From Generalised Stochastic Petri Nets to Markov Automata and MAPA specifications

Rob Bamberg and Mark Timmer

June 26, 2012

This article formalises the semantics of Generalised Stochastic Petri Nets (GSPNs) in terms of Markov Automata (MAs). Additionally, we provide a translation of GSPNs to specifications in the MAPA language. Since both are symbolic representations, this translation is fast and the resulting specifications are comparable smaller in size to the GSPN specifications. As MAPA also has semantics in terms of MAs, we can first translate a GSPN model to a MAPA specification and then generate the underlying MA. We prove that this MA corresponds to the MA representing the semantics of the original GSPN.

Although most GSPN representations require all immediate transitions to be accompanied by a weight, we are more liberal and consider the weights optional. Hence, this allows GSPN models in which several unweighted immediate transitions may fire from the same state. This introduces non-determinism, and hence motivates the choice to lift the semantics from MAs to CTMCs.

1 Preliminaries

1.1 Markov Automata

An MA is a transition system in which the set of transitions is partitioned into interactive transitions (which are equivalent to the transitions of a PA) and Markovian transitions (which are equivalent to the transitions of an IMC). The following definition formalises this, and provides notations for MAs. We assume a countable universe *Act* of actions, with $\tau \in Act$ the invisible internal action.

Definition 1 (MAs)**.** *A* Markov automaton (MA) *is a tuple* $M = \langle S, s^0, A, \hookrightarrow, \rightsquigarrow \rangle$*, where*

- S *is a countable set of* states
- $s^0 \in S$ *is the* initial state*;*
- $A \subseteq Act$ *is a countable set of* actions*;*
- $\hookrightarrow \subseteq S \times A \times \mathsf{Distr}(S)$ *is the* interactive transition relation*;*
- $\rightsquigarrow \subseteq S \times \mathbb{R}_{>0} \times S$ *is the* Markovian transition relation*.*

If $(s, a, \mu) \in \hookrightarrow$*, we write* $s \xhookrightarrow{a} \mu$ *and say that the action* a *can be* executed *from state* s*, after which the probability to go to* $s' \in S$ *is* $\mu(s')$*. If* $(s, \lambda, s') \in \rightsquigarrow$*, we write* $s \overset{\lambda}{\rightsquigarrow} s'$ *and say that* s *moves to* s' *with rate* λ*.*

1.2 Generalised Stochastic Petri Nets

GSPNs are among the most general variants of Petri Nets. Like other types of Petri Nets, they consist of *places* and *transitions*. The places can contain *tokens*, and the transitions move tokens between places. Each state of a GSPN is therefore given as a *marking*, i.e., a function that assigns a natural number to each place (the number of tokens is contains).

A marking m can evolve into another marking m' if there is a transition to make this happen. This can happen either immediately (by an *immediate transition*) or delayed (by a *rate transition*). The immediate transitions of a GSPN are assigned a priority, as well as a weight (although both are optional). If in a given state several immediate transitions are enabled, only the ones with the highest priority may fire. Since rate transitions are defined to have a priority of 0, and immediate transitions have to be assigned a priority higher than zero, this implies that immediate transition always take precedence over rate transitions (which is often called the *maximal progress property*).

If several immediate transitions with the same priority are enabled at the same time, any one of them can be chosen non-deterministically. However, if some of them have a weight assigned, those are merged into one transition that has a probabilistic outcome (based on the weights).

Definition 2 (GSPNs). *A GSPN is a 10-tuple* $G = (P, m_0, \mathit{IMT}, \mathit{PRI}, W, \mathit{RT}, R, I, F, \mathit{MU})$*, where*

- P *is a finite set of* places*;*
- $m_0 \colon P \to \mathbb{N}$ *is the* initial marking
- IMT *is a finite set of* immediate transitions*;*
- $\mathit{PRI} \colon \mathit{IMT} \to \mathbb{N}^{>0}$ *is the* priority function*;*
- $W \colon \mathit{IMT} \to \mathbb{R} \cup \{\bot\}$ *is the* weight function*;*
- RT *is a finite set of* rate transitions*;*
- $R \colon \mathit{RT} \to \mathbb{R}^{>0}$ *is the* rate function
- $I \colon P \times T$ *is a finite set of* inhibitor arcs
- $F \colon (P \times T) \cup (T \times P)$ *is a finite set of* regular arcs
- $\mathit{MU} \colon F \times \mathbb{N}^{>0}$ *is the* multiplicity function

where we use $T = \mathit{IMT} \cup \mathit{RT}$ *to denote all transitions.*

The set of all markings of a GSPN is defined as $\mathcal{M} = \{m \colon P \to \mathbb{N}\}$. For each immediate transition t, $\mathit{PRI}(t)$ provides its priority and $W(t)$ its weight. We use $W(i) = \bot$ to denote that t does not have a weight assigned. For each rate transition t, $R(t)$ provides its Markovian rate.

To give the precise semantics of G in terms of an MA, we first introduce some additional notation to deal with arcs and explain when transitions can be fired.

Definition 3 (Notations for arcs). *Let* $G = (P, M_0, \mathit{IMT}, \mathit{PRI}, W, \mathit{RT}, R, I, F, \mathit{MU})$ *be a GSPN. Then, given an arc* $a \in F \cup I$*, we write* $\mathit{src}(a)$ *for its first element and* $\mathit{target}(a)$ *for its second. Given a transition* $t \in T$*, we denote by* $\mathit{in}(t) = \{a \in F \mid \mathit{target}(a) = t\}$ *the set of all its incoming regular arcs. Similarly, we define* $\mathit{out}(t) = \{a \in F \mid \mathit{src}(a) = t\}$ *and* $\mathit{inhib}(t) = \{a \in I \mid \mathit{target}(t) = a\}$*.*

Note that for some arcs $\mathit{src}(a) \in P$ and $\mathit{target}(a) \in T$, while for others $\mathit{src}(a) \in T$ and $\mathit{target}(a) \in P$.

A transition can *fire* if all places associated with its incoming arcs have enough tokens (as indicated by the multiplicity function) and all places associated with its inhibitor arcs are empty. However, in the presence of priorities some transitions that in principle could fire are disabled nevertheless. If several transitions are enabled, only the one(s) with the highest priorities are actually enabled.

Definition 4 (Enabling). *Given a GSPN* $G = (P, M_0, \mathit{IMT}, \mathit{PRI}, W, \mathit{RT}, R, I, F, \mathit{MU})$ *and a markings* $m \in \mathcal{M}$*, the set of transitions that are basically enabled from* m *is given by*

$$\mathit{en}_{\mathit{basic}}(m) = \{t \in T \mid \forall a \in \mathit{in}(t) \,.\, m(\mathit{src}(a)) \geq \mathit{MU}(a) \wedge \forall a \in \mathit{inhib}(t) \,.\, m(\mathit{src}(a)) = 0\}.$$

The set of transitions that are actually enabled from m *is given by*

$$\mathit{en}(m) = \{t \in \mathit{en}_{\mathit{basic}}(m) \mid \forall t' \in \mathit{en}_{\mathit{basic}}(m) \,.\, \mathit{PRI}(t) \geq \mathit{PRI}(t')\}$$

If a transition is enabled, it can fire. In that case, the required tokens from the places associated with its incoming arcs are removed, and the places associated with its outgoing arcs gain additional tokens (as indicated by the multiplicity function). Hence, the transition moves the GSPN from one marking to another.

Definition 5 (Firings)**.** *Given a GSPN* $G = (P, M_0, \mathit{IMT}, \mathit{PRI}, W, \mathit{RT}, R, I, F, \mathit{MU})$ *and two markings* $m, m' \in \mathcal{M}$*, there is a firing from* m *to* m' *by transition* $t \in T$*, denoted* $m \xrightarrow{t} m'$*, if* $t \in \mathit{en}(m)$ *and*

$$\begin{aligned} &\forall a \in \mathit{in}(t) \,.\, m'(\mathit{src}(a)) = m(\mathit{src}(a)) - \mathit{MU}(a) \,\wedge \\ &\forall a \in \mathit{out}(t) \,.\, m'(\mathit{target}(a)) = m(\mathit{target}(a)) + \mathit{MU}(a) \end{aligned}$$

We define $\xrightarrow{t}^*$ *as the reflexive and transitive closure of the relation* $\xrightarrow{t}$*.*

1.3 Markov Automata Process Algebra

Markov Automata Process Algebra (MAPA) is a process-algebraic specification language in which all conditions, non-deterministic and probabilistic choices, and Markovian delays may depend on data parameters. It requires an external mechanism for the evaluation of expressions (e.g., equational logic, or a fixed data language), able to handle at least boolean and real-valued expressions. Here, we use an intuitive data language, containing basic arithmetic and boolean operators. We generally refer to data types with upper-case letters $D, E, \ldots$, and to variables with lower-case letters $u, v, \ldots$.

Definition 6 (Process terms)**.** *A* process term *in MAPA is any term that can be generated by the following grammar:*

$$p ::= Y(\boldsymbol{t}) \;\mid\; c \Rightarrow p \;\mid\; p + p \;\mid\; \textstyle\sum_{\boldsymbol{x}:\boldsymbol{D}} p \;\mid\; a(\boldsymbol{t}) \boldsymbol{\sum}_{\boldsymbol{x}:\boldsymbol{D}} f : p \;\mid\; (\lambda) \cdot p$$

Here, Y is a process name, $\boldsymbol{t}$ a vector of expressions, c a boolean expression, $\boldsymbol{x}$ a vector of variables ranging over a (possibly infinite) type $\boldsymbol{D}$, $a \in \mathit{Act}$ a (parameterised) atomic action, f a real-valued expression yielding values in $[0, 1]$, and λ an expression yielding positive real numbers (rates). We write $p = p'$ for syntactically identical process terms. Note that, if $|\boldsymbol{x}| > 1$, $\boldsymbol{D}$ is a Cartesian product, as for instance in $\sum_{(m,i):\{m_1,m_2\}\times\{1,2,3\}} \mathrm{send}(m, i) \ldots$.

Given an expression t, a process terms p and two vectors $\boldsymbol{x} = (x_1, \ldots, x_n)$, $\boldsymbol{d} = (d_1, \ldots, d_n)$, we use $t[\boldsymbol{x} := \boldsymbol{d}]$ to denote the result of substituting every x_i in t by d_i, and $p[\boldsymbol{x} := \boldsymbol{d}]$ for the result of applying this to every expression in p.

Definition 7 (Specifications)**.** *A* MAPA specification *is given by a tuple* $M = (\{X_i(\boldsymbol{x_i} : \boldsymbol{D_i}) = p_i\}, X_j(\boldsymbol{t}))$ *consisting of a set of uniquely-named* processes X_i*, each defined by a* process equation $X_i(\boldsymbol{x_i} : \boldsymbol{D_i}) = p_i$*, and an* initial process $X_j(\boldsymbol{t})$*. In a process equation,* $\boldsymbol{x_i}$ *is a vector of process variables with type* $\boldsymbol{D_i}$*, and* p_i *(the* right-hand side*) is a* process term *specifying the behaviour of* X_i*. A variable* v *in an expression in a right-hand side* p_i *is* bound *if it is an element of* $\boldsymbol{x_i}$ *or it occurs within a construct* $\sum_{\boldsymbol{x}:\boldsymbol{D}}$ *or* $\boldsymbol{\sum}_{\boldsymbol{x}:\boldsymbol{D}}$ *such that* v *is an element of* $\boldsymbol{x}$*. Variables that are not bound are said to be* free*.*

In a process term, $Y(\boldsymbol{t})$ denotes *process instantiation*, where $\boldsymbol{t}$ instantiates Y's process variables (allowing recursion). The term $c \Rightarrow p$ behaves as p if the *condition* c holds, and cannot do anything otherwise. The $+$ operator denotes *nondeterministic choice*, and $\sum_{\boldsymbol{x}:\boldsymbol{D}} p$ a (possibly infinite) *nondeterministic choice over data type* $\boldsymbol{D}$. The term $a(\boldsymbol{t}) \boldsymbol{\sum}_{\boldsymbol{x}:\boldsymbol{D}} f : p$ performs the action $a(\boldsymbol{t})$ and then does a *probabilistic choice* over $\boldsymbol{D}$. It uses the value $f[\boldsymbol{x} := \boldsymbol{d}]$ as the probability of choosing each $\boldsymbol{d} \in \boldsymbol{D}$. Finally, $(\lambda) \cdot p$ can behave as p after a delay, determined by a negative exponential distribution with rate λ.

We generally refer to process terms with lower-case letters p, q, r, and to processes with capitals X, Y, Z. Also, we will often write $X(x_1 : D_1, \ldots, x_n : D_n)$ for $X((x_1, \ldots, x_n) : (D_1 \times \cdots \times D_n))$. As syntactic sugar, we write $a(\boldsymbol{t}) \cdot p$ for the action $a(\boldsymbol{t})$ that goes to p with probability 1.

2 GSPN semantics as MA

We are now ready to define the semantics of a GSPN in terms of an MA. As stated earlier, transitions without a weight can fire non-deterministically. The transitions with weights are merged together in a single τ transition. The next state of this transition is probabilistically distributed according to the weights of the constituent transitions.

Definition 8 (GSPN semantics)**.** *Given a GSPN $G = (P, M_0, IMT, PRI, W, RT, R, I, F, MU)$, its semantics is given by the MA $M = (S, m_0, A, \hookrightarrow, \rightsquigarrow)$, where*

- $S = \{m' \in \mathcal{M} \mid m_0 \xrightarrow{t}{}^* m'\}$
- $A = \{\tau\}$
- $\hookrightarrow$ *is such that, for every $m, m' \in S, \mu \in \mathsf{Distr}(S)$,*

$$\begin{aligned} m \xhookrightarrow{\tau} \mathbb{1}_{m'} \quad &\iff \quad \exists t \in IMT \,.\, W(t) = \bot \land m \xrightarrow{t} m' \\ m \xhookrightarrow{\tau} \mu \quad &\iff \quad \exists t \in IMT \,.\, W(t) > 0 \land \\ &\qquad \forall m' \in S \,.\, \mu(m') = \frac{\sum \{\!|W(t) \mid t \in IMT \land W(t) > 0 \land m \xrightarrow{t} m'|\!\}}{\sum \{\!|W(t) \mid t \in IMT \land W(t) > 0 \land \exists m'' \in S \,.\, m \xrightarrow{t} m''|\!\}} \end{aligned}$$

- $\rightsquigarrow$ *is such that, for every $m, m' \in S$,*

$$m \overset{\lambda}{\rightsquigarrow} m' \quad \iff \quad \lambda = \sum \{\!|R(t) \mid t \in RT \land m \xrightarrow{t} m'|\!\} > 0$$

Note that the rate from a marking m to a marking m' is determined by the sum of all enabled rate transitions from m to m'.

3 Translation from GSPN to MAPA

We are now ready to specify a GSPN in terms of MAPA. The specification will consist of one process which will have the transitions as process terms. This process will have a variable for each place. A valuation of these variables corresponds to a marking of the GSPN. The initial marking is thus the initial state of the process.

Definition 9 (GSPN semantics)**.** *Given a GSPN $G = (P, M_0, IMT, PRI, W, RT, R, I, F, MU)$, with $P = p_0, p_1, \ldots, p_n$. Then the corresponding MAPA specification is a process $X(p_1 : \mathbb{N}, p_2 : \mathbb{N}, \ldots, p_n : \mathbb{N})$, with initial state $X(M_0(p_1), M_0(p_2), \ldots, M_0(p_n))$. The behaviour of X is given by a set of summands based on the transitions of G. More precisely, it is the smallest process such that:*

- *For each rate transition $t \in RT$, X has a summand*

$$cr_t \Rightarrow R(t) \cdot X(n_t)$$

where cr_t is the conjunction of all terms $\{src(a) \geq MU(a) \mid a \in in(t)\} \cup \{src(a) = 0 \mid a \in inhib(t)\}$. The next state n_t consists of all terms in $\{src(a) = src(a) - MU(a) \mid a \in in(t)\} \cup \{target(a) = target(a) + MU(a) \mid a \in out(t)\}$.

- *For each immediate transition $t \in IMT$ with $W(t) = \bot$, X has a summand*

$$ci_t \Rightarrow \tau \cdot X(n_t)$$

where ci_t is the conjunction of cr_t (as defined above) and all terms in $\{\neg(cr_{t'}) \mid t' \in IMT \land PRI(t') > PRI(t)\}$, and n_t is as defined above. This way, unweighted transitions are only enabled if none of the transitions with a higher priority are also enabled. Note that this is not needed for rate transitions, due to the maximal progress property.

- *Let* $t_p = \{t \mid t \in IMT \wedge W(t) > 0 \wedge PRI(t) = p\}$ *be the set of weighted transition with priority* p*. For each priority* p *such that* $|t_p| > 0$*,* X *has a summand*

$$cs_p \Rightarrow \tau \cdot \sum_{t:t_p} \frac{\text{if } ci_t \text{ then } W(t) \text{ else } 0}{totalWeight} : X(n_t)$$

where n_t *is as defined above, and* cs_p *is the disjunction of all terms in* $\{ci_t \mid t \in t_p\}$*, with* ci_t *also as defined above. Note that each transition only has a nonzero probability if it is enabled. Hence, the total weight is also defined conditionally, by*

$$totalWeight = \sum_{t \in t_p} (\text{if } ci_t \text{ then } W(t) \text{ else } 0)$$