Eleventh International Conference on Application of Concurrency to System Design

New Results on Abstract Probabilistic Automata

Benoît Delahaye[†] Joost-Pieter Katoen[‡] Kim G. Larsen[§] Axel Legay[†] Mikkel L. Pedersen[§] Falak Sher[‡] Andrzej Wąsowski[¶]

[†]INRIA/IRISA, Rennes, France
[‡]RWTH Aachen University, Germany
[§]Aalborg University, Denmark
[¶]IT University of Copenhagen, Denmark

***Abstract*—Probabilistic Automata (PAs) are a recognized framework for modeling and analysis of nondeterministic systems with stochastic behavior. Recently, we proposed Abstract Probabilistic Automata (APAs)—an abstraction framework for PAs. In this paper, we discuss APAs over dissimilar alphabets, a determinisation operator, conjunction of non-deterministic APAs, and an APA-embedding of Interface Automata. We conclude introducing a tool for automatic manipulation of APAs.**

I. Introduction

Probabilistic Automata (PAs), proposed by Segala [1], are a mathematical framework for rigorous specification and analysis of non-deterministic probabilistic systems, or more precisely systems that combine concurrent behaviour with discrete probabilistic choice. PAs are akin to Markov decision processes (MDPs). A detailed comparison with models such as MDPs, as well as generative and reactive probabilistic transition systems is given in [2]. PAs are recognized as an adequate formalism for various applications including randomized distributed algorithms and fault tolerant systems [3], [4], [5], [6], [7].

Recently [8], we have proposed Abstract Probabilistic Automata (APAs), that is a compact abstraction formalism for sets of PAs. The model is a marriage between our new abstract model for Markov chains [9] and modal automata, an abstraction for non-deterministic systems promoted by [10] and [11]. In an APA, non-deterministic behaviors are typed with may and must modalities. The must modalities identify those behaviors that must be present in any implementation, while the may modalities refer to those behaviors that are allowed to be omitted in an implementation. In APAs, probability distributions that govern the successor states are replaced by set of distributions, each of them representing a possible implementation of the abstraction.

One of the major contributions of [8] was to develop the first specification theory for PAs. This includes a satisfaction relation (to decide whether a PA is an implementation of an APA), a consistency check (to decide whether the specification admits an implementation), a refinement (to compare specifications in terms of inclusion of sets of implementations), logical composition (to compute the intersection of sets of implementations), and structural composition (to combine specifications). Our framework also supports incremental design [12]. In addition, we have proposed an abstraction mechanism that allows to simplify the design in an aggressive manner.

While the theory is already quite complete, some fundamental aspects have to be improved in order to make it attractive from a design point of view. First, our theory assumes that non-stochastic behaviors of the components are defined over the same alphabets. In various contexts this assumption is unrealistic. Indeed, one should be able to combine the existing design with new components whose ports and variables are not yet specified [13].

Second, the conjunction operation has only been defined for those systems whose non-stochastic behaviors are described in a deterministic manner. Again, from the practical point of view, one should be capable of handling non-determinism inherent to transition systems and concurrency.

Third, the existing composition operator for APAs assumes closed system composition, inhibiting reasoning about open systems. Support for open systems, enables incremental modeling and allows reasoning not only about stochastic components, but also about the requirements for their usage (environment).

The aim of this paper is to propose solutions to the above mentioned problems. Our contributions are described below.

- We extend the theory of APAs to support specifications over dissimilar alphabets. The principle is similar to what has been proposed for modal automata in [10]. Unfortunately due to interweaving of probabilistic and non-deterministic choices, proofs of correctness of [10] could not be reused.
- We show that the definition of conjunction proposed in [8] is too strong for non-deterministic APAs. We propose a more general construction that corresponds to the greatest lower bound with respect to a new refinement relation, more precise than refinements introduced before [8].
  This result is of additional theoretical interest. In [14] we have shown that such greatest lower bound generally does not exist for modal automata. Nevertheless it was possible to introduce it for APAs, which contain modal automata. The new construction works for modal automata encoded as APAs, because it potentially produces an APA which is not an encoding of any modal automaton.
- We present a determinization algorithm that given an APA whose non-stochastic behaviors are non-deterministic, computes its deterministic abstraction. We also show that there are APAs for which there exists no deterministic APAs accepting the same set of models (so determiniza-

1550-4808/11 $26.00 © 2011 IEEE
DOI 10.1109/ACSD.2011.10

tion must be lossy). This lossiness of the abstraction further motivates the need for the weaker conjunction operator mentioned above.

- We propose a translation of APAs to Abstract Probabilistic Interfaces (API) that is a stochastic extension of the classical game-based interface automata proposed by de Alfaro et al. APIs are similar to the stochastic I/O automata of Lynch except that they encompass a game-based semantics that allows for an optimistic composition. Given two APIs, one can compute the environment in where they can work together in a proper manner.
- We introduce the APAC tool, in which the APA theory has been implemented. APAC relies on the SMT solver Z3 [15] for checking relations between the probability distributions of the components. To the best of our knowledge, this is the first implementation of a theory that proposes both logical and structural compositions for Probabilistic Automata.

II. Background

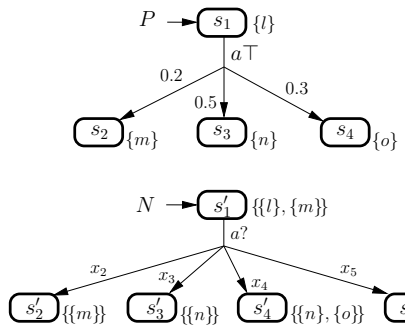
We now briefly introduce the specification theory of *Abstract Probabilistic Automata* as presented in [8]. We begin with the notion of a probabilistic automaton [1]. Let $\mathit{Dist}(S)$ denote a set of all discrete probability distributions over a finite set S, and let $\mathbb{B}_2 = \{\top, \bot\}$. Then:

Definition 1: A *probabilistic automaton* (PA) is a tuple (S, A, L, AP, V, s_0), where S is a finite set of states with the initial state $s_0 \in S$, A is a finite set of actions, L: $S \times A \times \mathit{Dist}(S) \to \mathbb{B}_2$ is a (two-valued transition) function, AP is a finite set of atomic propositions and V: $S \to 2^{AP}$ is a state-labeling function.

For a state s, an action a and a probability distribution μ, the value of $L(s, a, \mu)$ symbolizes the presence ($\top$) or absence ($\bot$) of a transition from s under action a to a distribution μ specifying possible successor states. In practice L may be a partial function—if a value of L is unspecified for a given combination of arguments, then it behaves as if it was specified to be $\bot$.

Example 1: The top of Figure 1 shows a PA P over the singleton set of actions $A = \{a\}$ and atomic propositions $AP = \{l, m, n, o\}$. In the figure, $\top$-transitions are drawn explicitly, and $\bot$-transitions are elided. For example, in P there is one a-transition from s_1 to the distribution $[0, 0.2, 0.5, 0.3]$.

An abstract probabilistic automaton relaxes the above definition to allow describing multiple probabilistic automata (including their probability distributions) within a single abstraction. Let $C(S)$ denote a set of all constraints over discrete probability distributions over a finite set S; so that each element $\varphi \in C(S)$ describes a set of distributions: $\mathit{Sat}(\varphi) \subseteq \mathit{Dist}(S)$. In this paper we do not fix the language of constraints used to generate $C(S)$. Instead, we just require that $C(S)$ is closed under usual Boolean connectives, that it includes equalities over summations and multiplications of probability values, and that it allows for existential quantification of variables. Also let $\mathbb{B}_3 = \{\top, ?, \bot\}$. Then:



$\varphi_x \equiv (x_2 + x_3 \geq 0.7) \wedge (x_4 + x_5 \geq 0.2) \wedge (x_2 + x_3 + x_4 + x_5 = 1)$

Fig. 1: Examples of a PA (top) and of an APA (bottom)

Definition 2: An *Abstract Probabilistic Automaton* (APA) is a tuple (S, A, L, AP, V, s_0), where S is a finite set of states, $s_0 \in S$, A is a finite set of actions, and AP is a finite set of atomic propositions. $L : S \times A \times C(S) \to \mathbb{B}_3$ is a *three*-valued distribution-constraint function, and $V : S \to 2^{2^{AP}}$ maps each state in S to a set of admissible labelings.

APAs play the role of specifications in our framework. An APA transition abstracts transitions of a certain unknown PA, called its implementation. Given a state s, an action a, and a constraint φ, the value of $L(s, a, \varphi)$ gives the modality of the transition. More precisely the value $\top$ means that transitions under a must exist in the PA to every distribution in $\mathit{Sat}(\varphi)$; ? means that these transitions are permitted to exist; $\bot$ means that such transitions must not exist. Again L may be partial. In practice, as will be seen in later definitions, a lack of value for given argument is equivalent to the $\bot$ value, so we will sometimes avoid defining $\bot$-value rules in constructions to avoid clutter, and occasionally will say that something applies if L takes the value of $\bot$, meaning that it is either taking this value or it is undefined. The function V labels each state with a subset of the powerset of AP, which models a disjunctive choice of possible combinations of atomic propositions.

We occasionally write $\mathsf{Must}(s)$ for the set of all actions a such that there exists φ, so that $L(s, a, \varphi) = \top$, and write $\mathsf{May}(s)$ for the set of all actions b such that there exists ψ, so that $L(s, b, \psi) = ?$.

Example 2: An example of an APA N, with the same signature as the PA P, is shown in the bottom of Figure 1. Again we follow a graphical convention of eliding $\bot$-transitions. Must ($\top$) and may (?) transitions are shown explicitly with modalities appended to the action label. Here, in the APA N, there is one allowed a-transition from N to a constraint φ_x (specified under the automaton).

A PA is essentially an APA in which every transition $L(s, a, \mu) = m$ is represented by the same modality transition $L(s, a, \varphi) = m$ with $\mathit{Sat}(\varphi) = \{\mu\}$, and each state-label consists of a single set of propositions.

As already mentioned, we relate APA specifications to PAs implementing them, by extending the definitions of satisfaction

introduced in [16]. We begin by relating distributions between sets of states [8]:

Definition 3: Let S and S' be non-empty sets, and μ, μ' be distributions; $\mu \in \mathit{Dist}(S)$ and $\mu' \in \mathit{Dist}(S')$. We say that μ is *simulated* by μ' with respect to a relation $R \subseteq S \times S'$ and a *correspondance function* $\delta : S \to (S' \to [0,1])$ iff

1) For all $s \in S$, $\delta(s)$ is a distribution on S' if $\mu(s) > 0$
2) For all $s' \in S'$, $\sum_{s \in S} \mu(s) \cdot \delta(s)(s') = \mu'(s')$,
3) Whenever $\delta(s)(s') > 0$ then $(s, s') \in R$.

We write $\mu \Subset^{\delta}_{R} \mu'$ meaning that μ is simulated by μ' with respect to R and δ, and we write $\mu \Subset_{R} \mu'$ iff there exists a function δ such that $\mu \Subset^{\delta}_{R} \mu'$.

Now, the following definition, originating in [8], formally establishes the roles of PAs and APAs as implementations and specifications respectively. For a PA P satisfying an APA N we require that any must-transition of N is matched by a must-transition of P agreeing with the distributions specified by the constraint, and any must-transition of P is matched by a may- or must-transition in N.

Definition 4: Let $P = (S, A, L, \mathit{AP}, V, s_0)$ be a PA and $N = (S', A, L', \mathit{AP}, V', s'_0)$ be an APA. A binary relation $R \subseteq S \times S'$ is a *satisfaction relation* iff, for any $(s, s') \in R$, the following conditions hold:

1) Whenever $L'(s', a, \varphi') = \top$ for some $a \in A$, $\varphi' \in C(S')$ then also $L(s, a, \mu) = \top$ for some distribution μ such that $\mu \Subset_R \mu'$ and $\mu' \in \mathit{Sat}(\varphi')$.
2) Whenever $L(s, a, \mu) = \top$ for some $a \in A$, $\mu \in \mathit{Dist}(S)$ then $L(s', a, \varphi')$ is defined with $L'(s', a, \varphi') \neq \bot$ for some $\varphi' \in C(S')$ and $\mu' \in \mathit{Sat}(\varphi')$ such that $\mu \Subset_R \mu'$.
3) $V(s) \in V'(s')$.

We say that P satisfies N, denoted $P \models N$, iff there exists a satisfaction relation relating s_0 and s'_0. If $P \models N$, then P is called an implementation of (specification) N.

Example 3: The relation $\mathcal{R} = \{(s_1 s'_1), (s_2 s'_2), (s_3 s'_3), (s_4 s'_4), (s_4 s'_5)\}$ is a satisfaction relation between P and N of Fig. 1. It is easy to see that all pairs in $R \setminus \{(s_1, s'_1)\}$ fulfill the definition, as they have no outgoing transitions and the labelings of states in P respect the labeling constraints of N.

So consider (s_1, s'_1). Condition 2 is satisfied vacuously. Take $\mu' = [0, 0.2, 0.5, 0.15, 0.15] \in \mathit{Sat}(\varphi_x)$. Let $\mu = [0, 0.2, 0.5, 0.3]$ be the distribution of the only a-transition of P. We are showing that condition 1 above is satisfied, i.e. that $\mu \Subset_R \mu'$. This is witnessed by the following correspondance function: $\delta(s_2, s'_2) = \delta(s_3, s'_3) = 1$, $\delta(s_4, s'_4) = \delta(s_5, s'_5) = 0.5$, and $\delta(s_i, s'_j) = 0$ for all remaining pairs of states. □

We denote the set of all implementations of N by $[\![N]\!] = \{P \mid P \models N\}$. An APA N is said to be *consistent* iff $[\![N]\!] \neq \emptyset$. A state s of an APA is called consistent if and only if $V(s) \neq \emptyset$ and $(L(s, a, \varphi) = \top \implies \mathit{Sat}(\varphi) \neq \emptyset)$. If all states of N are consistent then N is consistent, but not necessarily the other way around.

In [8], a pruning operator β is defined that filters out distributions leading to inconsistent states, making these states unreachable. After a single application of β to an APA N, it holds that $[\![N]\!] = [\![\beta(N)]\!]$. Pruning itself may introduce inconsistent states, so we apply β until a fixpoint is reached, which is guaranteed to happen after a finite number of steps.

We say that an APA N *thoroughly refines* another APA M iff $[\![N]\!] \subseteq [\![M]\!]$. Such notion of refinement, although theoretically satisfying, is not easy to establish algorithmically. For this reason [8] introduces a more syntactic refinement, called a weak refinement:

Definition 5: Let $N = (S, A, L, \mathit{AP}, V, s_0)$ and $N' = (S', A, L', \mathit{AP}, V', s'_0)$ be APAs. A binary relation $R \subseteq S \times S'$ is a *weak refinement* relation iff, for all $(s, s') \in R$, the following conditions hold:

1) Whenever $L'(s', a, \varphi') = \top$ for some action $a \in A$ and distribution constraint $\varphi' \in C(S')$ then $L(s, a, \varphi) = \top$ for some distribution constraint $\varphi \in C(S)$ such that $\forall \mu \in \mathit{Sat}(\varphi).\, \exists \mu' \in \mathit{Sat}(\varphi').\, \mu \Subset_R \mu'$
2) Whenever $L(s, a, \varphi) \neq \bot$ for some $a \in A$ and $\varphi \in C(S)$ then $L'(s', a, \varphi') \neq \bot$ for some constraint $\varphi' \in C(S')$ such that $\forall \mu \in \mathit{Sat}(\varphi).\, \exists \mu' \in \mathit{Sat}(\varphi').\, \mu \Subset_R \mu'$
3) $V(s) \subseteq V'(s')$.

We say that N weakly refines N', denoted $N \preceq N'$, iff there exists a weak refinement relation relating s_0 and s'_0.

The correspondence function δ is not fixed in advance, and can be chosen for each μ and μ' separately so that $\mu \Subset^{\delta}_{\mathcal{R}} \mu'$. The weak refinement is sound with respect to the thorough refinement: if $N \preceq N'$ then $[\![N]\!] \subseteq [\![N']\!]$ [8]. It is known that the two refinements coincide for deterministic APAs if the initial state admits exactly one labeling: ($|V(s_0)| = 1$) [8].

Definition 6: An APA $N = (S, A, L, \mathit{AP}, V, s_0)$ is deterministic if it satisfies the following two conditions:

[action-determinism] An action determines the successor: $\forall s \in S. \forall a \in A. |\{\varphi \in C(S) \mid L(s, a, \varphi) \neq \bot\}| \leq 1$.

[labeling-determinism] Labels discern possible successor states: $\forall s \in S. \forall a \in A. \forall \varphi \in C(S)$ if $L(s, a, \varphi) \neq \bot$ then:

$$\forall \mu', \mu'' \in \mathit{Sat}(\varphi), s', s'' \in S.$$
$$(\mu'(s') > 0 \wedge \mu''(s'') > 0 \;\Rightarrow V(s') \cap V(s'') = \emptyset)\,.$$

Example 4: The APA N in Fig. 1 is action-deterministic, but not labeling-deterministic. The distributions $\mu' = [0, 0.4, 0.4, 0.1, 0.1]$, $\mu'' = [0, 0.5, 0.2, 0.3, 0)$ are both in $\mathit{Sat}(\varphi_x)$ and give positive probability to s'_3 and s'_4, respectively, while their labeling constraints intersect on $\{\{n\}\}$.

To conclude this section, we present the definition of parallel composition, which is known to be a precongruence with respect to weak refinement [8].

Definition 7: Let $N = (S, A, L, \mathit{AP}, V, s_0)$ and $N' = (S', A', L', \mathit{AP}', V', s'_0)$ be APAs with $\mathit{AP} \cap \mathit{AP}' = \emptyset$. The parallel composition of $N \|_{\bar{A}} N'$ with respect to a synchronization set $\bar{A} \subseteq A \cap A'$ is given as $N \|_{\bar{A}} N' = (S \times S', A \cup A', \tilde{L}, \mathit{AP} \cup \mathit{AP}', \tilde{V}, (s_0, s'_0))$ and

1) For each $a \in \bar{A}$

$$\frac{\exists\varphi.\, L(s,a,\varphi) \neq \bot \qquad \forall\varphi'.\, L'(s',a,\varphi') \neq \bot}{\tilde{L}((s,s'),a,\tilde{\varphi}) = L(s,a,\varphi) \sqcap L'(s',a,\varphi')} \quad (1)$$

$$\frac{\forall\varphi.\, L(s,a,\varphi) = \bot \vee \forall\varphi'.\, L(s',a,\varphi') = \bot}{\forall\tilde{\varphi}'.\tilde{L}((s,s'),a,\tilde{\varphi}') = \bot} \quad (2)$$

where $\tilde{\varphi} \in C(S \times S')$ is so that $\tilde{\mu} \in \mathit{Sat}(\tilde{\varphi})$ iff there exists $\mu \in \mathit{Sat}(\varphi)$ and $\mu' \in \mathit{Sat}(\varphi')$ such that $\tilde{\mu}(u,v) = \mu(u) \cdot \mu'(v)$ for all $u \in S$ and $v \in S'$.

2) For each $a \in A \setminus A'$:

$$\frac{\mathit{Sat}(\tilde{\varphi}) = \{\tilde{\mu} \mid \tilde{\mu}(\cdot,s') \in \mathit{Sat}(\varphi), \tilde{\mu}(u,v) = 0 \text{ for } v \neq s'\}}{\tilde{L}((s,s'),a,\tilde{\varphi}) = L(s,a,\varphi)}$$

3) And symmetrically for each $a \in A' \setminus A$:

$$\frac{\mathit{Sat}(\tilde{\varphi}') = \{\tilde{\mu}' \mid \tilde{\mu}'(s,\cdot) \in \mathit{Sat}(\varphi'), \tilde{\mu}'(u,v) = 0 \text{ for } u \neq s\}}{\tilde{L}((s,s'),a,\tilde{\varphi}') = L'(s',a,\varphi')}$$

4) $\tilde{V}((s,s')) = \{B \cup B' \mid B \in V(s) \text{ and } B' \in V'(s')\}$.

III. Extensions of Alphabets

So far, the specification theory of APAs has required that all specifications share same alphabets of actions and labels. We are now going to lift this restriction, by introducing the alphabet extension mechanism. Just like for modal transition systems [17], for which there exist two ways of extending signatures [13], for APAs it is also necessary to choose the modality of transitions for new actions introduced, depending on the operation being applied to the result.

The weak extension is used when conjoining specifications with different signatures. This extension adds may loop transitions for all new actions and extends the sets of atomic propositions in a classical way:

Definition 8: Let $N = (S, A, L, AP, V, s_0)$ be an APA, and let A' and AP' be sets of actions and atomic propositions such that $A \subseteq A'$ and $AP \subseteq AP'$. Let the weak extension of N to (A', AP') be the APA $N \Uparrow (A', AP') = (S, A', L', AP', V', s_0)$ such that for all states $s \in S$:

- $L'(s,a,\varphi) = L(s,a,\varphi)$ if $a \in A$,
- $L'(s,a,\varphi) = ?$ if $a \in A' \setminus A$ and φ only admits a single point distribution μ such that $\mu(s) = 1$.
- $V'(s) = \{B \subseteq AP' \mid B \cap AP \in V(s)\}$.

A different extension, the strong one, is used in parallel composition. This extension adds must self-loops for all new actions and extends the sets of atomic propositions in a classical way. See [18] for a formal definition.

These different notions of extension give rise to different notions of satisfaction and refinement between structures with dissimilar sets of actions. Satisfaction (or refinement) between structures with different sets of actions is defined as the satisfaction (respectively refinement) between the structures after extension to a union of signatures.

IV. Conjunction

A. Incompleteness of Conjunction

A conjunction operator combines two specifications into a single one, ideally describing the intersection of their implementation sets (so $[\![N \wedge M]\!] = [\![N]\!] \cap [\![M]\!]$). In [8], conjunction was only defined for action-deterministic APAs with *identical alphabets*. In this paper, we first show that construction is incorrect for non-deterministic APAs. Then we generalize it to the non-deterministic case *with dissimilar alphabets*. Let's recall the definition given in [8]:

Definition 9: Let $N = (S, A, L, AP, V, s_0)$ and $N' = (S', A, L', AP, V', s'_0)$ be action-deterministic APAs. Their conjunction is the APA $N \wedge N' = (S \times S', A, \tilde{L}, AP, \tilde{V}, (s_0, s'_0))$ where $\tilde{V}((s,s')) = V(s) \cap V'(s')$ and $\tilde{L}$ is defined as follows. Given an action $a \in A$ and a state $(s,s') \in S \times S'$:

$$\frac{\exists\varphi.\, L(s,a,\varphi) = \top \qquad \forall\varphi'.\, L'(s',a,\varphi') = \bot}{\tilde{L}((s,s'),a,\text{false}) = \top} \quad (3)$$

$$\frac{\forall\varphi.\, L(s,a,\varphi) = \bot \qquad \exists\varphi'.\, L'(s',a,\varphi') = \top}{\tilde{L}((s,s'),a,\text{false}) = \top} \quad (4)$$

$$\frac{\forall\varphi.\, L(s,a,\varphi) \neq \top \qquad \forall\varphi'.\, L'(s',a,\varphi') = \bot}{\tilde{L}((s,s'),a,\_) = \bot} \quad (5)$$

$$\frac{\forall\varphi.\, L(s,a,\varphi) = \bot \qquad \forall\varphi'.\, L'(s',a,\varphi') \neq \top}{\tilde{L}((s,s'),a,\_) = \bot} \quad (6)$$

$$\frac{L(s,a,\varphi) \neq \bot \qquad L'(s',a,\varphi') \neq \bot}{\tilde{L}((s,s'),a,\tilde{\varphi}) = L(s,a,\varphi) \sqcup L'(s',a,\varphi')} \quad (7)$$

where $\tilde{\varphi} \in C(S \times S')$ such that $\tilde{\mu} \in \mathit{Sat}(\tilde{\varphi})$ iff both distribution $\mu : t \to \sum_{t' \in S'} \tilde{\mu}((t,t'))$ is in $\mathit{Sat}(\varphi)$ and distribution $\mu' : t' \to \sum_{t \in S} \tilde{\mu}((t,t'))$ is in $\mathit{Sat}(\varphi')$.

In [8] it is shown that this construction captures the greatest lower-bound with respect to weak refinement, i.e. For N, N', and N'' action-deterministic consistent APAs over the same action alphabet we have that:

- $\beta^*(N \wedge N') \preceq N$ and $\beta^*(N \wedge N') \preceq N'$
- If $N'' \preceq N$ and $N'' \preceq N'$ then $N'' \preceq \beta^*(N \wedge N')$.

At the same time this construction is inadequate for non-deterministic APAs. Combining one must-transition with several may-transitions using the same action is problematic. We show that the conjunction of non-deterministic APAs, using the definition above, is not a lower bound with respect to neither thorough refinement nor weak refinement.

Lemma 1: The construction of Def. 9 is strictly stronger than the greatest lower bound of the thorough refinement and of the weak refinement for non-deterministic APAs, in the following sense:

1) There exists a PA I, and APAs N and N' such that $I \models N$ and $I \models N'$ but not $I \models N \wedge N'$.

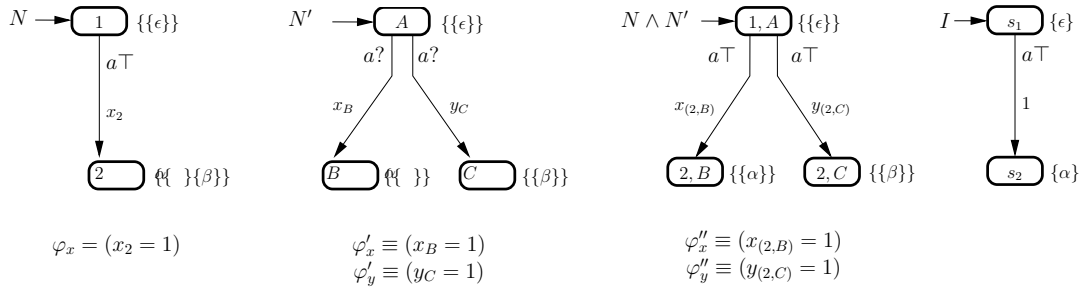


Fig. 2: Illustration that conjunction using Definition 9 is not a greatest lower bound.

2) There exists an APA M, such that $M \preceq N$ and $M \preceq N'$ but not $M \preceq N \wedge N'$.

Proof: Figure 2 presents two APAs $N = (\{1,2\}, \{a\}, L, \{\epsilon, \alpha, \beta\}, V, 1)$ and $N' = (\{A, B, C\}, \{a\}, L', \{\epsilon, \alpha, \beta\}, V', A)$ together with their conjunction $N \wedge N'$, constructed according to Definition 9. The rigthmost part of the figure shows a PA $I = (\{s_1, s_2\}, \{a\}, L_I, \{\epsilon, \alpha, \beta\}, V_I, s_1)$, which shows that the conjunction is too strong with respect to the thorough refinement. It holds that $I \models N$ and $I \models N'$, but $I \not\models (N \wedge N')$. The conjunction of N and N' has two must transitions, and the one leading to $(2, C)$ is not fulfilled by I.

For the second part of the theorem it is sufficient to interpret I as the APA M and the argument follows. ∎

For dissimilar alphabets, conjunction can be treated separately from alphabet extension. One first computes the (weak) alphabet extensions for both APAs, and then compute conjunction using the above definition 9. Formally:

Definition 10: Let $N_1 = (S_1, A_1, L_1, AP_1, V, s_1)$, $N_2 = (S_2, A_2, L_2, AP_2, V_2, s_2)$ be action-deterministic APAs. Their conjunction is the APA $N_1 \wedge N_2 = [N_1 \Uparrow \alpha] \wedge [N_2 \Uparrow \alpha]$, with $\alpha = (A_1 \cup A_2, AP \cup AP')$ and $\wedge$ defined as above.

Considering the above definition for APAs with dissimilar action sets, the following theorem trivially holds.

Theorem 1: Let N_1, N_2, and N_3 be action-deterministic consistent APAs over action alphabets A_1, A_2, A_3 and atomic proposition sets AP_1, AP_2 and AP_3 respectively. Let $\alpha_{ij} = (A_i \cup A_j, AP_i \cup AP_j)$, and $\alpha_{123} = (\bigcup_{i=1}^{3} A_i, \bigcup_{i=1}^{3} AP_i)$. Then:

1) $\beta^*(N_1 \Uparrow \alpha_{12} \wedge N_2 \Uparrow \alpha_{12}) \preceq N_1 \Uparrow \alpha_{12}$
2) If $N_3 \Uparrow \alpha_{123} \preceq N_1 \Uparrow \alpha_{123}$ and $N_3 \Uparrow \alpha_{123} \preceq N_2 \Uparrow \alpha_{123}$ then $N_3 \Uparrow \alpha_{123} \preceq \beta^*(N_1 \Uparrow \alpha_{12} \wedge N_2 \Uparrow \alpha_{12}) \Uparrow \alpha_{123}$

B. Weak weak Refinement

The weak refinement (Def. 5), along with the so called *strong refinement* [8], had been introduced for Constraint Markov Chains in [9], as syntax directed sound characterizations of thorough refinement. They were then generalized to APAs in [8] in a "natural" way.

As we see from Lemma 1 the conjunction construction of [8] is too strong with respect to the weak refinement for non-deterministic systems. In order to address this problem, one can (potentially) either weaken the construction or strenghten the refinement. There are issues with any of the solutions. First, strengthening the refinement makes it even more strong with respect to thorough refinement (so it becomes less precise which is undesirable), and moreover the known strong refinement [8] still violates Lemma 1 (i.e. it is too coarse).

Second, the natural weakening of the construction gives a resulting conjunction APA that is too weak with respect to the weak refinement.

Instead of fine tuning the construction, which could become very complicated, we decided to explore another possibility: namely propose a weaker and more precise refinement, the *weak weak refinement*, which is designed with APAs (and not CMCs) in mind. The weak weak refinement approximates thorough refinement even better than weak refinement, and it has a naturally characterized greatest lower bound.

In the weak refinement, cf. Def. 5, the correspondence function is established for two constraints: for each solution of one, there must be a correspondance to some solution of the other constraints. Weak weak refinement weakens this condition by allowing to choose, for each solution of the first constraint, both a different correspondence function *and* a different constraint (transition) to which it will be linked:

Definition 11: Let $N = (S, A, L, AP, V, s_0)$ and $N' = (S', A', L', AP', V', s'_0)$ be APAs with $AP = AP'$ and $A = A'$. A relation $R \subseteq S \times S'$ is a *weak weak refinement relation*, iff for all $(s, s') \in R$, the following conditions hold:

1) $\forall a \in A'. \forall \varphi' \in C(S'). L'(s', a, \varphi') = \top \implies \exists \varphi \in C(S). L(s, a, \varphi) = \top$ and $\forall \mu \in Sat(\varphi). \exists \mu' \in Sat(\varphi')$ such that $\mu \Subset_R \mu'$,
2) $\forall a \in A. \forall \varphi \in C(S). L(s, a, \varphi) \neq \bot \implies \forall \mu \in Sat(\varphi). \exists \varphi' \in C(S'). L'(s', a, \varphi') \neq \bot$ and $\exists \mu' \in Sat(\varphi')$ such that $\mu \Subset_R \mu'$, and
3) $V(s) \subseteq V'(s')$.

We say that N_1 weakly weakly refines N_2, denoted $N_1 \preceq_W N_2$, iff there exists a weak weak refinement relation relating s_0 and s'_0.

It follows directly that weak weak refinement is weaker than weak refinement and thus strong refinement. For action-deterministic APAs, weak weak refinement is equivalent to weak refinement.

C. Conjunction of Non-deterministic APAs

We thus propose the following definition for conjunction of possibly non-deterministic APAs.

Definition 12: Let $N=(S, A, L, AP, V, s_0)$ and $N'=(S', A, L', AP, V', s'_0)$ be APAs sharing action and proposition sets. Their conjunction $N \owedge N'$ is the APA $(S\times S', A\cup A', \tilde{L}, AP\cup AP', \tilde{V}, (s_0, s'_0))$ where $\tilde{V}((s,s')) = V(s)\cap V'(s')$ and

$$\frac{a \in (\mathsf{Must}(s')\backslash\mathsf{May}(s)) \cup (\mathsf{Must}(s)\backslash\mathsf{May}(s'))}{\tilde{L}((s,s'),a,\text{false}) = \top}, \quad (8)$$

$$\frac{a \in (\mathsf{May}(s)\backslash\mathsf{May}(s')) \cup (\mathsf{May}(s')\backslash\mathsf{May}(s))}{\tilde{L}((s,s'),a,\tilde{\varphi}) = \bot}, \quad (9)$$

$$\frac{a\in\mathsf{May}(s)\cap\mathsf{May}(s') \qquad L(s,a,\varphi)\neq\bot \qquad L'(s',a,\varphi')\neq\bot}{\tilde{L}((s,s'),a,\tilde{\varphi}) = ?}, \quad (10)$$

where $\tilde{\varphi} \in C(S\times S')$ such that $\tilde{\mu} \in \mathit{Sat}(\tilde{\varphi})$ iff both distribution $\mu : t \to \sum_{t'\in S'} \tilde{\mu}((t,t'))$ is in $\mathit{Sat}(\varphi)$ and distribution $\mu' : t' \to \sum_{t\in S} \tilde{\mu}((t,t'))$ is in $\mathit{Sat}(\varphi')$.

$$\frac{a \in \mathsf{Must}(s) \qquad L(s,a,\varphi) = \top}{\tilde{L}((s,s'),a,\tilde{\varphi}^{\top}) = \top}, \quad (11)$$

where $\tilde{\varphi}^{\top} \in C(S\times S')$ such that $\tilde{\mu} \in \mathit{Sat}(\tilde{\varphi})$ iff both the distribution $\mu : t \to \sum_{t'\in S'} \tilde{\mu}((t,t'))$ is in $\mathit{Sat}(\varphi)$, and there exists $\varphi' \in C(S')$ with $L'(s',a,\varphi') \neq \bot$ and the distribution $\mu' : t' \to \sum_{t\in S} \tilde{\mu}((t,t'))$ is in $\mathit{Sat}(\varphi')$.

$$\frac{a \in \mathsf{Must}(s') \qquad L'(s',a',\varphi') = \top}{\tilde{L}((s,s'),a,\tilde{\varphi}'^{\top}) = \top}, \quad (12)$$

where $\tilde{\varphi}^{\top} \in C(S\times S')$ is such that $\tilde{\mu} \in \mathit{Sat}(\tilde{\varphi})$ iff both there exists $\varphi \in C(S)$ such that $L(s,a,\varphi) \neq \bot$ and the distribution $\mu : t \to \sum_{t'\in S'} \tilde{\mu}((t,t'))$ is in $Sat(\varphi)$, and the distribution $\mu' : t' \to \sum_{t\in S} \tilde{\mu}((t,t'))$ is in $Sat(\varphi')$.

The apparent complexity of the new definition concurs our experience from specification theories for discrete systems. For example, in [19] the notion of conjunction is presented for nondeterministic Modal Automata, resulting in a similar sophistication for resolving nondeterminism (modal automata do not contain the probabilistic part).

Example 5: Following the example of Fig. 2, we build the conjunction of APAs N and N' using Definition 12. The APA $N \owedge N'$ is given in Figure 3. Clearly, the PA I given in Figure 2 is an implementation of $N \owedge N'$.

We now give the main result of the section: as expected, the conjunction operator, given in Definition 12 matches the greatest lower bound of the weak weak refinement.

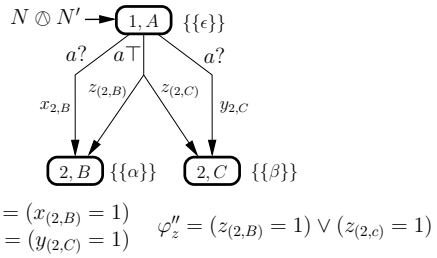


Fig. 3: APA $N \owedge N'$ obtained using Definition 12

Theorem 2: Let N_1, N_2, and N_3 be consistent APAs sharing action and atomic proposition sets. It holds that

- $\beta^*(N_1 \owedge N_2) \preceq_W N_1$ and $\beta^*(N_1 \owedge N_2) \preceq_W N_2$.
- If $N_3 \preceq_W N_1$ and $N_3 \preceq_W N_2$ then $N_3 \preceq_W \beta^*(N_1 \owedge N_2)$.

As expected, the new conjunction is weaker than the old one, thus it gives a more precise result:

Theorem 3: Let N_1 and N_2 be APAs. It holds that $N_1 \wedge N_2 \preceq N_1 \owedge N_2$.

Although the new notion of conjunction introduces some syntactic redundancy with the new must transitions, it agrees with the notion given in Definition 9 when considering action-deterministic APAs.

Theorem 4: Let N_1 and N_2 be action-deterministic APAs. We have $N_1 \owedge N_2 \preceq N_1 \wedge N_2$.

It follows from Theorems 3 and 4 that $[\![N_1 \wedge N_2]\!] = [\![N_1 \owedge N_2]\!]$ for any two action-deterministic APAs N_1 and N_2.

Finally, just like in the case of action-deterministic APAs, non-deterministic APAs with dissimilar alphabets can be handled by first equalizing their action and atomic proposition sets using weak extension.

V. Determinism

In the previous section we have seen that the use of non-determinism changes expressiveness of APAs with respect to the known conjunction operator. In fact, non-deterministic APAs are *generally* more expressive than deterministic ones. Fig. 4 presents a non-deterministic APA, whose set of implementations cannot be specified by a single deterministic APA. States 2 and 3 have overlapping labeling constraints (so state 1 has nondeterministic behaviour). We cannot put these states on two separate a-transitions as this introduces action nondeterminism. We cannot merge them either, as their subsequent evolutions are different (and for the same reason we cannot factor $\{\alpha, \gamma\}$ to a separate state).

Nevertheless use of deterministic abstractions of non-deterministic behaviours is an interesting alternative to relying on more complex refinements and more complex operators. Below, we present a determinization algorithm that can be applied to any APA N, producing a deterministic APA $\rho(N)$, such that: $N \preceq \rho(N)$.

Our algorithm is based on subset construction and it ressembles the determinization procedure for modal transition systems described in [20].

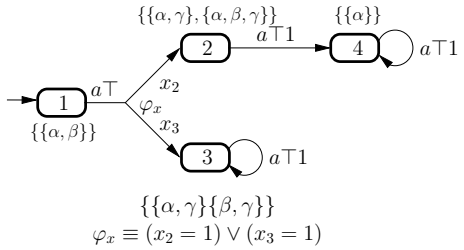


Fig. 4: A (labeling) nondeterministic APA whose set of implementations cannot be obtained with a deterministic APA.

Let $N = (S, A, L, AP, V, s_0)$ be a (consistent) APA in single valuation normal form (i.e. for all states s the set $V(s)$ is a singleton). Given a set of states $Q \subseteq S$, an action $a \in A$ and a valuation $\alpha \in AP$ we define 1-step reachability $\mathsf{Reach}(Q, a, \alpha)$ to be the maximal set of states with valuation α that can be reached with a non zero probability using a distribution π satisfying a constraint φ such that $L(q, a, \varphi) \neq \bot$ for some $q \in Q$. Formally, $\mathsf{Reach} : 2^S \times 2^A \times 2^{AP} \to 2^S$ and:

$$\mathsf{Reach}(Q, a, \alpha) = \bigcup \{s \in S \mid V(s) = \alpha \text{ and } \exists q \in Q.$$
$$\exists \varphi \in C(S).\, \exists \mu \in Sat(\varphi).\, L(q, a, \varphi) \neq \bot \text{ and } \mu(s) > 0\}$$

We lift this definition to all possible labelings as follows:

$$\mathsf{Reach}(Q, a) = \{\mathsf{Reach}(Q, a, \alpha) \mid \alpha \subseteq AP\}$$

Now define the n-step reachability as

$$\mathsf{Reach}^n(Q, a) = \mathsf{Reach}^{n-1}(Q, a) \cup \bigcup_{Q' \in \mathsf{Reach}^{n-1}(Q,a)} \mathsf{Reach}(Q', a)$$

where $\mathsf{Reach}^0(Q, a) = \{Q\}$ and denote its fixpoint as:

$$\mathsf{Reach}^*(Q, a) = \bigcup_{n=0}^{\infty} \mathsf{Reach}^n(Q, a).$$

Now, by construction, the following properties hold:

- For all $Q \subseteq S$ and $a \in A$, for all $Q', Q'' \in \mathsf{Reach}(Q, a)$, if $Q' \neq Q''$ then $Q' \cap Q'' = \emptyset$, and
- For all $Q \subseteq S$, $a \in A$ and $Q' \in \mathsf{Reach}^*(Q, a)$, there exists $\alpha \subseteq AP$ such that $\forall q' \in Q'$, we have $V(q') = \alpha$.

We will now use the notion of reachability in our determinisation construction. As already said, the algorithm works for APAs in the single valuation normal form. In [8] we show how every APA can be normalized without changing its semantics.

Definition 13: Let $N = (S, A, L, AP, V, s_0)$ be a consistent APA in single valuation normal form. A deterministic APA for N is the APA $\rho(N) = (S', A, L', AP, V', \{s_0\})$ such that

- $S' = \bigcup_{a \in A} \mathsf{Reach}^*(\{s_0\}, a)$
- V' is such that $V'(Q) = \alpha$ iff $\forall q' \in Q.\, V(Q) = \alpha$. There always exists exactly one such α by construction
- L' is defined as follows: Let $Q \in S'$ and $a \in A$.
  - If, for all $q \in Q$, we have that $\forall \varphi \in C(S)$, $L(q, a, \varphi) = \bot$, then define $L'(Q, a, \varphi') = \bot$ for all $\varphi' \in C(S')$.
  - Else, define $\varphi' \in C(S')$ such that $\mu' \in Sat(\varphi')$ iff (1) $\forall Q' \notin \mathsf{Reach}(Q, a)$, we have $\mu'(Q') = 0$, and (2) there exists $q \in Q$, $\varphi \in C(S)$ and $\mu \in Sat(\varphi)$ such that $L(q, a, \varphi) \neq \bot$ and $\forall Q' \in \mathsf{Reach}(Q, a)$, $\mu'(Q') = \sum_{q' \in Q'} \mu(q')$. Then define

$$L'(Q, a, \varphi') = \begin{cases} \top \text{ if } & \forall q \in Q, \exists \varphi \in C(S): \\ & L(q, a, \varphi) = \top \\ ? \text{ else} & \end{cases}$$

By construction, $\rho(N)$ is action- and labeling-deterministic. As expected, determinization is an abstraction. This is formalized in the following theorem.

Theorem 5: Let N be an APA in single valuation normal form. Then $N \preceq \rho(N)$.

VI. Composition and Games

So far APAs largely rely on the composition operation defined for modal transition systems. While, this operation mimics the classical composition between transition systems, it does not allow to reason about open systems, when some transitions are not in system's control. In a series of work [12], [21], de Alfaro and Henzinger proposed an approach based on game theory for doing so. More precisely, they introduced Interface Automata, or input/output automata [22] with a game semantic. When composing two such interfaces, the algorithm identifies bad states in where one of the components can send an output that cannot be catched by the other one. Then, it computes the set of states for which there is a possibility to avoid the set of bad states. Such strategies correspond to the environments in where the components can work together.

In [19], [23], we have proposed a game semantic to modal automata by labeling may and must with input and output. In this section, we extend this setup to APAs. This extension leads to the first theory for stochastic interface automata—an optimistic extension of stochastic I/O automata [24].

A. Abstract Probabilistic Interfaces

We begin by introducing profiles as presented in [19].

Definition 14: Given an alphabet of actions A, we define a profile as a function $\pi : A \to \{i, o\}$. We define $A^i = \{a \in A \mid \pi(a) = i\}$ and $A^o = \{a \in A \mid \pi(a) = o\}$, and write $\pi = (A^i, A^o)$.

Definition 15: Let $\pi_1 = (A_1^i, A_1^o)$ and $\pi_2 = (A_2^i, A_2^o)$ be profiles. We define the following operations:

- Refinement: We say that π_1 refines π_2, denoted $\pi_1 \preceq_p \pi_2$, if and only if $A_1 \supseteq A_2$ and $\pi_1(a) = \pi_2(a)$ for all $a \in A_2$
- Composition: If $A_1^o \cap A_2^o = \emptyset$, the composition of π_1 and π_2, denoted $\pi_1 \otimes \pi_2$, is defined as the profile $\pi_1 \otimes \pi_2 = (A^i, A^o)$ over $A_1 \cup A_2$, where $A^o = A_1^o \cup A_2^o$ and $A^i = (A_1^i \cup A_2^i) \setminus A^o$.

- Conjunction: If $\pi_1(a) = \pi_2(a)$ for all $a \in A_1 \cap A_2$, the conjunction of π_1 and π_2, denoted $\pi_1 \wedge \pi_2$, is defined as $A^o = A_1^o \cup A_2^o$ and $A^i = A_1^i \cup A_2^i$, where $A = A_1 \cup A_2$.

Lemma 2: Let $\pi_1 = (A_1^i, A_1^o)$ and $\pi_2 = (A_2^i, A_2^o)$ be profiles. If $\pi_1(a) = \pi_2(a)$ for all $a \in A_1 \cap A_2$, then

1) $\pi_1 \wedge \pi_2 \preceq_p \pi_1$ and $\pi_1 \wedge \pi_2 \preceq_p \pi_2$, and
2) whenever $\pi \preceq_p \pi_1$ and $\pi \preceq_p \pi_2$ then $\pi \preceq_p \pi_1 \wedge \pi_2$.

We are now ready to define Abstract Probabilistic Interfaces, that are APAs whose transitions are labeled by profiles.

Definition 16: Given an APA N with action set A and a profile $\pi : A \to \{i, o\}$, we call $\mathcal{N} = (N, \pi)$ an abstract probabilistic interface.

Given an APA N, we use A_N and AP_N to denote the action and atomic proposition set of N, respectively.

Let I be a PA. Given a profile $\pi_I : A_I \to \{i, o\}$ and an API (N, π), if $A_I \supseteq A_N$ and $AP_I \supseteq AP_N$, we say that $\mathcal{I} = (I, \pi_I)$ satisfies (N, π), denoted $\mathcal{I} \models (N, \pi)$, if and only if $I \models N \Uparrow (A_I, AP_I)$ and $\pi_I \preceq_p \pi$. We also say that $\mathcal{I}$ is called an implementation of (N, π).

Likewise, let (N', π') be an API. If $A_N \supseteq A_{N'}$ and $AP_N \supseteq AP_{N'}$, we say that (N, π) refines an (N', π'), denoted $(N, \pi) \preceq (N', \pi')$ iff $N \preceq N' \Uparrow (A_N, AP_N)$ and $\pi \preceq_p \pi'$.

Following the presentation in [23] it is possible to express an arbitrary inteface automaton as an abstract probabilistic automaton. We refer to [18] for the translation.

B. Parallel Composition of Abstract Probabilistic Interfaces

We now define an optimistic composition for APIs. We start with the definition of product of two APIs.

Definition 17: Given two APIs $\mathcal{N}_1 = (N_1, \pi_1)$ and $\mathcal{N}_2 = (N_2, \pi_2)$ with $AP_1 \cap AP_2 = \emptyset$, we define the product of $\mathcal{N}_1 = (N_1, \pi_1)$ and $\mathcal{N}_2 = (N_2, \pi_2)$ as $\mathcal{N}_1 \otimes \mathcal{N}_2 = (N_1 \| _{A_1 \cap A_2} N_2, \pi_1 \otimes \pi_2)$.

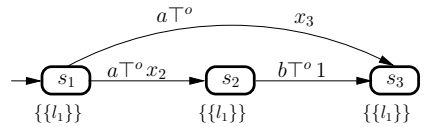
For two APIs N_1 and N_2 define the set of bad states $\mathsf{bad}_{\mathcal{N}_1 \otimes \mathcal{N}_2}$ as the set of pairs $(s_1, s_2) \in S_1 \times S_2$ satisfying one of the two following conditions:

1) There exists $a \in A_1^o \cap A_2^i$ and $\varphi_1 \in C(S_1)$ such that $L_1(s_1, a, \varphi_1) \geq ?$ and for all $\varphi_2 \in C(S_2)$, $L_2(s_2, a, \varphi_2) \neq \top$, or
2) There exists $a \in A_2^o \cap A_1^i$ and $\varphi_2 \in C(S_2)$ such that $L_2(s_2, a, \varphi_2) \geq ?$ and for all $\varphi_1 \in C(S_1)$, $L_1(s_1, a, \varphi_1) \neq \top$.

Basically, a state of the product is a bad if one of the operands can send an action that the other operand may avoid to catch.

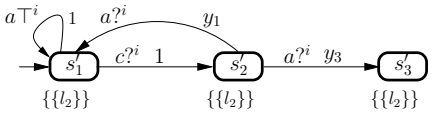
Example 6: Consider the APIs $\mathcal{N}_1$ and $\mathcal{N}_2$ given in Fig. 5 and Fig. 6. Their profiles are specified by attaching o and i letters to transition labels. The API $\mathcal{N}_1 \otimes \mathcal{N}_2$ is given in Fig. 7. Observe that $(s_1, s_2') \in \mathsf{bad}_{\mathcal{N}_1 \otimes \mathcal{N}_2}$. Indeed, the action a is an output action of $\mathcal{N}_1$ and an input action of $\mathcal{N}_2$. However, while there is a may-transition from s_1 on a, there is no must-transition on a from s_2'.

We now propose an algorithm that computes the set of states from where there is a way to reach the set of bad states. Given



$$\varphi \equiv (x_1 = 0 \wedge x_2 \leq 0.5 \wedge x_3 \geq 0.8 \wedge x_2 + x_3 = 1)$$

Fig. 5: $\mathcal{N}_1$



$$\varphi \equiv (y_2 = 0 \wedge y_1 \geq 0.7 \wedge y_3 \leq 0.4 \wedge y_1 + y_3 = 1)$$

Fig. 6: $\mathcal{N}_2$

an API $\mathcal{N} = (N, \pi)$ on state set S and action set A, the function pre of a set $S' \subseteq S$, $\mathsf{pre}^1(S')$ is defined as

$$\mathsf{pre}^1(S') = \{s \in S \mid \exists a \in A^o. \exists \varphi \in C(S). \exists \mu \in \mathit{Sat}(\varphi). L(s, a, \varphi) \geq ? \wedge \mu(S') > 0\}$$

We say that $\mathsf{pre}^0(S') = S'$, for $k \geq 0$, $\mathsf{pre}^{k+1}(S') = \mathsf{pre}^1(\mathsf{pre}^k(S'))$, and $\mathsf{pre}(S') = \bigcup_{k \geq 0} \mathsf{pre}^k(S')$.

Definition 18: We say that APIs $\mathcal{N}_1 = (N_1, \pi_1)$ and $\mathcal{N}_2 = (N_2, \pi_2)$ are compatible, if $(s_0^1, s_0^2) \notin \mathsf{pre}(\mathsf{bad}_{\mathcal{N}_1 \otimes \mathcal{N}_2})$.

After computing $\mathcal{N}_1 \otimes \mathcal{N}_2$ and $\mathsf{pre}(\mathsf{bad}_{\mathcal{N}_1 \otimes \mathcal{N}_2})$, the product is *relaxed*. For each state $s \in S$ and each action $a \in A$ perform the following: if it is possible to reach one of the states in $\mathsf{pre}(\mathsf{bad}_{\mathcal{N}_1 \otimes \mathcal{N}_2})$ with non-zero probability after issuing a then redefine s to have only a may-transition on a, with the constraint containing only a distribution giving probability 1 to a fresh state s_{may} not in S that allows everything but does not require anything.

Definition 19: Given two compatible APIs $\mathcal{N}_1 = (N_1, \pi_1)$ and $\mathcal{N}_2 = (N_2, \pi_2)$, we define the composition of $\mathcal{N}_1$ and $\mathcal{N}_2$, denoted as $\mathcal{N}_1 \| \mathcal{N}_2$, as the API obtained by substituting L and V in $\mathcal{N}_1 \otimes \mathcal{N}_2$ by L', a copy of L that is manipulated in the following way, and V', an extension of V: For all $(s_1, s_2) \in S_1 \times S_2$ and for all $a \in A$, if $(s_1, s_2) \notin \mathsf{pre}(\mathsf{bad}_{\mathcal{N}_1 \otimes \mathcal{N}_2})$ and there exists $\varphi \in C(S)$ and $\mu \in \mathit{Sat}(\varphi)$ such that

$$L((s_1, s_2), a, \varphi) \geq ? \wedge \mu(\mathsf{pre}(\mathsf{bad}_{\mathcal{N}_1 \otimes \mathcal{N}_2})) > 0$$

then let $L'((s_1, s_2), a, \varphi) = \bot$ and $L'((s_1, s_2), a, \varphi') = L((s_1, s_2), a, \varphi)$, where $\mathit{Sat}(\varphi') = \{\mu'\}$ and $\mu'(s_{\text{may}}) = 1$.

The new state s_{may}, not in $S_1 \times S_2$, is defined as, for all $a \in A$, $L'(s_{\text{may}}, a, \varphi'') = ?$, where $\varphi'' \in C(S)$ is such that $\mathit{Sat}(\varphi'') = \{\mu''\}$ and $\mu''(s_{\text{may}}) = 1$. The function V' is defined as $V'(s_1, s_2) = V(s_1, s_2)$ for all $(s_1, s_2) \in S_1 \times S_2$ and $V'(s_{\text{may}}) = 2^{AP}$.

Example 7: Returning to Example 6, the parallel composition of $\mathcal{N}_1$ and $\mathcal{N}_2$ is obtained as the API in Fig. 8. The profile of the composition is $\pi_1 \otimes \pi_2 = [a \mapsto o, b \mapsto o, c \mapsto i]$.

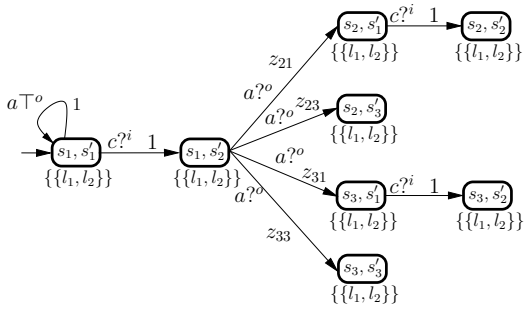


Fig. 7: $\mathcal{N}_1 \otimes \mathcal{N}_2$

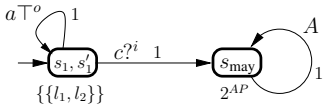


Fig. 8: $\mathcal{N}_1 \| \mathcal{N}_2$

Since (s_1, s_2') is a bad state it becomes unreachable (and thus most of other state pairs become unreachable). Instead a transition to the universal state s_{may} is inserted, modeling the fact that after receiving c the system becomes unpredictable.

Our composition for APIs satisfies classical theorems of independent implementability.

Lemma 3: Given two compatible APIs $\mathcal{N}_1$ and $\mathcal{N}_2$, it holds that $\mathcal{N}_1 \otimes \mathcal{N}_2 \preceq \mathcal{N}_1 \| \mathcal{N}_2$.

Theorem 6: Given two compatible APIs $\mathcal{N}_1$ and $\mathcal{N}_2$, and two implementations $\mathcal{I}_1$ and $\mathcal{I}_2$, such that $\mathcal{I}_1 \models \mathcal{N}_1$ and $\mathcal{I}_2 \models \mathcal{N}_2$, then $\mathcal{I}_1 \otimes \mathcal{I}_2 \models \mathcal{N}_1 \| \mathcal{N}_2$.

VII. Implementation

Some of the operations introduced in this paper have been implemented in a new tool, written in C# 4.0, called APAC.[1] To the best of our knowledge, this is the first implementation effort for stochastic interfaces. Presently the tool relies on the Z3 solver [15] for solving constraints. The tool implements the following operations: weak refinement checking, weak weak refinement checking, determinism checking, pruning (β^*), alphabet extensions, and conjunction.

Example 8: We present the input format of the tool in an example. We will be checking weak refinement between two APAs. The following code example will result in the definition of two models N_1 and N_2. The last line specifies that weak refinement will be checked. Notice, that we require states to be named with the natural numbers where 1 is the initial state.

```
Name: N1;
A:(a,b);
AP:(l,m,n,o);
state 1:((l)): a? -> x[1] = 0.0 && x[2] + x[3] >=
    7/10 && x[3] + x[4] >= 2/10;
state 2:((m)): b? -> x[1] = 0.0 && x[2] = 0.0 && (x
    [3] = 1.0 || x[4] = 1.0);
state 3:((n)): b? -> x[3] = 1.0;
state 4:((o)): b? -> x[4] = 1.0;

Name: N2;
A:(a,b);
AP:(l,m,n,o);
state 1:((l)): a? -> x[1] = 0.0 && x[2]+x[3] >= 7/10
     && x[4] + x[5] >= 2/10;
state 2:((m)): b? -> x[3] <= 1.0 && x[4] <= 1.0 && x
    [5] <= 1.0 && x[1] = 0.0 && x[2] = 0.0;
state 3:((n)): b? -> x[4] = 1.0;
state 4:((n)): b? -> x[3] = 1.0;
state 5:((o)): b? -> x[5] = 1.0;

check: N1 wref N2;
```

[1]The tool can be found on www.cs.aau.dk/~mikkelp/apac

It takes 179 milliseconds on a typical laptop before APAC reports that $\{(1,1),(2,2),(3,3),(3,4),(4,5)\}$ is a weak refinement relation.

At the moment APAC does not support parallel composition. This is because its definition requires use of multiplication, which is not supported by Z3. The situation could have been the same for refinement, but we have been able to use a different encoding. The idea is to let the correspondence functions give the actual value redistributed, and not the proportions. Still, this trick cannot be used for strong refinement, as defined in [8]. At the moment we do not know, whether strong refinement can be checked relying solely on solving linear arithmetic constraints.

We discuss the weak refinement in somewhat more details. The algorithm is implemented as a coinductive fixpoint iteration. Starting from the full relation, violating pairs are removed until a fixpoint is reached. Given a pair of states $(s, s') \in \mathcal{R}$ and constraints φ and φ', the pseudo-formula, Eq. (13), is passed to Z3. We invoke quantifier elimination, and since all variables are quantified, quantifier elimination will evaluate the formula to true or false.

$$
\begin{aligned}
\forall x : \varphi(x) \Rightarrow \exists \delta : S \to (S' \to [0,1]) : \quad & (13)\\
& \varphi'\left(t \mapsto \sum_{s \in S} \delta(s)(t)\right) \wedge \\
& \forall s \in S : x_s = \sum_{s' \in S'} \delta(s)(s') \wedge \\
& \forall (s,s') : [(s,s') \notin \mathcal{R} \vee V(s) \not\subseteq V'(s')] \\
& \qquad \Rightarrow \delta(s)(s') = 0 \wedge \\
& \forall (s,s') : [(s,s') \in \mathcal{R} \wedge V(s) \subseteq V'(s')] \\
& \qquad \Rightarrow 0 \leq \delta(s)(s') \leq 1
\end{aligned}
$$

Notice that, if a pair of states has conflicting labeling, we set the correspondence function to 0 for this pair. The tool can also synthesize a witness in case the refinement does not hold.

In order to evaluate the performance of the tool, we generate "random" APAs and measure the time for performing operations on these. Given a number of states and whether or not we are interested in simple or more elaborate constraints, we generate an APA with an action alphabet A and an atomic proposition alphabet *AP* on 5–10 members each, state valuations consisting of up to 0–4 members of 2^{AP}, 1–3 outgoing transitions for each state on random action and modality, and a

APA 1		APA 2		
states	simple	states	simple	time
10	yes	10	yes	10/20/72 ms
10	no	10	no	10/41/1121 ms
10	no	10	yes	20/1046/? ms
10	yes	10	no	18/94/4079 ms
15	yes	15	yes	125/140/? ms

TABLE I: Weak refinement

random choice between constraint designs for each transition. Given a state i there are three simple constraints and three more eleborate constraints:

- simple:
  - $x_{i+1} \geq 7/10 \wedge x_{i+2} \leq 3/10$,
  - $x_{i+1} = 7/10 \wedge x_{i+2} = 3/10$, and
  - $x_{i+1} = 1.0$
- more elaborate:
  - $x_{i+1} \geq 3/10 \wedge x_{i+1} \leq 4/10$,
  - true, and
  - $x_{i+1} = 1.0 \vee (x_{i+1} \geq 7/10 \wedge x_{i+2} \leq 3/10)$

The tests, that are summarized in Table I, are performed on an x64 Intel Core 2 Duo 2.2 GHz with 4 GB RAM running Windows 7, using version 2.16 of the Z3 API. The first line of the table gives execution times for three random input files (three times are reported, as the experiment was repeated three times, with different randomly generated instances). In each input file, weak refinement is checked on two random APAs on each 10 states with simple constraints.

This procedure is repeated for each line in the table. A question mark (?) means that the specific random input file does not stop executing within 5 minutes.

The above results are still preliminary and we hope to reduce the computation time by adapting classical heuristics for fixed-point computations [25].

VIII. Conclusion

In [8], we have introduced the first complete specification theory for PAs with a comparison operator and both logical and structural composition. In this paper, we have strengthened those results by extending the power of the operators as well as the expressiveness of the model. The results have been implemented in APAC, a prototype tool that has been evaluated on several case studies.

There are many directions for future research. First, one should pursue the development of APAC by adding the composition operators. Heuristics should also be implemented. Among them, one naturally thinks of the work by Henzinger et al.[25] that could be adapted to reduce the number of steps in the fixed-point algorithm for refinement. Another suggestion would be to adapt bisimulation quotient [26] in order to minimize the size of the APAs.

Another direction is to develop a generalized model checking procedure for APAs. We postulate that this could be done by extending results obtained for Hennessy-Milner logic and modal automata [10]. Finally, we are also considering to mix the results on APAs with those we obtained on timed interfaces. This would lead to the first specification theory for timed systems [27] with a stochastic semantics.

References

[1] R. Segala and N. A. Lynch, "Probabilistic simulations for probabilistic processes," *NJC*, vol. 2, no. 2, pp. 250–273, 1995.

[2] R. Segala, "Probability and nondeterminism in operational models of concurrency," in *CONCUR*, ser. LNCS, vol. 4173. Springer, 2006.

[3] A. Parma and R. Segala, "Axiomatization of trace semantics for stochastic nondeterministic processes," in *QEST*. IEEE, 2004, pp. 294–303.

[4] D. N. Jansen, H. Hermanns, and J.-P. Katoen, "A probabilistic extension of UML statecharts," in *FTRTFT*, ser. LNCS, vol. 2469. Springer, 2002.

[5] L. Cheung, N. A. Lynch, R. Segala, and F. W. Vaandrager, "Switched PIOA: Parallel composition via distributed scheduling," *TCS*, vol. 365, no. 1-2, pp. 83–108, 2006.

[6] R. Canetti, L. Cheung, D. K. Kaynar, M. Liskov, N. A. Lynch, O. Pereira, and R. Segala, "Analyzing security protocols using time-bounded task-PIOAs," *Discrete Event Dynamic Systems*, vol. 18, no. 1, 2008.

[7] L. Cheung, M. Stoelinga, and F. W. Vaandrager, "A testing scenario for probabilistic processes," *J. ACM*, vol. 54, no. 6, 2007.

[8] B. Delahaye, J.-P. Katoen, K. Larsen, A. Legay, M. Pedersen, F. Sher, and A. Wasowski, "Abstract probabilistic automata," in *VMCAI*, 2011.

[9] B. Caillaud, B. Delahaye, K. G. Larsen, A. Legay, M. L. Pedersen, and A. Wąsowski, "Compositional design methodology with constraint Markov chains," in *QEST*. IEEE, 2010.

[10] K. G. Larsen, "Modal specifications," in *AVMFSS*. Springer, 1989, pp. 232–246.

[11] J.-B. Raclet, "Residual for component specifications," in *FACS*, ser. Electronic Notes Theoretical Computer Science, vol. 215, 2008.

[12] L. de Alfaro and T. A. Henzinger, "Interface-based design," in *Engineering Theories of Software-intensive Systems*, ser. NATO Science Series: Mathematics, Physics, and Chemistry, vol. 195. Springer, 2005.

[13] J.-B. Raclet, E. Badouel, A. Benveniste, B. Caillaud, and R. Passerone, "Acsd," in *ACSD*. IEEE, 2009, pp. 119–127.

[14] B. Delahaye, K. G. Larsen, A. Legay, and A. Wąsowski, "On greatest lower bound of modal transition systems," Tech. Rep., 2010. [Online]. Available: http://www.irisa.fr/s4/people/benoit.delahaye/rapports/modalIPL.pdf

[15] L. De Moura and N. Bjørner, "Z3: an efficient smt solver," in *TACAS*. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 337–340. [Online]. Available: http://portal.acm.org/citation.cfm?id=1792734.1792766

[16] B. Jonsson and K. G. Larsen, "Specification and refinement of probabilistic processes," in *LICS*. IEEE, 1991, pp. 266–277.

[17] K. G. Larsen and B. Thomsen, "A modal process logic," in *LICS*. IEEE, 1988, pp. 203–210.

[18] B. Delahaye, J.-P. Katoen, K. G. Larsen, A. Legay, M. L. Pedersen, F. Sher, and A. Wąsowski, "New Results on Abstract Probabilistic Automata," http://www.irisa.fr/s4/people/benoit.delahaye/rapports/ACSD11-long.pdf, 2011.

[19] J.-B. Raclet, E. Badouel, A. Benveniste, B. Caillaud, A. Legay, and R. Passerone, "Modal interfaces: unifying interface automata and modal specifications," in *EMSOFT*. ACM, 2009, pp. 87–96.

[20] N. Benes, J. Kretínský, K. G. Larsen, and J. Srba, "On determinism in modal transition systems," *TCS*, vol. 410, no. 41, pp. 4026–4043, 2009.

[21] L. de Alfaro and T. A. Henzinger, "Interface automata," in *FSE*. ACM Press, 2001, pp. 109–120.

[22] N. Lynch and M. R. Tuttle, "An introduction to Input/Output automata," *CWI-quarterly*, vol. 2, no. 3, 1989.

[23] K. Larsen, U. Nyman, and A. Wąsowski, "Modal I/O automata for interface and product line theories," in *PLS*. Springer, 2007.

[24] N. Lynch, I. Saias, and R. Segala, "Proving time bounds for randomized distributed algorithms," in *PODC*. ACM Press, 1994, pp. 314–323.

[25] M. R. Henzinger, T. A. Henzinger, and P. W. Kopke, "Computing simulations on finite and infinite graphs," in *FOCS*, 1995, pp. 453–462.

[26] P. Crouzen and H. Hermanns, "Aggregation ordering for massively compositional models," in *ACSD*. IEEE Computer Society, 2010.

[27] A. David, K. G. Larsen, A. Legay, U. Nyman, and A. Wąsowski, "Timed I/O automata: a complete specification theory for real-time systems," in *HSCC*. ACM ACM, 2010, pp. 91–100.