2012 International Conference on Computer and Software Modeling (ICCSM 2012)
IPCSIT vol. 54 (2012) © (2012) IACSIT Press, Singapore
DOI: 10.7763/IPCSIT.2012.V54.08

Addressing State Space Explosion Problem in Performance Evaluation Using Maximality-based Labeled Stochastic Transition Systems

Mokdad AROUS [1+], Jean-Michel Ilié [2] and Djamel-Eddine Saïdouni [1]

[1] MISC Laboratory, Mentouri University, Constantine, 25000, Algeria.
[2] LIP6 Laboratory, Paris6 University, Paris, France.

Abstract. A new Stochastic Process Algebra called S-LOTOS is investigated, it extends LOTOS in order to specify the durations of actions in terms of generally distributed functions. We present its operational semantics and its underlying semantic model, called *Maximality-based Labeled Stochastic Transition System* (MLSTS). With regards to performance properties, we show that MLSTS and ST-semantics (*Start-Termination*) based models are equivalent, but the former brings more compact structure.

Keywords: Performance Evaluation, Maximality Semantics, ST-semantics, Stochastic Process Algebra, Stuttering Equivalence, (Stochastic) Labeled Transition System.

1. Introduction

The importance of studying the stochastic temporal behaviors of concurrent (stochastic) systems is widely recognized, e.g. [1, 12, 15]. In stochastic systems, the durations of actions are expressed probabilistically through probability distribution functions. Among the specification languages, the extension of the Process Algebras called Stochastic Process Algebras (SPAs) take advantage from its compositionality (model a system as the interaction of its components) and abstraction aspects (build up complex models from detailed components but disregarding internal behavior when it is appropriate to do so), whereas providing a formal description context.

Two main approaches have been adopted for expressing random time properties of stochastic systems. The first one considers exponential distributions for action durations. Markovian Process Algebras (MPAs) are SPAs where expressiveness is limited to exponential distributions for specifying the durations of actions, but are useful for many applications. MPA models accord with the interleaving semantics [1, 6, 9], such that the parallel execution of two actions *a* and *b* (see the expression *E* in Figure 1) is assumed to be equivalent to their interleaving execution (see the expression *F*). The semantic model of a MPA is a transition system wherein each transition is labeled with a pair (a, λ) representing the execution of the action *a* and the rate λ of the exponential distribution function governing the duration of *a*. Because of the memoryless property, exponential distributions yield analytically tractable models in the form of Continuous Time Markov Chains (CTMCs) [2, 16].

In the second approach, the durations of actions are specified by general distributions. In fact, exponential distribution appears to be not realistic in many concrete phenomena. It is only appropriate whether the mean values of random variables are known. This is not the case, for instance, when only the minimum and maximum values of the random variables are known.

+ Corresponding author. Tel.: (00213) 662 736 739
E-mail address : mokdad.a@gmail.com

Contrary to exponential laws, general distributions allow handling the residual durations of running actions. In Figure 1, it turns out that the two presented expressions are not behaviorally equivalent with regards to the temporal properties, although the same graph structure. This distinction is not possible under the memoryless property of the exponential distribution, since both models yield the same CTMC hence the same performance properties.

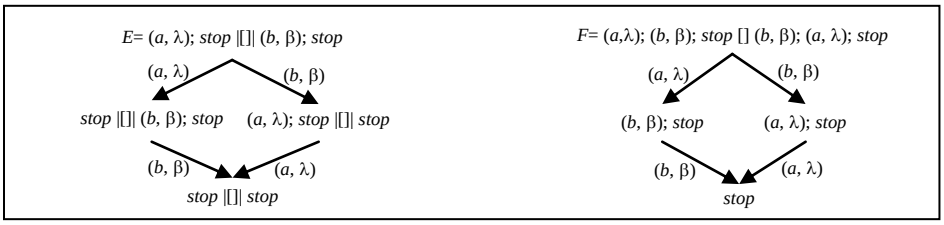


Fig. 1: Behaviors of *E* and *F* according to the interleaving semantics

For specifying generally distributed durations, the choice of true concurrency semantics appears to be more appropriate [7]. Actually, the system behaviors are never more represented like totally ordered sequences, but more adequately like partial order ones, allowing one to consider non-atomic actions. Therefore, many SPA papers adopt both, the general probability distributions and the true concurrency semantics, e.g. IGSMP (*Interactive Generalized Semi-Markov Process*) [10, 12], GSMPA (*Generalized Semi-Markovian Process Algebra*) [11, 12], SPADES (*Stochastic Process Algebra for Discrete Event Simulation*) [7, 14]. All approaches introduce an explicit representation (in the semantic models) of the start and end events of the running actions, hence accords with a specific true concurrency semantics called ST-Semantics (for *Start* and *Termination*). However, as a drawback, these approaches suffer from an increasing exponential blow up implied by the combinations of the start and termination events.

Alternatively, our approach [8] aims at handling true concurrency notions without being attacked by the state space explosion problem inherent to the splitting of actions. It is based on the true concurrency maximality semantics, proposed in [3, 4]. In this paper, we show how this approach is pragmatic and can solve the same performance properties as the ST-Semantics models, with a more compact structure.

The paper is organized as follows: In section 2, we present the principle of our modeling approach, namely S-LOTOS (for Stochastic LOTOS), and the semantic model of their expressions called Maximality-based Labeled Stochastic Transition System. The operational semantics of S-LOTOS is presented in Section 3, based on maximality principles. In Section 4, we show that our maximality semantics based model, is equivalent to the ST-semantic ones, up to some stuttering equivalence property, then our conclusion in Section 5 opens the discussion by comparing model conciseness and performance characteristics.

2. Principles of S-LOTOS

In this section, we introduce the main syntactical elements of S-LOTOS, and then we present its underlying maximality semantics. The reader is assumed to be familiar with the syntax of Basic LOTOS, a standard process algebra (PA), from which S-LOTOS derives. Let us first recall the Syntax of S-LOTOS. See [8] for details about the semantics of the different operators.

Considering some concurrent system, let *A* be the set of observable actions ranged over *a*, *b*, … and *L* denote any subset of *A*. The set of all actions that is finally considered, is denoted by *Act* (*Act* = $A \cup \{i, \delta\}$) where $\delta \notin A$ is a particular observable action used to notify the successful termination of processes, and *i* denotes any internal (unobservable) action. Introduce *DF* as the set of (continuous) probability distribution functions ($\Re \rightarrow [0, 1]$), ranged over *f*, *g*,… Then, define the set *B* ranged over *E*, *F*... of the behavior expressions that can specify the studied system, according to the following syntax of expressions:

$E ::= stop \mid exit \mid (a, f) ; E \mid (i, f) ; E \mid E [] E \mid E |[L]| E \mid hide\ L\ in\ E \mid E [b_1/a_1, ..., b_n/a_n]$

In S-LOTOS, stochastic time is handled by using arbitrary distribution functions. An action is represented by a pair (*a*, *f*), where *a* is the action name and *f* is the probability distribution function that governs the duration of *a*.

The semantic model of an S-LOTOS expression is a Labeled Transitions System (LTS), called Maximality-based Labeled Stochastic Transition System (MLSTS). Within this model, each transition only represents the start of an action execution. Since actions are not considered as atomic, the concurrent execution of multiple actions can be represented, and distinguishing between sequential and parallel executions is possible.

In the semantic model of S-LOTOS, the running actions are represented at the states level. Each instance of running actions is called a maximal event and is identified by a distinct name. In fact, each state of the system is featured by a unique configuration [3]. The configuration of a state s is denoted ${}_{M}[E]$ s.t. M is the set of maximal events in s and E is the behavior expression of s. In addition, a distinct clock is associated with each maximal event of M, to represent the time evolving. Every transition defined from s is labeled by ${}_{C}(a, f)_{x}$ whenever a is an action that can be activated from E iff. the maximal events of the subset $C \subseteq M$ are terminated. Further C is called the causality set of the transition. The symbol x is the name identifying the start event of the new execution of a. The event identification is required to avoid confusion since several instances of running actions can have the same action name.

To illustrate the principles of S-LOTOS and both concepts of maximality and configuration, consider the following two behavior expressions: E= (a,f); *stop* |[]| (b,g); *stop* and F= (a,f); (b,g); *stop* [] (b,g); (a,f); *stop*. Their respective MLSTSs, obtained by applying the maximality semantics, are represented in Figure 2. Initially, no action has yet been executed, then the set of maximal events is empty, and the initial configurations associated with E and F respectively, are ${}_{\varnothing}[E]$ and ${}_{\varnothing}[F]$. By assuming that the action a happens first from E and F, the corresponding transitions are respectively:

$$ {}_{\varnothing}[E] \xrightarrow{\varnothing(a,f)_x} {}_{\{x\}}[stop] \;|[]|\; {}_{\varnothing}[(b,g);\ stop] $$

$$ {}_{\varnothing}[F] \xrightarrow{\varnothing(a,f)_x} {}_{\{x\}}[(b,g);\ stop]. $$

x is the event name identifying the starting of a, and it represents a counting down clock which is initially set according to the distribution function f of the duration of a. In both new resulting configurations, x is said maximal.

From the new state of E (according to the semantic of the parallel operator []), the following transition occurs in case b starts: ${}_{\{x\}}[stop] \;|[]|\; {}_{\varnothing}[(b,g);\ stop] \xrightarrow{\varnothing(b,g)_y} {}_{\{x\}}[stop] \;|[]|\; {}_{\{y\}}[stop]$, where y is the maximal event name identifying the start of b, and it represents a clock set according to the distribution function g of the duration of b. In the resulting state, the clock of y starts counting down while the clock of x continues recording the time of a.

From the new state (${}_{\{x\}}[(b,g);\ stop]$) of F and the semantic of the prefix operator (;) expressing the sequentiality in execution, we deduce that the start of b is constrained by the causality dependence against x. Actually, it is submitted to the end of the execution of a, inducing that the clock of x has expired. This results in the following transition: ${}_{\{x\}}[(b,\ g);\ stop] \xrightarrow{\{x\}(b,g)_y} {}_{\{y\}}[stop]$, In the resulting state, the only maximal event is the one identified by y, representing the start of b, and the value of the associated clock is set according to the distribution function g of the duration of b.

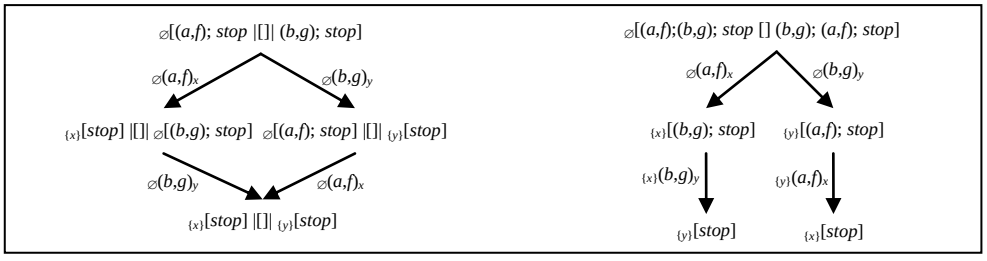


Fig. 2: Behaviors of E and F according to the maximality semantics

As this can be seen in Figure 2, the behaviors represented by the configurations ${}_{\{y\}}[stop]$ and ${}_{\{x\}}[stop]|[]|_{\{y\}}[stop]$ are rather different, in particular there is a single maximal event (identified by y) in the first one, whereas two maximal events appear in the second (identified by x and y). Observe that a symmetric scenario happens when the action b happens first.

3. Formal Aspects of S-Lotos

We briefly recall the definition of configurations [3], and then an operational semantics is defined for S-LOTOS to derive the possible transitions linking the configurations.

3.1. Configurations

Let $\mathcal{M}$ be the set of event names, ranged over *x*, *y* … Further, the notation $2^{\mathcal{X}}_{fn}$ represents the set of finite subsets of a set $\mathcal{X}$.

Definition 1. Configurations.

The set *C* of configurations is given by:

$\forall\ E \in \mathcal{B}, \forall\ M \in 2^{\mathcal{M}}_{fn}$: ${}_M[E] \in C$

$\forall\ P \in PN, \forall\ M \in 2^{\mathcal{M}}_{fn}$: ${}_M[P] \in C$

if $\varepsilon \in C$ then : *hide L in* $\varepsilon \in C$

if $\varepsilon, \mathcal{F} \in C$ then : $\varepsilon\ [\ ]\ \mathcal{F} \in C$, $\varepsilon\ |[L]|\ \mathcal{F} \in C$

if $\varepsilon \in C$ and $\{a_1, \ldots, a_n\}, \{b_1, \ldots, b_n\} \in 2^{A}_{fn}$, then: $\varepsilon[b_1/a_1, \ldots, b_n/a_n] \in C$ ♦

3.2. Derivation Rules

Let *S* be the set of states; transitions between states are also projected between configurations of these states. The transition relation between configurations is denoted $\rightarrow$. $\rightarrow \subseteq C \times Atm \times C$, where the set of atoms of support $(Act \times DF)$ is $Atm = 2^{\mathcal{M}}_{fn} \times (Act \times DF) \times \mathcal{M}$. For any subset of event names $M \in 2^{\mathcal{M}}_{fn}$, $(a,f) \in (Act \times DF)$ and $x \in \mathcal{M}$, the atom $(M, (a, f), x)$ will be denoted ${}_M(a,f)_x$. The choice of an event name can be realized deterministically by using any function *get*: $2^{\mathcal{M}} \setminus \{\emptyset\} \rightarrow \mathcal{M}$ satisfying $get(M) \in M$, for all $M \in 2^{\mathcal{M}} \setminus \{\emptyset\}$. The operational semantics of S-LOTOS is summarized in Table 1. Function ψ: $S \rightarrow 2^{\mathcal{M}}_{fn}$ is the function that associates every state with the finite set of its maximal events, and the predicate *Wait*: $2^{\mathcal{M}}_{fn} \rightarrow \{true, false\}$ characterizes the termination of maximal actions: $Wait(M) = true$ if there is at least one running action referenced in *M*.

Table. 1: Operational Semantics of S-LOTOS

$$\frac{\neg(wait(M))}{{}_M[stop] \xrightarrow{{}_M(\delta,0)_x} {}_\phi[exit]} \quad x = get(\mathcal{M})$$

$$\frac{\neg(wait(M))}{{}_M[(a,f);E] \xrightarrow{{}_M(a,f)_x} {}_{\{x\}}[E]} \quad x = get(\mathcal{M})$$

$$\frac{\neg(wait(M))}{{}_M[(i,f);E] \xrightarrow{{}_M(i,f)_x} {}_{\{x\}}[E]} \quad x = get(\mathcal{M})$$

$$\frac{\varepsilon \xrightarrow{{}_M(a,f)_x} \varepsilon' \quad a \notin L}{\mathcal{F}\,|[L]|\,\varepsilon \xrightarrow{{}_M(a,f)_x} \mathcal{F}\,|[L]|\,\varepsilon'}$$

$$\frac{\varepsilon \xrightarrow{{}_M(a,f)_x} \varepsilon' \quad a \notin L}{\varepsilon\,|[L]|\,\mathcal{F} \xrightarrow{{}_M(a,f)_x} \varepsilon'\,|[L]|\,\mathcal{F}}$$

$$\frac{\varepsilon \xrightarrow{{}_M(a,f)_x} \varepsilon' \quad \mathcal{F} \xrightarrow{{}_N(a,g)_y} \mathcal{F}' \quad a \in L}{\varepsilon\,|[L]|\,\mathcal{F} \xrightarrow{{}_{M \cup N}(a,h)_z} \varepsilon'[z/x]\,|[L]|\,\mathcal{F}'[z/y]} \quad z = get(\mathcal{M} - ((\psi(\varepsilon) \cup (\psi(\mathcal{F})))$$

$$\frac{\varepsilon \xrightarrow{{}_M(a,f)_x} \varepsilon'}{\mathcal{F}[\,]\,\varepsilon \xrightarrow{{}_M(a,f)_x} \varepsilon'}$$

$$\frac{\varepsilon \xrightarrow{{}_M(a,f)_x} \varepsilon'}{\varepsilon[\,]\,\mathcal{F} \xrightarrow{{}_M(a,f)_x} \varepsilon'}$$

$$\frac{\varepsilon \xrightarrow{{}_M(a,f)_x} \varepsilon' \quad a \notin \{a_1, \ldots, a_n\}}{\varepsilon[b_1/a_1, \ldots, b_n/a_n] \xrightarrow{{}_M(a,f)_x} \varepsilon'[b_1/a_1, \ldots, b_n/a_n]}$$

$$\frac{\varepsilon \xrightarrow{{}_M(a,f)_x} \varepsilon' \quad a = a_i \quad 1 \le i \le n}{\varepsilon[b_1/a_1, \ldots, b_n/a_n] \xrightarrow{{}_M(b_i,f)_x} \varepsilon'[b_1/a_1, \ldots, b_n/a_n]}$$

$$\frac{\varepsilon \xrightarrow{{}_M(a,f)_x} \varepsilon' \quad a \notin L}{hide\ L\ in\ \varepsilon \xrightarrow{{}_M(a,f)_x} hide\ L\ in\ \varepsilon'}$$

$$\frac{\varepsilon \xrightarrow{{}_M(a,f)_x} \varepsilon' \quad a \in L}{hide\ L\ in\ \varepsilon \xrightarrow{{}_M(i,f)_x} hide\ L\ in\ \varepsilon'}$$

4. Models based on Maximality Semantics vs. ST-Semantics

We briefly present the principles of some well-known ST-semantic models, and then show its equivalence with our performance model working under the maximality semantics.

4.1. ST-Semantics

The ST-semantics is used to describe the behavior of a concurrent system in terms of running actions, each one for a specific duration. Contrary to standard interleaving semantics, actions are not considered as atomic. The running of an action is split into two distinguished events, representing the start of the running and the corresponding end. Since the same actions can be launched several times concurrently (auto-concurrency), supplementary information is included in the semantic models to avoid confusion: maintain the correspondence between each start event of an action and its termination. In that purpose, two naming strategies were proposed for events.

The first one is based on static names, where names are defined at compiling time, according to their syntactical position (*left* or *right*) in the description of the initial parallel process specifying the whole system. The second strategy is based on dynamic names, that is: under a fixed rule, dynamically assign a different name to each new action that becomes active, assuming the names of the currently active actions are known.

Whatever the technique used to name the events, the ST-semantic model appears to be an LTS wherein states are labeled by pairs $\langle E, X\rangle$ mentioning for each state, the behavior expression E and the set X of activities under execution. Moreover, transitions between these states are of two types: start transitions labeled by a_x^+ where x is the name associated to the running action a, and termination transitions labeled by a_x^- where the name x determines exactly which action a is terminating.

4.2. Performance Equivalence

Regarding to any system specification P, we now prove that there is a stuttering equivalence between its maximality semantic model M_P and ST-semantic model ST_P, under the following hypothesis.

Hypothesis 1: The durations of transitions in semantic models are assumed to be null. ♦

The considered stuttering equivalence consists in aggregating the states of ST_P up to preserve its temporal properties. Such aggregations emerge from the observation of the start events, without regarding the end ones. From a performance view point, when focusing on ST_P, it appears that two successive states linked by a termination event are stutter, thus can be aggregated. Roughly speaking, when considering any transition $\langle E, X\rangle \xrightarrow{t} \langle E', X'\rangle$, s.t. t is a terminated event, we deduce that both source and target states preserve the same temporal properties because $E=E'$ and duration(t)=0 (according to Hypothesis 1). This adjacency relation can be extended to transitions sequence, hence the application of the stuttering equivalence on words, as follows: two words are stuttering equivalent if both can be partitioned into n blocks (having possibly different lengths), so that the states in the k^{th} block of one word are labeled the same as the states in the k^{th} block of the other word.

As an example, Figure 3-a presents the ST-semantic model corresponding to a system that concurrently runs two actions a and b, having f and g as respective duration distribution functions. The dashed lines bring out different groups of stutter states. A state aggregation w.r.t. these equivalences results in the graph of Figure 3-b, which can be obtained directly by considering the maximality semantics.

Considering the model ST_P of a concurrent system specification P, let *Start* and *End* be the two disjoint subsets of start and end events with regards to the behavior of P. Let L_P be the language generated by ST_P and $w=s_0s_1\ldots s_n$ be any word of L_P corresponding to the transition sequence $s_0 \rightarrow s_1 \ldots \rightarrow s_n$ enabled in P. The notation $w(i)$ and $w(last)$ respectively represent the i^{th} and the last state of w. Further, we consider the observational language L_P^{start} of L_P built from the observation of the start transitions as follows:

Definition 2: Observational language for ST-semantic model

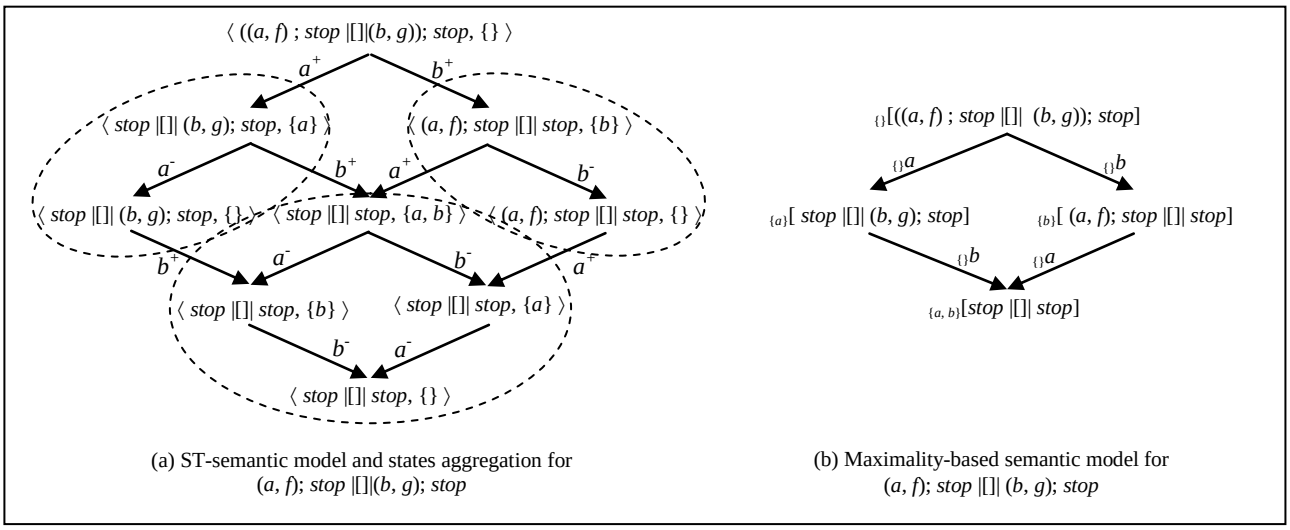
Let $w = s_0s_1\ldots s_n$ be a word in L_P, $\sigma=\sigma_0\sigma_1\ldots\sigma_m \in L_P^{start}$ is the observation of w w.r.t. *Start* iff. each σ_i corresponds to a sub-word of w, such that:

$$\sigma_0(1) = s_0$$
$$\sigma_i(j) \xrightarrow{t} \sigma_i(j+1) \qquad 1 \le j, 0 \le i \le n, t \in End$$
$$\sigma_i(last) \xrightarrow{d} \sigma_{i+1}(1) \qquad 0 \le i < n, d \in Start$$

♦

Lemma: Given a system specification P, let M_P and ST_P be the semantic models of P respectively based on the maximality and ST-semantics, then M_P and ST_P are stuttering equivalent with respect to the performance properties considered for P. ♦

We proceed the proof by induction on the length of transitions sequences of M_P and ST_P. A word $w = s_0 s_1 \ldots s_n$ is a sequence of (labeled) states such that there is a transitions sequence $s_0 \rightarrow s_1 \rightarrow s_2 \ldots \rightarrow s_n$ in the transition system. For sake of simplicity, we assimilate the performance properties that hold on some state to the state itself.



(a) ST-semantic model and states aggregation for (*a*, *f*); *stop* |[]|(*b*, *g*); *stop*

(b) Maximality-based semantic model for (*a*, *f*); *stop* |[]| (*b*, *g*); *stop*

Fig. 3: Example of ST-semantic model and the equivalent maximality-based semantic model.

Proof:

First, the property of stuttering equivalence straight fully holds, whether paths are reduced to their initial states, i.e. ${}_{\{\}}[P]$ in M_P and $\langle P, \{\}\rangle$ in ST_P.

Consider now that the following two words, $w_1 = s_0 s_1 \ldots s_n$ in ST_P and $w_2 = s'_0 s'_1 \ldots s'_m$ in M_P, are stuttering equivalent with respect to the performance properties of P. From the fact that w_1(*last*) and w_2 (*last*) are labeled by the same behavior expression, say E, we prove that the stuttering equivalence still holds after considering the next transitions in both models.

1) If the behavior expression E allows the execution of an action a with duration distributed by function f yielding to a behavior expression E': $E \xrightarrow{a} E'$, then we obtain in ST_P the transition: $s_n \xrightarrow{a_x^+} s_{n+1}$, and in M_P the transition : $s'_m \xrightarrow{\phi a_x} s'_{m+1}$

Both new states s_{n+1} and s'_{m+1} are labeled by the same behavior expression E'. Therefore, both new paths: $w_1' = s_0 s_1 \ldots s_n s_{n+1}$ and $w_2' = s'_0 s'_1 \ldots s'_m s'_{m+1}$ are stuttering equivalent, because we construct, in each path, a new block of one state labeled by the same behavior expression E', and executed action have same duration.

2) If some running action a terminates, this simply has an effect to retiring the corresponding event from the set of running events. We have the following transition in ST_P: $s_n \xrightarrow{a_x^-} s_{n+1}$, where state s_{n+1} is labeled by the behavior expression E. However, this transition is not represented explicitly in M_P. Hence, both paths w_1 and w_2 still stuttering equivalent, because we have not change their blocks.

Generally, If the behavior expression E have to terminate some actions b_i, $1 \leq i \leq k$, before the start event of an action a with duration distributed by function f yielding to a behavior expression E',

$$E \xrightarrow{b_1^-} E \xrightarrow{b_2^-} \ldots \xrightarrow{b_k^-} E \xrightarrow{a_x^+} E'$$

then we have in the ST_P the transitions:

$$s_n \xrightarrow{b_1^-} s_{n+1} \xrightarrow{b_2^-} \ldots \xrightarrow{b_k^-} s_{n+k} \xrightarrow{a_x^+} s_{n+k+1}$$

and in the M_P the transitions:

$$s'_m \xrightarrow{\{b_i, 1 \leq i \leq k\} a_x} s'_{m+1}$$

where the new states s_{n+i}, $1 \leq i \leq k$, are labeled by the behavior expression E (because, according to the ST-semantics, termination events does not alter the behavior expression), and the states s_{n+k+1} and s'_{m+1} are

both labeled by the behavior expression *E'*. Therefore, the new paths w_1'= $s_0 s_1\ s_n s_{n+1} \ldots s_{n+k+1}$ and w_2'= $s'_0 s'_1 \ldots s'_m s'_{m+1}$ stay stuttering equivalent because we increase the last block of w_1 by states s_{n+i}, $1 \leq i \leq k$, which are labeled by the same behavior expression than states in this block, and we construct a new block in each path of one state labeled by *E'*, and the duration added to total time of execution is the same in both models because termination event have null duration. ♦

Consequently, models M_P and ST_P for a specification expression *P* are stuttering equivalent, where blocks in a path of M_P are constructed each one of one state, and the corresponding blocks in ST_P are constructed according to the observational language on start transitions. The following transitions (1) in M_P and (2) in ST_P indicate the sequence of states of blocks constructed in both models. Termination transitions which are explicitly represented in ST_P (2) are abstracted in M_P. These transitions are in fact implicitly formalized in M_P through the notions of maximal events attached to states (i.e. a running action can terminate), nevertheless the transition can give the information about the termination of some running action if this last one belongs to the causality set *C* of the transition (1).

$$M[E] \xrightarrow{C\,a_x} M'[E'] \quad (1)$$

$$\langle E, M\rangle \xrightarrow{b_1^-} \langle E, M-\{b_1\}\rangle \xrightarrow{b_2^-} \ldots \langle E, M-C\rangle \xrightarrow{a^+} \langle E', M'\rangle \qquad C=\{b_1, b_2, b_3 \ldots\}, M' = M - C + \{x\} \quad (2)$$

In order to obtain its equivalent M_P, for some ST_P, we only observe the start transitions, thus aggregating states which are connected by termination transitions. In fact, the equivalence between points (1) and (2) defines the basis of the formal transformation from one model to the other.

4.3. Conclusion and Perspectives

The stochastic algebra named S-LOTOS extends existing LOTOS specification language by introducing general distribution functions to govern the action durations. By adopting the maximality-based semantics, we gave to S-LOTOS the ability to conform with the true concurrency paradigm of concurrent systems.

We showed that there is a stuttering equivalence between our Maximality-based Stochastic Labeled Transition System (MLSTS) and the existing ST-semantics based models. This allows one to re-use existing techniques and tools for performance evaluations with general distribution functions.

Our next perspective is to derive the performance properties from the MLSTS representations directly. Actually, the fact that these representations are less prone to the state space explosion problem due to the fact that the terminations of actions is no more explicit, made them attractive to deal with stochastic model checking problem.

5. References

[1] A. Clark, S. Gilmore, J. Hillston, and M. Tribastone, *Stochastic Process Algebras*, In. M. Bernardo and J. Hillston (Eds.): Formal Methods for Performance Evaluation, LNCS 4486, Springer-Verlag, 2007, pp. 132–179.

[2] B. R. Haverkort, *Markovian Models for Performance and Dependability Evaluation*, In. E. Brinksma, H. Hermanns, and J.-P. Katoen (Eds.): Lectures on Formal Methods and Performance Analysis, LNCS 2090, Springer-Verlag, 2001, pp. 38–83.

[3] D. E. Sa ïdouni and J. P. Courtiat, *Prise en compte des durées d'action dans les alg èbres de processus par l'utilisation de la sémantique de maximalité*, In. Proceedings of CFIP'2003. Hermes, France, 2003.

[4] D. E. Saidouni, *S émantique de maximalit é: Application au raffinement d'actions en LOTOS*, PhD thesis, LAAS-CNRS, 7 av. du Colonel Roche, 31077 Toulouse Cedex France, 1996.

[5] F. K. Dankar, *Approaches to Analysis and Simplification of non-Markovian System Models*, PhD. Thesis, School of Information Technology and Engineering (SITE), Faculty of Engineering, University of Ottawa, Canada, 2008.

[6] J. Hillston, *Process Algebras for Quantitative Analysis*, Proceedings of the 20th Annual IEEE Symposium on Logic in Computer Science (LICS'05), 2005, pp. 239-248.

[7] J.-P. Katoen and P.R, D'Argenio, *General Distributions in Process Algebra*, In. E. Brinksma, H. Hermanns, and J.-P. Katoen (Eds.): Lectures on Formal Methods and Performance Analysis, LNCS 2090, Springer-Verlag, 2001, pp. 375–429.

[8] M. Arous, D. E. Saidouni and J. M. Ilié, *Maximality Semantics based Stochastic Process Algebra for Performance Evaluation*. 1st IEEE International Conference on Communications, Computing and Control Applications (CCCA'11) March 3-5, 2011 at Hammamet, Tunisia. IEEE Catalog Number: CFP1154M-ART, ISBN: 978-1-4244-9796-6.

[9] M. Bernardo, *Theory and Application of Extended Markovian Process Algebra*, Ph.D. Thesis, University of Bologna (Italy), 1999.

[10] M. Bravetti and R. Gorrieri, *The theory of interactive generalized semi-Markov processes*, In. Theoretical Computer Science 282, 2002, pp. 5–32.

[11] M. Bravetti, M. Bernardo, and R. Gorrieri, *Towards Performance Evaluation with General Distributions in Process Algebra*, In. D. Sangiorgi and R. de Simone (Eds.): LNCS 1466, Proceedings of CONCUR'98, Nice (France), Springer, 1998, pp. 405-422.

[12] M. Bravetti, *Specification and Analysis of Stochastic Real-time Systems*, PhD thesis, Università di Bologna, Padova, Venezia, 2002.

[13] M. Kwiatkowska, G. Norman and A. Pacheco. *Model Checking CSL Until Formulae with Random Time Bounds*, In H. Hermanns and R. Segala (Eds.): 2nd Joint Int. Workshop on Process Algebra and Performance Modelling and Probabilistic Methods in Verification, LNCS 2399, Springer. July 2002, pp. 152–168.

[14] P. R. D'Argenio and J.-P. Katoen, *A theory of stochastic systems. Part II: Process algebra*, In. Information and Computation 203, Elsevier Inc, 2005, pp. 39–74.

[15] U. Herzog, *Formal Methods for Performance Evaluation*, In. E. Brinksma, H. Hermanns, and J.-P. Katoen (Eds.): Lectures on Formal Methods and Performance Analysis, LNCS 2090, Springer-Verlag, 2001, pp. 01–37.

[16] W. J. Stewart, *Performance Modelling and Markov Chains*, In. M. Bernardo and J. Hillston (Eds.): Formal Methods for Performance Evaluation, LNCS 4486, Springer-Verlag, 2007, pp. 1–33.