





Compositional Reasoning for Parametric Probabilistic Automata

Hannah Mertens¹  

RWTH Aachen University, Germany

Tim Quatmann  

RWTH Aachen University, Germany

Joost-Pieter Katoen  

RWTH Aachen University, Germany

Abstract

We establish an assume-guarantee (AG) framework for compositional reasoning about multi-objective queries in parametric probabilistic automata (pPA) – an extension to probabilistic automata (PA), where transition probabilities are functions over a finite set of parameters. We lift an existing framework for PA to the pPA setting, incorporating asymmetric, circular, and interleaving proof rules. Our approach enables the verification of a broad spectrum of multi-objective queries for pPA, encompassing probabilistic properties and (parametric) expected total rewards. Additionally, we introduce a rule for reasoning about monotonicity in composed pPAs.

2012 ACM Subject Classification Theory of computation → Logic

Keywords and phrases Verification, Probabilistic systems, Assume-guarantee reasoning, Parametric Probabilistic Automata, Parameter synthesis

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2025.31

Related Version *Full Version*: <https://arxiv.org/abs/2506.08525> [38]

Funding *Tim Quatmann*: This research was funded by a KI-Starter grant from the Ministerium für Kultur und Wissenschaft NRW.

1 Introduction

Probabilistic Model Checking [21, 31] studies the automated verification of Markov models for systems with random behavior. Applications include network and security protocols, biochemical processes, and planning under uncertainty [40, 33, 20]. Common properties such as reachability probabilities in Markov decision processes (MDPs) can often be verified efficiently in PTIME [3].

When the probabilities with which the system evolves are not known exactly, verification results must be robust towards slight perturbations. *Parametric Markov models* [14, 28] allow representing uncertain model quantities – for example the bias of a coin-flip or the probability of a sensor misreading – using parameters. *Feasibility* is a fundamental verification problem for parametric systems and asks whether there is an instantiation of the parameters under which a given specification holds. Deciding feasibility for reachability probability specifications in parametric MDPs is ETR-complete, i.e., at least NP-hard and within PSPACE [29]. The dual *verification* problem that asks if the specification holds under all instantiations is co-ETR complete. Checking parametric Markov models is therefore significantly more complex compared to Markov models without parameters.

¹ Corresponding author



© Hannah Mertens, Tim Quatmann, and Joost-Pieter Katoen;
licensed under Creative Commons License CC-BY 4.0

36th International Conference on Concurrency Theory (CONCUR 2025).

Editors: Patricia Bouyer and Jaco van de Pol; Article No. 31; pp. 31:1–31:20



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

The number of system states grows exponentially with the number of system components. The resulting *state-space explosion* is an omnipresent challenge when model checking complex systems, often rendering analysis computationally infeasible. Compositional verification techniques such as *assume-guarantee* (AG) reasoning [26, 44] address this problem by decomposing the verification task into smaller sub-tasks that consider individual components in isolation. This modular verification approach has been successfully applied in various domains, including service-based workflow verification [6], large-scale IT systems [7], and autonomous systems incorporating deep neural networks [42, 43]. Recent advancements include circular AG reasoning [16] and verification-repair techniques [22]. Extensions to probabilistic systems have further expanded the scope of AG reasoning, as demonstrated in works such as [35], where AG reasoning was applied to Segala’s probabilistic automata (PA) [47] – a compositional extension of Markov decision processes. Automated approaches to AG reasoning for probabilistic systems have also been explored [19, 36], enabling more scalable verification. Other works have considered parametric, but non-probabilistic timed automata [2] as well as parameterized programs [4, 48, 39] – where the concurrent system is parameterized by the number of processes or threads in a configured instance.

This work introduces a framework for compositional reasoning about parametric probabilistic automata (pPA). The case studies presented by Kwiatkowska et al. [35] demonstrate the practical applicability of AG reasoning within a non-parametric setting. These findings provide strong motivation for extending this approach to the parametric domain. In this work, we develop the theoretical foundations of an AG reasoning framework for pPAs, leveraging results from (non-parametric) PAs [35]. Due to the aforementioned ETR-hardness, compositional reasoning has a large potential and can be crucial to verify complex parameterized systems that are too large to handle monolithically.

► **Example 1.** Consider a communication system, where a *sender* \mathcal{S} broadcasts messages to a *receiver* \mathcal{R} through a *broadcast channel* \mathcal{B} . The system is modeled by the parallel composition $\mathcal{S} \parallel \mathcal{B} \parallel \mathcal{R}$. The components are faulty: \mathcal{S} might face a collision, broadcasting in \mathcal{B} might fail due to message loss, and \mathcal{R} might miss broadcasts. The reliability of \mathcal{S} , \mathcal{B} , and \mathcal{R} is influenced by parameters and the precise values of these parameters vary depending on network conditions, interference, or other factors. Our goal is to verify that under all parameter instantiations in a given parameter space R , the message is successfully *received* with at least probability 0.7, formally denoted by

$$(\mathcal{S} \parallel \mathcal{B} \parallel \mathcal{R}), R \models \mathbb{P}_{\geq 0.7}(\text{received}).$$

AG reasoning allows to verify the specification without explicitly considering the (potentially large) composition $\mathcal{S} \parallel \mathcal{B} \parallel \mathcal{R}$. To this end, assume that we have established the following statements:

- $\mathcal{S}, R \models \mathbb{P}_{< 0.1}(\text{collision})$ – the probability that \mathcal{S} faces a collision is below 0.1
- $\mathcal{B}, R \models \mathbb{P}_{< 0.1}(\text{collision}) \rightarrow \mathbb{P}_{\geq 0.8}(\text{broadcast})$ – if \mathcal{B} observes a collision with probability below 0.1, the message is broadcast with probability at least 0.8
- $\mathcal{R}, R \models \mathbb{P}_{\geq 0.8}(\text{broadcast}) \rightarrow \mathbb{P}_{\geq 0.7}(\text{received})$ – If the message is broadcast with probability at least 0.8, then \mathcal{R} receives the message with probability at least 0.7

We reason about the composed system using the AG rule stated in Theorem 33 in Section 5:

$$\frac{\mathcal{S}, R \models \mathbb{P}_{< 0.1}(\text{collision}) \quad \mathcal{B}, R \models \mathbb{P}_{< 0.1}(\text{collision}) \rightarrow \mathbb{P}_{\geq 0.8}(\text{broadcast})}{(\mathcal{S} \parallel \mathcal{B}), R \models \mathbb{P}_{\geq 0.8}(\text{broadcast})} \quad \frac{(\mathcal{S} \parallel \mathcal{B}), R \models \mathbb{P}_{\geq 0.8}(\text{broadcast}) \quad \mathcal{R}, R \models \mathbb{P}_{\geq 0.8}(\text{broadcast}) \rightarrow \mathbb{P}_{\geq 0.7}(\text{received})}{(\mathcal{S} \parallel \mathcal{B} \parallel \mathcal{R}), R \models \mathbb{P}_{\geq 0.7}(\text{received})}$$

Contributions. To the best of our knowledge, *we provide the first framework for compositional reasoning of parametric Markov models*. Our main contributions are as follows.

- We introduce and formalize pPAs, i.e., compositional probabilistic automata with parametric transitions.
- We provide a conservative extension of *strategy projections* [46, 35] to pPAs, including a more natural definition based on conditional probabilities. Strategy projections are essential for correctness of compositional reasoning as they allow to link measures of a composed model to measures of its constituting components.
- We present rules for assume-guarantee reasoning, generalizing an established framework by Kwiatkowska et al. [35] to the parametric setting – which requires some technically intricate proofs. The framework applies to ω -regular and expected total reward properties as well as multi-objective combinations thereof.
- We provide a new rule for compositional reasoning about monotonicity in pPAs. Knowing that a measure of interest – either the probability to satisfy an ω -regular specification or an expected total reward – is monotone in one or more parameters can significantly speed up verification [28, 50]. Our rule allows to derive monotonicity w.r.t. a composed pPA by only determining monotonicity for its components.

We introduce pPAs in Section 2 and discuss strategy projections in Section 3. Section 4 outlines properties of interest and Section 5 presents our AG rules. We outline results for monotonicity in Section 6 and related work in Section 7. Section 8 concludes the paper. Proofs omitted in the main part of the paper are given in the extended version [38].

2 Preliminaries

For sets X and Y , let $f: X \hookrightarrow Y$ denote a *partial function* from X to Y with domain $\text{dom}(f) \subseteq X$. The projection of f to a set Z is written as $f|_Z: (X \cap Z) \rightarrow Y$. Iverson brackets $\llbracket \varphi \rrbracket \in \{0, 1\}$ map a Boolean condition φ to 1 if φ holds and 0 otherwise.

$\mathbb{Q}[V]$ denotes the set of (multivariate) *polynomials* over a finite set of real-valued *parameters* $V = \{p_1, \dots, p_n\}$. A (parameter) *valuation* for V is a function $v: V \rightarrow \mathbb{R}$. Evaluating a polynomial $f \in \mathbb{Q}[V]$ at v yields $f[v] \in \mathbb{R}$. A *region* R for V is a set of valuations. For $p \in V$, we define the valuation e_p with $e_p(q) = \llbracket p=q \rrbracket$ for $q \in V$.

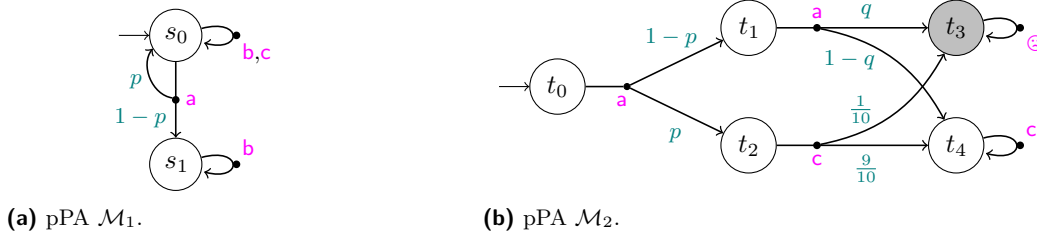
A *parametric distribution*² for V over a finite set S is a function $\mu: S \rightarrow (\mathbb{Q}[V] \cup \mathbb{R})$. Applying valuation v to μ yields $\mu[v]: S \rightarrow \mathbb{R}$ with $\mu[v](s) = \mu(s)[v]$ for all $s \in S$. We call $\mu: S \rightarrow [0, 1]$ a *subdistribution* if $\sum_{s \in S} \mu(s) \leq 1$ and a *distribution* if $\sum_{s \in S} \mu(s) = 1$. The sets of parametric distributions, subdistributions, and distributions over S are denoted by $\text{Dist}_V(S)$, $\text{SubDist}(S)$, and $\text{Dist}(S)$, respectively. For $s \in S$, $\mathbb{1}_s \in \text{Dist}(S)$ is the *Dirac* distribution with $\mathbb{1}_s(s') = \llbracket s'=s \rrbracket$. For sets S_1, S_2 , the *product* of $\mu_1 \in \text{Dist}_V(S_1)$ and $\mu_2 \in \text{Dist}_V(S_2)$ is the distribution $\mu_1 \times \mu_2 \in \text{Dist}_V(S_1 \times S_2)$ with $(\mu_1 \times \mu_2)(s_1, s_2) = \mu_1(s_1) \cdot \mu_2(s_2)$.

2.1 Parametric Probabilistic Automata

We combine probabilistic automata (PA) [47, 52] with parametric Markov models [28].

► **Definition 2.** A parametric probabilistic automaton (pPA) over a finite alphabet Σ is a tuple $\mathcal{M} = (S, s^{\text{init}}, V, \text{Act}, \mathbf{P}, L)$, where S , V , and Act are finite sets of states, parameters, and actions, respectively, $s^{\text{init}} \in S$ is an initial state, $\mathbf{P}: (S \times \text{Act}) \hookrightarrow \text{Dist}_V(S)$ is a parametric transition function, and $L: \text{dom}(\mathbf{P}) \rightarrow \Sigma$ is a labeling function.

² We use the term *parametric distribution* – rather than *parametric function* – to emphasize that we are typically interested in functions μ where $\mu[v]$ is a (sub)probability distribution.



■ **Figure 1** Example pPAs \mathcal{M}_1 and \mathcal{M}_2 .

Let $\mathcal{M} = (S, s^{init}, V, Act, \mathbf{P}, L)$ be a pPA. For $s \in S$, $Act(s) = \{\alpha \in Act \mid (s, \alpha) \in \text{dom}(\mathbf{P})\}$ denotes the set of enabled actions in s and \mathcal{M}_s is the pPA where the initial state is set to s . We set $\mathbf{P}(s, \alpha, s') = \mathbf{P}(s, \alpha)(s')$ if $(s, \alpha) \in \text{dom}(\mathbf{P})$ and otherwise $\mathbf{P}(s, \alpha, s') = 0$. \mathcal{M} is a (*non-parametric*) PA if $\mathbf{P}(s, \alpha) \in \text{Dist}(S)$ for all $(s, \alpha) \in \text{dom}(\mathbf{P})$. In this case, $\mathbf{P}(s, \alpha, s')$ is the probability to transition to successor state s' when action α is selected at state s .

The *instantiation* of \mathcal{M} at valuation v for V is the pPA $\mathcal{M}[v] = (S, s^{init}, \emptyset, Act, \mathbf{P}[v], L)$, where $\text{dom}(\mathbf{P}[v]) = \text{dom}(\mathbf{P})$ and $\mathbf{P}[v](s, \alpha) = \mathbf{P}(s, \alpha)[v]$. If $\mathcal{M}[v]$ is a non-parametric PA, we say v is *well-defined* for \mathcal{M} . A valuation v is *graph-preserving* for \mathcal{M} if it is well-defined and for all $s, s' \in S$ and $\alpha \in Act_{\mathcal{M}}$: $\mathbf{P}(s, \alpha, s')[v] = 0$ iff $\mathbf{P}(s, \alpha, s') = 0$. A region R is well-defined (graph-preserving) if all its valuations $v \in R$ are well-defined (graph-preserving).

► **Example 3 (pPA).** Consider the pPA \mathcal{M}_1 in Figure 1a and \mathcal{M}_2 in Figure 1b. $\mathcal{M}_1 = (S_1, s_1^{init}, V_1, Act_1, \mathbf{P}_1, L_1)$, where $S_1 = \{s_0, s_1\}$, $s_1^{init} = s_0$ and $V_1 = \{p\}$. Actions $Act_1 = \{a, b, c\}$ and alphabet $\Sigma_1 = \{a, b, c\}$. In this example, the alphabet coincides with actions as these uniquely define the transitions. Similarly, $\mathcal{M}_2 = (S_2, s_2^{init}, V_2, Act_2, \mathbf{P}_2, L_2)$, where $S_2 = \{t_0, \dots, t_4\}$, $s_2^{init} = t_0$, and $V_2 = \{p, q\}$. Again, $Act_2 = \Sigma_2 = \{a, c, \otimes\}$.

An *infinite path* of \mathcal{M} is an alternating sequence $\pi = s_0, \alpha_0, s_1, \alpha_1, \dots$ of states $s_i \in S$ and actions $\alpha_i \in Act$ such that $(s_i, \alpha_i) \in \text{dom}(\mathbf{P})$ for all $i \geq 0$. A finite path of length $n \in \mathbb{N}$ is a prefix $\hat{\pi} = s_0, \alpha_0, \dots, s_n$ of an infinite path, ending in a state $\text{last}(\hat{\pi}) = s_n \in S$. $\text{Paths}_{\mathcal{M}}^{inf}$ and $\text{Paths}_{\mathcal{M}}^{fin}$ are the sets of infinite and finite paths of \mathcal{M} , respectively. For a (finite or infinite) path $\pi \in \text{Paths}_{\mathcal{M}}^{inf} \cup \text{Paths}_{\mathcal{M}}^{fin}$, we write $|\pi| \in \mathbb{N} \cup \{\infty\}$ for its length and $\pi[0, j]$ for its prefix of length $j \leq |\pi|$. We deliberately allow paths that take transitions with probability 0. As a consequence, a path of a pPA \mathcal{M} is always also a path of any of its instantiations $\mathcal{M}[v]$ – even if v is not graph-preserving.

Strategies – also known as schedulers or adversaries – resolve nondeterminism by assigning (sub-)distributions over enabled actions based on the history – i.e., a finite path – observed so far. We allow for partial strategies that, intuitively, can choose none of the enabled actions to reflect the case that no further transition is executed.

► **Definition 4.** A (partial) strategy for \mathcal{M} is a function $\sigma: \text{Paths}_{\mathcal{M}}^{fin} \rightarrow \text{SubDist}(Act)$ such that $\sigma(\hat{\pi})(\alpha) > 0$ implies $(\text{last}(\hat{\pi}), \alpha) \in \text{dom}(\mathbf{P})$. A strategy $\sigma: \text{Paths}_{\mathcal{M}}^{fin} \rightarrow \text{Dist}(Act_{\mathcal{M}})$ is called complete. The set of all partial and complete strategies on \mathcal{M} are denoted by $\text{Str}_{\mathcal{M}}^*$, where $\star \in \{\text{prt}, \text{cmp}\}$, respectively. A memoryless strategy only depends on the last state of $\hat{\pi}$. The set of memoryless strategies on \mathcal{M} is denoted by $\text{Str}_{\mathcal{M}}^{\text{mless}, \star}$.

For a strategy σ for \mathcal{M} , we may write $\sigma(\hat{\pi}, \alpha)$ instead of $\sigma(\hat{\pi})(\alpha)$. If σ is memoryless, we write $\sigma(s_n, \alpha)$ instead of $\sigma(\hat{\pi}, \alpha)$, where $s_n = \text{last}(\hat{\pi})$.

A well-defined instantiation v and a strategy σ for \mathcal{M} yield a purely probabilistic process described by the (sub)probability measure $Pr_{\mathcal{M}}^{v,\sigma}$ on the measurable subsets of $Paths_{\mathcal{M}}^{inf}$, which is obtained by a standard cylinder set construction [3]:

$$\text{Cyl}(\hat{\pi}) = \{\pi \in Paths_{\mathcal{M}}^{inf} \mid \hat{\pi} \text{ is a prefix of } \pi\}$$

is the cylinder set of a finite path $\hat{\pi} = s_0, \alpha_0, \dots, s_n$ of \mathcal{M} and we set

$$Pr_{\mathcal{M}}^{v,\sigma}(\text{Cyl}(\hat{\pi})) = \llbracket s_0 = s^{init} \rrbracket \cdot \prod_{i=0}^{n-1} \sigma(\hat{\pi}[0, i], \alpha_i) \cdot \mathbf{P}(s_i, \alpha_i, s_{i+1})[v].$$

This definition extends uniquely to a probability measure on *all* measurable sets of infinite paths. We further lift $Pr_{\mathcal{M}}^{v,\sigma}$ to (sets of) finite paths and write, e.g., $Pr_{\mathcal{M}}^{v,\sigma}(\hat{\pi})$ for $\hat{\pi} \in Paths_{\mathcal{M}}^{fin}$ or $Pr_{\mathcal{M}}^{v,\sigma}(\Pi)$ for $\Pi \subseteq Paths_{\mathcal{M}}^{fin}$ – implicitly referring to (unions of) cylinder sets. If \mathcal{M} is a (non-parametric) PA, we may omit v and write $Pr_{\mathcal{M}}^{\sigma}$. Well-defined v yields $Pr_{\mathcal{M}[v]}^{\sigma} = Pr_{\mathcal{M}}^{v,\sigma}$.

► **Example 5.** For the pPA \mathcal{M}_2 from Figure 1b and a well-defined valuation v , the probability to reach the state t_3 under valuation v is $(1 - v(p)) \cdot v(q) + v(p) \cdot \frac{1}{10}$.

► **Remark 6.** Our definition of PA slightly deviates from related work [47, 35, 32, 36], which commonly define a transition relation $\delta \subseteq S \times \Sigma \times Dist(S)$ instead of functions \mathbf{P} and L . In our setting, a pair $(s, \alpha) \in \text{dom}(\mathbf{P})$ uniquely identifies both, a label $L(s, \alpha) \in \Sigma$, and a distribution over successor states $\mathbf{P}(s, \alpha) \in Dist(S)$, which significantly simplifies formalizations related to pPAs. In particular, any strategy for \mathcal{M} immediately also applies to instantiations of \mathcal{M} and vice versa, i.e., we have $Str_{\mathcal{M}} = Str_{\mathcal{M}[v]}$ for any valuation v . On the other hand, our variant does not affect expressiveness of non-parametric PA as one can convert between the two formalisms.

We lift parallel composition of PA [47] to pPAs. Composed pPAs synchronize on common transition labels while behaving autonomously on non-common labels. For simplicity, we assume that composed pPAs consider a common set of parameters V .³

► **Definition 7 (Parallel Composition).** For $i = 1, 2$, let $\mathcal{M}_i = (S_i, s_i^{init}, V, Act_i, \mathbf{P}_i, L_i)$ be two pPAs over alphabets Σ_i with $Act_i \cap (\Sigma_1 \cup \Sigma_2) = \emptyset$. The parallel composition of \mathcal{M}_1 and \mathcal{M}_2 is given by the pPA $\mathcal{M}_1 \parallel \mathcal{M}_2 = (S_1 \times S_2, (s_1^{init}, s_2^{init}), V, Act_{\parallel}, \mathbf{P}_{\parallel}, L_{\parallel})$ over $\Sigma_1 \cup \Sigma_2$, where

- $Act_{\parallel} = (Act_1 \times Act_2) \cup (Act_1 \times \Sigma_1 \setminus \Sigma_2) \cup (\Sigma_2 \setminus \Sigma_1 \times Act_2)$,
- for each $(s_1, \alpha_1) \in \text{dom}(\mathbf{P}_1)$, $(s_2, \alpha_2) \in \text{dom}(\mathbf{P}_2)$ with $L_1(s_1, \alpha_1) = L_2(s_2, \alpha_2) \in \Sigma_1 \cap \Sigma_2$:

$$\mathbf{P}_{\parallel}((s_1, s_2), (\alpha_1, \alpha_2)) = \mathbf{P}_1(s_1, \alpha_1) \times \mathbf{P}_2(s_2, \alpha_2) \quad \text{and} \quad L_{\parallel}((s_1, s_2), (\alpha_1, \alpha_2)) = L_1(s_1, \alpha_1),$$

- for each $(s_1, \alpha_1) \in \text{dom}(\mathbf{P}_1)$, $s_2 \in S_2$ with $L_1(s_1, \alpha_1) = \mathbf{a}_1 \in \Sigma_1 \setminus \Sigma_2$:

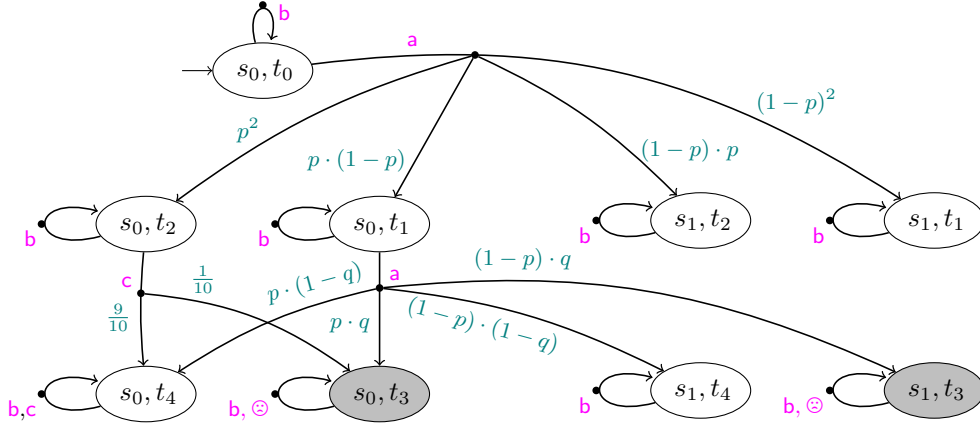
$$\mathbf{P}_{\parallel}((s_1, s_2), (\alpha_1, \mathbf{a}_1)) = \mathbf{P}_1(s_1, \alpha_1) \times \mathbb{1}_{s_2} \quad \text{and} \quad L_{\parallel}((s_1, s_2), (\alpha_1, \mathbf{a}_1)) = \mathbf{a}_1 = L_1(s_1, \alpha_1),$$

- for each $s_1 \in S_1$, $(s_2, \alpha_2) \in \text{dom}(\mathbf{P}_2)$ with $L_2(s_2, \alpha_2) = \mathbf{a}_2 \in \Sigma_2 \setminus \Sigma_1$:

$$\mathbf{P}_{\parallel}((s_1, s_2), (\mathbf{a}_2, \alpha_2)) = \mathbb{1}_{s_1} \times \mathbf{P}_2(s_2, \alpha_2) \quad \text{and} \quad L_{\parallel}((s_1, s_2), (\mathbf{a}_2, \alpha_2)) = \mathbf{a}_2 = L_2(s_2, \alpha_2).$$

The parallel composition of parametric probabilistic automata (pPAs) is associative, meaning that $(\mathcal{M}_1 \parallel \mathcal{M}_2) \parallel \mathcal{M}_3$ and $\mathcal{M}_1 \parallel (\mathcal{M}_2 \parallel \mathcal{M}_3)$ are equivalent up to state renaming. Therefore, we denote this composition as $\mathcal{M}_1 \parallel \mathcal{M}_2 \parallel \mathcal{M}_3$.

³ If two pPAs have different parameter sets $V_1 \neq V_2$, the assumption can be established by considering $V = V_1 \cup V_2$ instead, potentially adding (unused) parameters to the individual pPAs.



■ **Figure 2** Parallel composition of pPAs \mathcal{M}_1 and \mathcal{M}_2 from Figure 1.

► **Example 8 (Parallel Composition).** Figure 2 shows the composition of pPAs \mathcal{M}_1 and \mathcal{M}_2 from Figure 1. Actions with labels a or c are synchronized while b and \oplus are asynchronous. Similar to [35, Section 3.5], we sometimes assume fairness of strategies, meaning that specific sets of labels $\Sigma_i \subseteq \Sigma$ are visited infinitely often.

► **Definition 9 (Fair Strategy).** Let $\mathcal{C} \subseteq 2^\Sigma$ and v be a well-defined valuation for \mathcal{M} over Σ . A complete strategy $\sigma \in \text{Str}_{\mathcal{M}[v]}^{\text{cmp}}$ is fair w.r.t. \mathcal{C} (denoted $\text{fair}_{\mathcal{C}}$) if

$$\Pr_{\mathcal{M}}^{v, \sigma} \left(\{s_0, \alpha_0, s_1, \alpha_1, \dots \in \text{Paths}_{\mathcal{M}}^{\text{inf}} \mid \forall \Sigma_i \in \mathcal{C}: \forall j \in \mathbb{N}: \exists k \geq j: L(s_k, \alpha_k) \in \Sigma_i\} \right) = 1.$$

The set of all $\text{fair}_{\mathcal{C}}$ strategies of $\mathcal{M}[v]$ is denoted $\text{Str}_{\mathcal{M}[v]}^{\text{fair}_{\mathcal{C}}}$.

Almost-sure repeated reachability in PA only depends on the graph structure, which yields:

► **Proposition 10.** For any graph-preserving valuations v, v' for \mathcal{M} we have $\text{Str}_{\mathcal{M}[v]}^{\text{fair}_{\mathcal{C}}} = \text{Str}_{\mathcal{M}[v']}^{\text{fair}_{\mathcal{C}}}$, i.e., a strategy is $\text{fair}_{\mathcal{C}}$ for $\mathcal{M}[v]$ iff it is $\text{fair}_{\mathcal{C}}$ for $\mathcal{M}[v']$.

3 Strategy Projections

In this section, we define the projection of a strategy of a composite pPA $\mathcal{M} = \mathcal{M}_1 \parallel \mathcal{M}_2$ onto a single component \mathcal{M}_i for $i = 1, 2$. Projections for PA are defined in [35, Definition 6] and originate from [46, page 65, Definition of Projection]. They are intuitively used to relate probability measures for \mathcal{M} and \mathcal{M}_i .

The projection of a finite path $\pi \in \text{Paths}_{\mathcal{M}}^{\text{fin}}$ onto component \mathcal{M}_i is the finite path $\pi|_i \in \text{Paths}_{\mathcal{M}_i}^{\text{fin}}$ obtained by restricting π to the steps performed by \mathcal{M}_i . Formally, $(s_1^{\text{init}}, s_2^{\text{init}})|_i = s_i^{\text{init}}$ and for $\pi = \pi', (\alpha_1, \alpha_2), (s_1, s_2)$:

$$\pi|_i = \begin{cases} \pi'|_i, \alpha_i, s_i & \text{if } \alpha_i \in \text{Act}_i \\ \pi'|_i & \text{otherwise.} \end{cases}$$

Path projections are neither injective nor surjective, i.e., we might have $\pi|_i = \pi'|_i$ for two distinct paths $\pi \neq \pi'$ of \mathcal{M} , and for some $\pi_i \in \text{Paths}_{\mathcal{M}_i}^{\text{fin}}$ there might not be any $\pi \in \text{Paths}_{\mathcal{M}}^{\text{fin}}$ with $\pi_i = \pi|_i$. We define the set of paths of $\mathcal{M}_1 \parallel \mathcal{M}_2$ that are projected to $\pi_i \in \text{Paths}_{\mathcal{M}_i}^{\text{fin}}$ as

$$\pi_i \otimes \mathcal{M}_{3-i} = \{\pi \in \text{Paths}_{\mathcal{M}}^{\text{fin}} \mid \pi_i = \pi|_i\}.$$

We first focus on strategy projections for non-parametric PA. Then, we lift our notions to the parametric setting.

3.1 Projections for non-parametric PAs

We fix the parallel composition $\mathcal{N} = \mathcal{N}_1 \parallel \mathcal{N}_2 = (S_{\parallel}, s_{\parallel}^{init}, Act_{\parallel}, \mathbf{P}_{\parallel}, L_{\parallel})$ of (non-parametric) PAs \mathcal{N}_1 and \mathcal{N}_2 with $\mathcal{N}_i = (S_i, s_i^{init}, Act_i, \mathbf{P}_i, L_i)$ and alphabets Σ_i for $i = 1, 2$.

► **Definition 11** (Strategy Projection for PA). *The projection of a strategy $\sigma \in Str_{\mathcal{N}}^{prt}$ to \mathcal{N}_i is the strategy $\sigma|_i^{\mathcal{N}} \in Str_{\mathcal{N}_i}^{prt}$, where for $\pi_i \in Paths_{\mathcal{M}_i}^{fin}$ and $\alpha_i \in Act_i$:*

$$\sigma|_i^{\mathcal{N}}(\pi_i, \alpha_i) = \begin{cases} \frac{\sum_{s_i \in S_i} Pr_{\mathcal{N}}^{\sigma}((\pi_i, \alpha_i, s_i) \otimes \mathcal{N}_{3-i})}{Pr_{\mathcal{N}}^{\sigma}(\pi_i \otimes \mathcal{N}_{3-i})} & \text{if } Pr_{\mathcal{N}}^{\sigma}(\pi_i \otimes \mathcal{N}_{3-i}) > 0 \\ 0 & \text{otherwise.} \end{cases}$$

If \mathcal{N} is clear, we simply write $\sigma|_i$ instead of $\sigma|_i^{\mathcal{N}}$. Lemmas 12 and 13 below yield that Definition 11 is equivalent to the projection defined in [35, Def. 6]. We argue that our variant is more intuitive since the numerator and denominator of the fraction consider the same probability measure $Pr_{\mathcal{N}}^{\sigma}$. Intuitively, $\sigma|_i(\pi_i, \alpha_i)$ coincides with the conditional probability that – under $Pr_{\mathcal{N}}^{\sigma}$ and given that a path π with projection $\pi|_i = \pi_i$ is observed – the next action of component \mathcal{N}_i is α_i . We now provide an alternative characterization for the numerator given in Definition 11.

► **Lemma 12.** *For $\sigma \in Str_{\mathcal{N}}^{prt}$, $\pi_i \in Paths_{\mathcal{N}_i}^{fin}$, and $\alpha_i \in Act_i$:*

$$\sum_{s_i \in S_i} Pr_{\mathcal{N}}^{\sigma}((\pi_i, \alpha_i, s_i) \otimes \mathcal{N}_{3-i}) = \sum_{\pi \in (\pi_i \otimes \mathcal{N}_{3-i})} \sum_{(\hat{\alpha}_1, \hat{\alpha}_2) \in Act_{\parallel}, \hat{\alpha}_i = \alpha_i} Pr_{\mathcal{N}}^{\sigma}(\pi) \cdot \sigma(\pi, (\hat{\alpha}_1, \hat{\alpha}_2)).$$

The next lemma is the key observation for strategy projections as it connects the probability measure $Pr_{\mathcal{N}}^{\sigma}$ for \mathcal{N} and $Pr_{\mathcal{N}_i}^{\sigma|_i}$ for each component \mathcal{N}_i .

► **Lemma 13.** *For $\sigma \in Str_{\mathcal{N}}^{prt}$ and $\pi_i \in Paths_{\mathcal{N}_i}^{fin}$: $Pr_{\mathcal{N}_i}^{\sigma|_i}(\pi_i) = Pr_{\mathcal{N}}^{\sigma}(\pi_i \otimes \mathcal{N}_{3-i})$.*

The following result lifts [35, Lemma 2] to our setting and states that fair (and therefore also complete) strategies have fair and complete projections.

► **Lemma 14.** *Let $\mathcal{C}_1 \subseteq 2^{\Sigma_1}$ and $\mathcal{C}_2 \subseteq 2^{\Sigma_2}$. If $\sigma \in Str_{\mathcal{N}}^{fair_{\mathcal{C}_1 \cup \mathcal{C}_2}}$, then $\sigma|_i \in Str_{\mathcal{N}_i}^{fair_{\mathcal{C}_i}}$.*

The converse of Lemma 14 does not hold: The projection $\sigma|_i$ might not be a complete strategy for \mathcal{N}_i , even though σ is complete for \mathcal{N} . Furthermore, $\sigma|_i$ might not be memoryless, even if σ is memoryless. The following example shows both cases.

► **Example 15** (Strategy Projection for pPA). Consider the pPA $\mathcal{M} = \mathcal{M}_1 \parallel \mathcal{M}_2$ from Figure 2, using \mathcal{M}_1 and \mathcal{M}_2 depicted in Figure 1. We fix the valuation v with $v(p) = v(q) = 0.1$ and set $\mathcal{N} = \mathcal{M}[v]$ and $\mathcal{N}_i = \mathcal{M}_i[v]$ for $i = 1, 2$. Let $\sigma \in Str_{\mathcal{N}_1 \parallel \mathcal{N}_2}^{mless, cmp}$ that always selects the actions with label **a**, **c**, or \odot with probability 1 when available; otherwise, it chooses **b**.

We compute the projection $\sigma|_2$ to the PA $\mathcal{M}_2[v]$:

- $\sigma|_2(t_0, \mathbf{a}) = 1$
- $\sigma|_2((t_0, \mathbf{a}, t_2), \mathbf{c}) = \frac{(v(p))^2}{v(p)} = v(p) = 0.1$
- $\sigma|_2((t_0, \mathbf{a}, t_2, \mathbf{c}, t_3), \odot) = \frac{(v(p))^2 \cdot \frac{1}{10}}{v(p) \cdot \frac{1}{10}} = v(p) = 0.1$

Similarly, $\sigma|_2((t_0, \mathbf{a}, t_1), \mathbf{a}) = v(p) = 0.1$ and $\sigma|_2((t_0, \mathbf{a}, t_1, \mathbf{a}, t_3), \odot) = v(p) = 0.1$. We observe that the projection is not a complete strategy as \odot is the only available action in t_3 for \mathcal{M}_2 . Moreover, the projection is not memoryless, because, for example, $\sigma|_2((t_0, \mathbf{a}, t_2, \mathbf{c}, t_4), \mathbf{c}) = v(p) = 0.1$ is not equal to $\sigma|_2((t_0, \mathbf{a}, t_1, \mathbf{a}, t_4), \mathbf{c}) = 1$.

3.2 Projections for Parametric PAs

We now lift strategy projections to the parametric case. We fix the parallel composition $\mathcal{M} = \mathcal{M}_1 \parallel \mathcal{M}_2$ of two pPAs \mathcal{M}_1 and \mathcal{M}_2 with a common set of parameters V . Let $i = 1, 2$ and let $v_i: V \rightarrow \mathbb{R}$ be a well-defined valuation for \mathcal{M}_i . We define strategy projections for pPAs in terms of the instantiated PAs.

► **Definition 16** (Strategy Projection for pPA). *The projection of a strategy $\sigma \in \text{Str}_{\mathcal{M}}^{\text{prt}}$ to pPA \mathcal{M}_i w.r.t. v_1, v_2 is the strategy $\sigma|_i^{v_1, v_2} \in \text{Str}_{\mathcal{M}_i}^{\text{prt}}$, defined by $\sigma|_i^{v_1, v_2} = \sigma|_i^{\mathcal{M}_1[v_1] \parallel \mathcal{M}_2[v_2]}$.*

If the valuations coincide, i.e., $v_1 = v_2 = v$, we write $\sigma|_i^v$ instead of $\sigma|_i^{v_1, v_2}$. Strategy projections depend on the parameter instantiations as the following example illustrates. Such strategies are also referred to as (*parameter*) *dependent strategies* [49, Def. 2.6].

► **Example 17** (Strategy Projection for pPA). Consider $\mathcal{M}_1 \parallel \mathcal{M}_2$ from Figure 2 and the strategy σ as in Example 15. Let v_1 and v_2 be the valuations used for \mathcal{M}_1 and \mathcal{M}_2 , respectively, with $v_1(p) = v_1(q) = 0.1$, and $v_2(p) = v_2(q) = 0.9$. For the projection $\sigma|_2^{v_1, v_2}$ to \mathcal{M}_2 , the following values are obtained:

- $\sigma|_2^{v_1, v_2}(t_0, a) = 1$
- $\sigma|_2^{v_1, v_2}((t_0, a, t_2), c) = \frac{v_1(p) \cdot v_2(p)}{v_2(p)} = v_1(p) = 0.1$
- $\sigma|_2^{v_1, v_2}((t_0, a, t_2, c, t_3), \odot) = \frac{v_1(p) \cdot v_2(p) \cdot \frac{1}{10}}{v_2(p) \cdot \frac{1}{10}} = v_1(p) = 0.1$,
- $\sigma|_2^{v_1, v_2}((t_0, a, t_1), a) = v_1(p) = 0.1$, and
- $\sigma|_2^{v_1, v_2}((t_0, a, t_1, a, t_3), \odot) = v_1(p) = 0.1$.

Note that the resulting projection coincides with the one computed in Example 15, where both components were instantiated using v_1 , i.e., $\sigma|_2^{v_1, v_2} = \sigma|_2^{v_1} = \sigma|_2^{v_1, v_1}$.

The next lemma states that – when restricting to valuations that yield the same non-zero transitions – the strategy projection to \mathcal{M}_i only depends on the parameter instantiation applied to \mathcal{M}_{3-i} . This observation is the key insight for the correctness of the proof rule in Theorem 41, which enables compositional reasoning about monotonicity.

► **Lemma 18.** *For $i = 1, 2$, and well-defined valuations $v_i, v'_i: V \rightarrow \mathbb{R}$ for \mathcal{M}_i such that $\mathbf{P}_i(s_i, \alpha_i, s'_i)[v_i] = 0$ iff $\mathbf{P}_i(s_i, \alpha_i, s'_i)[v'_i] = 0$ we have: $\sigma|_1^{v_1, v_2} = \sigma|_1^{v'_1, v_2}$ and $\sigma|_2^{v_1, v_2} = \sigma|_2^{v_1, v'_2}$.*

4 Verification Objectives for pPAs

We define properties of interest for pPA verification. Let Σ be a finite alphabet. $\Sigma^\infty = \Sigma^* \cup \Sigma^\omega$ denotes the set of all finite and infinite words over Σ . For a word $\rho = a_0, a_1, \dots \in \Sigma^\infty$ and another alphabet $\hat{\Sigma}$, let $\rho|_{\hat{\Sigma}} \in \hat{\Sigma}^\infty$ denote the projection of ρ onto $\hat{\Sigma}$ – obtained by dropping all $a_i \in \Sigma \setminus \hat{\Sigma}$ from ρ . The restriction $\rho|_{\hat{\Sigma}}$ can be finite, even if ρ is infinite.

We fix a pPA $\mathcal{M} = (S, s^{\text{init}}, V, \text{Act}, \mathbf{P}, L)$. The trace of $\pi = s_0, \alpha_0, s_1, \alpha_1, \dots \in \text{Paths}_{\mathcal{M}}^{\text{inf}}$ is the sequence $\text{tr}(\pi) = L(s_0, \alpha_0), L(s_1, \alpha_1), \dots$ of transition labels. The *probability of a language* $\mathcal{L} \subseteq \Sigma^\infty$ at a well-defined valuation v under strategy σ of \mathcal{M} is given by

$$\text{Pr}_{\mathcal{M}}^{v, \sigma}(\mathcal{L}) = \text{Pr}_{\mathcal{M}}^{v, \sigma} \left(\{ \pi \in \text{Paths}_{\mathcal{M}}^{\text{inf}} \mid \text{tr}(\pi)|_{\Sigma} \in \mathcal{L} \} \right).$$

We also consider (parametric) *expected total reward properties*. Let V be a set of parameters. A *reward function* $\mathcal{R}: \Sigma \rightarrow \mathbb{Q}[V] \cup \mathbb{R}_{\geq 0}$ over Σ assigns a (potentially parametric) reward to each symbol $a \in \Sigma$. *Instantiation* of \mathcal{R} at a valuation $v: V \rightarrow \mathbb{R}$ yields $\mathcal{R}[v]$ with $\mathcal{R}[v](a) = \mathcal{R}(a)[v]$ for all $a \in \Sigma$. Valuation v is *well-defined* for \mathcal{R} if $\mathcal{R}[v]: \Sigma \rightarrow \mathbb{R}_{\geq 0}$. In this case, the accumulated reward for a word $\rho = a_0, a_1, \dots \in \Sigma^\infty$ is given by $\mathcal{R}[v](\rho) = \sum_{i=0}^{|\rho|} \mathcal{R}[v](a_i) \in \mathbb{R}_{\geq 0} \cup \{\infty\}$.

When applied to a pPA \mathcal{M} , a reward function \mathcal{R} assigns the reward $\mathcal{R}(L(s, \alpha))$ to the enabled state-action-pairs $(s, \alpha) \in \text{dom}(\mathbf{P})$ with $L(s, \alpha) \in \Sigma$. For a well-defined valuation v for \mathcal{M} and \mathcal{R} , we define the *expected total reward* under strategy σ as

$$Ex_{\mathcal{M}}^{v, \sigma}(\mathcal{R}) = \int_{\pi \in \text{Paths}_{\mathcal{M}}^{\text{inf}}} \mathcal{R}[v](tr(\pi) \upharpoonright_{\Sigma}) dPr_{\mathcal{M}}^{v, \sigma}(\pi).$$

We consider probabilistic and reward-based objectives as well as their multi-objective combinations.

► **Definition 19 (Objectives).** For $\sim \in \{>, \geq, <, \leq\}$, $p \in [0, 1]$, and $r \in \mathbb{R}_{\geq 0}$, we denote

- a probabilistic objective over $\mathcal{L} \subseteq \Sigma^{\infty}$ by $\mathbb{P}_{\sim p}(\mathcal{L})$ and
- a reward objective over $\mathcal{R}: \Sigma \rightarrow \mathbb{Q}[V] \cup \mathbb{R}_{\geq 0}$ by $\mathbb{E}_{\sim r}(\mathcal{R})$.

Their satisfaction for a well-defined valuation v and strategy σ is defined by

$$\mathcal{M}, v, \sigma \models \mathbb{P}_{\sim p}(\mathcal{L}) \Leftrightarrow Pr_{\mathcal{M}}^{v, \sigma}(\mathcal{L}) \sim p \quad \text{and} \quad \mathcal{M}, v, \sigma \models \mathbb{E}_{\sim r}(\mathcal{R}) \Leftrightarrow Ex_{\mathcal{M}}^{v, \sigma}(\mathcal{R}) \sim r.$$

Let $\varphi \in \{\mathbb{P}_{\sim p}(\mathcal{L}), \mathbb{E}_{\sim r}(\mathcal{R})\}$ refer to a (probabilistic or reward) objective. If neither \mathcal{M} nor φ consider any parameters, we may drop the valuation from the notation and just write $\mathcal{M}, \sigma \models \varphi$. We lift the satisfaction relation \models to regions, i.e., sets of valuations.

► **Definition 20 (Region Satisfaction Relation).** Let $\star \in \{prt, cmp\} \cup \{fair_{\mathcal{C}} \mid \mathcal{C} \subseteq 2^{\Sigma_{\mathcal{M}}}\}$. For objective φ and well-defined region R for \mathcal{M} – and \mathcal{R} if $\varphi = \mathbb{E}_{\sim r}(\mathcal{R})$ – the region satisfaction relation \models^{\star} is given by:

$$\mathcal{M}, R \models^{\star} \varphi \Leftrightarrow \forall v \in R : \forall \sigma \in \text{Str}_{\mathcal{M}}^{\star} : \mathcal{M}, v, \sigma \models \varphi.$$

Satisfaction under memoryless strategies – denoted by $\models^{\text{mless}, \star}$ – is defined similarly.

► **Remark 21.** For $\star \in \{prt, cmp\}$ and any well-defined valuation v we have $\text{Str}_{\mathcal{M}[v]}^{\star} = \text{Str}_{\mathcal{M}}^{\star}$. Thus, for well-defined R , we can swap the quantifiers in Definition 20:

$$\mathcal{M}, R \models^{\star} \varphi \Leftrightarrow \forall \sigma \in \text{Str}_{\mathcal{M}}^{\star} : \forall v \in R : \mathcal{M}, v, \sigma \models \varphi.$$

However, this is not the case for *fair* strategies and regions that are not graph-preserving: A strategy that is not *fair_C* for \mathcal{M} (under graph-preserving instantiations) might be *fair_C* for $\mathcal{M}[v]$ if v is not graph-preserving, because states that violate the fairness condition might not be reachable in $\mathcal{M}[v]$. For a *graph-preserving* region R and all $v \in R$, we have $\text{Str}_{\mathcal{M}}^{\text{fair}_C} = \text{Str}_{\mathcal{M}[v]}^{\text{fair}_C}$. Thus, we can swap quantifiers as above.

Our framework also handles conjunctions of multiple objectives.

► **Definition 22 (MO-Query).** A multi-objective query (mo-query) is a set $X = \{\varphi_1, \dots, \varphi_n\}$ of n probabilistic or reward objectives with $\mathcal{M}, v, \sigma \models X \Leftrightarrow \mathcal{M}, v, \sigma \models \varphi_i$ for all $\varphi_i \in X$.

The conjunction of two mo-queries is a union of sets: $X_1 \wedge X_2 = X_1 \cup X_2$. We lift objective satisfaction for regions (Definition 20) to mo-queries in a straightforward way.

► **Remark 23.** In [38] it is shown that model checking under partial strategies in \mathcal{M} for probabilistic properties, rewards, and multi-objective queries reduces to model checking under complete strategies in a modified pPA, denoted \mathcal{M}_{τ} . This result extends [35, Proposition 2] to pPA while preserving memorylessness of the strategies.

We consider *safety objectives* as a special type of probabilistic objectives.

► **Definition 24** (Safety Objective). $\mathbb{P}_{\geq p}(\mathcal{L})$ is a safety objective⁴ if \mathcal{L} can be characterized by a DFA $\mathcal{A}_{\mathcal{L}}^{bad}$ accepting a language of finite words (bad prefixes):

$$\mathcal{L} = \{w \in \Sigma^\infty \mid \text{no prefix of } w \text{ is accepted by } \mathcal{A}_{\mathcal{L}}^{bad}\}.$$

A mo-query is called *safe*, denoted \mathbf{X}^{safe} , if each φ_i is a probabilistic safety objective.

For PA, computing the probability for a safety objective reduces to maximal reachability properties in the PA-DFA product [35, Lemma 1]. This result can be lifted to pPA in a straightforward manner; see [38]. For reachability and safety objectives, it is equivalent to quantify over complete or partial strategies. Lemma 25 lifts [35, Proposition 1] to pPA.

► **Lemma 25.** *Let \mathcal{M} be a pPA, let R be a well-defined region and let $\mathbb{P}_{\geq p}(\mathcal{L})$ be a safety objective for \mathcal{M} . It holds that: $\mathcal{M}, R \models^{cmp} \mathbb{P}_{\geq p}(\mathcal{L}) \Leftrightarrow \mathcal{M}, R \models^{prt} \mathbb{P}_{\geq p}(\mathcal{L})$. Same for $\models^{mless, \star}$.*

4.1 Preservation Under Projection

We generalize the result from [35, Lemma 3] – originally stated in [46, Lemma 7.2.6] – to the parametric setting. In particular, we show that probabilistic and reward properties in a composed pPA $\mathcal{M} = \mathcal{M}_1 \parallel \mathcal{M}_2$ under a strategy σ are preserved when projecting to either component \mathcal{M}_i over Σ_i , assuming a well-defined valuation v .

► **Theorem 26.** *For $i = 1, 2$, let \mathcal{L} be a language over Σ_i and \mathcal{R} be a reward function over Σ_i . Then, for a well-defined valuation v :*

$$Pr_{\mathcal{M}_i}^{v, \sigma \upharpoonright_i^v}(\mathcal{L}) = Pr_{\mathcal{M}_1 \parallel \mathcal{M}_2}^{v, \sigma}(\mathcal{L}) \quad \text{and} \quad Ex_{\mathcal{M}_i}^{v, \sigma \upharpoonright_i^v}(\mathcal{R}) = Ex_{\mathcal{M}_1 \parallel \mathcal{M}_2}^{v, \sigma}(\mathcal{R})$$

► **Example 27.** Let $\mathcal{L} = \{w \in \{\mathbf{a}, \mathbf{c}, \odot\}^\infty \mid \exists i \in \mathbb{N} : w_i = \odot\}$ be the language of words in which \odot occurs. Reconsider the strategy σ of $\mathcal{M}_1 \parallel \mathcal{M}_2$ from Example 17 and the projection $\sigma \upharpoonright_2^v$ to \mathcal{M}_2 . We have $Pr_{\mathcal{M}_1 \parallel \mathcal{M}_2}^{v, \sigma}(\mathcal{L}) = Pr_{\mathcal{M}_2}^{v, \sigma \upharpoonright_2^v}(\mathcal{L}) = (v(p))^2 \cdot \frac{1}{10} + v(p) \cdot (1 - v(p)) \cdot v(q)$.

Theorem 26 assumes that the property only involves action labels from a single component \mathcal{M}_i . To allow objectives over arbitrary alphabets Σ , we can add a self-loop labeled \mathbf{a} at every state, for each label $\mathbf{a} \notin \Sigma_i$.

► **Definition 28** (Alphabet Extension). *Let $\mathcal{M} = (S, s^{init}, V, Act, \mathbf{P}, L)$ be a pPA over $\Sigma_{\mathcal{M}}$ and let Σ be an alphabet with $Act \cap (\Sigma \setminus \Sigma_{\mathcal{M}}) = \emptyset$. The alphabet extension of \mathcal{M} with respect to Σ is the pPA $\mathcal{M}(\Sigma) = (S, s^{init}, V, Act \cup (\Sigma \setminus \Sigma_{\mathcal{M}}), \mathbf{P}_\Sigma, L_\Sigma)$ over alphabet $\Sigma_{\mathcal{M}} \cup \Sigma$, where*

- $\mathbf{P}_\Sigma(s, \alpha) = \mathbf{P}(s, \alpha)$ and $L_\Sigma(s, \alpha) = L(s, \alpha)$ for all $(s, \alpha) \in \text{dom}(\mathbf{P})$ and
- $\mathbf{P}_\Sigma(s, \mathbf{a}) = \mathbb{1}_s$ and $L_\Sigma(s, \mathbf{a}) = \mathbf{a}$ for all $s \in S$ and $\mathbf{a} \in \Sigma \setminus \Sigma_{\mathcal{M}}$.

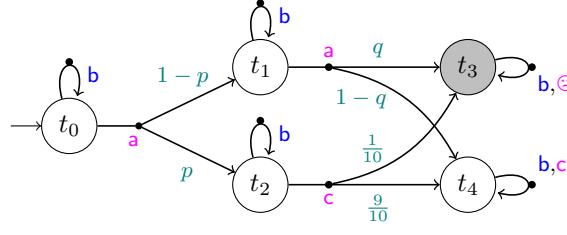
► **Example 29** (Alphabet Extension). Figure 3 shows $\mathcal{M}_2(\{\mathbf{a}, \mathbf{b}\})$ for pPA \mathcal{M}_2 over $\Sigma_2 = \{\mathbf{a}, \mathbf{c}, \odot\}$ from Figure 1b. Transitions with label \mathbf{a} remain unchanged as $\mathbf{a} \in \Sigma_2$, but an additional self-loop with action $\mathbf{b} \notin \Sigma_2$ is added to every state.

We now lift [35, Lemma 3] to the parametric setting, covering properties and mo-queries over an alphabet that is not necessarily shared by \mathcal{M}_i :

► **Theorem 30.** *Let $\Sigma \subseteq \Sigma_{\mathcal{M}_1 \parallel \mathcal{M}_2}$, and σ be a strategy for $\mathcal{M}_1(\Sigma) \parallel \mathcal{M}_2(\Sigma)$. Let \mathcal{L} be a language over Σ and \mathcal{R} be a reward function over Σ . Then, for well-defined valuation v :*

$$Pr_{\mathcal{M}_i(\Sigma)}^{v, \sigma \upharpoonright_i^v}(\mathcal{L}) = Pr_{\mathcal{M}_1(\Sigma) \parallel \mathcal{M}_2(\Sigma)}^{v, \sigma}(\mathcal{L}) \quad \text{and} \quad Ex_{\mathcal{M}_i(\Sigma)}^{v, \sigma \upharpoonright_i^v}(\mathcal{R}) = Ex_{\mathcal{M}_1(\Sigma) \parallel \mathcal{M}_2(\Sigma)}^{v, \sigma}(\mathcal{R})$$

⁴ Note that safety objectives contain all finite prefixes of words in \mathcal{L} , i.e., they are prefix-closed. This is different in [35], where only infinite words are considered, leading to technical problems. See [38].



■ **Figure 3** Alphabet extension $\mathcal{M}_2\langle\{a, b\}\rangle$ of the pPA \mathcal{M}_2 from Figure 1b to the alphabet $\{a, b\}$.

Let X be a mo-query over Σ . Then, for any well-defined valuation v :

$$\mathcal{M}_i\langle\Sigma\rangle, v, \sigma \models_i^v X \Leftrightarrow (\mathcal{M}_1\langle\Sigma\rangle \parallel \mathcal{M}_2\langle\Sigma\rangle), v, \sigma \models X$$

► **Remark 31.** Since alphabet extensions add self-loop transitions for new labels, and thus do not change the system's state, the pPAs $\mathcal{M}_1\langle\Sigma\rangle \parallel \mathcal{M}_2\langle\Sigma\rangle$ and $\mathcal{M}_1 \parallel \mathcal{M}_2$ satisfy the same properties and mo-queries over the alphabet $\Sigma \subseteq \Sigma_{\mathcal{M}_1 \parallel \mathcal{M}_2}$.

Theorems 26 and 30 play a key role in the proof of the AG framework for reasoning about mo-queries and monotonicity, which will be established in Sections 5 and 6.

5 Assume-Guarantee Reasoning for pPA

Kwiatkowska et al. [35] introduced assume-guarantee (AG) reasoning proof rules for PA. This section extends their proof rules to the parametric setting. We first generalize the concept of AG triples to pPAs in Section 5.1. Then, we extend the asymmetric and circular proof rule in Section 5.2. Additional proof rules from [35] are presented in [38].

5.1 Assume-Guarantee Triples for pPA

We extend compositional reasoning to the parametric setting by generalizing assume-guarantee (AG) triples. Intuitively, an AG triple states that if a component satisfies an assumption, it also satisfies the guarantee under the same strategy and valuation.

► **Definition 32** (AG Triple). *The assume-guarantee triple for \mathcal{M} , (parametric) mo-queries A (assumption) and G (guarantee), well-defined region R , and $\star \in \{cmp, prt, fair_C\}$ is*

$$\mathcal{M}, R \models^\star A \rightarrow G \Leftrightarrow \left(\forall v \in R : \forall \sigma \in Str_{\mathcal{M}[v]}^\star : \mathcal{M}, v, \sigma \models A \rightarrow \mathcal{M}, v, \sigma \models G \right)$$

5.2 Assume-Guarantee Rules for pPA

We present AG proof rules for the compositional verification of parametric probabilistic automata (pPAs). In the remainder of this section, we fix two pPAs \mathcal{M}_1 and \mathcal{M}_2 with alphabets Σ_1 and Σ_2 , respectively. Further, let R_i be a well-defined region for \mathcal{M}_i .

First, we establish the asymmetric proof rule for safety and mo-queries – based on [35, Theorem 1 and 2], respectively – for pPA.

► **Theorem 33** (Asymmetric Rule). *Let A and G be mo-queries over $\Sigma_A \subseteq \Sigma_1$ and $\Sigma_G \subseteq \Sigma_2 \cup \Sigma_A$, respectively. Let $\mathcal{C}_1 \subseteq 2^{\Sigma_1}$ and $\mathcal{C}_2 \subseteq 2^{\Sigma_2 \cup \Sigma_A}$. Then, the two proof rules holds:*

$$\frac{\mathcal{M}_1, R_1 \models^{cmp} A^{safe} \quad \mathcal{M}_2\langle\Sigma_A^{safe}\rangle, R_2 \models^{prt} A^{safe} \rightarrow G^{safe}}{\mathcal{M}_1 \parallel \mathcal{M}_2, R_1 \cap R_2 \models^{cmp} G^{safe}} \quad \frac{\mathcal{M}_1, R_1 \models^{fair_{C_1}} A \quad \mathcal{M}_2\langle\Sigma_A\rangle, R_2 \models^{fair_{C_2}} A \rightarrow G}{\mathcal{M}_1 \parallel \mathcal{M}_2, R_1 \cap R_2 \models^{fair_{C_1 \cup C_2}} G}$$

Proof sketch. Let $v \in R_1 \cap R_2$, and σ be a strategy for the composed pPA $\mathcal{M}_1 \parallel \mathcal{M}_2$. To prove validity of the rule, we need to show that $\mathcal{M}_1 \parallel \mathcal{M}_2$ instantiated with v satisfies G^{safe} .

1. Since $v \in R_1$, the first premise implies $\mathcal{M}_1, v \models^{cmp} A^{safe}$, which is equivalent to $\mathcal{M}_1, v \models^{prt} A^{safe}$ by Lemma 25. This implies that \mathcal{M}_1 under the partial strategy $\sigma|_{\mathcal{M}_1}^v$ also satisfies A^{safe} . Since strategies and their projections satisfy the same properties (as shown in Theorem 30), we conclude that $\mathcal{M}_1 \parallel \mathcal{M}_2$ instantiated at v under the strategy σ satisfies A^{safe} .

2. As $v \in R_2$, the second premise implies that $\mathcal{M}_2[v]$ under the strategy $\sigma|_{\mathcal{M}_2}^v$ satisfies G^{safe} . Again, Theorem 30 implies that $(\mathcal{M}_1 \parallel \mathcal{M}_2)[v]$ under the strategy σ satisfies G^{safe} .

Thus, we conclude that $\mathcal{M}_1 \parallel \mathcal{M}_2$ instantiated at v under σ satisfies G^{safe} . The rule on the right holds by a similar reasoning, where, in addition, Lemma 14 is used to establish that projections of fair strategies remain fair. \blacktriangleleft

► **Example 34** (Asymmetric Rule). We illustrate the left proof rule from Theorem 33 for the pPA $\mathcal{M}_1 \parallel \mathcal{M}_2$ in Figure 2 – composed of the pPAs \mathcal{M}_1 and \mathcal{M}_2 depicted in Figure 1 – and w.r.t. $G = \mathbb{P}_{\geq 0.8}(\mathcal{L}_G)$, where $\mathcal{L}_G = \{w \in \{a, b, c, \odot\}^\infty \mid |w|_\odot = 0\}$. Let $A = \{\mathbb{P}_{\geq 0.9}(\mathcal{L}_A)\}$, where $\mathcal{L}_A = \{w \in \{a, b\}^\infty \mid |w|_a \leq 1\}$. The pPA $\mathcal{M}_2(\{a, b\})$ is depicted in Figure 3. For the premises of the proof rule, we obtain that the (largest) region R for which $\mathcal{M}_1, R_1 \models^{cmp} \mathbb{P}_{\geq 0.9}(\mathcal{L}_A)$ is $R_1 = \{v: \{p, q\} \rightarrow \mathbb{R} \mid v(p) \in [0, 0.1]\}$ and the (largest) region R_2 for which $\mathcal{M}_2(\{a, b\}), R_2 \models^{prt} A \rightarrow G$ holds, is $R = \{v: \{p, q\} \rightarrow \mathbb{R} \mid (v(p) \in [0, 0.5], v(q) \in [0, 1]) \vee (v(p) \in (0.5, 1), v(q) \in [0, 2 - 2 \cdot p])\}$. The intersection $R_1 \cap R_2$ – for which $\mathcal{M}_1 \parallel \mathcal{M}_2, R_1 \cap R_2 \models^{prt} G$ holds by [38, Theorem 52] – contains all valuations with $v(p) \in [0, 0.1], v(q) \in [0, 1]$. The (largest) region R for which $\mathcal{M}_1 \parallel \mathcal{M}_2, R \models^{prt} G$ is $R = \{v: p, q \rightarrow \mathbb{R} \mid (v(p) \in ([0, \frac{1}{4}] \cup \{1\}), v(q) \in [0, 1]) \vee (v(p) \in (\frac{1}{4}, 1), v(q) \in [0, \frac{p+1}{5 \cdot p}])\}$. This satisfies $(R_1 \cap R_2) \subset R$.

The proof rules in Theorem 33 can be extended to systems with more than two components, as detailed in [38, Theorem 52]. Next, we lift the circular proof rule given in [35, Theorem 5] to pPAs:

► **Theorem 35** (Circular Rule). *Let A_1, A_2 and G be (parametric) mo-queries over $\Sigma_{A_1} \subseteq \Sigma_2$, $\Sigma_{A_2} \subseteq \Sigma_1 \cup \Sigma_{A_1}$ and $\Sigma_G \subseteq \Sigma_2 \cup \Sigma_{A_2}$, respectively. Let $\mathcal{C}_i \in 2^{\Sigma_i \cup \Sigma_{A_i}}$ for $i \in \{1, 2\}$, and $\mathcal{C}_3 \in 2^{\Sigma_2}$. Then:*

$$\frac{\begin{array}{l} \mathcal{M}_1 \langle \Sigma_{A_1^{safe}} \rangle, R_1 \models^{prt} A_1^{safe} \rightarrow A_2^{safe} \\ \mathcal{M}_2 \langle \Sigma_{A_2^{safe}} \rangle, R_2 \models^{prt} A_2^{safe} \rightarrow G^{safe} \\ \mathcal{M}_2, R_3 \models^{cmp} A_1^{safe} \end{array}}{\mathcal{M}_1 \parallel \mathcal{M}_2, R_1 \cap R_2 \cap R_3 \models^{cmp} G^{safe}} \quad \frac{\begin{array}{l} \mathcal{M}_1 \langle \Sigma_{A_1} \rangle, R_1 \models^{fair_{\mathcal{C}_1}} A_1 \rightarrow A_2 \\ \mathcal{M}_2 \langle \Sigma_{A_2} \rangle, R_2 \models^{fair_{\mathcal{C}_2}} A_2 \rightarrow G \\ \mathcal{M}_2, R_3 \models^{fair_{\mathcal{C}_3}} A_1 \end{array}}{\mathcal{M}_1 \parallel \mathcal{M}_2, R_1 \cap R_2 \cap R_3 \models^{fair_{\mathcal{C}_1 \cup \mathcal{C}_2 \cup \mathcal{C}_3}} G}$$

Proof sketch. Similar to Theorem 33, the proof of the circular rules makes use of Theorem 30, which establishes that the composition under a strategy satisfies the same properties as the individual components under their corresponding projections. For safety, Lemma 25 allows us to verify the condition for complete strategies rather than partial strategies in the third premise. For fairness, Lemma 14 ensures that strategy projections remain fair. \blacktriangleleft

► **Remark 36.** The inclusion of fairness in the premises of the right rules in Theorem 33 and Theorem 35 enables recursive application and thus supports the compositional verification of systems with more than two components. In the case of a single application of one of the rules, it is sufficient to verify with respect to complete strategies, which, while a stronger condition, simplifies the verification process.

6 Compositional Reasoning about Monotonicity

Exploiting monotonicity can significantly enhance the efficiency of parameter synthesis [51]. However, determining monotonicity is computationally hard⁵ and it would be beneficial to determine monotonicity in a compositional way. Additionally, monotonicity in composed pPA is challenging due to the complexities introduced by parameter dependencies and interactions among components. While we focus on global monotonicity, the following results can be extended to *local monotonicity*, which considers only the first transition from a given state. See [49, Definitions 4.4 and 4.5].

The probability of a language or the expected total reward for a pPA \mathcal{M} can be viewed as a function – called *solution function* – that maps a well-defined parameter valuation to the corresponding probability or expected total reward, respectively [27, Definition 4.7].

► **Definition 37** (Solution Function). *Let \mathcal{M} be a pPA over Σ , let $\sigma \in \text{Str}_{\mathcal{M}}$ and let R be a well-defined region. The solution function for \mathcal{M} and language $\mathcal{L} \subseteq \Sigma^\infty$ is $\text{sol}_{\mathcal{M},\sigma}^{\text{Pr}(\mathcal{L})} : R \rightarrow [0, 1]$, where $\text{sol}_{\mathcal{M},\sigma}^{\text{Pr}(\mathcal{L})}(v) = \text{Pr}_{\mathcal{M}}^{v,\sigma}(\mathcal{L})$. The solution function for \mathcal{M} and a reward function \mathcal{R} over Σ is $\text{sol}_{\mathcal{M},\sigma}^{\text{Ex}(\mathcal{R})} : R \rightarrow \mathbb{R}_{\geq 0}$, where $\text{sol}_{\mathcal{M},\sigma}^{\text{Ex}(\mathcal{R})}(v) = \text{Ex}_{\mathcal{M}}^{v,\sigma}(\mathcal{R})$.*

When referring to a solution function without specifying whether it pertains to probabilities or expected rewards, we simply write $\text{sol}_{\mathcal{M},\sigma}$.

► **Example 38** (Solution Function). Consider the pPA $\mathcal{M}_1 \parallel \mathcal{M}_2$ in Figure 2 and the region $R = \{v : \{p, q\} \rightarrow [0, 1]\}$ which is well-defined for $\mathcal{M}_1 \parallel \mathcal{M}_2$. Let $\mathcal{L} = \{w \in \{a, c, \odot\}^\infty \mid |w|_{\odot} = 0\}$ be the language of words over $\{a, b, c, \odot\}$ that do not contain \odot . Let σ be the complete strategy of $\mathcal{M}_1 \parallel \mathcal{M}_2$ from Example 17, which always selects action a, c or \odot with probability 1 whenever any of them is enabled; otherwise, it chooses b with probability 1. The solution function $\text{sol}_{\mathcal{M}_1 \parallel \mathcal{M}_2, \sigma}^{\text{Pr}(\mathcal{L})} : R \rightarrow [0, 1]$ is defined by $\text{sol}_{\mathcal{M}_1 \parallel \mathcal{M}_2, \sigma}^{\text{Pr}(\mathcal{L})}(p, q) = 1 - (p^2 \cdot \frac{1}{10} + p \cdot (1 - p) \cdot (p \cdot q + (1 - p) \cdot q)) = 1 - (p^2 \cdot \frac{1}{10} + (p - p^2) \cdot q)$.

We extend the standard notion of monotonicity [49] by differentiating between different strategy classes, including complete, partial, and fair strategies.

► **Definition 39** (Monotonicity). *Let σ be a strategy of \mathcal{M} . A solution function $\text{sol}_{\mathcal{M},\sigma}$ is monotonic increasing in $p \in V$ on region R – denoted $\text{sol}_{\mathcal{M},\sigma} \uparrow_{p,R}$ – if for all $v, v_+ \in R$ with $v_+(q) = v(q) + x \cdot \llbracket p=q \rrbracket$ for $q \in V$ and some $x \geq 0$, we have: $\text{sol}_{\mathcal{M},\sigma}(v) \leq \text{sol}_{\mathcal{M},\sigma}(v_+)$.*

For $\star \in \{\text{prt}, \text{cmp}\}$, we write $\text{sol}_{\mathcal{M}} \uparrow_{p,R}^\star$ if $\text{sol}_{\mathcal{M},\sigma} \uparrow_{p,R}$ for all $\sigma \in \text{Str}_{\mathcal{M}}^\star$. If R is graph-preserving, we write $\text{sol}_{\mathcal{M}} \uparrow_{p,R}^{\text{fair}_c}$ if $\text{sol}_{\mathcal{M},\sigma} \uparrow_{p,R}$ holds for all fair strategies $\sigma \in \text{Str}_{\mathcal{M}[v]}^{\text{fair}_c}$, $v \in R$. Notations $\text{sol}_{\mathcal{M},\sigma} \downarrow_{p,R}$ and $\text{sol}_{\mathcal{M}} \downarrow_{p,R}^\star$ for monotonic decreasing $\text{sol}_{\mathcal{M},\sigma}$ are defined analogously.

We require the region to be graph-preserving when defining monotonicity w.r.t. fair strategies. This ensures that for any two valuations, v, v_+ , we have $\text{Str}_{\mathcal{M}[v]}^{\text{fair}_c} = \text{Str}_{\mathcal{M}[v_+]}^{\text{fair}_c}$; see Proposition 10.

► **Remark 40.** Monotonicity for partial strategies w.r.t. general properties is equivalent to monotonicity for complete strategies in a modified pPA; see [38].

The following theorem states that monotonicity of a composed system can be verified by analyzing its individual components.

⁵ For deterministic pPA (Markov chains) determining monotonicity is coETR-hard [49, Sec. 3.4].

► **Theorem 41** (Monotonicity). Let $\mathcal{M}_1, \mathcal{M}_2$ be pPAs with alphabets Σ_1 and Σ_2 and R_i be a graph-preserving region for \mathcal{M}_i . Let $\text{sol} \in \{\text{sol}^{Pr(\mathcal{L})}, \text{sol}^{Ex(\mathcal{R})}\}$ be a solution function w.r.t. the language \mathcal{L} or reward function \mathcal{R} over $\Sigma \subseteq (\Sigma_1 \cup \Sigma_2)$ and let $\uparrow \in \{\uparrow, \downarrow\}$. Let $\mathcal{C}_i \subseteq 2^{\Sigma_1 \cup \Sigma_2}$. Then the following two proof rules hold:

$$\frac{\text{sol}_{\mathcal{M}_1 \langle \Sigma \rangle \downarrow_{p, R_1}^{prt} \quad \text{sol}_{\mathcal{M}_2 \langle \Sigma \rangle \downarrow_{p, R_2}^{prt}}}{\text{sol}_{\mathcal{M}_1 \parallel \mathcal{M}_2 \downarrow_{p, R_1 \cap R_2}^{prt}} \quad \frac{\text{sol}_{\mathcal{M}_1 \langle \Sigma \rangle \downarrow_{p, R_1}^{fair_{\mathcal{C}_1}}} \quad \text{sol}_{\mathcal{M}_2 \langle \Sigma \rangle \downarrow_{p, R_2}^{fair_{\mathcal{C}_2}}}{\text{sol}_{\mathcal{M}_1 \parallel \mathcal{M}_2 \downarrow_{p, R_1 \cap R_2}^{fair_{\mathcal{C}_1 \cup \mathcal{C}_2}}}}$$

Proof. We show the premises imply $\text{sol}_{\mathcal{M}_1 \langle \Sigma \rangle \parallel \mathcal{M}_2 \langle \Sigma \rangle \downarrow_{p, R_1 \cap R_2}^{\star}$ for $\star \in \{prt, fair_{\mathcal{C}_1 \cup \mathcal{C}_2}\}$, which directly implies that $\text{sol}_{\mathcal{M}_1 \parallel \mathcal{M}_2 \downarrow_{p, R_1 \cap R_2}^{\star}$ holds, see Remark 31. We focus on the left rule, i.e., $\star = prt$.

► **Remark 42.** The proof for $\star = fair_{\mathcal{C}_1 \cup \mathcal{C}_2}$ is similar but additionally requires Lemma 14 in [38, Appendix A.4].

We further consider $\uparrow = \uparrow$. The case $\uparrow = \downarrow$ follows analogously. Our proof is by contradiction. Assume that the premises hold but $\text{sol}_{\mathcal{M}_1 \langle \Sigma \rangle \parallel \mathcal{M}_2 \langle \Sigma \rangle \uparrow_{p, R_1 \cap R_2}^{\star}$ does not hold. Thus, there is a strategy $\sigma \in \text{Str}_{(\mathcal{M}_1 \langle \Sigma \rangle \parallel \mathcal{M}_2 \langle \Sigma \rangle)}^{\star}$ and valuations $v, v_+ \in R_1 \cap R_2$ with $v_+(q) = v(q) + x \cdot \llbracket p=q \rrbracket$ for $q \in V$ and some $x \geq 0$ and

$$\text{sol}_{(\mathcal{M}_1 \langle \Sigma \rangle \parallel \mathcal{M}_2 \langle \Sigma \rangle), \sigma}(v) > \text{sol}_{(\mathcal{M}_1 \langle \Sigma \rangle \parallel \mathcal{M}_2 \langle \Sigma \rangle), \sigma}(v_+). \quad (1)$$

Theorem 30 yields $\text{sol}_{(\mathcal{M}_1 \langle \Sigma \rangle \parallel \mathcal{M}_2 \langle \Sigma \rangle), \sigma}(v) = \text{sol}_{\mathcal{M}_1 \langle \Sigma \rangle, \sigma \uparrow_1^{v, v_+}}(v)$. We have $\mathbf{P}_{\mathcal{M}_1 \langle \Sigma \rangle}(s, \alpha, s')[v] = 0$ iff $\mathbf{P}_{\mathcal{M}_1 \langle \Sigma \rangle}(s, \alpha, s')[v_+] = 0$ as v and v_+ are graph preserving for \mathcal{M}_1 . Thus, we can apply Lemma 18 and obtain

$$\begin{aligned} \text{sol}_{\mathcal{M}_1 \langle \Sigma \rangle, \sigma \uparrow_1^{v, v_+}}(v) &= \text{sol}_{\mathcal{M}_1 \langle \Sigma \rangle, \sigma \uparrow_1^{v_+, v_+}}(v) && \text{(by Lemma 18)} \\ &\leq \text{sol}_{\mathcal{M}_1 \langle \Sigma \rangle, \sigma \uparrow_1^{v_+, v_+}}(v_+) && (\sigma \uparrow_1^{v_+, v_+} \in \text{Str}_{\mathcal{M}_1 \langle \Sigma \rangle}^{prt}, \text{sol}_{\mathcal{M}_1 \langle \Sigma \rangle} \uparrow_{p, R_1}^{prt}) \\ &= \text{sol}_{(\mathcal{M}_1 \langle \Sigma \rangle \parallel \mathcal{M}_2 \langle \Sigma \rangle)[v], \sigma}(v_+) && \text{(by Theorem 30)} \end{aligned}$$

We observe that

$$(\mathcal{M}_1 \langle \Sigma \rangle \parallel \mathcal{M}_2 \langle \Sigma \rangle)[v][v_+] = (\mathcal{M}_1 \langle \Sigma \rangle[v_+] \parallel \mathcal{M}_2 \langle \Sigma \rangle[v]) = (\mathcal{M}_1 \langle \Sigma \rangle[v_+] \parallel \mathcal{M}_2 \langle \Sigma \rangle)[v].$$

Consequently, $\text{sol}_{(\mathcal{M}_1 \langle \Sigma \rangle \parallel \mathcal{M}_2 \langle \Sigma \rangle)[v], \sigma}(v_+) = \text{sol}_{(\mathcal{M}_1 \langle \Sigma \rangle[v_+] \parallel \mathcal{M}_2 \langle \Sigma \rangle), \sigma}(v)$. By a similar reasoning as above, we obtain

$$\begin{aligned} \text{sol}_{(\mathcal{M}_1 \langle \Sigma \rangle[v_+] \parallel \mathcal{M}_2 \langle \Sigma \rangle), \sigma}(v) &= \text{sol}_{\mathcal{M}_2 \langle \Sigma \rangle, \sigma \uparrow_2^{v, v_+}}(v) && \text{(by Theorem 30)} \\ &\leq \text{sol}_{\mathcal{M}_2 \langle \Sigma \rangle, \sigma \uparrow_2^{v_+, v_+}}(v_+) && (\sigma \uparrow_2^{v_+, v_+} \in \text{Str}_{\mathcal{M}_2 \langle \Sigma \rangle}^{prt}, \text{sol}_{\mathcal{M}_2 \langle \Sigma \rangle} \uparrow_{p, R_2}^{prt}) \\ &= \text{sol}_{\mathcal{M}_2 \langle \Sigma \rangle, \sigma \uparrow_2^{v_+, v_+}}(v_+) && \text{(by Lemma 18)} \\ &= \text{sol}_{(\mathcal{M}_1 \langle \Sigma \rangle \parallel \mathcal{M}_2 \langle \Sigma \rangle), \sigma}(v_+) && \text{(by Theorem 30)} \end{aligned}$$

Thus, $\text{sol}_{(\mathcal{M}_1 \langle \Sigma \rangle \parallel \mathcal{M}_2 \langle \Sigma \rangle), \sigma}(v) \leq \text{sol}_{(\mathcal{M}_1 \langle \Sigma \rangle \parallel \mathcal{M}_2 \langle \Sigma \rangle), \sigma}(v_+)$, violating Equation (1). ◀

► **Example 43.** Reconsider the pPA $\mathcal{M}_1 \parallel \mathcal{M}_2$ in Figure 2, the region $R = \{v: \{p, q\} \rightarrow [0, 1]\}$, and the language $\mathcal{L} = \{w \in \{a, c, \odot\}^\infty \mid |w|_{\odot} = 0\}$ from Example 38. The pPA $\mathcal{M}_1 \parallel \mathcal{M}_2$ is composed of the pPAs \mathcal{M}_1 and \mathcal{M}_2 shown in Figure 1. The region R is well-defined for \mathcal{M}_1 and \mathcal{M}_2 . We check whether $\text{sol}_{\mathcal{M}_1 \parallel \mathcal{M}_2}^{Pr(\mathcal{L})}$ is monotonic in q on R via Theorem 41. Since the premises $\text{sol}_{\mathcal{M}_i \langle \Sigma_{\mathcal{L}} \rangle \downarrow_{q, R}^{prt}}$ for $i \in \{1, 2\}$ are satisfied, we conclude $\text{sol}_{\mathcal{M}_1 \parallel \mathcal{M}_2 \downarrow_{q, R}^{Pr(\mathcal{L})}}$.

7 Related Work

Compositional verification has been widely studied in both probabilistic and non-probabilistic systems. We summarize key contributions related to our work.

Jones’ rely-guarantee method [26] and Pnueli’s compositional proof system [44] for temporal logic laid the foundation for AG reasoning. Subsequent work focused on AG rules for systems using CTL* [12] and developed AG reasoning for *reactive modules* [25, 1]. Automated AG reasoning techniques – developed by Pasareanu et al. [13, 41] – include learning-based assumption generation. More recent work has focused on circular AG reasoning [16] and verification-repair approaches [22].

AG reasoning has been lifted to probabilistic settings. Initial work by de Alfaro et al. [15] introduced AG rules for a probabilistic extension of reactive modules [25, 1]. Their model is similar to PA [47, 46], but limited to synchronous composition.

Kwiatkowska et al. [34, 21] generalized AG verification for PA, allowing more flexible parallel compositions and extending AG reasoning to probabilistic safety properties. Their approach reduces AG verification to multi-objective model checking, as proposed by Etessami et al. [17]. This was further refined in [35], enabling AG reasoning over a broader class of quantitative properties, including conjunctions over probabilistic liveness and expected rewards. Algorithmic learning-based assumption generation techniques [13, 23] were later adapted for AG reasoning in probabilistic settings [18, 19, 36]. Other assumption generation approaches include abstraction-refinement methods [32, 10], based on the CEGAR paradigm [11], and symbolic learning-based methods [24, 5]. AG reasoning has been applied to various real-world domains, including service-based workflow verification [6], large-scale IT systems [7], and autonomous systems with deep neural networks [42, 43].

AG reasoning has also been extended to systems with uncertainty, for example, [55] introduced an AG framework for verifying systems with components modeled by MDPs and *partially observable MDPs* (POMDPs). In contrast, our work considers a different type of uncertainty; We extend AG reasoning to parametric probabilistic automata (PA), leveraging research on parametric MDPs [27, 45, 28] and previous AG verification techniques [35]. Our framework allows to reason about monotonicity [49, 50, 51] in a compositional manner. To the best of our knowledge, this is the first AG-based compositional verification framework for parametric PA. Existing modular proof systems have focused on parametric timed automata [2] or non-probabilistic parameterized systems [4, 48, 39], where concurrent programs are parameterized by the number of processes or threads in a configured instance.

Another recent line of research focuses on the *sequential* composition of MDPs rather than parallel decomposition: Junges and Spaan [30] introduced an abstraction-refinement approach for hierarchical probabilistic models, leveraging parametric MDPs to represent sets of similar subroutines. Recent work by Watanabe et al. [53] on mean-payoff games, applies category-theoretic string diagrams to the verification of sequentially composed MDPs.

8 Conclusion

We presented an assume-guarantee framework for compositional verification of parametric probabilistic automata, building on the proof rules for Segala’s PA by Kwiatkowska et al. [35]. In addition, we introduced new compositional proof rules to reason about monotonicity in composed systems. These contributions lay the theoretical foundations for modular verification of pPA. To the best of our knowledge, these are the first AG proof rules for probabilistic models with parametric transition probabilities.

Future work involves implementing the framework and demonstrating its effectiveness through case studies. Another direction is to deduce additional assume-guarantee rules – for example, reasoning about robust valuations or strategies, i.e., properties of the form: $\exists v \in R : \forall \sigma \in \text{Str}_{\mathcal{M}} : \mathcal{M}[v], \sigma \models X$. Additionally, interesting directions include the modular verification of other properties, such as long-run average rewards or expected visiting times [37]. Other areas include extending verification to logics such as parametric LTL [9] and probabilistic CTL. Further research could also explore Markov automata with parameters, building on preliminary work in modular reasoning for continuous-time and continuous-space models [8]. Another interesting direction is adapting assume-guarantee reasoning for stochastic games [54] to a parametric setting.

References

- 1 Rajeev Alur and Thomas A. Henzinger. Reactive modules. *Formal Methods Syst. Des.*, 15(1):7–48, 1999. doi:10.1023/A:1008739929481.
- 2 Lacramioara Astefanoaei, Saddek Bensalem, Marius Bozga, Chih-Hong Cheng, and Harald Ruess. Compositional parameter synthesis. In John S. Fitzgerald, Constance L. Heitmeyer, Stefania Gnesi, and Anna Philippou, editors, *FM 2016: Formal Methods - 21st International Symposium, Limassol, Cyprus, November 9-11, 2016, Proceedings*, volume 9995 of *Lecture Notes in Computer Science*, pages 60–68, 2016. doi:10.1007/978-3-319-48989-6_4.
- 3 Christel Baier and Joost-Pieter Katoen. *Principles of model checking*. MIT Press, 2008.
- 4 Samik Basu and C. R. Ramakrishnan. Compositional analysis for verification of parameterized systems. *Theor. Comput. Sci.*, 354(2):211–229, 2006. doi:10.1016/J.TCS.2005.11.016.
- 5 Redouane Bouchekir and Mohand Cherif Boukala. Toward implicit learning for the compositional verification of Markov decision processes. In Mohamed Faouzi Atig, Saddek Bensalem, Simon Bliudze, and Bruno Monsuez, editors, *Verification and Evaluation of Computer and Communication Systems - 12th International Conference, VECoS 2018, Grenoble, France, September 26-28, 2018, Proceedings*, volume 11181 of *Lecture Notes in Computer Science*, pages 200–217. Springer, 2018. doi:10.1007/978-3-030-00359-3_13.
- 6 Redouane Bouchekir, Saïda Boukhedouma, and Mohand Cherif Boukala. Automatic compositional verification of probabilistic safety properties for inter-organisational workflow processes. In Yuri Merkuriev, Tuncer I. Ören, and Mohammad S. Obaidat, editors, *Proceedings of the 6th International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH 2016), Lisbon, Portugal, July 29-31, 2016*, pages 244–253. SciTePress, 2016. doi:10.5220/0005978602440253.
- 7 Radu Calinescu, Shinji Kikuchi, and Kenneth Johnson. Compositional reverification of probabilistic safety properties for large-scale complex IT systems. In Radu Calinescu and David Garlan, editors, *Large-Scale Complex IT Systems. Development, Operation and Management - 17th Monterey Workshop 2012, Oxford, UK, March 19-21, 2012, Revised Selected Papers*, volume 7539 of *Lecture Notes in Computer Science*, pages 303–329. Springer, 2012. doi:10.1007/978-3-642-34059-8_16.
- 8 Luca Cardelli, Kim G. Larsen, and Radu Mardare. Modular Markovian logic. In Luca Aceto, Monika Henzinger, and Jiri Sgall, editors, *Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part II*, volume 6756 of *Lecture Notes in Computer Science*, pages 380–391. Springer, 2011. doi:10.1007/978-3-642-22012-8_30.
- 9 Souymodip Chakraborty and Joost-Pieter Katoen. Parametric LTL on Markov chains. In Josep Díaz, Ivan Lanese, and Davide Sangiorgi, editors, *Theoretical Computer Science - 8th IFIP TC 1/WG 2.2 International Conference, TCS 2014, Rome, Italy, September 1-3, 2014. Proceedings*, volume 8705 of *Lecture Notes in Computer Science*, pages 207–221. Springer, 2014. doi:10.1007/978-3-662-44602-7_17.

- 10 Krishnendu Chatterjee, Martin Chmelik, and Przemyslaw Daca. CEGAR for compositional analysis of qualitative properties in Markov decision processes. *Formal Methods Syst. Des.*, 47(2):230–264, 2015. doi:10.1007/S10703-015-0235-2.
- 11 Edmund M. Clarke, Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith. Counterexample-guided abstraction refinement. In E. Allen Emerson and A. Prasad Sistla, editors, *Computer Aided Verification, 12th International Conference, CAV 2000, Chicago, IL, USA, July 15-19, 2000, Proceedings*, volume 1855 of *Lecture Notes in Computer Science*, pages 154–169. Springer, 2000. doi:10.1007/10722167_15.
- 12 Edmund M. Clarke, David E. Long, and Kenneth L. McMillan. Compositional model checking. In *Proceedings of the Fourth Annual Symposium on Logic in Computer Science (LICS '89), Pacific Grove, California, USA, June 5-8, 1989*, pages 353–362. IEEE Computer Society, 1989. doi:10.1109/LICS.1989.39190.
- 13 Jamieson M. Cobleigh, Dimitra Giannakopoulou, and Corina S. Pasareanu. Learning assumptions for compositional verification. In Hubert Garavel and John Hatcliff, editors, *Tools and Algorithms for the Construction and Analysis of Systems, 9th International Conference, TACAS 2003, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2003, Warsaw, Poland, April 7-11, 2003, Proceedings*, volume 2619 of *Lecture Notes in Computer Science*, pages 331–346. Springer, 2003. doi:10.1007/3-540-36577-X_24.
- 14 Conrado Daws. Symbolic and parametric model checking of discrete-time Markov chains. In *ICTAC*, volume 3407 of *Lecture Notes in Computer Science*, pages 280–294. Springer, 2004. doi:10.1007/978-3-540-31862-0_21.
- 15 Luca de Alfaro, Thomas A. Henzinger, and Ranjit Jhala. Compositional methods for probabilistic systems. In Kim Guldstrand Larsen and Mogens Nielsen, editors, *CONCUR 2001 - Concurrency Theory, 12th International Conference, Aalborg, Denmark, August 20-25, 2001, Proceedings*, volume 2154 of *Lecture Notes in Computer Science*, pages 351–365. Springer, 2001. doi:10.1007/3-540-44685-0_24.
- 16 Karam Abd Elkader, Orna Grumberg, Corina S. Pasareanu, and Sharon Shoham. Automated circular assume-guarantee reasoning. *Formal Aspects Comput.*, 30(5):571–595, 2018. doi:10.1007/S00165-017-0436-0.
- 17 Kousha Etessami, Marta Z. Kwiatkowska, Moshe Y. Vardi, and Mihalis Yannakakis. Multi-objective model checking of Markov decision processes. *Log. Methods Comput. Sci.*, 4(4), 2008. doi:10.2168/LMCS-4(4:8)2008.
- 18 Lu Feng, Marta Z. Kwiatkowska, and David Parker. Compositional verification of probabilistic systems using learning. In *QEST 2010, Seventh International Conference on the Quantitative Evaluation of Systems, Williamsburg, Virginia, USA, 15-18 September 2010*, pages 133–142. IEEE Computer Society, 2010. doi:10.1109/QEST.2010.24.
- 19 Lu Feng, Marta Z. Kwiatkowska, and David Parker. Automated learning of probabilistic assumptions for compositional reasoning. In Dimitra Giannakopoulou and Fernando Orejas, editors, *Fundamental Approaches to Software Engineering - 14th International Conference, FASE 2011, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2011, Saarbrücken, Germany, March 26-April 3, 2011. Proceedings*, volume 6603 of *Lecture Notes in Computer Science*, pages 2–17. Springer, 2011. doi:10.1007/978-3-642-19811-3_2.
- 20 Lu Feng, Clemens Wiltsche, Laura R. Humphrey, and Ufuk Topcu. Controller synthesis for autonomous systems interacting with human operators. In *ICCPs*, pages 70–79. ACM, 2015. doi:10.1145/2735960.2735973.
- 21 Vojtech Forejt, Marta Z. Kwiatkowska, Gethin Norman, and David Parker. Automated verification techniques for probabilistic systems. In Marco Bernardo and Valérie Issarny, editors, *Formal Methods for Eternal Networked Software Systems - 11th International School on Formal Methods for the Design of Computer, Communication and Software Systems, SFM 2011, Bertinoro, Italy, June 13-18, 2011. Advanced Lectures*, volume 6659 of *Lecture Notes in Computer Science*, pages 53–113. Springer, 2011. doi:10.1007/978-3-642-21455-4_3.

- 22 Hadar Frenkel, Orna Grumberg, Corina S. Pasareanu, and Sarai Sheinvald. Assume, guarantee or repair: a regular framework for non regular properties. *Int. J. Softw. Tools Technol. Transf.*, 24(5):667–689, 2022. doi:10.1007/S10009-022-00669-9.
- 23 Anubhav Gupta, Kenneth L. McMillan, and Zhaohui Fu. Automated assumption generation for compositional verification. *Formal Methods Syst. Des.*, 32(3):285–301, 2008. doi:10.1007/S10703-008-0050-0.
- 24 Fei He, Xiaowei Gao, Miaofei Wang, Bow-Yaw Wang, and Lijun Zhang. Learning weighted assumptions for compositional verification of Markov decision processes. *ACM Trans. Softw. Eng. Methodol.*, 25(3):21:1–21:39, 2016. doi:10.1145/2907943.
- 25 Thomas A. Henzinger, Shaz Qadeer, and Sriram K. Rajamani. You assume, we guarantee: Methodology and case studies. In Alan J. Hu and Moshe Y. Vardi, editors, *Computer Aided Verification, 10th International Conference, CAV '98, Vancouver, BC, Canada, June 28 - July 2, 1998, Proceedings*, volume 1427 of *Lecture Notes in Computer Science*, pages 440–451. Springer, 1998. doi:10.1007/BFB0028765.
- 26 Cliff B. Jones. Tentative steps toward a development method for interfering programs. *ACM Trans. Program. Lang. Syst.*, 5(4):596–619, 1983. doi:10.1145/69575.69577.
- 27 Sebastian Junges. *Parameter synthesis in Markov models*. PhD thesis, RWTH Aachen University, Germany, 2020. URL: <https://publications.rwth-aachen.de/record/783179>.
- 28 Sebastian Junges, Erika Ábrahám, Christian Hensel, Nils Jansen, Joost-Pieter Katoen, Tim Quatmann, and Matthias Volk. Parameter synthesis for Markov models: covering the parameter space. *Formal Methods Syst. Des.*, 62(1):181–259, 2024. doi:10.1007/S10703-023-00442-X.
- 29 Sebastian Junges, Joost-Pieter Katoen, Guillermo A. Pérez, and Tobias Winkler. The complexity of reachability in parametric Markov decision processes. *J. Comput. Syst. Sci.*, 119:183–210, 2021. doi:10.1016/J.JCSS.2021.02.006.
- 30 Sebastian Junges and Matthijs T. J. Spaan. Abstraction-refinement for hierarchical probabilistic models. In Sharon Shoham and Yakir Vizel, editors, *Computer Aided Verification - 34th International Conference, CAV 2022, Haifa, Israel, August 7-10, 2022, Proceedings, Part I*, volume 13371 of *Lecture Notes in Computer Science*, pages 102–123. Springer, 2022. doi:10.1007/978-3-031-13185-1_6.
- 31 Joost-Pieter Katoen. The probabilistic model checking landscape. In *LICS*, pages 31–45. ACM, 2016. doi:10.1145/2933575.2934574.
- 32 Anvesh Komuravelli, Corina S. Pasareanu, and Edmund M. Clarke. Assume-guarantee abstraction refinement for probabilistic systems. In P. Madhusudan and Sanjit A. Seshia, editors, *Computer Aided Verification - 24th International Conference, CAV 2012, Berkeley, CA, USA, July 7-13, 2012 Proceedings*, volume 7358 of *Lecture Notes in Computer Science*, pages 310–326. Springer, 2012. doi:10.1007/978-3-642-31424-7_25.
- 33 Marta Z. Kwiatkowska, Gethin Norman, and David Parker. Using probabilistic model checking in systems biology. *SIGMETRICS Perform. Evaluation Rev.*, 35(4):14–21, 2008. doi:10.1145/1364644.1364651.
- 34 Marta Z. Kwiatkowska, Gethin Norman, David Parker, and Hongyang Qu. Assume-guarantee verification for probabilistic systems. In Javier Esparza and Rupak Majumdar, editors, *Tools and Algorithms for the Construction and Analysis of Systems, 16th International Conference, TACAS 2010, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2010, Paphos, Cyprus, March 20-28, 2010. Proceedings*, volume 6015 of *Lecture Notes in Computer Science*, pages 23–37. Springer, 2010. doi:10.1007/978-3-642-12002-2_3.
- 35 Marta Z. Kwiatkowska, Gethin Norman, David Parker, and Hongyang Qu. Compositional probabilistic verification through multi-objective model checking. *Inf. Comput.*, 232:38–65, 2013. doi:10.1016/J.IC.2013.10.001.
- 36 Rui Li and Yang Liu. Compositional stochastic model checking probabilistic automata via symmetric assume-guarantee rule. In *17th IEEE International Conference on Software Engineering Research, Management and Applications, SERA 2019, Honolulu, HI, USA, May 29-31, 2019*, pages 110–115. IEEE, 2019. doi:10.1109/SERA.2019.8886808.

- 37 Hannah Mertens, Joost-Pieter Katoen, Tim Quatmann, and Tobias Winkler. Accurately computing expected visiting times and stationary distributions in Markov chains. In Bernd Finkbeiner and Laura Kovács, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 30th International Conference, TACAS 2024, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2024, Luxembourg City, Luxembourg, April 6-11, 2024, Proceedings, Part II*, volume 14571 of *Lecture Notes in Computer Science*, pages 237–257. Springer, 2024. doi:10.1007/978-3-031-57249-4_12.
- 38 Hannah Mertens, Tim Quatmann, and Joost-Pieter Katoen. Compositional reasoning for parametric probabilistic automata, 2025. arXiv:2506.08525.
- 39 Kedar S. Namjoshi and Richard J. Treller. Parameterized compositional model checking. In Marsha Chechik and Jean-François Raskin, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 22nd International Conference, TACAS 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2-8, 2016, Proceedings*, volume 9636 of *Lecture Notes in Computer Science*, pages 589–606. Springer, 2016. doi:10.1007/978-3-662-49674-9_39.
- 40 Gethin Norman and Vitaly Shmatikov. Analysis of probabilistic contract signing. *J. Comput. Secur.*, 14(6):561–589, 2006. doi:10.3233/jcs-2006-14604.
- 41 Corina S. Pasareanu, Dimitra Giannakopoulou, Mihaela Gheorghiu Bobaru, Jamieson M. Cobleigh, and Howard Barringer. Learning to divide and conquer: applying the L* algorithm to automate assume-guarantee reasoning. *Formal Methods Syst. Des.*, 32(3):175–205, 2008. doi:10.1007/S10703-008-0049-6.
- 42 Corina S. Pasareanu, Divya Gopinath, and Huafeng Yu. Compositional verification for autonomous systems with deep learning components. *CoRR*, abs/1810.08303, 2018. arXiv:1810.08303.
- 43 Corina S. Pasareanu, Ravi Mangal, Divya Gopinath, and Huafeng Yu. Assumption generation for learning-enabled autonomous systems. In Panagiotis Katsaros and Laura Nenzi, editors, *Runtime Verification - 23rd International Conference, RV 2023, Thessaloniki, Greece, October 3-6, 2023, Proceedings*, volume 14245 of *Lecture Notes in Computer Science*, pages 3–22. Springer, 2023. doi:10.1007/978-3-031-44267-4_1.
- 44 Amir Pnueli. In transition from global to modular temporal reasoning about programs. In Krzysztof R. Apt, editor, *Logics and Models of Concurrent Systems - Conference proceedings, Colle-sur-Loup (near Nice), France, 8-19 October 1984*, volume 13 of *NATO ASI Series*, pages 123–144. Springer, 1984. doi:10.1007/978-3-642-82453-1_5.
- 45 Tim Quatmann, Christian Dehnert, Nils Jansen, Sebastian Junges, and Joost-Pieter Katoen. Parameter synthesis for Markov models: Faster than ever. In Cyrille Artho, Axel Legay, and Doron Peled, editors, *Automated Technology for Verification and Analysis - 14th International Symposium, ATVA 2016, Chiba, Japan, October 17-20, 2016, Proceedings*, volume 9938 of *Lecture Notes in Computer Science*, pages 50–67, 2016. doi:10.1007/978-3-319-46520-3_4.
- 46 Roberto Segala. *Modeling and verification of randomized distributed real-time systems*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 1995. URL: <https://hdl.handle.net/1721.1/36560>.
- 47 Roberto Segala and Nancy A. Lynch. Probabilistic simulations for probabilistic processes. *Nord. J. Comput.*, 2(2):250–273, 1995.
- 48 Antti Siirtola and Keijo Heljanko. Parametrised compositional verification with multiple process and data types. In Josep Carmona, Mihai T. Lazarescu, and Marta Pietkiewicz-Koutny, editors, *13th International Conference on Application of Concurrency to System Design, ACSD 2013, Barcelona, Spain, 8-10 July, 2013*, pages 60–69. IEEE Computer Society, 2013. doi:10.1109/ACSD.2013.9.
- 49 Jip Spel. *Monotonicity in Markov models*. PhD thesis, RWTH Aachen University, Germany, 2023. URL: <https://publications.rwth-aachen.de/record/974903>.
- 50 Jip Spel, Sebastian Junges, and Joost-Pieter Katoen. Are parametric Markov chains monotonic? In Yu-Fang Chen, Chih-Hong Cheng, and Javier Esparza, editors, *Automated Technology*

- for Verification and Analysis - 17th International Symposium, ATVA 2019, Taipei, Taiwan, October 28-31, 2019, Proceedings*, volume 11781 of *Lecture Notes in Computer Science*, pages 479–496. Springer, 2019. doi:10.1007/978-3-030-31784-3_28.
- 51 Jip Spel, Sebastian Junges, and Joost-Pieter Katoen. Finding provably optimal Markov chains. In Jan Friso Groote and Kim Guldstrand Larsen, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 27th International Conference, TACAS 2021, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2021, Luxembourg City, Luxembourg, March 27 - April 1, 2021, Proceedings, Part I*, volume 12651 of *Lecture Notes in Computer Science*, pages 173–190. Springer, 2021. doi:10.1007/978-3-030-72016-2_10.
- 52 Mariëlle Stoelinga. An introduction to probabilistic automata. *Bull. EATCS*, 78:176–198, 2002.
- 53 Kazuki Watanabe, Clovis Eberhart, Kazuyuki Asada, and Ichiro Hasuo. Compositional solution of mean payoff games by string diagrams. In Nils Jansen, Sebastian Junges, Benjamin Lucien Kaminski, Christoph Matheja, Thomas Noll, Tim Quatmann, Mariëlle Stoelinga, and Matthias Volk, editors, *Principles of Verification: Cycling the Probabilistic Landscape - Essays Dedicated to Joost-Pieter Katoen on the Occasion of His 60th Birthday, Part III*, volume 15262 of *Lecture Notes in Computer Science*, pages 423–445. Springer, 2024. doi:10.1007/978-3-031-75778-5_20.
- 54 Clemens Wiltsche. *Assume-guarantee strategy synthesis for stochastic games*. PhD thesis, University of Oxford, UK, 2015. URL: <https://ethos.bl.uk/OrderDetails.do?uin=uk.bl.ethos.719857>.
- 55 Xiaobin Zhang, Bo Wu, and Hai Lin. Assume-guarantee reasoning framework for MDP-POMDP. In *55th IEEE Conference on Decision and Control, CDC 2016, Las Vegas, NV, USA, December 12-14, 2016*, pages 795–800. IEEE, 2016. doi:10.1109/CDC.2016.7798365.