Finite-Trace Analysis of Stochastic Systems with Silent Transitions (Extended Abstract)*

Sander J.J. Leemans[1] , **Fabrizio M. Maggi**[2] and **Marco Montali**[2]
[1]RWTH Aachen, Germany
[2]Free University of Bozen-Bolzano, Italy
s.leemans@bpm.rwth-aachen.de, {maggi, montali@inf.unibz.it}

Abstract

In this paper, we summarise the main technical results obtained for *specification probability*. That is, we compute the probability that if a bounded stochastic Petri net produces a trace, that trace satisfies a given specification.

1 Introduction

Dealing with actions and processes under a stochastic interpretation of (different forms of) nondeterminism is a long-standing problem within AI and business process management (BPM). In AI, Markov chains and richer stochastic models such as MDPs are at the core of a number of tasks related to reasoning, planning, and learning in dynamic systems. In BPM, stochastic extensions of Petri nets, studied from the foundational point of view in the 70s and 80s, have been recently revived in the context of process mining, where models of processes and data about the execution of processes are related to obtain insights about how organisations work [van der Aalst, 2016].

Interestingly, both fields traditionally progressed with an inherent dichotomy on the semantics of such stochastic processes. On the one hand, when an agent has to *plan* how to achieve a desired goal, or a work process is *instantiated* on a particular case that has to be progressed towards one of the final states of the process, the semantics is inherently that of *finite traces* - that is, each trace consists of an unbounded, yet finite, number of steps, each witnessing the execution of an action or task. On the other hand, key reasoning tasks such as static analysis/verification typically work under an *infinite-trace* semantics, where temporal/dynamic properties of interests are checked by assuming that every execution of the system under scrutiny is non-terminating (see, e.g., [Bustan *et al.*, 2004; Baier and Katoen, 2008]). Such a dichotomy is witnessed in several related lines of research, such as for example that of linear temporal logics, which have been traditionally studied under an infinite-trace semantics, in recent times also deeply exploring them under a finite-trace interpretation [De Giacomo and Vardi, 2013a], which radically changes the resulting framework [De Giacomo *et al.*, 2014].

*This paper summarises [Leemans *et al.*, 2022; Leemans *et al.*, 2023].

The formal analysis of such systems is not only deeply affected by the adopted trace semantics, but also by the presence or absence of other features. In particular, when using Petri nets to represent work processes, it is important to support the labelling of transitions with activity names, to account for duplicate labels (witnessing that the same activity may be executed in different states of the process), and to account for silent steps, in the style of ϵ-moves in automata (to deal with internal state changes that do not represent the execution of an activity). This calls for a careful reconsideration of the system under scrutiny, as a visible trace of the system (consisting of sequences of visible activities) may in general correspond to infinitely many distinct underlying system runs (consisting of sequences of transitions).

Our work [Leemans *et al.*, 2022; Leemans *et al.*, 2023] starts precisely from these two considerations: we are interested in the *formal analysis of stochastic systems*, under the following working hypotheses:

1. the system may contain *duplicate activities* - i.e., the same activity may be executed in different system states;
2. the system is equipped with *silent transitions* - i.e., internal orchestration steps inducing state changes that are not cause by a visible activity;
3. the system produces *finite-length traces*.

We ground these general notions within the framework of (bounded generalised) stochastic Petri nets, extensively studied in the 70s and 80s [Ajmone Marsan, 1988; Ajmone Marsan *et al.*, 1995] but without considering the three working hypotheses listed above. By formal analysis, we consider the following essential problems, that are in turn relevant to solve further problems in the spectrum of stochastic process mining:

1. **Outcome probability**: what is the probability that the stochastic system evolves from the initial marking to one (or a subset) of its final markings?
2. **Trace probability**: what is the probability of a given trace of the stochastic system?
3. **Specification probability**: what is the probability that the stochastic system produces a trace that satisfies a given qualitative specification that captures desired behaviour, expressed using a deterministic finite-state automaton?

4. **Stochastic compliance**: is the stochastic system of interest compatible, in behavioural and stochastic terms, to a probabilistic declarative specification [Alman *et al.*, 2022] indicating which temporal properties are expected to hold, and with which probability?
5. **Stochastic conformance checking**: how can we employ the previous analysis questions, in particular trace probability, to improve the correctness and applicability of existing stochastic conformance checking techniques [Leemans *et al.*, 2019] relating a reference stochastic process model to a recorded log?

Our main contributions are the following. First, we show that *outcome probability* can be solved through standard analytic techniques from Markov chains. This calls for revisiting the connection between generalised stochastic Petri nets and discrete-time Markov chains [Marsan *et al.*, 1984; Ajmone Marsan *et al.*, 1995] for our working hypotheses, in particular the class of absorbing Markov chains [Grinstead and Snell, 1997, Chapter 11], which naturally capture the computation of probabilities for finite traces. We also show that when the input system is livelock-free (i.e., every intermediate state can potentially reach one of the final states), then the collective probability of all model traces (i.e., finite-length traces of the system leading from the initial to one of the final states) is indeed 1. Second, we show that the *specification probability* problem can be solved in three steps. Firstly, we enrich the automaton of the property with additional transitions that handle silent steps of the system, thus moving from an automaton that expresses desired traces to an automaton that expresses desired runs. Secondly, we demonstrate that standard cross product-based techniques can be used to obtain the system runs that satisfy the desired properties. Thirdly, we show that the problem can now be solved as an outcome probability problem over the cross product. Third, we show that the other mentioned problems can be reduced to corresponding specification probability and, in turn, outcome probability problems.

2 Related Work

Stochastic process-based models have been studied extensively in literature. We use formal, Petri net-based stochastic models that are used by some recent approaches in stochastic process discovery [Burke *et al.*, 2020; Rogge-Solti *et al.*, 2013; Burke *et al.*, 2021] and conformance checking [Leemans *et al.*, 2021; Leemans and Polyvyanyy, 2020; Bergami *et al.*, 2021b; Alkhammash *et al.*, 2022; Burke *et al.*, 2022]. Such approaches all refer to the model of (generalised) stochastic Petri nets, or fragments thereof. A first version of this model was proposed in [Molloy, 1982], extending Petri nets by assigning exponentially distributed firing *rates* to transitions. This was extended in [Marsan *et al.*, 1984] by distinguishing timed (as in [Molloy, 1982]) and immediate transitions. Immediate transitions have priority over timed ones, and have *weights* to define their relative likelihood. As these two types of transitions, abstracting from time, behave homogeneously, we may capture the stochastic behaviour of the net through a discrete-time Markov chain [Marsan *et al.*, 1984].

Several variants of stochastic Petri nets have been investigated starting from the seminal work in [Marsan *et al.*, 1984]. These variants differ from each other depending on the features they support (e.g., arbiters to resolve non-determinism, immediate vs timed transitions) and the way they express probabilities. Such nets may aid modellers in expressing certain constructs. An orthogonal, important dimension is to ensure that probabilities and concurrency interact properly. This can be achieved through good modelling principles [Marsan *et al.*, 1984; Chiola *et al.*, 1993] or automated techniques [Bruni *et al.*, 2019].

Contrasting these formal models with recent works in stochastic process mining, key differences exist. Traditional stochastic nets do not support transition labels nor silent transitions, and put emphasis on recurring, infinite executions and the so-called steady-state analysis, focused on calculating the probability that an execution is currently placed in a given state. This is done by constructing a discrete-time Markov chain that characterises the stochastic behaviour of the net [Molloy, 1982; Marsan *et al.*, 1984]. Finding the probability of a finite-length trace in such nets is trivial, as every trace corresponds to a single path. However, no transition labels or silent steps are supported, which limits their usefulness for process mining due to the omnipresence of such transitions in process models. On the other hand, when these features are incorporated in stochastic Petri nets, which is precisely what we target in this paper, computing the probability of a trace cannot be approached directly anymore, as infinitely many paths would potentially need to be inspected. At the same time, in business processes we are interested in behaviour at the trace level rather than at the process level – that is, we are not interested in the *state* that a process can be in, but rather on the *path* that a trace follows through the model – thus the large body of work on steady-state-based analyses on Markov models does not apply for our purposes. This explains why reasoning on the stochastic behaviour of such extended nets has been conducted in an approximated way [Leemans *et al.*, 2021; Leemans and Polyvyanyy, 2020; Bergami *et al.*, 2021a], or by imposing restrictions on the model [Bergami *et al.*, 2021b].

In this paper we take the basic stochastic Petri nets: we do not consider time or priority, but we add (duplicate) labels and silent transitions. Importantly, *our results seamlessly carry over to bounded, generalised stochastic Petri nets*, thanks to the fact that incorporating priorities in bounded nets is harmless, and that timed and immediate transitions are homogeneous from the stochastic point of view. To the best of our knowledge, outside of recent work using stochastic Petri nets with silent transitions [Leemans *et al.*, 2021; Leemans *et al.*, 2019; Bergami *et al.*, 2021b], such nets have not been defined or studied before.

While intuitively stochastic conformance checking techniques need to obtain the probability of a given trace in a stochastic process model (for instance, [Leemans *et al.*, 2019] explicitly obtains this probability to compute a distance measure between a log and a stochastic process model), some stochastic conformance checking techniques avoid computing the probability for a single trace, for instance by playing out the model to obtain a sample of executions [Leemans

et al., 2021], or by assuming that the model is deterministic [Leemans and Polyvyanyy, 2020]. The results presented in this paper therefore enable the practical application of [Leemans *et al.*, 2019], and may enable new types of analyses and stochastic conformance checking techniques.

Silent steps have been studied in the context of automata. For instance, in [Hanneforth and De La Higuera, 2010] an ad-hoc method is described to iteratively remove all silent steps from a stochastic automaton. Due to concurrency and confusion, such techniques are not directly applicable to stochastic Petri nets. A result of this paper is that silent steps can be handled directly, without the need for ad-hoc techniques.

3 Preliminaries

We first coin some multiset machinery: let X be a set, then $X^{\mathbb{N}}$ is the set of all multisets over the elements of X. Given two multisets A and B, for an element x, $A(x)$ denotes the multiplicity of x in A. Furthermore, we denote $A \leq B \equiv \forall_{x \in A} A(x) \leq B(x)$, $\forall_x (A + B)(x) = A(x) + B(x)$ and, when $B \leq A$, then $\forall_x (A - B)(x) = A(x) - B(x)$.

Given an alphabet Σ of process steps (activities), an event is the execution of an activity. A sequence of events, denoting all activities executed for a particular process instance such as an order, claim or application is a trace. A multiset of traces is an event log.

Definition 1 (labelled stochastic Petri net)**.** *Let Σ be an alphabet of activities, such that $\tau \notin \Sigma$. Then, a* labelled stochastic Petri net *is a tuple $(P, T, F, m_0, M_f, \ell, w)$ in which P is a set of places, T is a set of transitions such that $F \cap T = \emptyset$, $F \subseteq (P \times T) \cup (T \times P)$ is a flow relation, $m_0 \in P^{\mathbb{N}}$ is an initial marking, $M_f \subseteq P^{\mathbb{N}}$ is a set of final markings, $\ell \colon T \to \Sigma \cup \{\tau\}$ is a labelling function that maps each transition to either an activity of Σ or to the silent label τ, and $w \colon T \to \mathbb{R}^+$ is a weight function.*

Semantically, the net starts its execution in the initial marking m_0. In a particular marking m, a transition $t \in T$ is *enabled* if ${}^\bullet t \leq m$. Let $E(m) = \{t \in T \mid {}^\bullet t \leq m\}$ denote the set of enabled transitions. An enabled transition t can *fire*, which results in the marking $m' = m + t^\bullet - {}^\bullet t$ and, if $\ell(t) \neq \tau$, in the emitting of the activity $\ell(t)$. In a marking m, the probability that an enabled transition t fires is proportional to the weight of t vs. the weight of all enabled transitions: $\mathbb{P}_m(t) = \frac{w(t)}{\sum_{t' \in E(m)} w(t')}$.

A *run* of the net is a sequence of transitions $\langle t_0 \dots t_n \rangle$ that brings the net from its initial marking m_0 to a marking in which no transitions are enabled: $\exists^1_{m_1 \dots m_{n+1}} \forall_{0 \leq i \leq n} t_i \in E(m_i) \wedge m_{i+1} = m_i + t^\bullet - {}^\bullet t \wedge E(m_{n+1}) = \emptyset$. A corresponding *trace* of the net is a sequence of activities that denotes the visible behaviour of the net: $\langle a_i \mid a_i = \ell(t_j) \wedge \ell(t_j) \neq \tau \rangle$. The probability of the run is $\prod_{0 \leq i \leq n} \mathbb{P}_{m_i}(t_i)$.

In this paper, we consider labelled stochastic Petri nets where each marking that is reachable from the initial marking m_0 has a bounded number of tokens. We refer to such nets as bounded stochastic Petri net-based processes (bounded stochastic PNPs). Figure 1 shows an example of a bounded stochastic PNP. The circles are places, while the boxes are transitions. Silent transitions are filled boxes.

Definition 2 (DFA, acceptance)**.** *A* deterministic finite-state automaton *(DFA) over an alphabet Σ is a tuple $A = \langle S, s_0, S_f, \delta \rangle$, where: (i) S is a finite set of* states*, with $s_0 \in S$ the* initial state *and $S_f \subseteq S$ the set of* final states*; (ii) $\delta : S \times \Sigma \to S$ is a transition function that, given a state $s \in S$ and an activity $a \in \Sigma$, returns the successor state $\delta(s, a)$. A accepts a trace $\sigma = \langle a_0, \dots, a_n \rangle$ over $\Sigma^\star$ if there exists a sequence of states $s_0, \dots, s_{n+1}$ starting from the initial state and such that: (i) $s_{n+1} \in S_f$, and (ii) for every $0 \leq i < n$, we have $s_{i+1} = \delta(s_i, l_i)$.*

Figure 2 shows an example, stating that after every open there is a pay. This definition accounts for non-deterministic automata (NFAs), which can be encoded as DFAs. The same holds for regular expressions, LTLf/LDLf temporal formulae over finite traces [De Giacomo and Vardi, 2013b], and Declare models extended with meta-constraints [De Giacomo *et al.*, 2022]. We refer to a DFA representing a certain desired behaviour as a specification.

4 Specification Probability

In this section, we tackle a fundamental problem: *computing specification probabilities* VERIFY-PROB$(\mathcal{N}, A)$, that is, computing what the probability is that an arbitrary process instance of the bounded stochastic PNP is accepted by the specification:

Input: Bounded stochastic PNP $\mathcal{N}$, DFA A;
Output: Probability $\sum_{\sigma \text{ trace of } \mathcal{N} s.t.\ \sigma \in \mathcal{L}_A} \mathbb{P}_{\mathcal{N}}(\sigma)$.

For example, we may be interested in the probability that the bounded stochastic PNP $\mathcal{N}_{\text{order}}$ of our running example (Figure 1) evolves an order from opening to payment, encoded in the DFA shown in Figure 2.

To solve the problem, we need to account for three different aspects: *(i)* deal with the mismatch between runs over $\mathcal{N}$ and traces of A; *(ii)* single out all and only those model traces of $\mathcal{N}$ that are also traces of A; and *(iii)* compute the collective probability of all such traces.

Where runs of a bounded stochastic PNP may contain silent transitions, traces of a DFA do not. Therefore, we adapt the DFA to allow arbitrary τs. We take the input DFA A over Σ and turn it into a corresponding automaton $\bar{A}$ over $\Sigma \cup \{\tau\}$, by adding τ-labelled edges connecting every state to itself.

We now consider $RG(\mathcal{N})$ and $\bar{A}$. Since they both generate runs, we can obtain a representation of all the runs of $\mathcal{N}$ by constructing a *product* stochastic transition system that generates the runs that are common to $\mathcal{N}$ and $\bar{A}$, which in turn are the runs of $\mathcal{N}$ that induce traces of A. This can be done by the usual product automaton construction, with the only difference that we need to retain the stochastic information coming from $\mathcal{N}$. This is straightforward as $\bar{A}$ does not contain probabilities. (For a formal definition, please refer to [Leemans *et al.*, 2022].) Such a product system is not necessarily a complete stochastic transition system, as there may be states whose successor probabilities do not sum to one; correcting this is not necessary for the consequent computation (as the state variable for such a sink state would be equal to 0).

Lastly, we reduce the VERIFY-PROB problem to the problem of computing the probability that a marking from a set F

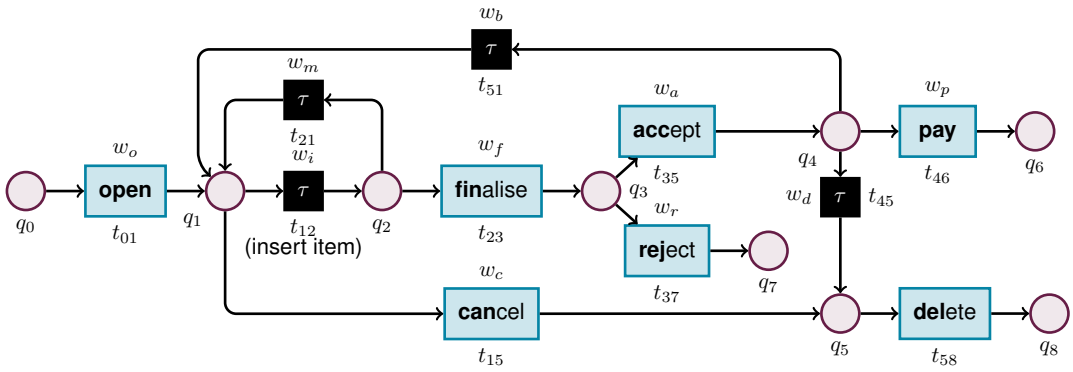

Figure 1: Example of a bounded stochastic PNP ($\mathcal{N}_{\text{order}}$) of an order-to-cash process. Transition t_{12} denotes the insertion of an item, but as it is not logged, it has been modelled using a silent transition. From [Leemans *et al.*, 2022].

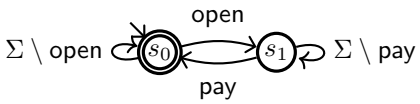

Figure 2: DFA, stating that after every open there is a pay.

is reached (OUTCOME-PROB):

Input: Bounded stochastic PNP $\mathcal{N}$, set $F \subseteq M_f$ of desired final states;

Output: $\mathbb{P}_{\mathcal{N}}(F|m_0) = \sum_{\eta \text{ run of } \mathcal{N} \text{ ending in } m \in F} \mathbb{P}_{\mathcal{N}}(\eta)$.

OUTCOME-PROB cannot be obtained through enumeration of the potentially infinitely many runs. Instead, we build upon the connection between bounded stochastic PNPs and discrete-time Markov chains [Durrett, 2012], lifting [Marsan *et al.*, 1984] to our setting[1]. We exploit that, noticing that the OUTCOME-PROB problem corresponds to the problem of calculating *exit distributions* in a discrete-time Markov chain [Durrett, 2012] (also called the problem of calculating *absorption/hit probabilities* [Fewster, 2008]). To analytically solve the problem, we create a system of equations, starting from the reachability graph. Specifically, each state of the reachability graph gets a variable x_{s_i} denoting the probability of reaching one of the states in F from that state. After solving, x_{s_0} contains the solution of the problem. The equations define the value of one of a state variable x_s as follows:

Base case if s has no successor states, then $x_{s_i} = 1$ if s corresponds to a final marking, otherwise $x_{s_i} = 0$;

Inductive case if s has successors, x_{s_i} is equal to sum of the state variables of its successor states, weighted by the transition probability to move to that successor.

There is always at least a solution, though there may be infinitely many, requiring to pick the least committing (i.e., minimal non-negative) solution. The latter case happens when $\mathcal{N}$ contains livelock markings. For correctness, uniqueness of solutions in the case of livelock-free systems, and techniques to suitably handle livelocks, please refer to [Leemans *et al.*, 2023].

[1]In case of generalised stochastic Petri nets, the resulting discrete-time Markov chain is the so-called *embedded/jump* chain obtained from the continuous-time Markov chain capturing the execution semantics of the net [Molloy, 1982; Marsan *et al.*, 1984].

Example 1. *Consider bounded stochastic PNP $\mathcal{N}_{order}$ (Figure 1). We want to solve the problem* OUTCOME-PROB$(\mathcal{N}_{order}, [q_6])$ *to compute the probability that a created order eventually completes the process by being paid (Figure 2). To do so, we solve the following equations, derived from its reachability graph (shown in [Leemans* et al.*, 2022]) into:*

$$
\begin{array}{lll}
x_{s_8} = 0 & x_{s_5} = x_{s_8} & x_{s_2} = \rho_m x_{s_1} + \rho_f x_{s_3} \\
x_{s_7} = 0 & x_{s_4} = \rho_b x_{s_1} + \rho_d x_{s_5} + \rho_p x_{s_6} & x_{s_1} = \rho_i x_{s_2} + \rho_c x_{s_5} \\
x_{s_6} = 1 & x_{s_3} = \rho_a x_{s_4} + \rho_r x_{s_7} & x_{s_0} = x_{s_1}
\end{array}
$$

This yields $x_{s_0} = \frac{\rho_i\rho_f\rho_a\rho_p x_{s_6} + \rho_i\rho_f\rho_r x_{s_7} + (\rho_i\rho_f\rho_a\rho_d + \rho_c)x_{s_8}}{1 - \rho_i\rho_m - \rho_i\rho_f\rho_a\rho_b} = \frac{\rho_i\rho_f\rho_a\rho_p}{1 - \rho_i\rho_m - \rho_i\rho_f\rho_a\rho_b}$*, which is the only solution. If we assume that the weights of $\mathcal{N}_{order}$ are all equal, the probability distributions for choosing the next transition are all uniform, leading to* $\rho_i = \rho_f = \rho_m = \rho_a = \frac{1}{2}$ *and* $\rho_p = \rho_b = \frac{1}{3}$*, and, in turn, that the probability of completing the process by paying the order is* $x_{s_0} = \frac{1}{17} \sim 0.06$*. With an analogous approach, we can prove that the probability that an order gets deleted is* $\frac{13}{17}$*, and the one that an order gets rejected is* $\frac{3}{17}$*. Notice that the sum of all such probabilities is, as expected, 1, that is, every order gets paid, deleted or rejected.*

5 Conclusion

Stochastic Petri nets have been used extensively to model business processes. In this paper, we introduced a method to compute the probability that such a net produces a trace that satisfies a given specification.

Notably, solving such a problem under our working hypotheses comes with two side results. On the one hand, we obtain (to the best of our knowledge) the first technique for model checking stochastic systems against qualitative properties expressed in Linear Temporal Logic and Linear Dynamic Logic over finite traces. On the other hand, we give an alternative solution to an existing problem related to the removal of silent transitions in stochastic finite-state automata, so far only solved through ad-hoc algorithms [Hanneforth and De La Higuera, 2010].

Ethical Statement

There are no ethical issues.

Acknowledgements

This research has been partially supported by the UNIBZ projects ADAPTERS, SMART-APP, REKAP, VERBA and DUB, as well as the PRIN MIUR project PINPOINT Prot. 2020FNEB27.

References

[Ajmone Marsan *et al.*, 1995] Marco Ajmone Marsan, Gianfranco Balbo, Gianni Conte, Susanna Donatelli, and Giuliana Franceschinis. *Modelling with Generalized Stochastic Petri Nets*. Wiley Series in Parallel Computing. John Wiley and Sons, 1995.

[Ajmone Marsan, 1988] Marco Ajmone Marsan. Stochastic petri nets: an elementary introduction. In *Advances in Petri Nets 1989*, volume 424 of *LNCS*, pages 1–29. Springer, 1988.

[Alkhammash *et al.*, 2022] Hanan Alkhammash, Artem Polyvyanyy, Alistair Moffat, and Luciano García-Bañuelos. Entropic relevance: A mechanism for measuring stochastic process models discovered from event data. *Inf. Syst.*, 107:101922, 2022.

[Alman *et al.*, 2022] Anti Alman, Fabrizio Maria Maggi, Marco Montali, and Rafael Peñaloza. Probabilistic declarative process mining. *Inf. Syst.*, 109:102033, 2022.

[Baier and Katoen, 2008] Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT Press, 2008.

[Bergami *et al.*, 2021a] Giacomo Bergami, Fabrizio Maria Maggi, Marco Montali, and Rafael Peñaloza. Probabilistic trace alignment. In *ICPM*, pages 9–16. IEEE, 2021.

[Bergami *et al.*, 2021b] Giacomo Bergami, Fabrizio Maria Maggi, Marco Montali, and Rafael PeÃ±aloza. A tool for probabilistic trace alignments. In *CAiSE Forum*. Springer, 2021.

[Bruni *et al.*, 2019] Roberto Bruni, Hernán C. Melgratti, and Ugo Montanari. Concurrency and probability: Removing confusion, compositionally. *Log. Methods Comput. Sci.*, 15(4), 2019.

[Burke *et al.*, 2020] Adam Burke, Sander J. J. Leemans, and Moe T. Wynn. Stochastic process discovery by weight estimation. In *PQMI*, 10 2020.

[Burke *et al.*, 2021] Adam Burke, Sander J. J. Leemans, and Moe Thandar Wynn. Discovering stochastic process models by reduction and abstraction. In *Application and Theory of Petri Nets and Concurrency*, volume 12734 of *Lecture Notes in Computer Science*, pages 312–336. Springer, 2021.

[Burke *et al.*, 2022] Adam Burke, Sander J. J. Leemans, Moe Thandar Wynn, Wil M. P. van der Aalst, and Arthur H. M. ter Hofstede. Stochastic process model-log quality dimensions: An experimental study. In *Proceedings of the 4th International Conference on Process Mining, (ICPM 2022)*, pages 80–87. IEEE, 2022.

[Bustan *et al.*, 2004] Doron Bustan, Sasha Rubin, and Moshe Y. Vardi. Verifying omega-regular properties of markov chains. In *Proceedings of the 16th International Conference on Computer Aided Verification*, volume 3114 of *Lecture Notes in Computer Science*, pages 189–201. Springer, 2004.

[Chiola *et al.*, 1993] Giovanni Chiola, Marco Ajmone Marsan, Gianfranco Balbo, and Gianni Conte. Generalized stochastic Petri nets: A definition at the net level and its implications. *IEEE Trans. on soft. eng.*, 19(2):89–107, 1993.

[De Giacomo and Vardi, 2013a] Giuseppe De Giacomo and Moshe Y. Vardi. Linear temporal logic and linear dynamic logic on finite traces. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI 2013)*, pages 854–860. IJCAI/AAAI, 2013.

[De Giacomo and Vardi, 2013b] Giuseppe De Giacomo and Moshe Y. Vardi. Linear temporal logic and linear dynamic logic on finite traces. In *Proceedings IJCAI*. AAAI Press, 2013.

[De Giacomo *et al.*, 2014] Giuseppe De Giacomo, Riccardo De Masellis, and Marco Montali. Reasoning on LTL on finite traces: Insensitivity to infiniteness. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI 2014)*, pages 1027–1033. AAAI Press, 2014.

[De Giacomo *et al.*, 2022] Giuseppe De Giacomo, Riccardo De Masellis, Fabrizio Maria Maggi, and Marco Montali. Monitoring constraints and metaconstraints with temporal logics on finite traces. *ACM TOSEM*, 2022.

[Durrett, 2012] Richard Durrett. *Essentials of Stochastic Processes - 2nd Edition*. Springer, 2012.

[Fewster, 2008] Rachel Fewster. *Stochastic Processes*. Course Notes Stas 325. University of Auckland, 2008.

[Grinstead and Snell, 1997] Charles Miller Grinstead and J. Laurie Snell. *Introduction to Probability*. American Mathematical Society, 2nd edition, 1997.

[Hanneforth and De La Higuera, 2010] Thomas Hanneforth and Colin De La Higuera. Epsilon-removal by loop reduction for finite-state automata over complete semirings. *Studia Grammatica*, 72:297–312, 2010.

[Leemans and Polyvyanyy, 2020] Sander J. J. Leemans and Artem Polyvyanyy. Stochastic-aware conformance checking: An entropy-based approach. In *Advanced Information Systems Engineering*, pages 217–233. Springer, 2020.

[Leemans *et al.*, 2019] Sander J. J. Leemans, Anja F. Syring, and Wil M. P. van der Aalst. Earth movers' stochastic conformance checking. In *Proceedings of the BPM Forum*, volume 360, pages 127–143, 2019.

[Leemans *et al.*, 2021] Sander J. J. Leemans, Wil M. P. van der Aalst, Tobias Brockhoff, and Artem Polyvyanyy. Stochastic process mining: Earth movers' stochastic conformance. *Inf. Syst.*, 102:101724, 2021.

[Leemans *et al.*, 2022] Sander J. J. Leemans, Fabrizio Maria Maggi, and Marco Montali. Reasoning on labelled petri

nets and their dynamics in a stochastic setting. In *BPM*, volume 13420 of *LNCS*, pages 324–342. Springer, 2022.

[Leemans *et al.*, 2023] Sander J. J. Leemans, Fabrizio M. Maggi, and Marco Montali. Enjoy the silence: Analysis of stochastic petri nets with silent transitions. In *cs.LO 2306.06376*. arXiv, 2023.

[Marsan *et al.*, 1984] Marco Ajmone Marsan, Gianni Conte, and Gianfranco Balbo. A class of generalized stochastic Petri nets for the performance evaluation of multiprocessor systems. *ACM TOCS*, 2(2):93–122, 1984.

[Molloy, 1982] Michael Karl Molloy. Performance analysis using stochastic petri nets. *IEEE Transactions on Computers*, 31:913–917, 1982.

[Rogge-Solti *et al.*, 2013] Andreas Rogge-Solti, Wil M. P. van der Aalst, and Mathias Weske. Discovering stochastic Petri nets with arbitrary delay distributions from event logs. In *BPMW13*, pages 15–27, 2013.

[van der Aalst, 2016] Wil M. P. van der Aalst. *Process Mining - Data Science in Action, Second Edition*. Springer, 2016.