completely Axiomatisable (Extended Abstract)"*, in Proc. of the *9th Int. Conf. on Concurrency Theory (CONCUR '98)*, LNCS 1466:237-252, Nice (France), 1998

[9] X. Nicollin, J. Sifakis, *"An Overview and Synthesis on Timed Process Algebras"*, in Real-Time: Theory in Practice, LNCS 600, 1991