Probe Automata for Passage Time Specification

Elvio Gilberto Amparore, Marco Beccuti, Susanna Donatelli, Giuliana Franceschinis

1 Dipartimento di Informatica, Università di Torino, Italy
2 Dipartimento di Informatica, Università del Piemonte Orientale, Italy
{amparore, beccuti, susi}@di.unito.it, giuliana.franceschinis@di.unipmn.it

Abstract

Passage time distribution has drawn increasing attention over the past years as an important measure to define and verify service level agreements.

The definition of passage time requires the specification of a condition to start/stop the computation, and possibly a restriction on the system behavior to be considered between start and stop. Different characterizations have been defined in the past, either state-based, action-based or a mix of the two, either for Markov chains, or for stochastic Petri nets and process algebras.

In this paper we propose probe automata as a way to specify passage time for GSPNs that allows one to select entering, goal, and forbidden states, as well as paths of interest starting from any reachable state. The specification is in terms of conditions over the current marking, the transition (sequence) being fired, as well as over the marking reached through the firing. Probe automata subsume previous definitions of passage time for GSPNs and for Tagged GSPNs, the extension of GSPNs that was defined in the past for computing passage time of a tagged token in a GSPN.

Keywords: passage time, Markov chains, GSPN.

1. Introduction

Performance analysis of systems often concerns the ability to react in due time to external events. In service oriented systems a measure of reactivity is the response time as a function of the workload intensity or as a function of the operational conditions of the system, that may operate in degraded mode due to the occurrence of failures. In safety critical systems a measure of interest is the time required to reach a safe state after having detected a dangerous event. In many cases the mean of the time to be measured is not adequate: quantile estimates are more appropriate since performance indicators like “90% of the requests obtain an answer within $n$ time units” are typically used when the goal of the study is to check if the system achieves a given safety level or customer satisfaction, or a condition mentioned in a Service Level Agreement.

When the system is described by a Continuous Time Markov Chain (CTMC) the appropriate technique to estimate these measures is first passage time analysis. Passage time definition requires the identification of three sets of states, namely the entering states $E$, the goal states $G$ and the forbidden states $D$. Given an initial distribution $\pi(x), x \in E$ on the entering states, the computation of the Passage Time results in the Cumulative Distribution Function (CDF) of the time required to reach any of the goal states from the entering states without passing through any forbidden state: in practice this can be achieved by making both goal and forbidden states absorbing, and computing the probability of being in such states at any time $t$.

When the system under study is described using a higher level formalism from which a CTMC can be automatically derived, like Queueing Networks, Stochastic Petri Nets, or Stochastic Process Algebras, it is desirable to have a language to express passage time in terms of the elements of the high level formalism. Example of such a measure is the response time distribution in queueing networks [14] and in TGSPN [5], where the high level elements involved are a tagged customer (or token under observation) and a subnet: the problem is to compute the distribution of the time for the tagged customer/token to traverse the subnet.

Examples of languages for path-based measures and passage time specifications for Petri Nets and Process Algebras are Path Automata (PA) [15] and eXtended Stochastic Probes (XSP) [8] for PEPA [12]. PA have been defined as a way to define path-based reward variables. PA automata describe paths of interest as the event sequences accepted by an automaton defined over the state-transition graph of a Stochastic Activity Network (SAN) [16]. Measures are computed over a product state space of the SAN with the PA automaton. XSP have been developed for the stochastic process algebra PEPA, to define passage time measures. An XSP probe can be interpreted as an external observer,
encoding the rules that allow one to decide when a passage time must start and stop. XSP translate to PEPA terms, thus allowing the use of PEPA derivation rules to build the state space on which the passage time is computed.

The goal that motivated our research is to define a language for passage time specification for Generalized Stochastic Petri Nets (GSPNs) [1] that can extend the language of passage time specification in Tagged GSPNs (TGSPNs), an extension of GSPNs to deal with tagged customers. As discussed in [8] it is necessary to have a language integrating both state based and activity based conditions to be able to characterize the system behavior of interest without putting the burden of adapting the model to support the measure on the modeler.

In this paper we define Probe Automata (PrA) as a way to specify passage time measures. The proposal is inspired by XSP and other approaches that use an automata to specify performance measures (like PA) and to specify path formulas in stochastic logics (like CSLTA [11]).

PrA allow one to specify entering, goal and forbidden states in terms of conditions over markings, but also in terms of the paths connecting those markings. Paths are characterized in terms of GSPN extended firings, with pre-conditions for the transition to be considered (boolean function over the marking enabling it), but also with post-conditions (boolean function over the marking reached by the firing of the transition). This characteristic is particularly useful for GSPNs because the transition from one tangible state to its successors, involving one start or stop activity, may include the firing of a possibly long sequence of immediate transitions, hence potentially causing a state change that is not easy to predict. This is particularly relevant for the computation of the tagged token passage time in tagged GSPNs, since it can be used to require that, in presence of a start transition, the tagged token be out of the considered subnet before the firing, and into the subnet after the firing (vice versa for stop transitions). XSP and PA only allow preconditions.

A significant difference of PrA and of the associated Passage Time definition is that the paths specification is matched against the whole behavior of the model, and it is not specifically tied to the model initial state, as it is the case for XSP probes and PA automata. As we shall see in the paper, this difference is not a minor one.

Finally, while it has been developed for (tagged) GSPNs, the specification language and analysis method developed in this paper could be applied to any formalism whose semantics is provided by a labeled CTMC, therefore also to SAN and PEPA.

The paper is organized as follows. Section 2 introduces the background: the (T)GSPN formalisms, the passage time definition and a flexible manufacturing system example. A number of passage time properties are introduced that motivate the definition of PrA in Section 3. The computation of passage time distributions is addressed in Section 4, and the differences with the computation defined for XSP and PA are discussed in Section 5. Finally Section 6 summarizes the paper contribution and discusses future research lines for the PrA specification and (T)GSPNs.

2. Background

In this section we review some of the background material useful for the PrA definition. We start with a formal definition of passage time in the context of CTMCs, to then recall the definition of GSPNs and of TGSPNs, the extension of GSPNs to deal with tagged customers.

Passage time definition. The First Passage Time measure, passage time for short, for CTMCs is defined using three sets of states: entering states \( E \), goal states \( G \) and forbidden states \( D \). Given an initial distribution \( \pi(x), x \in E \), the passage time distribution is a CDF defined as follows [2]:

\[
\text{Passage}(E, G, D, t) = \sum_{x \in E} \pi(x) \cdot \text{Prob}(P_{x,G,D}^{E,G,D} \leq t)
\]

where \( P_{x,G,D}^{E,G,D} \) is the random variable expressing the passage time from state \( x \) and terminating in one of the states in \( G \), and without passing through \( D \) states:

\[
P_{x,G,D}^{E,G,D} = \min\{u : Z(u) \in G, N(u) > 0, Z(0) = x
\]

and \( \forall v \leq u, Z(v) \notin D \}

with \( Z(u) \) the state of the CTMC at time \( u \) and \( N(u) \) the number of events occurred until time \( u \).

Observe that \( \text{Passage}(E, G, D, t) \) may be defective, since there is a non-null probability of never reaching any goal state from an entering state or of reaching a forbidden state. The definition assumes that an initial distribution \( \pi(x) \) over the entering states \( E \) is given, and different choices have been considered in the literature. If the CTMC has a steady state behavior, an option is to use the steady state probability of the states in \( E \). An alternative choice is to use the steady state probability of entering the \( E \) states, computed as the steady state solution on the embedded Discrete Time Markov chain, thus disregarding the sojourn time in states. The two alternatives correspond to two different assumptions on the observation start time: in the first case it is a random instant during the steady state evolution of the system, in the second case it is the precise moment at which a state is entered. Another option is to condition the measure on a precise time \( t \), which requires to use the distribution at time \( t \), considering that the system started at 0.

Generalized Stochastic Petri Nets. (GSPNs) extend classical Place/Transition (Petri) nets with stochastic timing specifications. A detailed introduction to the formalism and
Definition 1 (GSPN). A GSPN system \( \mathcal{G} \) is a tuple \( \langle P, T, I^-, I^+, H, \Pi, w, m_0 \rangle \) where: \( P \) is the set of places; \( T = T_R \cup T_I \) is the set of exponential \((T_R)\) and immediate \((T_I)\) transitions; \( I^-, I^+, H : T \times P \rightarrow \mathbb{N} \) are the input, output, and inhibition arcs of the net; \( \Pi : T \rightarrow \mathbb{N} \) is the priority function. \( \forall t \in T_I, \Pi(t) = 0; \forall t \in T_R, \Pi(t) > 0; \) \( w : \mathbb{N}^P \times T \rightarrow \mathbb{R} \) is a function that assigns a rate to each \( t \in T \) and a weight to each \( t \in T_I \); and \( m_0 : P \rightarrow \mathbb{N} \) is the initial marking of the net.

A marking \( m \) (or state) of a GSPN \( \mathcal{G} \) is a multiset on \( P \). A transition \( t \) has concession at marking \( m \) iff \( I^-(t, p) \leq m(p) < H(t, p), \forall p \in P \). A transition \( t \) is enabled in marking \( m \) if no higher priority transitions have concession in that marking. The occurrence of transition \( t \) in \( m \) yields a marking \( m^t = m + I^+(t) - I^-(t) \), and we write \( m \circ t m^t \). The occurrence of a sequence \( \sigma \) of transitions enabled at \( m \) and yielding \( m' \) is denoted similarly: \( m(\sigma)m' \), and means that \( m' \) is reachable from \( m \), by firing the transitions in \( \sigma \).

The reachability set \( \text{RS}(\mathcal{G}) \) of a GSPN \( \mathcal{G} \) is the set of all the possible markings that can be reached from the initial marking \( m_0 \) with a sequence of zero or more transition firings. The Reachability Graph \( \text{RG}(\mathcal{G}) \) is defined by the pair \( \langle M, A \rangle \), where \( M = \text{RS}(\mathcal{G}) \) is the reachable state space and \( A \subseteq M \times T \times M \) is a finite set of arcs: an arc \( (m, t, m') \) is in \( A \) iff \( m(t)m', \) and \( m, m' \in M \).

A marking where only exponential transitions are enabled is called tangible, otherwise vanishing. The subset of tangible states of \( \text{RS}(\mathcal{G}) \) is called Tangible Reachability Set, \( \text{TRS}(\mathcal{G}) \). On the tangible states a Tangible Reachability Graph can be built:

Definition 2 (Tangible Reachability Graph). The tangible reachability graph of a GSPN \( \mathcal{G} \), is defined as: \( \text{TRG}(\mathcal{G}) = (M, A, \rho) \), where \( M = \text{TRS}(\mathcal{G}) \) is the tangible reachable state space, \( A \subseteq M \times T^+ \times M \) is a finite set of arcs, and \( \rho : A \rightarrow \mathbb{R}_{>0} \) is the transition rate function, computed in the usual way (see [1]) from transition rates and weights \( w \). Let \( \nu = (t, t') \) be an extended firing consisting of a timed transition \( t \) and a (possibly empty) sequence \( t' \) of immediate transitions, and let \( \Sigma_T \) be the set of all possible extended firings. An arc \( (m, \nu, m') \) is in \( A \) iff the extended firing \( \nu \) leads from \( m \) to \( m' \), i.e. \( m(\nu)m' \), and \( m, m' \in \text{TRS}(\mathcal{G}) \).

The states in the TRS correspond to the states of an underlying Markov Chain (MC) describing the stochastic behavior of the system. The infinitesimal generator \( Q \) can be automatically derived from the TRG [1].

A tangible path \( \tau \) (or simply path) of the TRG \( \mathcal{G} \), starting in a tangible marking \( m_1 \), is a sequence of markings and extended firings:

\[ \tau : m_1[\nu_1]m_2[\nu_2]m_3[\nu_3] \ldots \]

for all \( i \geq 1 \), \( m_i \in \text{TRS}(\mathcal{G}) \) and the arc \( (m_i, \nu_i, m_{i+1}) \) belongs to \( A \). A strongly connected TRG corresponds to an irreducible CTMC, and we shall consider only GSPN models producing finite and strongly connected TRGs. We also assume that the RG has no cycles of immediate transitions (although this assumption may be relaxed, see [1]).

Tagged GSPN models. If the measure of interest for a GSPN is the distribution of the time spent by a “specific token” in traversing a subnet, it is necessary to distinguish the token being measured (called the tagged token) from the others, as explained in [5, 10].

In Petri nets there is no concept of “token/customer flow”, as for clients in queueing networks, but it is possible to define a similar concept through the identification of the “conservative components” of a net. In [5] the authors suggest the use of p-semiflows, and introduce the concept of TGSPN, which is a pair \( \langle \mathcal{G}, f \rangle \) of a GSPN \( \mathcal{G} \) and a p-semiflow \( f \) (chosen among the minimal p-semiflows of \( \mathcal{G} \)). The concept of p-semiflow is standard in Petri nets, and for the sake of reasoning on TGSPNs, a p-semiflow \( f \) is a \( |P| \)-component vector over natural numbers, such that \( f \cdot m = f \cdot m_0 = K \). A generative family of minimal p-semiflows can be computed on the Petri net incidence matrix [1].

The procedure of computing a measure with a tagged customer in a GSPN is composed of three steps: all minimal p-semiflows of the GSPN are computed, to recognize the potential customer flows in the model; a p-semiflow of interest is identified, such that it includes the places of the subnet representing the service sequence to be analyzed, as well as the entry and exit points (transitions) of such subnet; finally the model is automatically unfolded to separate one of the tokens circulating in the selected p-semiflow (that will be the tagged token) from the others, in order to compute the passage time between any two events “tagged token entering the subnet” and “tagged token exiting the subnet”. The modified GSPN obtained from the unfolding procedure for a given p-semiflow \( f \) contains tagged replicas of the places in \( f \) and of the transitions connected to such places: hereafter we denote such replicas \( p_{i, \text{tag}} \) and \( t_{i, \text{tag}} \).

The passage time specification for a TGSPN introduced in [5] is given in terms of three sets of triplets \( \{t, C_{\text{in}}(t), C_{\text{out}}(t)\} \): \( \text{Entry, Exit, and Forbid} \), identifying respectively the transitions corresponding to the start and stop of passage time count, and those causing the abort of the measurement (i.e. allowing to discard the paths where any forbidden transition fires). \( C_{\text{in}}(t) \) has to be satisfied in the tangible marking where (the sequence containing) \( t \) is fired, and \( C_{\text{out}}(t) \) has to be satisfied in the tangible marking reached after firing it. The sets \( E, G \) and \( D \) for the passage time computation are derived respectively from the \( \text{Entry, Exit and Forbid} \) triplets as the tangible states reached by the firing of the specified transition \( t \), when the source state sat-
isfies $C_{in}$ and the state reached by the firing satisfies $C_{out}$. The definition of passage time in [5] assumes that any reference to transitions in the $Entry$, $Exit$ and $Forbid$ triplets is implicitly a reference to the corresponding tagged transition, which may be a limitation.

The Flexible Manufacturing System example. The running example used in this paper is the GSPN $G_{FMS}$ of a flexible manufacturing system (FMS) shown in Figure 1 (taken from [5]). The FMS comprises a Load/Unload (L/U) station and four manufacturing stations $M_1$–$M_4$, two of which, $M_2$ and $M_3$, can fail. $M_2$ can be locally repaired if a spare part is available, if this is not the case a repairman has to come into action. The same repairman is also taking care of machine $M_3$, for which no spare parts are locally available.

Passage times of interest require to work on the tagged version of the FMS. The model has twelve minimal $p$-semiflows, and we consider three $p$-semiflows that may be interpreted as the set of possible states of machine $M_2$ ($f_1$) and of a part along its processing cycle ($f_2$ and $f_3$):

\[
\begin{align*}
  f_1 &= p_{10} + p_{13} + p_{23} \\
  f_2 &= p_0 + p_1 + p_2 + p_3 + p_4 + p_5 + p_6 + p_7 + p_8 \\
  f_3 &= p_0 + p_1 + p_2 + p_3 + p_13 + p_{23} + p_5 + p_16 + p_17 + p_18 + p_7 + p_8
\end{align*}
\]

$f_3$ refines $f_2$, since it can distinguish the case where the part at machine $M_2$ is actually being processed (token in $p_{13}$) or is blocked due to a failure in $M_2$ awaiting for a repairman (token in $p_{23}$), similarly for $M_3$. The unfolding of the TGSPN ($G_{FMS}, f_2$) is shown in Figure 6 of [5].

Using the three $p$-semiflows, the following properties can be defined (recall that transition names that appear in $Entry$, $Exit$ and $Forbid$ sets refer to the tagged replicas):

- $P_1$: (for $f_1$) the waiting time in place $p_{23}$ (time spent by machine $M_2$ awaiting for the spare parts to be replaced by a repairman): $Entry = \{(T_5, p_{23} = 0, p_{23} > 0)\}$, $Exit = \{(T_8, -, -)\}$. In this particular case, since $f_1 \cdot m_0 = 1$ there is no need to unfold $G$. Note that the current definition of passage time for TGSPNs will in any case require the choice of a $p$-semiflow.

- $P_2$: (for $f_2$) passage time for the tagged token to complete the processing at $L/U$, $M_1$ and $M_2$: $Entry = \{(T_4, -, -)\}$, $Exit = \{(T_2, -, -)\}$.

- $P_3$: (for $f_3$): passage time for the tagged token to complete the processing at $L/U$, $M_1$ and $M_2$, conditioned on machine $M_2$ not breaking down while the tagged token is at $M_2$, resulting in: $Entry = \{(T_4, -, -)\}$, $Exit = \{(T_2, -, -)\}$ and $Forbid = \{(T_5, -, -)\}$.

- $P_4$: (for $f_2$): As for $P_3$ but conditioned on no breakdown occurring at $M_2$ while the tagged token is in $L/U$, $M_1$ or $M_2$. In this property $Forbid = \{(T_5, -, -)\}$, independently on whether it is firing for the tagged token or not.

- $P_5$: (for $f_2$): cycle time of the tagged token; this measure requires the entry point be equal to the exit point, for example the firing of $T_0^{lo}$.

- $P_6$ (for $f_3$): cycle time for the tagged token, conditioned on at most one breakdown occurring during the cycle. The specification of "at most one breakdown" cycle requires to keep memory of the failures occurred in the past.

Measures $P_4$, $P_5$ and $P_6$ cannot be computed with the TGSPN proposal in [5], since the specification language is not flexible enough.

3. Probe Automata

The FMS properties introduced in Section 2 require the ability to describe paths of interest in terms of transitions that may or may not fire, given certain pre and post conditions for their firing, as well as the ability to keep memory of events happened along a path (as in $P_6$). A reasonable way for doing this is to enrich the PA specification with pre and post conditions along the arcs, and the possibility to identify multiple initial and final states. We shall also add the possibility, included in XSP, of requiring certain path conditions to be met before the passage time computation starts.

A $Probe$ Automaton ($PrA$) is a finite automaton which reads, as input, a tangible path in the TRG of a GSPN system. A $PrA$ is a transition-diagram with constraint-labeled edges. Inside each TRG path, a $PrA$ identifies a set of passage subruns, that are used to compute the passage time distribution.

Given an alphabet of place names $P$, the language of boolean marking constraints $B(P)$ over $P$ is the set of boolean functions defined inductively by the language:

$$B(P) ::= p \otimes n \mid \neg \Phi \mid \Phi \land \Phi \mid true$$

where $p \in P$ is a place name, $\otimes \in \{<,\leq,=,\geq,>\}$ is a comparison operator, $n \in \mathbb{N}$ is a natural number and $\Phi \in B(P)$. Given a marking $m$ of a GSPN, the statement $p \otimes n$ is true if the comparison of the token assignment $m(p)$ of place $p$ in $m$ with the natural number $n$ is true. We write $m \models \Phi$ to say that the marking constraint $\Phi \in B(P)$ is satisfied in $m$.

**Definition 3 (Probe Automaton).** A $probe$ automaton $P$ is a tuple $(P, T, L, L_P, L_0, L_F, init, E)$, where:

- $P$ is a finite alphabet of place names;
- $T$ is a finite alphabet of transition names;
- $L$ is a finite set of probe automaton locations;
- $L_P \subseteq L$ is the subset of passage locations;
- $L_0 \subseteq L$ is the subset of initial locations;
- $L_F \subseteq L_P$ is the subset of final locations;
- $init : L_0 \to B(P)$ is the initial location constraint;
- $E \subseteq (L \setminus L_F) \times B(P) \times 2^{\Sigma_T} \times B(P) \times L$ is an edge relation. An edge $e = (l, \gamma_{pre}, X, \gamma_{post}, l')$, $e \in E$ represents a transition from a location $l$ to a new location $l'$, where $\gamma_{pre} \in B(P)$ is the pre-condition, $A \subseteq \Sigma_T$.
is the \textit{activating extended firing set}, and \( \gamma_{\text{post}} \in \mathcal{B}(P) \) is the post-condition. For each edge \( e \in E \), it should hold that \( l \in L_P \Rightarrow l' \in L_P \).

Final locations \( L_F \) select the states of the TRG that conclude the passage time computation, passage locations \( L_P \) select states of the TRG along which the passage time is computed, while initial locations in \( L_0 \) and \( \text{init} \) identify, for each TRG marking, the starting location of the probe. Once a passage location is reached, the probe may no longer visit non-passage locations until a final location is reached (see the last condition in the definition of the edges \( E \)). For GSPNs, the \( P \) and \( T \) alphabets coincide with the place and the transition sets of the GSPN (possibly derived from the unfolding of a TGSPN). The state of the probe automaton \( P \) is given entirely by its current location \( l \in L \).

Given a TRG arc \((m, \nu, m')\) and a probe edge \( e = (l, \gamma_{\text{pre}}, X, \gamma_{\text{post}}, l'), e \in E \), we say that the edge constraints of \( e \) are satisfied by the TRG arc, written as \((m, \nu, m') \models e\), if \( m \models \gamma_{\text{pre}} \land \nu \in X \land m' \models \gamma_{\text{post}} \). A probe automaton \( P \) accepts the language of tangible paths. A tangible path \( \tau = m_1 [\nu_1] m_2 [\nu_2] \ldots \) beginning in \( m_1 \in \text{TRS}(G) \), the initial state of the probe \( P \) for \( \tau \) is a location \( l_0 \in L_0 \) chosen such that \( m_1 \models \text{init}(l_0) \). We say that:

\[
\begin{align*}
\gamma : (l_0, k_0) \xrightarrow{k_1} (l_1, k_1) \xrightarrow{k_2} (l_2, k_2) \xrightarrow{k_3} \ldots
\end{align*}
\]

is a run over the path \( \tau \), with \( k_i \in \mathbb{N}, k_0 = 1 \), provided that \( l_0 \) is an initial location for \( m_1 \), and, for all \( i \geq 1 \), there exists an edge \( e = (l_{i-1}, \gamma_{\text{pre}}, X, \gamma_{\text{post}}, l_i) \in E \) such that \((m_{k_{i-1}}, \nu_{k_{i-1}}, m_{k_i}) \models e\).

A run \( \gamma \) over a path \( \tau \) is an accepting run for a target final location \( l_F \in L_F \) if \( \exists k \geq 0 \) such that \( \forall i \geq k \) the probe location \( l_i \) is \( l_F \). The language \( L_\gamma(\tau) \) is the set of all accepting runs that end in \( l_F \), and \( \mathcal{L}(\tau) = \bigcup_{l \in L_F} \mathcal{L}(\tau, l) \) is the union of the language of all the languages accepted by \( P \).

Figure 2 illustrates a simple PrA. Non-passage \((l_0)\) and passage \((l_1, l_2, l_3)\) locations are drawn with diamonds and circles, and are labeled with a name. Final locations have a double border. Edges are labeled with constraints written as \( \gamma_{\text{pre}} / X / \gamma_{\text{post}} \). Each \( l_0 \) location has an associated init constraint, shown as an entering arrow \((l_0, l_1)\). The minus sign denotes the absence of a pre/post condition, and an asterisk denotes the entire extended firing set \( \Sigma_T \). For example, the edge from \( l_0 \) to \( l_1 \) is triggered by the firing of any extended transition, as soon as the post-condition \( P_0 \neq 0 \) holds in a tangible state. An extended firing set is denoted as a list inside curly braces (as in the edge from \( l_1 \) to \( l_2 \)). We use the shorthand notation of writing a transition \( t \) (without curly braces) to indicate the set of all extended firings that contain \( t \), as in the edge from \( l_1 \) to \( l_3 \). This probe automaton has two final locations and therefore accepts two languages. For instance, the language \( \mathcal{L}(l_2, \tau) \) contains all the runs that first match a \( T_1 t_2 \) firing when \( P_1 > 0 \), after a state where \( P_0 \neq 0 \) has been visited.

![Figure 1. A FMS with machines breakdown and repair.](image)

**Figure 2.** Probe automaton example.

\[
\begin{align*}
P_0 = 0 & \quad \xrightarrow{\text{entry}} l_0 \quad \xrightarrow{-/\text{P}_0 \neq 0} l_1 \quad \xrightarrow{-/\{T_1 t_2\}/P_1 > 0} l_2 \quad \text{Exit condition} \quad \xrightarrow{C_{\text{out}}(T_{\text{out}})/C_{\text{in}}(T_{\text{in}})} l_{k_0} \quad \xrightarrow{C_{\text{in}}(T_{\text{in}})/C_{\text{out}}(T_{\text{out}})}/C_{\text{out}}(T_{\text{out}}) \xrightarrow{l_k} \text{Forbid condition} \quad \xrightarrow{C_{\text{in}}(T_{\text{in}})/C_{\text{out}}(T_{\text{out}})} l_{k_0}
\end{align*}
\]

**Figure 3.** Passage time in GSPNs

Given a run \( \gamma \), let \( i \) be the minimum index s.t. \( l_i \in L_P \), and let \( j \) be the index of the final location \( l_f \in L_F \) that ends \( \gamma \) (or \( j = \infty \) if the run \( \gamma \) is infinite and does not end in
any final location). The part of \( r \) that goes from \( l_i \) to \( l_j \) (excluded) is the passage subrun \( \bar{r} \) of \( r \). Passage time is the time elapsed in passage subruns, until a (given) final location is encountered.

Figure 3 illustrates the PrA that translate the generic passage time expression for TGSPNs based on the three triplets \( \text{Entry, Exit, and Forbid} \).

Figure 4 depicts the probes for the FMS properties of Section 2. The automata for \( P_1, P_2, \) and \( P_5 \), are instantiations of the automaton in Figure 3, according to their definition of \( \text{Entry, Exit, and Forbid} \) triplets. Automaton for \( P_3 \) is not shown, since it is the same as \( P_4 \) with \( T_3 \) replaced by \( T_5^{\text{tag}} \). Automaton \( P_6 \) depicts the use of probe locations to keep track of what has happened along a cycle: indeed after a first breakdown, a second breakdown leads to \( l_{ko} \). If the cycle concludes without a second breakdown, the probe goes to location \( l_{ok} \). Finally \( P_7 \) is a property which is here introduced to exemplify the use of multiple non-passage locations: it is a modification of \( P_5 \) to account only for those paths that have experienced, before starting the computation of the passage time, at least one real breakdown of machine \( M_2 \) (a breakdown when no spare is available, i.e. leaving \( p_{23} > 0 \)).

4. Passage time distribution computation

A probe automaton \( P \) follows passively the firings of a TRG. Each firing is read by the probe, that has a chance of moving from its current location according to the conditions specified on the outgoing edges of \( l \). This results in joint process that can be constructed as a synchronized product of both state spaces.

Let \( G = \langle P, T, I^-, I^+, H, \Pi, w, m_0 \rangle \) be a GSPN system, let \( \langle M, A, \rho \rangle = \text{TRG}(G) \) be its tangible reachability graph, and let \( P = \langle P, T, L, L_P, L_0, L_F, \text{init, } E \rangle \) be a probe automaton for \( G \). The initial states set \( S_0 \) of the product is a set of marking and location pairs defined as:

\[
S_0 = \{ (m, l) \in M \times L_0 \mid m \models \text{init}(l) \}
\]

\( S_0 \) contains the initial configuration of the probe in each reachable marking of the GSPN. We call product of the probe \( P \) with the TRG the union of all synchronized products of \( P \) starting from each initial state in \( S_0 \).

Definition 4 (Product of \( P \) with a TRG): A Product of a TRG \( \langle M, A, \rho \rangle \) with a probe automaton \( P \) is a tuple \( T = \langle S, S_0, S_P, S_F, A, \rho \rangle \), where:

- \( S \subseteq M \times L \) is a finite set of states, given by the cross product of the tangible reachability set \( M = \text{TRS}(G) \) with the location set \( L \);
- \( S_0 \subseteq S \) is the set of initial states, defined before;
- \( S_P = (M \times L_P) \cap S \) is the set of passage states;
- \( S_F = (M \times L_F) \cap S \) is the set of final states;
- \( A \subseteq S \times A \times S \) is the transition relation;
- \( \rho : A \to \mathbb{R}_{\geq 0} \) is the transition rate function;

defined inductively according to these rules:

1. The set of initial states is in \( S \): \( (S \cap S_0) = S_0 \).
2. Transition rule “probe moves”: let \( s = \langle m, l \rangle \) be a state in \( S \), \( a = \langle m, \nu, m' \rangle \) a TRG arc in \( A \), and \( e = (l, \gamma_{\text{pre}}, X, \gamma_{\text{post}}, l') \) a probe edge in \( E \), such that \( a \models e \). Then the state \( s' = \langle m', l' \rangle \) is also in \( S \) and the arc \( t = (s, a, s') \) is in \( A \), with its exponential transition rate \( \dot{\rho}(t) \) equal to \( \rho(a) \).
3. Transition rule “probe stands still”: let \( s = \langle m, l \rangle \) be a state in \( S \) with \( l \not\in L_F \), let \( a = \langle m, \nu, m' \rangle \) be a TRG arc in \( A \), such that there is no probe edge \( e \in E \) where \( a \models e \) is true. Then the state \( s' = \langle m', l \rangle \) belongs to \( S \), and the arc \( t = (s, a, s') \) is in \( A \), with rate \( \dot{\rho}(t) = \rho(a) \).

The product \( T \) is deterministic if and only if:

- Initial location determinism: for each \( m \in \text{TRS}(G) \), there exists exactly one initial location \( l \in L_0 \) such that \( m \models \text{init}(l) \).
- Probe edges determinism: for each state \( s = \langle m, l \rangle \) and for each TRG arc \( a = \langle m, \nu, m' \rangle \) leaving \( m \), there exists at most one location \( l' \in L \) s.t. any probe edges \( e \in E \) that satisfies the arc \( a \), i.e. \( a \models e \), reaches location \( l' \).
When $T$ is deterministic, each path $\tau$ in the TRG($G$) is matched by a single run $r$ of the probe. Therefore, the stochastic behavior of $T$ is fully determined and can be described as a CTMC.

There is a strong correspondence between TRG paths and paths in a deterministic product $T$. Given an (infinite) path $\tau$ in the TRG($G$) starting in $m$, there exists a unique path $\sigma$ rooted in $(m,l)$, with $m \models \text{init}(l)$, that follows each transition in $\tau$ until the probe reaches a final location. Such path $\sigma$ is unique due to determinism constraints. Vice versa, the sequence of markings of each path $\sigma$ rooted in $(m,l)$ corresponds to the (infinite) set $J(\sigma)$ of TRG($G$) paths, which share the same prefix. Since transition rates are "copied" by the construction rules of $T$, the continuous-time behavior of each $T$ path $\sigma$ is the same behavior of the common prefix of each TRG($G$) path in $J(\sigma)$.

The state space of $T$ has an asymptotical cost of $O(|M| \cdot |L|)$, since the state space $S$ is a subset of the Cartesian product of the marking set $M$ with the location set $L$. A deterministic product $T$ may be treated as a TRG with reachable state space $S$, are set $A$ and rate function $\rho$.

Figure 5 shows two simple probes (A and B), a cyclic TRG C; and the products of both probes with C. The sets $S_E$ and $S_F(l_{\text{ok}})$ highlighted in the products will be explained later, and constitutes a sort of "barriers" used for the passage time computation.

Observe that we are considering all the possible initial states $S_0$ for the product state space, since we are assuming that the probe may start "randomly" in any possible state of the model. The probe partitions the state space $S$ into the passage set $S_P$ (on the right of the $S_E$ line) and the non-passage set (on the left). The reader may check that all the possible runs are indeed in the product TRG $T$.

From a TRG $(M,A,\rho)$ of a GSPN system $G$, a CTMC is derived, as usual, in the domain space of $M$ by summing up the outgoing transition rates that leave each marking. Let $\{Z(t) \mid t \geq 0\}$ be the stochastic process that behaves according to the CTMC described by the TRG, and let $Q$ be its infinitesimal generator. Entries of $Q$ are:

$$Q_{i,j} = \begin{cases} \sum_{a \in (i,\nu,j) \in A} \rho(a) & i \neq j \\ -\sum_{k \neq i} q_{i,k} & i = j \end{cases}$$

In order to compute the passage time on the underlying CTMC, we need the set of initial states $S_0$ and two additional state subsets: the set of start states $S_E$, where the measure time starts, and the target states $S_F(l)$, for a given final location $l \in L_F$. Hence:

$$S_E = \{s' \in S_P \mid s' \in S_0 \lor (\exists s \in (S \setminus S_P) \land \exists(s,a,s') \in A)\}$$

is the set of states where the probe enters the passage set for the first time, starting the passage time computation. Let:

$$S_F(l) = \{(m,l') \in S \mid l' = l\}, \quad l \in L_F$$

be the set of states where the joint process accepts a run of $G$ in the specified final location $l$, thus ending the passage time computation. Note that $S_E$ is the set $E$ in the passage time definition of Section 2, while the sets $G$ and $D$ of goal and forbidden states are represented by the $S_F(l)$ sets (typically $G$ is represented by $l_{\text{ok}}$ and $D$ by $l_{\text{k}}$).

We consider that a probe may start at any time in the process described by the GSPN, in stationary conditions, so a marking $m$ is chosen according to its probability in the steady state distribution $\pi$ of TRG($G$). The probe starts in the initial location $l_0$ satisfied by the init constraint. Therefore, the joint process $T$ begins in the state $i = (m,l_0)$, as shown in Figure 6. The stochastic process $Z(t)$ then continues its execution until an $S_P$ state is reached, i.e. a state $k \in S_E$ is first entered. If the initial state $i$ is already an entering state (i.e. $i \in S_E$), then $k = i$. At this point a clock starts, and the execution continues in the passage subset $S_P$. The measure Passage($t,l$) is the probability of reaching a state $j \in S_F(l)$ before time $t$, starting at time 0 from any $k$.

Passage time measure for a probe automaton $P$ in a GSPN system $G$ can be easily described by means of accepted runs. The measure Passage : $\mathbb{R}_{\geq 0} \times L_F \rightarrow \mathbb{R}_{[0,1]}$
describes the cumulative distribution function for the passage time that starts in an $S_E$-state and ends in the specified $S_F(l)$-state in less that $t \in \mathbb{R}_{\geq 0}$, conditioned on an initial probability distribution over $S_0$-states. Let $\pi$ be such distribution over $S_0$-states in the product TRG $\mathcal{T}$. Then the Passage($t, l$) at time up to $t$ to a set of final states $S_F(l) = \{(m, l') \in S\}$ is:

$$\text{Passage}(t, l) = \sum_{i \in S_0} \pi(i) \cdot \sum_{k \in S_E} H_{i,k}^{S,S_F} \cdot \int_0^t \int_{S_F(l)} H_{k,j}^{S_F}(x)dx,$$

where $H_{i,k}^{S,S_F}$ is the long-run probability of going from state $i$ to state $k$ passing through $(S \setminus S_F)$ states only (note that $k \in S_E$, which implies that $k \in S_P$), and $H_{k,j}^{S_F}(x)$ is the probability of going from state $k$ to state $j$ at time $x$ passing though $S_P$-states only.

The term $H_{i,j}^{S_S}(t) = P_r\{Z'(t) = j \mid Z'(0) = i\}$ is the kernel matrix of the stochastic process $\{Z'(t) \mid t \geq 0\}$. Such process $Z'(t)$ behaves like $Z(t)$ restricted to the subset $S' \subseteq S$, which means that transitions may occur in $S'$-states only. Therefore, the infinitesimal generator $Q'$ of $Z'(t)$ has non-zero rows only for $S'$-states. The term $H_{i,j}^{S_S'}$ is the limit $\lim_{t \to \infty} H_{i,j}^{S_S'}(t)$.

Remember that all $S_F$-states are absorbing by construction, and $S_F(l) \subseteq S_F$. The integral (3) is the cumulative passage time distribution up to time $t$ from an entering state $k \in S_E$ to any final state in the target set $S_F(l)$.

Procedure (Computation of Passage($t, l$)). The passage time distribution computation at time $t$ for a specified final location $l \in L_F$ can be computed with these steps:

1. Generate the TRG$\mathcal{G}$ and get the steady state distribution $\pi$ over $\text{TRS}(\mathcal{G})$ states.
2. Generate the cross-product process $\mathcal{T}$ of TRG$\mathcal{G}$ with the PR$\mathcal{A}$ automaton $\mathcal{P}$.
3. Starting from $S_0$-states with initial steady state probability distribution $\pi$, compute the probability distribution of reaching the entering set $S_E$, assuming that $S_E$ states are made absorbing.
4. Compute the transient probability of reaching, from the $S_E$ state subset of $\mathcal{T}$, the subset $S_F(l)$ at time $t$.

As the reader may observe, the measure Passage($t, l$) can be seen as a steady-state measure, in the sense that the accepted sequences do not depend on the initial marking of the GSPN. Instead, it is more like inserting the probe in the GSPN in steady-state, and observing the probe behavior from that point on.

Tool support and experimentation. We have implemented a prototype solution as an add-on of GreatSPN [3]. It generates the CTMC derived by the product between the probe automaton and TRG, and exports it to HYDRA [9] for the computation of the (first) passage time distribution (from $S_E$ to $S_F(l)$), while GreatSPN is used to compute the initial distribution of the entering states (up to $S_E$).

The TRG generation and the product between TRG and the probe is based on Meddly [4], an existing open-source library for Decision Diagrams. The solution takes advantage (in terms of both memory utilization and execution time) of the efficiency of Multi-valued Decision Diagrams (MDDs) and of the saturation approach [6] for state space generation. The implemented solution approach is not as fast as it could be since the passage time distribution is not computed symbolically, but is passed on to HYDRA by writing the generated CTMC (which is built symbolically) on a file according to the HYDRA input format.

Figures 7(a), (b) and (c) report the CDF for the properties $P_1$, $P_2$, $P_4$, $P_5$ and $P_6$ considering a scenario with 9 pallets, where the timed transitions $T_{01}$, $T_2$, $T_3$, $T_4$ and $T_5$ are Single Server with rates 0.7, 2.0, 3.0, 4.0 and 0.5, and all the others are Infinite Server with rate 1. All immediate transitions have weight 1. The TRG has 146 526 states and requires less than 1s for its generation; while the number of states in $T_{P_1}$, $T_{P_2}$, $T_{P_4}$, $T_{P_5}$ and $T_{P_6}$ are 187 533, 240 669, 284 262, 310 902 and 409 845, respectively. Each of these TRGs is generated in less than 50s, with a memory peak < 1.8 MBs.

Figure 7(a) shows the CDF of property $P_1$ measuring the time spent by machine $M_2$ waiting for the spare parts to be replaced by a repairman. Observe that this property does not require to unfold the GSPN model as explained in Section 2.

Figure 7(b) shows the CDF of properties $P_2$ and $P_4$, both referred to the tagged token. The first one measures the passage time for the processing sequence at machines $L/U$, $M_1$ and $M_2$, while the measure $P_4$ does not consider paths in which machine $M_2$ breaks down during service. This makes the measure defective: in Figure 7(b) this distribution is shown also after normalization (with respect to the long-run probability of following a path without any $M_2$ breakdown). Observe that, as expected, the normalized CDF lays above the CDF obtained for $P_2$.

Figure 7(c) shows the passage time computation for properties $P_5$ and $P_6$, that are related to the distribution of the cycle time of the tagged token. Indeed, the former computes the unconditioned distribution of the cycle time of the
tagged token, while the latter the cycle time of the tagged token conditioned on at most one breakdown occurring during the cycle. For this case the normalized CDF for $P_6$ is not plotted since it is very close to the one of $P_5$.

5. Comparison with XSP and PA

XSP probes and PA automata, although similar to PrA automata, have some significant differences both in the **specification** of the paths to be considered and in the passage time **computation**.

In terms of paths description PrA also includes post-conditions on arcs, which turns out to be particularly useful to deal with immediate transitions, but it could be an interesting extension for XSP, since it may be used to discriminate, in a PEPA term, the path taken upon a choice between two actions with the same name. All three formalisms allow for multiple final states (considering the extended XSP definition in [7]): interpreting one of the final states as a reject state allows to discard undesired paths/behaviors from the passage time computation.

PrA have been defined for GSPNs, and therefore deal with immediate transitions. This is particularly useful for discriminating in a TGSPN the case in which the firing of a timed transition $t$ may lead to a entering state or not, depending on a specific sequence of immediate transitions activated by $t$. This could distinguish in a TGSPN whether the tagged token is in the subnet or not. XSP have been defined in the context of PEPA and therefore do not deal with immediate transitions, while PA deal with **stable steps** (the counterpart of extended firings for SAN).

What distinguishes PrA from similar techniques is the passage time computation method, in which the matching of the probe starts in any state of the CTMC and not only in the initial one, moreover the probe may start in different locations depending on the marking being considered, thanks to the possibility of defining multiple initial locations. Multiple initial locations are particularly interesting when $L_0$ contains both passage and non-passage locations.

As we shall see in the next example the proposed computation also allows a natural definition of cycle time (passage time for paths in which the start and stop actions/transitions are the same). An XSP probe (and a PA automaton) goes in parallel with a PEPA term (or a SAN) from their initial state, while a PrA automaton selects subpaths that may not start in the GSPN/CTMC initial marking/state. To better explain the consequences, consider this model, given as a PEPA term:

$$m_1 = (a, \lambda_1).m_2, \quad m_2 = (a, \lambda_2).m_3, \quad m_3 = (a, \lambda_3).m_4, \quad m_4 = (b, \mu).m_1$$

which has a derivation graph isomorphic to the CTMC given in Figure 5(c). The PrA automaton $A$ of Figure 5, leads to a passage time computation which is a weighted sum of the passage time distributions of the paths from $m_2$ to $m_1$, from $m_3$ to $m_1$, and from $m_4$ to $m_1$. This is not equivalent to the XSP probe $R = a:start,b:stop$, that can informally be described as: “start counting after $a$ is encountered, and stop after $b$ is encountered”, which computes the passage time only on the path from $m_2$ to $m_1$. It is our understanding that there is no way to do a similar computation in XSP which is not ad-hoc for the specific term $m_1$. If we consider instead PA, the probe $A$ of Figure 5, with a self loop on $l_1$, can be considered as a PA automaton, but the computation of passage time will produce the same results as in XSP.

If we instead want to compute the passage time that starts in a state entered with an $a$, and that observes exactly one $a$ followed by a $b$, this can be represented by the PrA automaton $B$ of Figure 5. This leads to the computation of the passage time distribution exactly from $m_2$ to $m_1$ (the only path of the CTMC in which we have exactly two $a$ followed by a $b$). Both XSP and PA fail to identify this path, since they are constrained to start from the initial marking.

There is another interesting case of a different behavior
between XSP/PA and PrA. Consider the PEPA term:

\[ P = (T0, \lambda_1).P1, \quad P1 = (T0, \lambda_2).P \]

and an XSP probe \( R = T0:start, T0:stop \), which could be thought adequate for computing the distribution of the cycle time for action \( T0 \), similar to the PrA automaton of property \( P_T \) in Figure 4. If our interpretation of XSP is correct, probe \( R \) will identify a set of start and stop states such that only \emph{even} occurrences of \( T0 \) (the ones of rate \( \lambda_2 \)) will be counted for the passage time, while \emph{odd} occurrences are not considered. This happens because every other occurrence of \( T0 \) is “consumed” to identify the stop state. A PA automaton suffers from the same problem, since it would read every even occurrence of \( T0 \) to enter the final state.

### 6. Conclusions

In this paper we have introduced a language for passage time specifications, that allows to express complex passage time measures based on path acceptance. The language has been defined for GSPNs, but it is rather straightforward to adapt it to other formalisms, since the semantics is defined at the state-space (TRG/CTMC) level. Associated with the probe language, there is a new proposal for computing passage time on all sub-paths satisfying the automaton.

A prototype tool has been implemented, that computes the product state space and the passage time distribution; it is built on top of GreatSPN and HYDRA. Results for a medium-sized FMS model are reported in Section 4.

Many interesting future developments are possible. We plan to do a thorough comparison of PrA with XSP, in particular to compare the tagged token approach of TGSPNs with the location-aware probes of XSP. We shall also investigate whether PrA could be used to specify complex reward structures, as it is the case for PA.

With respect to the application of PrA we plan to pursue two lines of research: evolution of the method proposed in this paper towards SWNs, a colored version of GSPNs, and implicit tagged token measures specification. For SWNs the issue is not to destroy the symmetry exploitation typical of SWN solution (a similar problem has been tackled in the context of PA), while for tagged token measure it would be nice to have a way to “instantiate” the probe depending on the semiflow associated to the TGSPN, and, vice versa, given a specific “tagged” probe and a GSPN, to define the TGSPN adequate for the probe computation (by selecting the appropriate p-semiflow).

The prototype implementation can be enhanced by making it symbolic in the stochastic computation of passage time distributions, which requires the replacement of HYDRA with a stochastic solver based on some form of Decision Diagram.

### References


