RWTH Aachen

Department of Computer Science
Technical Report

Abstraction and Refinement of Probabilistic Automata using Modal Stochastic Games

Falak Sher

ISSN 0935–3232 · Aachener Informatik-Berichte · AIB-2015-10

RWTH Aachen · Department of Computer Science · April 2015

Abstraction and Refinement of Probabilistic Automata using Modal Stochastic Games

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der RWTH Aachen University zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften genehmigte Dissertation

vorgelegt von

Falak Sher, M.Sc.
aus
Baqarpur, Daska, Sialkot, Pakistan

Berichter: Prof. Dr. Ir. Joost-Pieter Katoen
Prof. Dr. Lijun Zhang

Tag der mündlichen Prüfung: 24. April 2015

Falak Sher
Lehrstuhl Informatik 2
`chfalak@gmail.com`

Aachener Informatik Bericht AIB-2015-10

Herausgeber: Fachgruppe Informatik
RWTH Aachen University
Ahornstr. 55
52074 Aachen
GERMANY

ISSN 0935-3232

Abstract

Formal methods are mathematical techniques used in the development of trustworthy ICT systems. Their application ensures correct specifications and error-free implementations of systems providing guarantees on their *functional* requirements and *probabilistic* behaviours, e.g., performance, reliability, etc. *Model checking* is one of such techniques that systematically and exhaustively explores all possible configurations of systems' models to verify certain properties. One of the challenges faced by this approach is to design efficient algorithms for exploring all configurations of systems' model. The number of states for realistic systems is usually extremely high and sometimes even infinitely-many — known as the *state space explosion* problem. This problem restricts the applicability of model checking algorithms.

Abstraction is a reduction technique that removes information from system models irrelevant to the property of interest; and consequently induces additional behaviour. The abstract models over- and/or under-approximate the behaviour of concrete models. For example, *existential abstract* models like Segala's probabilistic automata (PA) over-approximate while *modal abstract* models over- and under-approximate the behaviour of concrete models. The *game-based abstract* models, such as Condon's stochastic games (SGs), are different in a sense that they keep the behaviour from the abstraction process (handled by, say, player-two) separate from the concrete behaviour (handled by, say, player-one). Both modal and game-based abstract models allow for bounding the *reachability probabilities* in concrete models from below and from above. In fact, abstraction-refinement algorithms initially construct coarser abstract models and then gradually refine them until the bounds on the reachability probabilities of concrete models are sufficiently tight.

We orthogonally combine the techniques of *modal* and *game-based* abstractions. This yields *modal* stochastic games in which player-two completely handles behaviour induced by abstraction whereas player-one handles behaviour induced by abstraction and from concrete models. Due to this additional non-deterministic behaviour in player-one states, bounds on reachability probabilities in modal stochastic games are at most as tight as in stochastic games, but modal games are comparatively smaller in size. Moreover, our *modal game-based* abstraction is *compositional* in a sense that individual components can be abstracted separately and then plugged together constituting an overall model of a system.

Existing abstraction techniques in the literature are *state-based*. That is to say, abstract models derive their transitions from that of the concrete states and, thus, simulate concrete models in a step-wise manner. Exploiting the fact that probabilistic systems are not just stochastic processes but transformers of probabilities as well, we treat distributions over states (rather than states) as first-class citizens and lift the notion of abstraction from states to *distributions* over states. We also define (*alternating*) *simulation* relations between concrete and abstract models. We show that game-based abstraction is not the optimal abstraction preserving extremal reachability probabilities. Furthermore, we illustrate that our distribution-based abstraction may induce more precise and concise models than state-based abstraction.

Finally, we propose a state-based and a distribution-based *abstraction-refinement framework* for PA. It refines a modal stochastic game in two nested loops. The inner-loop iteratively refines player-one states until the effect of non-deterministic behaviour induced in them by the abstraction process has no impact anymore on the reachability probabilities of player-two states. The outer-loop refines player-two states until their reachability probabilities bound that of their corresponding concrete states within a certain given range. This yields the smallest possible modal abstraction for a given state-space partitioning, that bounds the reachability probabilities of concrete model within a given range.

Zusammenfassung

Formale Methoden sind mathematische Techniken, die in der Entwicklung zuverlässiger Systeme angewandt werden. Ihre Anwendung garantiert korrekte Spezifikationen und fehlerfreie Implementierungen von Systemen dadurch, dass Garantien für *funktionale* Anforderungen und *probabilistisches* Verhalten gegeben werden, zum Beispiel bezüglich Performanz, Zuverlässigkeit usw. Model Checking is eine solche Technik die systematisch und vollständig alle möglichen Systemkonfigurationen eine Systemmodels untersucht um bestimmte Eigenschaften zu verifizieren. Eine der Herausforderungen für diesen Ansatz ist es, effiziente Algorithmen zu entwerfen um alle möglichen Systemkonfigurationen zu untersuchen. Die Anzahl an Zuständen für realistische Systeme ist im Allgemeinen extrem hoch und manchmal sogar unendlich — dies ist bekannt als die *Explosion des Zustandsraumes*. Dieses Problem schränkt die Anwendbarkeit von Model Checking ein.

Abstraktion ist eine Reduzierungstechnik welche Informationen aus dem Systemmodell entfernt, welche irrelevant für die untersuchte Eigenschaft sind; konsequenterweise wird zuätzliches Verhalten hinzugefügt. Das abstrakte Modell über- und/oder unterapproximiert das Verhalten konkreter Modelle. Zum Beispiel, *existentielle abstrakte* Modelle wie Segalas probabilistische Automaten (PA) überapproximieren das Verhalten konkreter Modelle während *modale abstrakt* Modelle es über- und unterapproximieren. *Spielbasierte* Modelle, wie zum Beispiel Condon's stochastische Spiele (SGs), unterscheiden sich dadurch dass sie das Verhalten abstrakter Prozesse (kontrolliert durch Spieler 2) von dem konkreten Verhalten (kontrolliert durch Spieler 1) getrennt wird. Sowohl modale und spielbasierte Abstraktionsmodelle ermöglichen es, *Erreichbarkeitswahrscheinlichkeiten* in konkreten Modellen von unten und oben zu beschränken. Tatsächlich konstruieren Abstraktionsverfeinerungsalgorithmen initial gröbere abstrakte Modelle und verfeinern diese dann schrittweise bis die Grenzen für Erreichbarkeitswahrscheinlichkeiten ausreichend genau sind.

Wir kombinieren die Techniken *modaler* und *spielbasierter* Abstraktion in orthogonaler Art und Weise. Dies ergibt *modale* stochastische Spiele wo Spieler 2 vollständig das Verhalten kontrolliert, welches durch Abstraktion entsteht, während Spieler 1 Verhalten kontrolliert, welches durch Abstraktion und durch das konkrete Modell entsteht. Durch dieses zusätzliche nichtdeterministische Verhalten in Spieler 1-Zuständen sind Grenzen für Errreichbarkeitswahrscheinlihkeiten in modalen stochastischen Spielen mindestens so genau wie in stochastischen Spielen, während modale Spiele vergleichsweise kleiner sind. Zudem ist unsere *modale spielbasierte* Abstraction kompositionell in dem Sinne, dass einzelne Komponenten auch einzeln abstrahiert und dann zusammengefügt werden können, was ein Gesamtmodell des Systems ergibt.

Vorhandene Abstraktionstechniken aus der Literatur sind *zustandsbasiert*. Das heißt, abstrakte Modelle leiten ihre Transitionen von denen konkreter Zustände ab und simulieren daher schrittweise konkrete Modelle. Wir nutzen aus, dass probabilistische Systeme nicht nur stochastische Prozesse sind sondern auch Wahrscheinlichkeitstransformer und behandeln Verteilungen über Zuständen (anstatt Zuständen) als Bürger erster Klasse und heben das Konzept der Abstraktion von Zuständen auf *Verteilungen* über Zuständen. Außerdem definieren wir (*alternierende*) *Simulationsrelationen* zwischen konkreten und abstrakten Modellen. Wir zeigen, dass spielbasierte Abstraktion nicht die optimale Strategie ist um extreme Erreichbarkeitswahrscheinlichkeiten zu erhalten. Darüber hinaus zeigen wir, dass unsere verteilungsbasierte Abstraktion präzisere und prägnantere Modelle erzeugen kann als zustandsbasierte Abstraktion.

Schlußendlich stellen wir ein zustandsbasiertes und ein verteilungsbasiertes *Abstraktionsverfeinerungs-Rahmenwerk* für PAs vor. Es verfeinert ein modales stochastisches Spiel in zwei geschachtelten Schleifen. Die innere Schleife verfeinert schrittweise Spieler 1-Zustände solange bis nichtdeterministisches Verhalten induziert durch den Abstraktionsprozess keinen Einfluss mehr auf Erreichbarkeitswahrscheinlichkeiten von Spieler 2-Zuständen hat. Die äußere Schleife verfeinert Spieler 2-Zustände bis ihre Erreichbarkeitswahrscheinlichkeit die von zugehörigen Zuständen innerhalb eines bestimmten Bereiches beschränkt. Dies ergibt die kleinste mögliche modale Abstraktion für eine gegebene Zustandspartitionierung, welche Erreichbarkeitswahrscheinlichkeiten innerhalb eines bestimmten Bereiches beschränkt.

Acknowledgements

Man is so helpless in this world that he cannot take even a single breath but with the consent of Allah. All praises goes to Him Who gave me health, resources and nice people around that made my life comfortable and enjoyable. Regarding my studies at RWTH Aachen University, I am thankful to my supervisor, Joost-Pieter Katoen, whose guidance and support helped me completing my thesis successfully. Moreover, I am thankful to all my colleagues for their help in studies and in other activities like reading letters for me, filling out forms in German language :), etc. Particularly, Souymodip Chakraborty and Christoph Matheja helped me a lot in preparing for my final Ph.D. exam.

Last but not the least, I would like to thank my family members for their unconditional love and support in my life; my parents, my brother and sister, my wife and my little princes, Shanaya Falak Sher, all helped me in one way or the other. Particularly, the sacrifices of my father and mother are uncountable and I will never be able to pay them back. Without their encouragement, patience and commitment, I would not have achieved this success.

Contents

1 Introduction

In today's world, almost every aspect of our society, e.g. energy requirements, medical treatments, communication, etc., is directly or indirectly dependent on information and communication technology (ICT) systems. With the advancement in science and technology, these systems not only get bigger and bigger but their interaction with each other also increases. This trend poses multidimensional challenges to the developers of these systems, for example, how to correctly specify the requirements of systems, how to design systems from their requirements, how to verify that the resultant systems comply with their requirements, etc. This is because a flaw in the development of a system may result in catastrophic events with severe financial penalties, loss of lives and lasting damages to the environment.

Conventionally, in industry the flaws at each stage of a development process are figured out by conducting a large number of tests. As it is not possible to test every scenario, assuring development of error-free and reliable systems through this process alone is impossible. This, therefore, requires the development of systems in some formal ways in order to have reliable and trust-worthy ICT systems; thus, necessitating the use of *Formal Methods* in the development of systems.

Formal methods are mathematical techniques used in the production of trustworthy ICT systems, e.g., *static analysis*, *abstract interpretation* [CC92, CC77], *model checking* [BK08], etc. Their application during the development cycle of systems ensures their correct specifications and error-free implementations; and provides guarantees on their *qualitative* and *quantitative* aspects.

By qualitative aspects, we mean the *functional behaviours* of systems. For example, "the gate of a railway crossing is closed when a train passes by", "in case of fire, the fire alarm starts", etc; and by quantitative aspects, we mean *probabilistic behaviours* of systems. For example, "in case of fire, the fire alarm starts with probability at least 0.95", "a message is delivered with probability at least 0.9", etc. Therefore, the guarantees on qualitative aspects of system models help identifying whether the functional requirements are correctly implemented, whereas guarantees on quantitative aspects help so for non-functional requirements such as performance, reliability, robustness, etc., — which are only relevant for systems with probabilistic behaviour.

Of the many formal methods techniques used for the verification of systems, this thesis mainly improves upon model checking — a framework for the verification of systems models for certain aspects (properties) — of probabilistic systems. To be precise, we propose different techniques to reduce the size of huge (even infinite) systems models (with probabilistic behaviour) such that they can be verified; thus pushing the existing boundaries of model checking techniques. Moreover, among the different techniques used for reduction, we deal with *abstraction* for discrete-time systems, i.e., systems evolving at discrete time points.

1.1 Model Checking

Model checking approach has been used to verify qualitative as well as quantitative aspects of systems. The basic concept behind this approach is to systematically and exhaustively explore all possible configurations of systems' models and check whether certain requirements are fulfilled.

Of the challenges faced by model checking approach, three are the important ones: getting a mathematical model of systems, formally specifying requirements (properties) to be verified, and designing efficient algorithms that thoroughly explore all possible configurations of systems' model.

In the literature, different ways to build mathematical models of systems have been suggested, e.g., models are developed in high-level modeling formalism like (probabilistic) state-charts [Har87, JHK02], Petri nets [Pet62, Kud05, LMZL11], guarded-command languages [Dij75, HSM97], etc; or extracted from a program code [CKSY05], etc. The properties of systems are given as formulas of some (probabilistic) temporal logic — (probabilistic) linear temporal logic, (probabilistic) computation tree logic, etc), as (probabilistic) automata or as statements of high-level specification languages. Similarly, different model checking algorithms have been proposed in the literature.

To explore all configurations of systems models (specified in high-level formalisms), the model checking algorithms need to convert them into underlying low-level models, i.e. labelled transition systems (LTS), discrete-time Markov chains (DTMC) [BK08], etc. These underlying models are usually manifolds (or may be infinitely) larger than their high-level descriptions — in case of probabilistic systems, as each configuration keeps probabilities of taking transitions to next configurations, the models are relatively bigger. This is known as the *state space explosion* problem, that restricts the applicability of model checking approach to systems not bigger than a certain size. For relatively bigger systems, the model checking algorithms say nothing about the satisfaction or refutation of properties, thus generating *out-of-memory* error instead of *yes* or *no*. However, with certain reduction techniques such as bisimulation minimization [KKZJ07], partial order reduction [GB06], abstraction [CGL94, DJL01] etc., systems with billions of states are verifiable [CCG$^+$02, KNP09, KZH$^+$11].

1.2 Abstraction

Informally speaking, abstraction is a generalization that allows omitting details from the models of systems that are not relevant for the verification of the properties under consideration. The models induced as a result are finite, small in size and have less information as compared to the concrete models (that may be infinitely large). Usually, model checking frameworks construct coarser abstractions in the beginning, and then gradually refine them until the abstract models have enough details for the verification of properties. In the literature, different abstraction techniques have been proposed, we discuss a few of them that are relevant to this thesis.

Existential abstraction: The abstract models obtained by this technique over-approximate the behaviour of concrete models [CGL94, Seg95], i.e., for every (probabilistic) execution [Seg95] in the concrete model, there is a (probabilistic) execution in the abstract model. This, therefore, allows for the verification of those properties on concrete models that can be refuted by single (probabilistic) executions, i.e., *safety properties* in non-probabilistic systems, whereas one-sided bounds on *quantitative properties* — *upper/lower* bound of *maximum/minimum* reachability probabilities — in probabilistic systems. That is to say, if an abstract model satisfies a property, the concrete model also does so. But if the abstract model

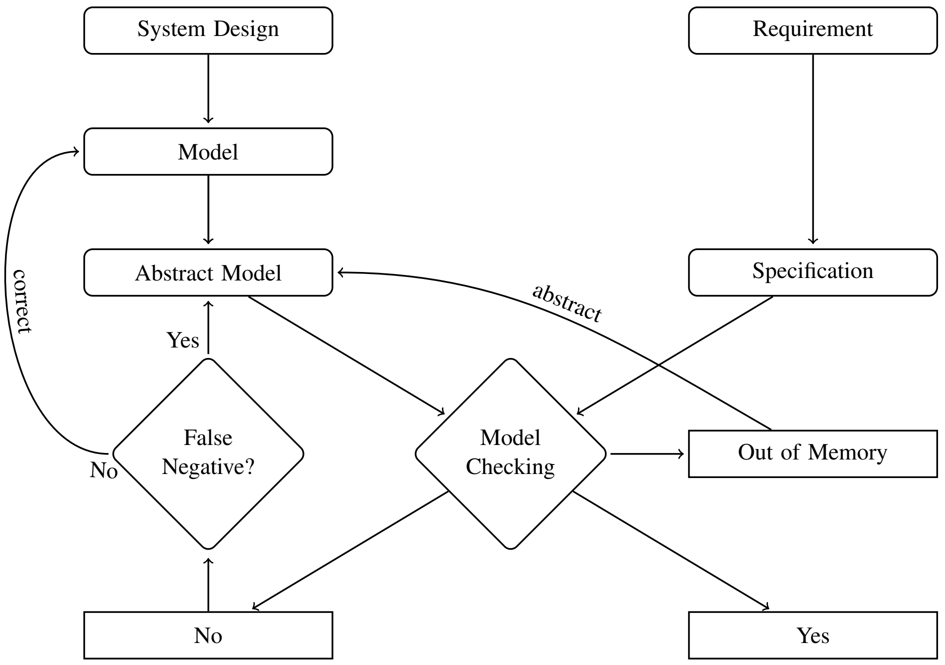


Figure 1.1: Existential abstraction in the context of model checking.

refutes, no conclusion can be drawn about the validity in the concrete model. Therefore, in case of refutation an abstract counterexample that refutes the property is analysed on the concrete model to find out the source of this behaviour. If the abstract counterexample corresponds to some concrete behaviour, then the model is faulty and needs to be corrected. Otherwise, if the refutation is due to over-approximation of concrete behaviour, then the abstraction is too coarse and needs to be refined. This procedure forms the basis of the CEGAR approach (counter-example-guided abstraction-refinement) (CEGAR)[CGJ$^+$03] for non-probabilistic systems, and has been extended for probabilistic systems in [HWZ08] (see Fig. 1.1).

Modal abstraction: The abstract models [LT88b, DKL$^+$13, KKLW12, KKN09] obtained by this technique have two transition functions: one over-approximates the behaviour of the concrete model — as in existential abstraction — whereas the other under-approximates. Alternatively, a modal abstraction, in fact, represents two abstractions of a concrete model: one over-approximates and the other under-approximates its behaviour. Therefore, if an over(under)-approximating abstraction satisfies (refutes) a safety property or a one-sided bound on a quantitative property, so does the concrete model. However, if a property is neither satisfied nor refuted by the over- and under-approximating abstraction, e.g. modal transition systems [LT88a], then an abstraction is too coarse, its verification is inconclusive and it needs to be refined, as given in [SG07, dAR07] for non-probabilistic systems and in [KKLW12] for probabilistic systems (see Fig. 1.2). Intuitively, an abstract model is analysed and those states which neither satisfy nor refute the property are refined. By this way, refinement is done locally and the size of an abstract model does grow very fast in an abstraction-refinement loop.

Game-based abstraction: This technique [KKNP10, WZ10] induces two-player turn-based games [Sha53, Con92] as abstract models in which state spaces are partitioned into two sets, one for each player. One player deals with the behaviour from the abstraction and the other from states of concrete

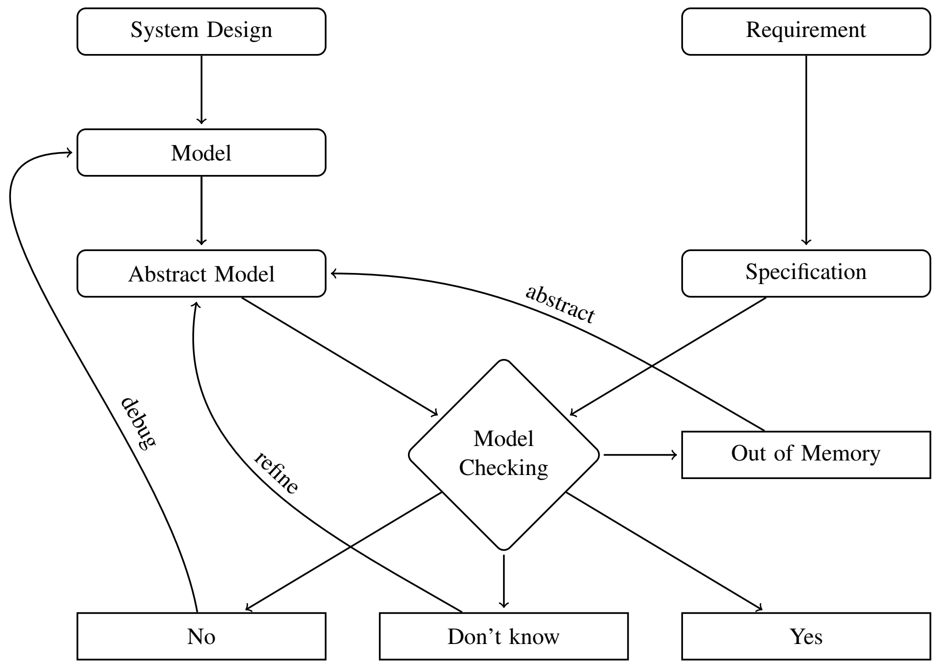


Figure 1.2: Modal/Game-based abstraction in the context of model checking.

models. This, therefore, allows to keep the behaviour from abstraction separate from the behaviour from concrete models. In probabilistic systems, game-based abstractions preserve *extremal* (maximum and minimum) reachability probabilities to sets of states, i.e., the reachability probabilities in the abstract models bound those in concrete models. For example, if the maximum probability to a set of, say, *bad* states in a concrete model is 0.8, in game-based abstract models it would be given as an interval $[x-0.8, y+0.8]$ for some $x \in [0, 0.8]$ and $y \in [0, 0.2]$. The reason of given the maximum probability as an interval in the abstract models is the additional behaviour induced by abstraction — if no additional behaviour is induced by abstraction, the values of both x and y would be zero. Moreover, if the difference between probability bounds is above a certain threshold for some states in the abstract model, they are refined as in *game-based abstraction refinement frameworks* [KKNP10, WZ10] (see Fig. 1.2). In case of non-probabilistic systems, the applicability of this technique is in those models where stake holders compete with each other to achieve their goals [HJM03].

Compositional abstraction: In another approach to tackle the state space explosion problem, the models of huge systems are not build in a monolithic way; rather using *divide-and-conquer* approach the systems are broken down into components and each component is modelled and then abstracted individually; and at the end abstract models are plugged together to get abstract models of complete systems. This strategy, therefore, calls for high level formalisms to support compositional modeling as in [BHH$^+$09, HHK02] and abstraction techniques to be compositional [Kli10].

1.3 Contributions

The formalism that we deal with in this thesis for modeling discrete-time probabilistic systems is Segala's *probabilistic automata* (PA) [Seg95], that extend labelled transition systems by allowing targets of transitions to be distributions over states rather than simply states. PA are a slight extension of Markov decision processes (MDPs) [Put94]. They allow for modeling functional and probabilistic behaviour of systems in a compositional manner, which have been successfully exploited in modeling and verification of randomized distributed algorithms, security protocols, etc.

To mention our contributions we first explain some existing game-based abstraction techniques of PA and then describe how we improve upon these techniques.

In the literature, *stochastic games* (SGs) [Sha53, Con92] have been used as abstract models of PA. SGs are turn-based two-player games in which one player has only non-deterministic behaviour whereas the other player has non-deterministic as well as probabilistic behaviour.

In [KKNP10], SG-based abstractions, called *game-based* abstractions, of Markov decision processes (MDPs) — PA with singleton action sets — have been proposed. In a game-based abstraction, one set of states, say player-two states, represents the partition of the state space of an MDP and the other set, say player-one states, represents the sets of states of the MDP that have the same step-wise behaviour after abstraction.

In a slightly different way, [WZ10] proposes SG-based abstractions of PA, called *menu-based* abstractions. In a menu-based abstraction, like game-based abstraction, player-two states represent the partition of the state space of a PA whereas player-one states represent the sets of transitions, for a particular action, of the associated concrete states. Thus, each player-two state is associated with only those player-one states that derive their transitions from the concrete states of the player-two state.

Both SG-based abstractions preserve *extremal* (maximum and minimum) reachability probabilities. Though menu-based abstractions are easier to implement than game-based abstractions, their reachability probability bounds are at most as tight as that of game-based abstractions. Moreover, [WZ10] claims that game-based abstraction is the optimal abstraction preserving reachability probabilities (best transformers in terms of abstract interpretation), i.e., no abstraction induces tighter bounds than game-based abstraction. Therefore, we concentrate on game-based abstraction [KKNP10] and describe how our techniques compare to [KKNP10].

Next, we briefly explain our contributions towards improving upon the game-based abstraction for PA. Broadly speaking, we devise *compositional* abstraction techniques that induce more *precise* (in terms of reachability probabilities) as well as *concise* (in terms of number of states and transitions) abstract models of PA than game-based abstraction, i.e., our abstract models are relatively smaller in size with reachability probability bounds at least as tight as that of game-based abstractions. This shows that game-based abstraction is not the optimal abstraction preserving reachability probabilities. Moreover, our abstractions can be plugged together, thus generating the abstractions of systems in a compositional way. In the sequel, SG-based abstractions mean game-based abstractions of [KKNP10].

Preliminary results of the material in this dissertation have been published in the following publications:

1. Falak Sher, Joost-Pieter Katoen. *Tight Game Abstractions of Probabilistic Automata.* Concurrency Theory (CONCUR), Volume 8704 of LNCS, pages 576-592, 2014.

2. Benoit Delahaye, Joost-Pieter Katoen, Kim G. Larsen, Axel Legay, Mikkel Pedersen, Falak Sher, Andrzej Wasowski. *Abstract Probabilistic Automata.* Information and Computation, Volume 232, pages 66-116, 2013.

3. Falak Sher, Joost-Pieter Katoen. *Compositional Abstraction Techniques for Probabilistic Automata.* IFIP Conference on Theoretical Computer Science (TCS), Springer Berlin Heidelberg, Volume 7604 of LNCS, pages 325-341, 2012.

4. Benoit Delahaye, Joost-Pieter Katoen, Kim G. Larsen, Axel Legay, Mikkel Pedersen, Falak Sher, Andrzej Wasowski. *New Results on Abstract Probabilistic Automata.* Applications of Concurrency to System Design (ACSD), IEEE, pages 118-127, 2011.

5. Benoit Delahaye, Joost-Pieter Katoen, Kim G. Larsen, Axel Legay, Mikkel Pedersen, Falak Sher, Andrzej Wasowski. *Abstract Probabilistic Automata.* Verification, Model Checking and Abstract Interpretation (VMCAI), Springer Berlin Heidelberg, Volume 6538 of LNCS, pages 324-339, 2011.

In one direction, we propose non-game-based abstractions of PA, i.e., the state space of abstract models is not partitioned into two sets, and in the other direction, we extend the game-based abstraction technique of [KKNP10] for PA.

- In [DKL$^+$13] (2nd publication above), we consider an extension of PA with modalities, i.e., *required* and *possible* transitions as in modal transition systems (MTS) [LT88a]. Abstract PA (APA) have as semantics a (possibly infinite) set of PA, namely all PA that have at least all *required* transitions and zero or more *possible* ones. Whereas sets of transitions are modelled by modalities, sets of distributions are represented by *constraint functions* as in constraint Markov chains [CDL$^+$11]. APA thus provide a model/framework for designing abstraction techniques for PA.

- In [SK12] (3rd publication above), we propose APA-based compositional abstraction techniques of PA such that the same transitions of concrete states become the *required* transitions in the associated abstract states, and the abstract states have at least the step-wise behaviour of concrete states. We show that APA-based abstract models bound the reachability probabilities of PA.

Although the aforementioned SG-based as well as APA-based abstraction techniques are different in nature, they have in common that the abstraction is *state-based*. That is to say, abstract models derive their transitions from that of the concrete states and, thus, simulate concrete models in a step-wise manner [JL91]. In more recent work, we propose an extension of game-based abstractions [KKNP10] of PA by lifting abstraction from states to *distributions* over states.

- In [SK14] (1st publication above), we treat distributions rather than states as first-class citizens, and relax state-based simulation to *distribution*-based simulations. Our abstractions yield (simple) probabilistic game automata (PGA) [CL88], 2-player turn-based stochastic games in which moves of both players — as opposed to classical stochastic games (SGs) [Sha53, Con92] — yield distributions over states. The new abstraction technique yields tighter upper and lower bounds on (extremal) reachability probabilities than state-based abstraction. This shows the potential superiority over *state*-based game-based MDP abstraction [KKNP10], and puts the optimality result of [WZ10] in perspective.

- In [SK14], we define two distribution-based pre-orders between abstract and concrete PGA: *simulation* and *alternating simulation* relations. Simulation relations are of interest when both players have identical objectives, whereas alternating simulation relations are useful for competitive objectives. Both relations are shown to be pre-congruences w.r.t. parallel composition of (a class of) PGA, enabling *compositional* abstraction of P(G)A. The pre-orders are the key to distribution-based abstraction, a technique distinguishing the non-deterministic behaviour of concrete distributions from that of the distributions induced by the abstraction. This enables merging concrete distributions having similar behaviour in the abstraction.

Our APA-based and PGA-based abstractions of PA are not separately discussed in this thesis. The reason being they are the special cases of the abstraction techniques that we discuss in this thesis. In this thesis, we propose a new game-based modeling formalism that generalizes APA and PGA, called *abstract probabilistic game automata* (APGA). Therefore, PA, SGs, APA and PGA are subclasses of APGA. Moreover, APGA-based abstractions of PA are the natural extensions of our APA-based [SK12] and PGA-based [SK14] abstractions.

- We consider an extension of probabilistic game automata (PGA) with modalities (i.e., *required* and *possible* transitions) and *constraint functions* as for APA [DKL$^+$13]. Abstract PGA (APGA) have as semantics a set of PGA, and are equipped with the notions of *state-based* and *distribution-based* refinement, showing that refinement relations between APGA imply (alternating) simulation relations between their implementations. We show that maximal and minimal reachability probabilities in APGA can be bound by considering extremal games – those PGA that besides all *required* transitions contain all *possible* transitions, and those that contain only *required* transitions for one set and all *possible* transitions for the other set of states. Moreover, we define a composition operator for a class of APGA that act as abstract models of PA, and show that our refinement relations are pre-congruences w.r.t. it, thus facilitating compositional abstraction of PA as APGA.

To define the APGA-based compositional abstraction techniques of PA, we combine the techniques of [KKNP10], [SK12] and [SK14] in two ways.

- First, we combine the techniques of [KKNP10] and [SK12]; the abstract models are then a class of APGA in which one of the players have only non-deterministic behaviour as in SGs. This is called *state-based abstraction* of APGA. State-based abstraction differs from [KKNP10] in the sense that the non-deterministic behaviour in concrete systems is not completely handled by one set of states: in state-based APGA-based abstraction, concrete states are merged if they have the same step-wise behaviour after abstraction; whereas in [KKNP10], they are merged iff they have the same step-wise behaviour. Because of this, the bounds on extremal reachability probabilities in state-based APGA-based models are at most as tight as in SG-based models, however, they are at most the size of SG-based models.

- Second, we combine the techniques of [SK12] and [SK14]; the induced models are APGA with both players having non-deterministic and probabilistic behaviour as in PGA. This is called *distribution-based abstraction* of APGA. The difference between distribution-based abstraction and [SK14] is the same as between state-based abstraction and [KKNP10], i.e., in distribution-based APGA-based abstraction, (support sets of) concrete distributions are merged if they have the same step-wise behaviour; whereas in [SK14], they are merged iff they have the same step-wise behaviour. Thus, the bounds on extremal reachability probabilities in distribution-based APGA-based models are at most as tight as in PGA-based models; and they are at most the size of PGA-based models.

- Both state-based and distribution-based abstractions are comparable with concrete models using state-based and distribution-based refinement relations respectively, that are shown to be pre-congruences w.r.t. parallel composition of (a large class of) APGA, enabling *compositional* abstraction of AP(G)A. Moreover, SG-based and APA-based abstractions are special cases of our state-based abstraction; and similarly PGA-based abstraction is a special case of distribution-based abstraction. We show that game-based abstraction of [KKNP10] is not the optimal abstraction preserving extremal reachability probabilities. Furthermore, we illustrate with examples that our distribution-based abstraction may induce more precise as well as concise models than our state-based abstraction of APGA.

- We propose a *state-based* and a *distribution-based abstraction-refinement framework* for PA. These frameworks are, in fact, inspired by the frameworks of [KKLW12] and [KKNP10] for *modal* and *game-based* abstractions respectively. Intuitively, it iteratively refines a modal game-based model until the effect of non-deterministic behaviour from the abstraction process in player-one states have no impact on the reachability probabilities of player-two states; and, moreover, the reachability probabilities of player-two states bound that of their corresponding concrete states within a certain given range. Therefore, the resulting models are at least as large as (PGA-)SG-based models, but they have the same precision as (PGA-)SG-based models. Moreover, we illustrate with examples that our distribution-based abstraction-refinement framework may induce abstract models having the tightest bounds on extremal reachability probabilities so far.

1.4 Outline of the thesis

- In **Chapter 2**, we define some basic notations and give background on probabilistic models used in the remaining of the thesis. We also recall Segala's simulation (bisimulation) relations for PA [Seg95], and bisimulation minimization as a reduction technique.
- In **Chapter 3**, we discuss *state-based* and *distribution-based* (*alternating*) *simulation* relations for PGA, and show that (alternating) simulation relations between PGA preserve their reachability probabilities.
- In **Chapter 4**, we define *abstract probabilistic game automata* (APGA) — an extension of PGA with *required* and *possible* modalities as well as constraint functions. We define *state-based* and *distribution-based refinement* relations for APGA, and discuss approximation techniques for APGA such that extremal reachability probabilities in APGA can be bound by considering their extremal implementations.
- In **Chapter 5**, we define two compositional abstraction techniques for APGA — *state-based* and *distribution-based abstraction* — and show that the abstract models bound the extremal reachability probabilities of concrete models.
- In **Chapter 6**, we discuss *state-based* and *distribution-based abstraction-refinement frameworks* of PA. We also show that our distribution-based framework may induce abstract models having the tightest bounds on extremal reachability probabilities so far.

2 Preliminaries

This chapter prepares a ground for this thesis by giving some notations and definitions relevant to this thesis. It recaps the concept of some probabilistic modeling formalisms like Markov chains, probabilistic automata, stochastic games, probabilistic game automata, etc. along with the reachability analysis framework for probabilistic game automata. At the end, Segala's simulation and bisimulation relations are discussed along with the limitations of bisimulation minimization to tackle the state space explosion problem in model checking of probabilistic systems.

2.1 Notations

Distributions. A *distribution* μ is a function on a countable set S iff $\mu : S \to [0,1]$ and $0 < \sum_{s \in S} \mu(s) \leq 1$; its support set is given as $\mathrm{Supp}(\mu) = \{s \in S \mid \mu(s) > 0\}$; and its mass w.r.t. set $S' \subseteq S$ is given as $\mu(S') = \sum_{s \in S'} \mu(s)$. Let $|\mu| = \mu(S)$ denote the size of the distribution μ; μ is a *full distribution* iff $|\mu| = 1$, otherwise, it is a sub-distribution. Let $\mathrm{Dist}(S)$ and $\mathrm{SDist}(S)$ denote the set of full and sub-distributions over S respectively. Let $\iota_s \in \mathrm{Dist}(S)$ denote the *Dirac* distribution for $s \in S$, i.e., $\iota_s(s) = 1$. Let $\mathrm{Dirac}(S) \subseteq \mathrm{Dist}(S)$ denote the set of Dirac distributions over S.

A distribution μ'' can be split into sub-distributions μ and μ', say, represented as $\mu'' = \mu \oplus \mu'$, iff $\mu''(s) = \mu(s) + \mu'(s)$ for $s \in S$. Since $\oplus$ is associative and commutative, we use the notation $\bigoplus$ for finite sums. Let $\mathrm{SDist}(\mu)$ denote the set of all sub-distributions that have a support included in $\mathrm{Supp}(\mu)$. A distribution is sometimes represented as $\mu = \llbracket \mu(s)s \mid s \in \mathrm{Supp}(\mu) \rrbracket$, where $\llbracket$ and $\rrbracket$ differentiate a set of probabilities from an ordinary set.

For $0 \leq c \leq 1$, $c{\cdot}\mu$ denotes the distribution defined by: $(c{\cdot}\mu)(s) = c \cdot \mu(s)$. For a distribution μ, the conditional distribution w.r.t. a set $A \subseteq \mathrm{Supp}(\mu)$ is given as: $\mu_{\downarrow_A}(s) = \frac{\mu(s)}{\mu(A)}$ for $s \in A$, and $\mu_{\downarrow_A}(s) = 0$ if $s \notin A$; if $A = \mathrm{Supp}(\mu)$, we omit A and simply write $\mu_{\downarrow}$. For example, for a sub-distribution $\mu = \llbracket 0.1 s_1, 0.3 s_2 \rrbracket$ over $S = \{s_1, s_2\}$, the conditional distribution $\mu_{\downarrow}$ is given as $\mu_{\downarrow}(s_1) = \frac{\mu(s_1)}{\mu(S)} = \frac{0.1}{0.4} = 0.25$ and $\mu_{\downarrow}(s_2) = \frac{\mu(s_2)}{\mu(S)} = \frac{0.3}{0.4} = 0.75$.

Constraint functions. A *constraint function*, denoted φ, is an arithmetic expression on variables denoting probabilities over S. A set of distributions over S satisfying φ is the *satisfaction set* of φ, denoted $sat(\varphi)$. For example, $\varphi = \big((x_1 \leq 0.5, x_2 \geq 0.5, 0 \leq x_3 \leq 1) \vee (x_1 = 0.2, x_2 = 0.2, x_3 = 0.6)\big) \wedge (x_1 + x_2 + x_3 = 1)$ is a linear constraint function on the variables x_1, x_2 and x_3 denoting probabilities over $S = \{s_1, s_2, s_3\}$ respectively, with $sat(\varphi) = \{\llbracket 0.5 s_1, 0.5 s_2 \rrbracket, \llbracket 0.2 s_1, 0.2 s_2, 0.6 s_3 \rrbracket\}$. For two constraint functions ϕ and φ, we write $\phi = \varphi$ iff $sat(\phi) = sat(\varphi)$. Let $\mathrm{CFunc}(S)$ denote the set of (not necessarily linear) constraint functions over S with non-empty satisfaction sets.

Probability measures and spaces. Let Ω be a non-empty set and $\mathscr{F} \subseteq 2^{\Omega}$. $\mathscr{F}$ is a σ-field on Ω iff: (1) $\emptyset \in \mathscr{F}$; (2) $A \in \mathscr{F} \Rightarrow \Omega \backslash A \in \mathscr{F}$; (3) $A_1, A_2, A_3, \ldots \in \mathscr{F} \Rightarrow \bigcup_{i \geq 1} A_i \in \mathscr{F}$. The elements of $\mathscr{F}$ are *measurable sets* and $(\Omega, \mathscr{F})$ is a *measurable space*. A function $\Pr : \mathscr{F} \to [0,1]$ is a *probability measure* on $(\Omega, \mathscr{F})$ iff $\Pr(\Omega) = 1$ and if $A_1, A_2, \ldots$ are disjoint elements in $\mathscr{F}$, then $\Pr(\bigcup_i A_i) = \sum_i \Pr(A_i)$. $(\Omega, \mathscr{F}, \Pr)$ is called a *measurable space*. For any $\mathscr{A} \subseteq \mathscr{F}$, there exists a unique smallest σ-field that contains $\mathscr{A}$ [ADD00]; and given that $\mathscr{A}$ satisfies certain conditions [ADD00], a *probability measure* defined on $\mathscr{A}$ can be uniquely extended to the σ-field containing $\mathscr{A}$.

2.2 Stochastic Processes

The models that we deal with in this thesis are stochastic in nature, therefore, we recap the concept of *stochastic processes* along with some of their properties as per the need of this work.

Definition 1. *A stochastic process is a collection of random variables $\{X(t) \mid t \in T\}$ defined over a probability space.*

For a stochastic process that evolves discretely like every second, every hourly, etc., we usually have $T = \mathbb{N}$, and it is called *discrete-time stochastic process*; otherwise, we have $T = \mathbb{R}_{\geq 0}$ that shows continuous evolution of the process, called *continuous-time stochastic process*. In this thesis, we only deal with discrete-time processes, moreover, we also assume the domain of $X(t)$ — called *state space* — to be countable.

Markov and time-homogeneity properties. During the evolution of a stochastic process, if the probability to reach any state s' after t' time units only depends on the current state s (and not on the states that have been traversed to reach s), the process is said to hold *Markov property*. Formally,

$$\Pr(\{X(t+t') = s' \mid \forall y \leq t : X(y) = s_y\}) = \Pr(\{X(t+t') = s' \mid X(t) = s_t\})$$

Moreover, if the above probability is even unaffected by the time span elapsed after reaching the current state s at time t, then the process possesses the *time-homogeneity property*. Formally,

$$\Pr(\{X(t+t') = s' \mid X(t) = s\}) = \Pr(\{X(t') = s' \mid X(0) = s\})$$

2.3 Discrete-time Stochastic Models

In this section, we introduce some discrete-time stochastic models that are related to this thesis. These models have been extensively used in modelling and quantitative verification of probabilistic systems and algorithms like randomized distributed algorithms, communication protocols, etc, [Her90, Rab82]. We start with the simplest models *discrete-time Markov chains*.

2.3.1 Discrete-time Markov chains (DTMC)

Discrete-time Markov chains are fully probabilistic models suitable for modeling probabilistic systems with no notion of non-determinism. They usually act as semantic models for other higher-level probabilistic models. Formally,

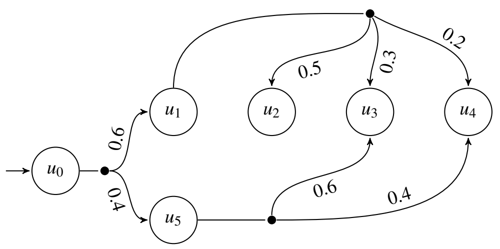


Figure 2.1: A DTMC $\mathscr{L}$

Definition 2. (Discrete-time Markov Chains). *A* discrete-time Markov chain (DTMC) *is a tuple $\mathscr{L} = (S, \Delta, s_0)$ where S is a non-empty countable set of states with initial state $s_0 \in S$ and $\Delta : S \to \mathrm{Dist}(S)$ is a total probabilistic transition function.*

The restriction of totality on transition functions ensures the deadlock freeness of DTMC. Note that our assumption of unique initial states for DTMC does not restrict their modeling power. This is because any model with a probability distribution μ_0 over initial states can be converted into a system with a single initial state s_0 having $\Delta(s_0) = \mu_0$. In the sequel, $\mathscr{L} = (S, \Delta, s_0)$ is a DTMC. We assume that in figures a state without any outgoing transition is equipped with a self-loop, and we adopt this convention for all models that we discuss in this thesis. In the figures, we depict the states of DTMC as circles.

Example 1. *Consider the DTMC $\mathscr{L}$ in Fig. 2.1 in which for every state, the transitions to next states are defined by a unique probability distribution function. Note that as per our assumption of deadlock-freeness, a state u_i is equipped with a self-loop, where $i \in \{1, 2, 3, 4\}$, but for simplicity we do not draw them.*

2.3.2 Probabilistic Automata (PA)

Segala's *probabilistic automata* [Seg95] are used for modelling probabilistic systems that have non-deter-ministic behaviour and, therefore, can act *asynchronously* as well as *synchronously*. Informally, PA are extensions of labelled transition systems (LTS) in which the target of an action-labelled transition is a distribution over states instead of a single state. Let UAct be a countable universe actions including the internal action τ. Formally,

Definition 3. (Probabilistic Automata). *A* Probabilistic Automaton (PA) *is a tuple $\mathscr{M} = (S, A, \Delta, s_0)$ where S is a non-empty, countable set of states with initial state $s_0 \in S$; $A \subseteq \mathrm{UAct}$; and $\Delta \subseteq S \times A \times \mathrm{Dist}(S)$ is a set of transitions.*

We denote $(s, a, \mu) \in \Delta$ by $s \xrightarrow{a} \mu$ and $\mathrm{Act}(s)$ as the set of enabled actions from state s, i.e., $\mathrm{Act}(s) = \{a \in A \mid s \xrightarrow{a} \mu\}$; and $\mathrm{Succ}(s) = \{u \in S \mid \exists s \to \mu : \mu(u) > 0\}$ as the set of successor states of s. We say a PA is *finite* if S and Δ are finite sets. We assume that each state in PA has at least one action enabled from it to ensure deadlock freeness. Note that DTMC is the subclass of PA in which $A = \{\tau\}$ and $|\mathrm{Act}(s)| = 1$ for $s \in S$. Moreover, *Markov decision processes* (MDPs) are also the subclass of PA in which $a \in \mathrm{Act}(s)$ implies $|\Delta(s, a)| = 1$ for $s \in S$. In the sequel, $\mathscr{M} = (S, A, \Delta, s_0)$ is an infinitely branching PA. For depicting PA, we use the same convention as for DTMC.

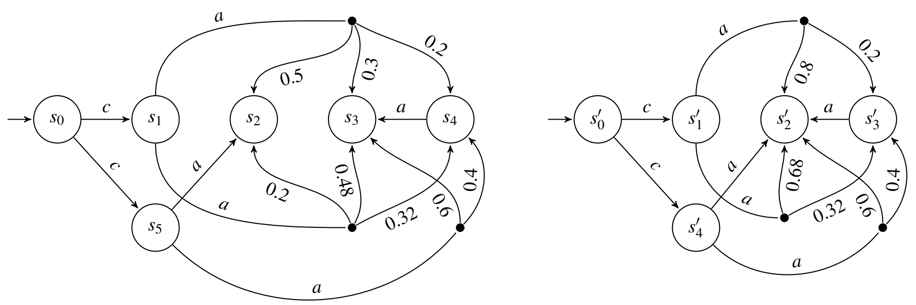


Figure 2.2: PA $\mathscr{M}$ (left) and $\mathscr{M}'$ (right) are bisimilar, i.e., $\mathscr{M} \sim_{\text{pa}} \mathscr{M}'$

Example 2. *Consider the PA $\mathscr{M}$ in Fig. 2.2. The target of the transitions $s_1 \xrightarrow{a} [\![0.5s_2, 0.3s_3, 0.2s_4]\!]$, $s_1 \xrightarrow{a} [\![0.2s_2, 0.48s_3, 0.32s_4]\!]$ and $s_5 \xrightarrow{a} [\![0.6s_3, 0.4s_4]\!]$ are distributions over states; whereas for other transitions the targets are Dirac distributions e.g. the a-transition from s_4 has ι_{s_3} as the target distribution (for simplicity we omit "black dots" to represent Dirac distributions). Note that s_0, s_1 and s_5 have more than one target distributions for one action.*

2.3.3 Alternating Two-Player Stochastic Games (SGs)

PA are not suitable for modeling probabilistic systems in which stakeholders compete among themselves for certain objectives. For such system, *stochastic games (SGs)* have been proposed [Sha53, Con92].

An alternating two-player SG is a game of chance played between two players, say, player one and player two. The game arena is a bipartite graph – having, say, S_1 and S_2 as sets of vertices – in which each player owns a specific set of vertices; say, the players one and two own S_1 and S_2 respectively. The game is started by player one and evolves in a turn-based fashion. Starting from the initial state in S_1, player one non-deterministically chooses an action-distribution pair. Based on the selected distribution, a state in S_2, say s_2, is randomly selected and control is passed to player two. Player two non-deterministically selects an enabled action in s_2, uniquely picks a successor of s_2 and passes control back to player one. This goes on until some goal is achieved either by player one or player two.

Definition 4. (Stochastic Games). *A* Stochastic Game (SG) *is a tuple $\mathscr{S} = (S, \{S_1, S_2\}, A, \Delta, s_0)$ where S is a non-empty, countable set of states, disjointly partitioned into S_1 and S_2, with $s_0 \in S_1$; $A \subseteq \mathrm{UAct}$; and $\Delta \subseteq (S_1 \times A \times \mathrm{Dist}(S_2)) \cup (S_2 \times A \times \mathrm{Dirac}(S_1))$ is a set of probabilistic transitions.*

We assume that a game is started by player one in s_0. Note that PA are SGs in which $\forall s \in S_2$, $a, b \in A$: $(s \xrightarrow{a} \mu \wedge s \xrightarrow{b} \nu)$ implies $\mu = \nu$ and $|\mathrm{Supp}(\mu)| = 1$. For depicting SGs we represent states in S_1 and S_2 as rectangles and double rectangles respectively. Moreover, we show player-one states inside player-two states for simplicity. In the sequel, $\mathscr{S} = (S, \{S_1, S_2\}, A, \Delta, s_0)$ is an (possibly infinitely branching) SG.

Example 3. *Consider the SG $\mathscr{S}$ in Fig. 2.3 with $S_1 = \{s_0, s_1, s_2, s_3, s_4\}$ and $S_2 = \{t_1, t_2, t_3\}$. Note that the target of a transition from every state in S_2 is a Dirac distribution which is not the case for states in S_1. For example, s_2 has an a-transition to $[\![0.5t_2, 0.5t_3]\!]$. For simplicity, we do not draw transitions from*

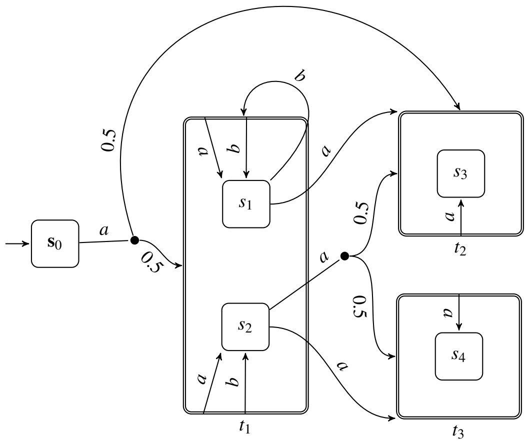


Figure 2.3: An SG $\mathscr{S}$

s_3 and s_4.

Let player one start the game and choose state t_1 with probability 0.5. It then passes control to player two. Player two has two possibilities (s_1 or s_2). Let she choose state s_2 and pass control back to player one. Player one now has two choices, she can non-deterministically select one of the a-transitions and continue the game. In this way, (possibility) infinite runs in $\mathscr{S}$ can be generated.

2.3.4 (Simple) Probabilistic Game Automata (PGA)

In SGs, player-one moves yield distributions over states, while player-two moves yield states. In PGA, player-two moves also yield distributions over states. Formally,

Definition 5. (Simple Probabilistic Game Automata). *A* Simple Probabilistic Game Automaton (PGA) *is a tuple $\mathscr{G} = (S, \{S_1, S_2\}, A, \Delta, s_0)$ where S, S_1, S_2, A and s_0 are as for* SGs*, and $\Delta \subseteq (S_1 \times A \times \mathrm{Dist}(S_2)) \cup (S_2 \times A \times \mathrm{Dist}(S_1))$ is a set of probabilistic transitions.*

PGA are simplified versions of the probabilistic game automata in [CL88]. SGs are the subclass of PGA in which $\mathrm{Dist}(S_1)$ is a set of Dirac distributions. In the sequel, $\mathscr{G} = (S, \{S_1, S_2\}, A, \Delta, s_0)$ is a (possibly infinitely branching) PGA. For depicting PGA, we use the same convention as for SGs.

Example 4. *Consider the PGA $\mathscr{G}$ in Fig. 2.4 with $S_1 = \{s_0, s_1, s_2, s_3, s_4\}$ and $S_2 = \{t_1, t_2, t_3\}$. Note that in $\mathscr{G}$, both the players can make a non-deterministic as well as probabilistic choices. For example, unlike in Fig. 2.3, the target of the a-transition from t_1 is a distribution, i.e., $[\![0.5 s_1, 0.5 s_2]\!]$. The rest of the example is trivial.*

Paths: If $|\mathrm{Act}(s)| > 1$ for a state s, a non-deterministic choice among the enabled actions in s occurs. A path (also known as *play*) in a PGA represents a particular resolution of non-determinism by players one and two at each state, as well as a resolution of the probabilistic choices. Formally,

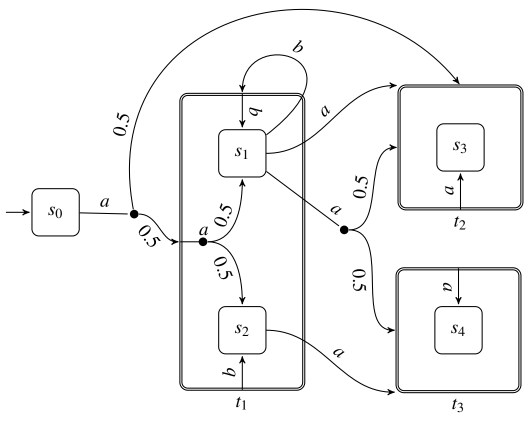


Figure 2.4: A PGA $\mathscr{G}$

Definition 6. *For PGA $\mathscr{G}$, a path from $s_{1_0} \in S_1$ is given as:*

$$\pi = s_{1_0} \xrightarrow{a_{1_0},\mu_{1_0}} s_{2_0} \xrightarrow{a_{2_0},\mu_{2_0}} s_{1_1} \dots$$

where $s_{i_k} \in S_i$, $a_{i_k} \in \mathrm{Act}(s_{i_k})$, $(s_{i_k}, a_{i_k}, \mu_{i_k}) \in \Delta$, $\mu_{1_k}(s_{2_k}) > 0$ and $\mu_{2_k}(s_{1_{k+1}}) > 0$ for all $i \in \{1,2\}$ and $k \geq 0$. A path is finite *if it has a finite number of transitions, otherwise* infinite*.*

For a finite path π_{fin}, let $\text{last}_i(\pi_{\text{fin}})$ denote the last S_i state in π_{fin} for $i \in \{1,2\}$. Let $\text{Path}_{\text{fin}}(\mathscr{G})$ and $\text{Path}_{\text{inf}}(\mathscr{G})$ denote the set of finite and infinite paths in a PGA $\mathscr{G}$ respectively, and $\text{Paths}(\mathscr{G}) = \text{Path}_{\text{fin}}(\mathscr{G}) \cup \text{Path}_{\text{inf}}(\mathscr{G})$.

Schedulers: In order to analyse reachability properties on $\mathscr{G}$, we resolve non-determinism at all game states by means of a *scheduler* (also known as *policy, strategy or adversary*). Let κ_i be the scheduler for S_i states, $i \in \{1,2\}$. We consider *deterministic memoryless* (DM) schedulers as they suffice for reachability probabilities on PGA. DM-schedulers select an action-distribution pair only on the basis of the current state. More specifically, for bit x, a *deterministic* scheduler $\kappa_{(1+x)}$ maps a finite path π_{fin} to a pair in $\mathrm{Act}(last_{(1+x)}(\pi_{\text{fin}})) \times \mathrm{Dist}(S_{(2-x)})$; and a *memoryless* scheduler $\kappa_{(1+x)}$ assures that for finite paths π_{fin} and π'_{fin}, $last_{(1+x)}(\pi_{\text{fin}}) = last_{(1+x)}(\pi'_{\text{fin}})$ implies $\kappa_{(1+x)}(\pi_{\text{fin}}) = \kappa_{(1+x)}(\pi'_{\text{fin}})$.

A path π under a pair of DM-schedulers (κ_1, κ_2) is of the form $\pi = s_{1_0} \xrightarrow{a_{1_0},\mu_{1_0}} s_{2_0} \xrightarrow{a_{2_0},\mu_{2_0}} s_{1_1} \dots$ where $\kappa_i(s_{i_k}) = (a_{i_k}, \mu_{i_k})$ for $i \in \{1,2\}$ and $k \geq 0$. Let $\text{Paths}^{\kappa_1}_{\kappa_2}(\mathscr{G})$ be the set of paths of PGA $\mathscr{G}$ under DM-schedulers (κ_1, κ_2). The DM-schedulers (κ_1, κ_2) on PGA $\mathscr{G}$ induce a Markov chain with countably-many states. This allows us to construct a measurable space $(\text{Paths}^{\kappa_1}_{\kappa_2}(\mathscr{G}), \mathscr{F}^{\kappa_1}_{\kappa_2}, \Pr^{\kappa_1}_{\kappa_2})$ over the (infinite) paths of $\mathscr{G}$ under (κ_1, κ_2) [BK08, Ch. 10]; and thus determining the probability for a certain set of paths of $\mathscr{G}$ under (κ_1, κ_2). In the sequel, we consider only DM-schedulers.

Reachability Probabilities in PGA

This section discusses how optimal (i.e., maximal and minimal) reachability probabilities are defined for PGA. We first define some notations and definitions.

Optimal reachability probabilities: Let $\Pr_{\kappa_2}^{\kappa_1}(T)$ be the probability of the set of paths from the initial state s_0 that reach some set of states $T \subseteq S$ under schedulers (κ_1, κ_2) for PGA $\mathscr{G}$.

Definition 7. [CL88] *For PGA $\mathscr{G}$, the* optimal probabilities *of reaching $T \subseteq S$ for players one and two are defined respectively as:* $\sup_{\kappa_1} \inf_{\kappa_2} \Pr_{\kappa_2}^{\kappa_1}(T)$ *and* $\inf_{\kappa_1} \sup_{\kappa_2} \Pr_{\kappa_2}^{\kappa_1}(T)$.

Intuitively, the reachability probability to a set T of target states is optimal for player one under scheduler κ iff for every scheduler κ_2 of player two, $\inf_{\kappa_2} \Pr_{\kappa_2}^{\kappa}(T) = \sup_{\kappa_1} \inf_{\kappa_2} \Pr_{\kappa_2}^{\kappa_1}(T)$. Similarly, we can define optimal reachability probability for player two. For PGA $\mathscr{G}$ and $T \subseteq S$, we write:

- $\max^{\blacktriangledown}(T) = \sup_{\kappa_1} \inf_{\kappa_2} \Pr_{\kappa_2}^{\kappa_1}(T)$ and $\max^{\blacktriangle}(T) = \sup_{\kappa_1} \sup_{\kappa_2} \Pr_{\kappa_2}^{\kappa_1}(T)$
- $\min^{\blacktriangledown}(T) = \inf_{\kappa_1} \inf_{\kappa_2} \Pr_{\kappa_2}^{\kappa_1}(T)$ and $\min^{\blacktriangle}(T) = \inf_{\kappa_1} \sup_{\kappa_2} \Pr_{\kappa_2}^{\kappa_1}(T)$.

Note that the values $\max^{\blacktriangledown}(T)$ and $\min^{\blacktriangle}(T)$ are the optimal reachability probabilities for players one and two respectively, which can be achieved by DM-schedulers [CL88]. The values $\max^{\blacktriangle}(T)$ and $\min^{\blacktriangledown}(T)$ – for which both players collaborate with each other – can be obtained similarly. For games with finite state spaces these values can be computed through value iteration [BT91, Alf99] or by linear programming.

Closed PGA: To calculate the reachability probabilities of a PGA, we assume that it does not interact with its environment. Therefore, we define a function that yields closed-versions of PGA — PGA in which all actions are replaced with the internal action τ.

Definition 8. *For* PGA $\mathscr{G}$, *let PGA* $\tau(\mathscr{G}) = \mathscr{G}' = (S', \{S'_1, S'_2\}, A', \Delta', s'_0)$ *with* $S' = S$, $s'_0 = s_0$, $A' = \{\tau\}$ *and* $\Delta' = \{(s, \tau, \mu) \mid (s, a, \mu) \in \Delta\}$.

Probability valuation transformer: Let $w : S \to [0,1]$ be a probability valuation function mapping a state s to the probability of reaching target states $T \subseteq S$ from s. The probability valuation functions $W = \{w \mid w : S \to [0,1]\}$ form a complete lattice $(W, \leq, \bot, \top)$ with order $\leq$, bottom element $\bot \in W$ and top element $\top \in W$. We write $w \leq w'$ iff $w(s) \leq w'(s)$; $\bot(s) = 0$ and $\top(s) = 1$ for $s \in S$. For a set $M \subseteq W$, the least upper bound is given as $\bigsqcup M(s) = \max_{w \in M} w(s)$, and the greatest lower bound as $\bigsqcap M(s) = \min_{w \in M} w(s)$ for $s \in S$. Let $w(\mu) = \sum_{s \in S} \mu(s) \cdot w(s)$ for $\mu \in \mathrm{Dist}(S)$.

Definition 9. *In PGA $\tau(\mathscr{G})$ with goal states $T \subseteq S$, let $T_0 \subseteq S$ be the set of states without outgoing transitions. For reachability objectives $\mathbf{1}, \mathbf{2} \in \{\min, \max\}$ for players one and two respectively, the* probability valuation transformer $\mathrm{Prt}_{\mathbf{2}}^{\mathbf{1}} : W \to W$ *is defined for $w \in W$ and $s \in S$ as:*

$$\mathrm{Prt}_{\mathbf{2}}^{\mathbf{1}}(w)(s) = \begin{cases} 1 & \text{if } s \in T \\ \mathbf{1} = \max ?\, 0 : 1 & \text{if } s \in S_1 \cap T_0 \\ \mathbf{2} = \max ?\, 0 : 1 & \text{if } s \in S_2 \cap T_0 \\ \mathbf{1}\{w(\mu) \mid s \xrightarrow{\tau} \mu\} & \text{if } s \in S_1 \setminus (T \cup T_0) \\ \mathbf{2}\{w(\mu) \mid s \xrightarrow{\tau} \mu\} & \text{if } s \in S_2 \setminus (T \cup T_0) \end{cases}$$

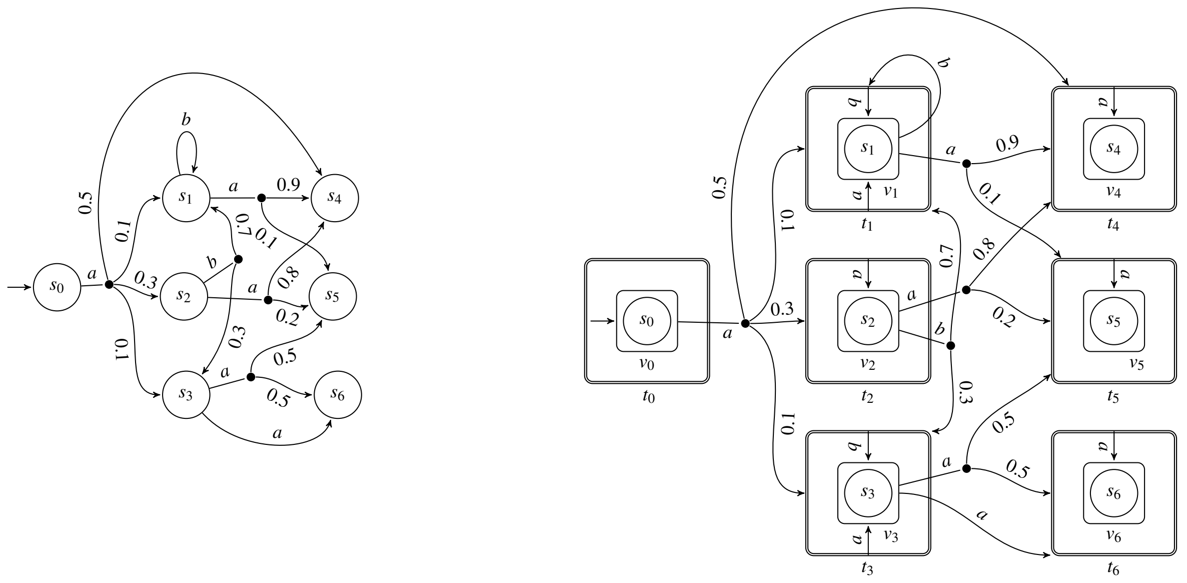


Figure 2.5: A PA $\mathscr{M}$ (left) and its embedding $\mathscr{G} = \alpha_{\text{PA}}(\mathscr{M})$ (right)

For a state $s \in T_0$, the reachability probability is set depending on the objective of its associated player: if it is to maximize then the reachability probability is set to 0; otherwise 1. If $s \in S_1 \setminus (T \cup T_0)$ then for the next iteration the reachability probability of s is the optimal value of the set $\{w(\mu) \mid s \xrightarrow{\tau} \mu\}$ w.r.t. the objective $\mathbf{1}$; and if $s \in S_2 \setminus (T \cup T_0)$, it is w.r.t. the objective $\mathbf{2}$. Note that $\text{Prt}^{\mathbf{1}}_{\mathbf{2}}$ is a monotonic function over W and, by Tarski's theorem [T+55], has a least and a greatest fixpoint. This definition provides the basis to compute reachability probabilities.

Example 5. *For PGA $\mathscr{G}$ in Fig. 2.4, let $\mathbf{1} = \max$, $\mathbf{2} = \min$, $T = \{t_3\}$ and w_0 be a probability valuation function with $w_0(t_3) = 1$ and $w_0(v) = 0$ for every $v \in S \backslash \{t_3\}$. Then $w_1 = \text{Prt}^{\mathbf{1}}_{\mathbf{2}}(w_0)$ with $w_1(t_3) = 1$, $w_1(s_1) = 0.5$, $w_1(s_2) = 1$ and $w_1(v) = 0$ for every $v \in S \backslash \{t_3, s_1, s_2\}$. Note that the probability valuation function w with $w(s_0) = 0.25$, $w(s_1) = 0.5$, $w(s_2) = 1$, $w(s_3) = 0$, $w(s_4) = 0$, $w(t_1) = 0.5$, $w(t_2) = 0$, and $w(t_3) = 1$ is a fixpoint of* $\text{Prt}^{\mathbf{1}}_{\mathbf{2}}$.

Embedding of PA into PGA

In the following, we show how a PA can be embedded into a PGA. For a state $s \in S$, let $\bar{s}$ be a copy of s.

Definition 10. *For* PA $\mathscr{M}$*, the bijective* embedding *function $\alpha : S \to S'_2$ induces the* PGA $\alpha(\mathscr{M}) = \mathscr{G}' = (S', \{S'_1, S'_2\}, A', \Delta', s'_0)$ *where $A' = A$, $S'_1 = \{\overline{s'} \mid s' \in S'_2\}$ — S'_1 is a copy of S'_2 —, $s'_0 = \overline{\alpha(s_0)}$ and for every $s' \in S'_2$:*

1. *$\overline{s'} \xrightarrow{a} \mu'$ iff $\alpha^{-1}(s') \xrightarrow{a} \mu$ and $\mu'(u') = \mu(\alpha^{-1}(u'))$ for all $u' \in S'_2$,*
2. *$s' \xrightarrow{a} \iota_{\overline{s'}}$ iff $\alpha^{-1}(s') \in \text{Supp}(\mu)$ for some $u \in S$ such that $(u, a, \mu) \in \Delta$ in $\mathscr{M}$.*

Let α_{PA} denote an embedding *function for* PA.

Example 6. *Let $\mathcal{G} = \alpha_{\mathrm{PA}}(\mathcal{M})$ (see Fig. 2.5) with $S_2 = \{t_0, \dots, t_6\}$ and $S_1 = \{v_0, \dots, v_6\}$, $\alpha_{\mathrm{PA}}^{-1}(t_i) = s_i$, and $\overline{t_i} = v_i$, for $i = 0$ to 6. For convenience, the s_i states are depicted inside the corresponding states v_i and t_i. We have e.g., $v_2 \xrightarrow{b} \mu'$ with $\mu'(t_1) = \frac{7}{10}$ and $\mu'(t_3) = \frac{3}{10}$ and $t_1 \xrightarrow{b} v_1$ and $t_3 \xrightarrow{b} v_3$, as in PA $\mathcal{M}$ we have $s_2 \xrightarrow{b} \mu$ with $\mu(s_1) = \frac{7}{10}$ and $\mu(s_3) = \frac{3}{10}$.*

Combined hyper-transitions

We now adapt hyper and combined transitions – convex combinations of sets of transitions – for PA [Seg95, LSV07] to PGA.

Definition 11. *For* PGA $\mathcal{G}$ *with $s \in S$ and $\mu \in \mathrm{Dist}(S)$, we write:*

- $\mu \xrightarrow{a} \eta$ *is a* hyper-transition *iff $\eta = \bigoplus\{\mu(s) \cdot \rho \mid \exists s \in \mathrm{Supp}(\mu) : s \xrightarrow{a} \rho\}$. Let $\Delta(\mu, a) = \{\eta \mid \exists \eta \in \mathrm{Dist}(S) : \mu \xrightarrow{a} \eta\}$; and $\Delta(\mu) = \{(a, \nu) \mid \exists a \in A, \nu \in \mathrm{Dist}(S) : \mu \xrightarrow{a} \nu\}$.*

- $s \xrightarrow{a}_{\mathrm{c}} \eta$ *is a* combined transition *iff there is a finite indexed set $\{(c_i, \eta_i)\}_{i \in I}$ such that $s \xrightarrow{a} \eta_i$ and $c_i \in \mathbb{R}_{\geq 0}$ for all $i \in I$, $\sum_{i \in I} c_i = 1$ and $\eta = \bigoplus_{i \in I} c_i \cdot \eta_i$.*

- $\mu \xrightarrow{a}_{\mathrm{c}} \eta$ *is a* combined hyper-transition *iff $\eta = \bigoplus\{\mu(s) \cdot \rho \mid \exists s \in \mathrm{Supp}(\mu) : s \xrightarrow{a}_{\mathrm{c}} \rho\}$.*

2.4 Simulation Relations on PA

Simulation relations are used to compare the behaviour of systems. They have been extensively discussed both for non-probabilistic [Mil89, LV92] as well as probabilistic systems [Seg95, SL95, DKL$^+$11, SK12]. For non-probabilistic systems, they are defined over the states of systems; however, for probabilistic cases, they have also been defined over the distributions over states [Seg95, EHZ10, DHR08].

The state-based notion of simulation for probabilistic systems [JL91] is a preorder on a state space requiring that whenever state u simulates state s, then u can mimic the stepwise behaviour of s but may have more behaviour. This notion can be lifted to distributions over states using weight functions [JL91] as:

Definition 12. *Let S be a finite, non-empty set of states, and let $\mu, \mu' \in \mathrm{Dist}(S)$. For $R \subseteq S \times S$, μ' simulates μ w.r.t. R, denoted $\mu R \mu'$, iff there exists a weight function $\delta : S \times S \to [0, 1]$ such that for all $u, v \in S$:*

1. *$\delta(u, v) > 0 \Rightarrow uRv$,*
2. *$\sum_{s \in S} \delta(u, s) = \mu(u)$, and*
3. *$\sum_{s \in S} \delta(s, v) = \mu'(v)$.*

In [BEMC00], it has been shown that simulation preorders can be computed by reducing them to maximum-flow problems in suitable networks.

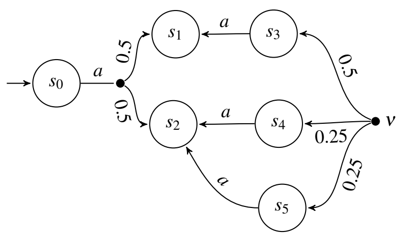


Figure 2.6: $s_0 \prec_{\text{paf}} [\![0.5s_3, 0.25s_4, 0.25s_5]\!]$

2.4.1 Segala's Probabilistic (Bi)Simulation Relations

We now recall *Segala's probabilistic simulation*, *forward simulation* and *bisimulation* [Seg95, LSV07] relations for PA.

Definition 13. (Simulation). *$R \subseteq S \times S$ is a* simulation *relation on PA $\mathscr{M}$ iff for every sRs', $s \xrightarrow{a} \mu$ implies $s' \xrightarrow{a}_c \mu'$ such that $\mu R \mu'$. If it also holds that $s' \xrightarrow{a} \mu'$ implies $s \xrightarrow{a}_c \mu$ such that $\mu R \mu'$, then R is a* bisimulation *relation.*

We can lift $\prec_{\text{pa}}$ ($\sim_{\text{pa}}$) to PA in the usual way: $\mathscr{M} \prec_{\text{pa}} \mathscr{M}'$ ($\mathscr{M} \sim_{\text{pa}} \mathscr{M}'$) for PA $\mathscr{M}$ and $\mathscr{M}'$, with initial states s_0 and s_0', iff $s_0 \prec_{\text{pa}} s_0'$ ($s_0 \sim_{\text{pa}} s_0'$) in the disjoint union of $\mathscr{M}$ and $\mathscr{M}'$. In the sequel, we will adopt this convention for all relations.

Note that R is defined at the level of states; whereas $\mu R \mu'$ denotes μ and μ' are related by Def. 12 w.r.t. R.

Example 7. *Ignoring the action labels of PA $\mathscr{M}$ (Fig. 2.2), $\mathscr{M}$ simulates the DTMC $\mathscr{L}$ (Fig. 2.1), i.e., $\mathscr{L} \prec_{\text{pa}} \mathscr{M}$ as $R = \bigcup_{i=0\ldots5}\{(u_i, s_i)\}$ is a simulation relation between states of $\mathscr{L}$ and $\mathscr{M}$. Moreover, PA $\mathscr{M}$ and $\mathscr{M}'$ (Fig. 2.2) are bisimilar, i.e., $R' = \{(s_0, s_0'), (s_1, s_1'), (s_2, s_2'), (s_3, s_2'), (s_4, s_3'), (s_5, s_4')\}$ is a bisimulation relation relating the initial states of $\mathscr{M}$ and $\mathscr{M}'$.*

Segala's probabilistic forward simulation is based on distributions over states rather than states.

Definition 14. (Forward Simulation). *$R \subseteq S \times \mathrm{Dist}(S)$ is a* probabilistic forward simulation *relation on PA $\mathscr{M}$ iff for every $sR\mu$, $s \xrightarrow{a} \eta$ implies $\mu \xrightarrow{a} \eta'$ such that $\forall u \in \mathrm{Supp}(\eta)$, $\exists \eta_u' \in \mathrm{SDist}(\eta') : uR\eta'_{u\downarrow}$. Let $\prec_{\text{paf}}$ be the largest* forward simulation *relation.*

Note that for $sR\mu$, an a-transition from s to *some* η implies a hyper a-transition from μ to η' such that η' splits into sub-distributions as per the support of η, i.e., for every $u \in \mathrm{Supp}(\eta)$, there exists a sub-distribution η_u' of η', and the conditional distribution of η_u' is related to u. Recall that every state can also be represented by a Dirac distribution, and hence R is directly defined at the level of distributions over states instead of states. Note that in $uR\eta'_{u\downarrow}$, u and $\eta'_{u\downarrow}$ are not related by Def. 12.

Example 8. *In Fig. 2.6, $s_0 \prec_{\mathrm{paf}} [\![0.5s_3, 0.25s_4, 0.25s_5]\!] = \nu$ as $R = \{(s_1, \iota_{s_1}), (s_2, \iota_{s_2}), (s_0, [\![0.5s_3, 0.25s_4, 0.25s_5]\!])\}$ is a probabilistic forward simulation relation. Let us check the conditions of Def. 14 for s_0 and ν. For the a-transition from s_0 to $[\![0.5s_1, 0.5s_2]\!]$, there is an a-transition from ν to $[\![0.5s_1, 0.5s_2]\!]$, and the condition of splitting the target distributions into sub-distributions trivially holds.*

2.5 Bisimulation Minimization

Bisimulation relations, in fact, relate those states of a system that cannot be distinguished apart in any aspect, i.e, they as well as their successors mimic the step-wise behaviour of each other. This provides a recipe to reduce the state space of a system, by merging bisimilar states, without compromising its behaviour. Therefore, bisimulation is an important minimization technique that has been used for all kinds of models. In fact, bisimulation is the coarsest equivalence that is compatible with trace equivalence, that is, there cannot exist another equivalence that implies trace equivalence and that collapses more states. Moreover, bisimulation minimization is a fully automated technique that can be done compositionally and it preserves exact probabilities (of almost all interesting measures).

However, bisimulation is not always a suitable technique for reduction. It may, sometimes, not induce a required reduction — in case of infinite state models, it is not always possible to reduce them to finite state models using bisimulation minimization — that otherwise is possible using other aggressive reduction techniques based on grouping (possibly) non-bisimilar states like *three-valued abstraction* [Kli10], etc. Such aggressive reduction techniques preserve simulation relations between concrete and abstract models, and they are not always fully automated (or involve more complex algorithms).

In the next chapter, we introduce (alternating) simulation relations for PGA and show that they preserve reachability probabilities. Later on in Ch. 4 and 5, these relations compare abstract with concrete models.

3 Relations on Stochastic Games

In this chapter, we discuss relations for comparing stochastic games. Stochastic games have the notion of players which, therefore, calls for defining two preorders (and equivalences) on their state spaces: *simulation* and *alternating simulation* relations. Simulation relations are of interest when both of the players have identical objectives, whereas alternating simulation relations are useful for competitive objectives. We define simulation and alternating simulation relations at the level of states as well as distributions over states and study their ordering. Moreover, we show that these relations preserve reachability probabilities. Later on in Ch. 5, we discuss abstraction of PGA and show that (some implementations of) abstract models are related to concrete models through these simulation relations. In Ch. 4, we also show these relations to be pre-congruences w.r.t. parallel composition for (a class of) probabilistic game automata (PGA), thus, enabling *compositional* abstraction of P(G)A.

3.1 Simulation Relations

Simulation relations are typically defined over the states of models; however in the probabilistic settings, coarser relations have been considered over the distributions over states [Seg95, EHZ10, DHR08]. We define simulation relations for PGA, that are state-based as well as distribution-based, and prove them to be preorders.

3.1.1 State-based Simulation Relation

State-based simulation relations are preorders on a state space requiring that whenever a state s' simulates a state s, then s' can mimic at least the step-wise behaviour of s. Formally,

Definition 15. (State-based Simulation). *$R \subseteq \bigcup_{j\in\{1,2\}} S_j \times S_j$ is a* state-based simulation (SBS) *relation on a PGA $\mathcal{G}$ iff for every sRs',*

- *$s \xrightarrow{a} \mu$ implies $s' \xrightarrow{a}_{\mathrm{c}} \mu'$ with $\mu R \mu'$.*

Let $\prec_{\mathrm{sb}}$ be the largest SBS relation.

Def. 15 asserts that, for sRs', an a-transition from s implies a combined a-transition from s' such that the resulting distributions are related (by Def. 12) w.r.t. R. Note that player-one(two) states can only be related with player-one(two) states.

Proposition 1. *For PGA $\mathcal{G}$ and $\mathcal{G}'$, $\mathcal{G} \prec_{\text{sb}} \mathcal{G}'$ implies $\tau(\mathcal{G}) \prec_{\text{sb}} \tau(\mathcal{G}')$.*

Lemma 1. *Let $\mu_1 \in \text{Dist}(U_1)$, $\mu_2 \in \text{Dist}(U_2)$ and $\mu_3 \in \text{Dist}(U_3)$. Let $R \subseteq U_1 \times U_2$ and $R' \subseteq U_2 \times U_3$ be relations such that $R'' \subseteq U_1 \times U_3$ and $R'' = R \circ R'$ (composition of R and R'), then:*

$$\mu_1 R \mu_2 \textit{ and } \mu_2 R' \mu_3 \textit{ implies } \mu_1 R'' \mu_3$$

Proof. In order to prove that $\mu_1 R'' \mu_3$, we need to show that there exists a weight function for distributions μ_1 and μ_3 w.r.t. relation R''. Let δ and δ' be the weight functions for distributions μ_1 and μ_2 w.r.t. relation R, and μ_2 and μ_3 w.r.t. relation R' respectively. We define a weight function for distributions μ_1 and μ_3 w.r.t. relation R'' such that for all $u_1 \in U_1$, $u_2 \in U_2$ and $u_3 \in U_3$:

$$\delta''(u_1,u_3) = \sum_{u_2 \in U_2} \frac{\delta(u_1,u_2)\cdot\delta'(u_2,u_3)}{\mu_2(u_2)}$$

and prove that it fulfils the three conditions of a weight function (Def. 12).

1. It follows trivially from the definition.
2. The proof of $\delta''(u_1,U_3) = \mu_1(u_1)$ goes as follows:

$$\begin{aligned}\sum_{u_3 \in U_3} \delta''(u_1,u_3) &= \sum_{u_3 \in U_3}\sum_{u_2 \in U_2} \frac{\delta(u_1,u_2)\cdot\delta'(u_2,u_3)}{\mu_2(u_2)} \\ &= \sum_{u_2 \in U_2} \frac{\delta(u_1,u_2)\cdot\sum_{u_3 \in U_3}\delta'(u_2,u_3)}{\mu_2(u_2)} \\ &= \sum_{u_2 \in U_2} \frac{\delta(u_1,u_2)\cdot\delta'(u_2,U_3)}{\mu_2(u_2)} \\ &= \sum_{u_2 \in U_2} \frac{\delta(u_1,u_2)\cdot\mu_2(u_2)}{\mu_2(u_2)} \\ &= \mu_1(u_1)\end{aligned}$$

3. This is proven along the same lines as case two.

□

Theorem 1. $\prec_{\text{sb}}$ *is a preorder.*

Proof. Reflexivity: It follows trivially from Def. 15.

Transitivity: Let $\mathscr{G} = (S, \{S_1, S_2\}, A, \Delta, s_0)$, $\mathscr{G}' = (S', \{S'_1, S'_2\}, A, \Delta', s'_0)$ and $\mathscr{G}'' = (S'', \{S''_1, S''_2\}, A, \Delta'', s''_0)$ be PGA. Let $\mathscr{G} \prec_{\text{sb}} \mathscr{G}'$ and $\mathscr{G}' \prec_{\text{sb}} \mathscr{G}''$, then we prove that $\mathscr{G} \prec_{\text{sb}} \mathscr{G}''$ holds. Let R_1 be an SBS relation between $\mathscr{G}$ and $\mathscr{G}'$, and R_2 between $\mathscr{G}'$ and $\mathscr{G}''$. We define the relation $R \subseteq (S_1 \times S''_1) \cup (S_2 \times S''_2)$ as:

$$R = \{(s, s'') \mid sR_1s', s'R_2s'' \textit{ for } s' \in S'\}$$

and show that it fulfils the conditions of Def. 15.

Assume sRs'', and let $s \xrightarrow{a} \mu$. As sR_1s', by Def. 15, $s' \xrightarrow{a}_{\text{c}} \mu'$ such that $\mu R_1 \mu'$. Similarly, as $s' \xrightarrow{a}_{\text{c}} \mu'$ and $s'R_2s''$, it implies by Def. 15 that $s'' \xrightarrow{a}_{\text{c}} \mu''$ with $\mu' R_2 \mu''$. As $\mu R_1 \mu'$ and $\mu' R_2 \mu''$, this implies by Lem. 1 that $\mu R \mu''$. □

Note that for PA, SBS relations coincide with *Segala's simulation* (see Def. 13 on page 18) relations; in that case R in Def. 15 is defined only for $j = 1$. The following proposition trivially follows from the definitions of $\prec_{\text{sb}}$ and $\prec_{\text{pa}}$.

Proposition 2. $\prec_{\text{sb}} = \prec_{\text{pa}}$ *for PA.*

3.1.2 Distribution-based Simulation Relation

Distribution-based simulation relations are preorders on distributions over a state space requiring that whenever a distribution μ' simulates a distribution μ, then μ' can mimic at least the step-wise behaviour of μ. Formally,

Definition 16. (Distribution-based Simulation). $R \subseteq \bigcup_{j \in \{1,2\}} \text{Dist}(S_j) \times \text{Dist}(S_j)$ *is a* distribution-based simulation (DBS) *relation on a PGA $\mathscr{G}$ iff for every $\mu R \mu'$,*

1. $\mu = \bigoplus_{s' \in \text{Supp}(\mu')} \mu_{s'}$ *and* $\forall s' \in \text{Supp}(\mu') : (\mu'(s') = |\mu_{s'}|$ *and* $\mu_{s'\downarrow} R \iota_{s'})$*, and*
2. $\mu \xrightarrow{a} \rho$ *implies* $\mu' \xrightarrow{a}_{\text{c}} \rho'$ *such that* $|\rho| \leq |\rho'|$ *and* $\rho_{\downarrow} R \rho'_{\downarrow}$.

Let $\prec_{\text{db}}$ be the largest DBS relation. We write $s \prec_{\text{db}} s'$ iff $\iota_s \prec_{\text{db}} \iota_{s'}$.

By condition (1), μ splits into sub-distributions as per the support of μ', i.e., for every $s' \in \text{Supp}(\mu')$, there exists a sub-distribution $\mu_{s'}$ of μ such that the conditional distribution of $\mu_{s'}$ is related to $\iota_{s'}$. By condition (2), an a-transition from μ to *some* ρ implies a combined a-transition from μ' to ρ' such that the mass of ρ' is at least that of ρ and their conditional distributions are related.

Example 9. *In Fig. 3.1, $\mu = [\![0.3u_3, 0.3u_4, 0.4u_5]\!] \prec_{\text{db}} \iota_{u_0}$ as $R = \{(\iota_{u_1}, \iota_{u_1}), (\iota_{u_2}, \iota_{u_2}), ([\![0.3u_3, 0.3u_4, 0.4u_5]\!], \iota_{u_0}), ([\![0.5u_1, 0.5u_2]\!], [\![0.5u_1, 0.5u_2]\!])\}$ is a DBS relation. Let us check the conditions of Def. 16 for μ and ι_{u_0}. The condition (1) trivially holds for μ and ι_{u_0}. For the a-transition from μ to $\rho = [\![0.3u_1, 0.3u_2]\!]$, there is an a-transition from ι_{u_0} to $\rho' = [\![0.5u_1, 0.5u_2]\!]$ such that $|\rho| \leq |\rho'|$ and $\rho_{\downarrow} R \rho'$. The same holds for the b-transitions from μ and ι_{u_0}, thus fulfilling condition (2). Note that no SBS relation exists associating u_0 with any other state in Fig. 3.1.*

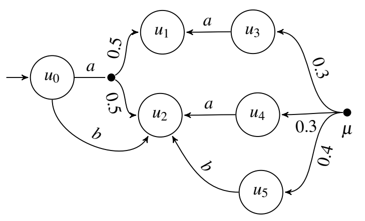


Figure 3.1: $[\![0.5u_3, 0.5u_4]\!] \prec_{\text{db}} \iota_{u_0}$ but $u_i \not\prec_{\text{sb}} u_0$ for $i \in \{3,4\}$.

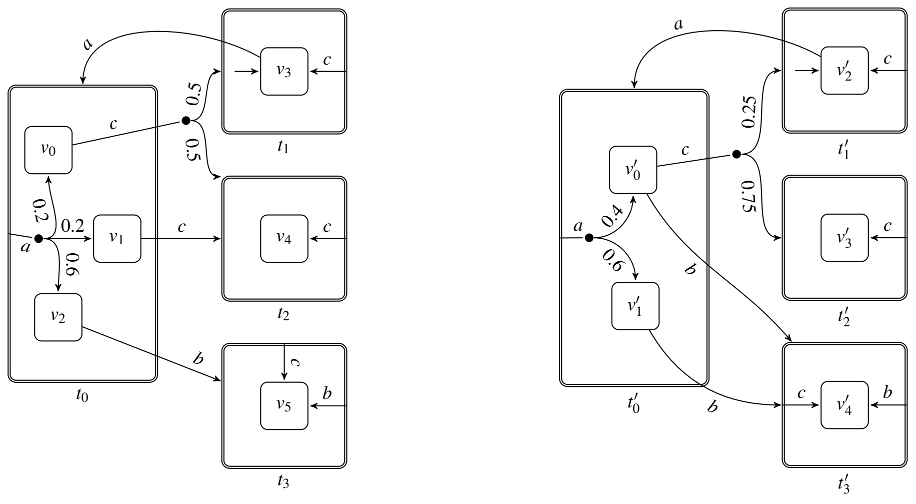
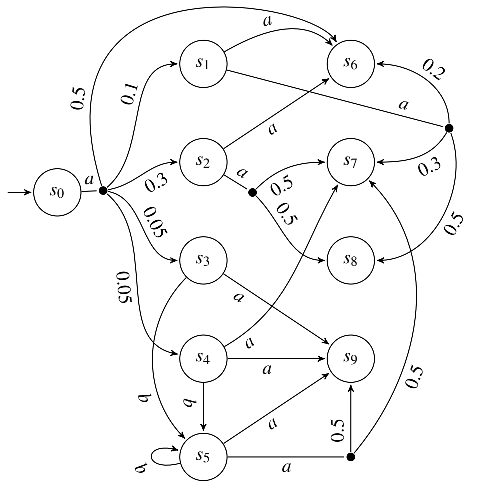


Figure 3.2: $\mathscr{G}$ (left) $\prec_{\text{db}}$ $\mathscr{G}'$ (right).

The following example illustrates a DBS relation between two PGA which are not related by any SBS relation.

Example 10. *Consider PGA $\mathscr{G}$ and $\mathscr{G}'$ in Fig. 3.2. $\mathscr{G} \prec_{\text{db}} \mathscr{G}'$ as $R = \{(\iota_{v_3}, \iota_{v'_2}), (\iota_{v_2}, \iota_{v'_1}), (\iota_{v_5}, \iota_{v'_4}), (\iota_{v_4}, \iota_{v'_3})\}$ $\cup\{([\![0.2v_0, 0.2v_1, 0.6v_2]\!], [\![0.4v'_0, 0.6v'_1]\!]), ([\![0.2v_0, 0.2v_1]\!]_{\downarrow}, \iota_{v'_0}), ([\![0.25t_1, 0.75t_2]\!], [\![0.25t'_1, 0.75t'_2]\!]), ([\![0.25v_3,$ $0.75v_4]\!], [\![0.25v'_2, 0.75v'_3]\!])\} \cup \bigcup_{i=0\ldots3}\{(t_i, t'_i)\}$ is a DBS relation. Let us consider the distributions $\mu =$ $[\![0.2v_0, 0.2v_1, 0.6v_2]\!]$ and $\mu' = [\![0.4v'_0, 0.6v'_1]\!]$, and check the conditions of Def. 16. For $v'_0 \in supp(\mu')$, $[\![0.2v_0, 0.2v_1]\!]$ is the sub-distribution of μ and $[\![0.2v_0, 0.2v_1]\!]_{\downarrow} R \iota_{v'_0}$ holds. Similarly, for $v'_1 \in \text{Supp}(\mu')$, we have $[\![0.6v_2]\!]$ as the sub-distribution of μ and $\iota_{v_2} R \iota_{v'_1}$ holds. Now for the b-transition from μ to $[\![0.6t_3]\!]$, there is a b-transition from μ' to $\iota_{t'_3}$ such that $|[\![0.6t_3]\!]| \leq 1$ and $\iota_{t_3} R \iota_{t'_3}$ hold. Similarly, for the c-transition from μ to $\rho = [\![0.1t_1, 0.3t_2]\!]$, there is a c-transition from μ' to $\rho' = [\![0.1t'_1, 0.3t'_2]\!]$ such that $|\rho| = |\rho'|$ and $\rho R \rho'$ hold , thus, fulfilling the conditions of Def. 16. Note that no SBS relation exists between $\mathscr{G}$ and $\mathscr{G}'$ as v_0 and v_1 are not simulated by any state in $\mathscr{G}'$.*

Unlike SBS relations (see Proposition 1), a DBS relation between two PGA does not imply a DBS relation between their closed versions.



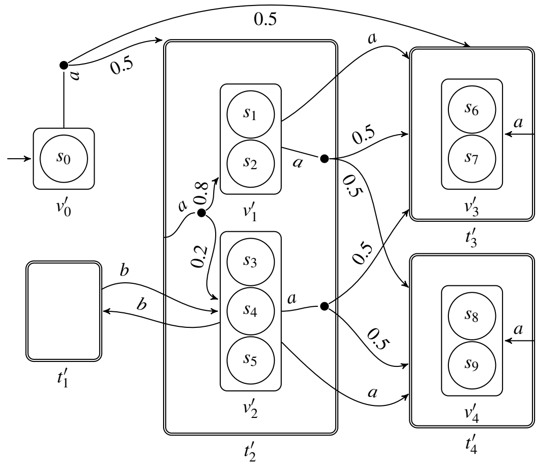


Figure 3.3: For PA $\mathscr{M}$ (left) with $\mathscr{G} = \alpha_{\mathrm{PA}}(\mathscr{M})$ and PGA $\mathscr{G}'$ (right), $\mathscr{G} \prec_{\mathrm{db}} \mathscr{G}'$ and $\tau(\mathscr{G}) \not\prec_{\mathrm{db}} \tau(\mathscr{G}')$.

Proposition 3. *For PGA $\mathscr{G}$ and $\mathscr{G}'$, $\mathscr{G} \prec_{\mathrm{db}} \mathscr{G}'$ does not imply $\tau(\mathscr{G}) \prec_{\mathrm{db}} \tau(\mathscr{G}')$.*

Example 11. *Consider PA $\mathscr{M}$ (left) with $\mathscr{G} = \alpha_{\mathrm{PA}}(\mathscr{M})$ and PGA $\mathscr{G}'$ (right) in Fig. 3.3. Note that $\mathscr{G} \prec_{\mathrm{db}} \mathscr{G}'$ whereas $\tau(\mathscr{G}) \not\prec_{\mathrm{db}} \tau(\mathscr{G}')$. Let us consider the distribution $\mu = [\![0.5v_3, 0.5v_4]\!]$ in PGA $\mathscr{G}$; note that $\mu \prec_{\mathrm{db}} \iota_{v'_2}$ in PGA $\mathscr{G}'$. But in $\tau(\mathscr{G})$, there is a transition $\mu \to [\![0.5t_5, 0.5t_7]\!]$ that is not simulated by any transition from $\iota_{v'_2}$ in $\tau(\mathscr{G}')$.*

Moreover, DBS relations are not comparable to Segala's probabilistic forward simulation relations (see Def. 14 on page 18).

Proposition 4. *$\prec_{\mathrm{db}}$ and $\prec_{\mathrm{paf}}$ are incomparable for PA.*

Example 12. *Consider the distributions ν and μ in Fig. 2.6 and 3.1 respectively. By checking the conditions of Def. 14 and 16, we find that $s_0 \prec_{\mathrm{paf}} [\![0.5s_3, 0.25s_4, 0.25s_5]\!]$ whereas $s_0 \not\prec_{\mathrm{db}} [\![0.5s_3, 0.25s_4, 0.25s_5]\!]$; and $[\![0.3u_3, 0.3u_4, 0.4u_5]\!] \prec_{\mathrm{db}} \iota_{u_0}$ whereas $[\![0.3u_3, 0.3u_4, 0.4u_5]\!] \not\prec_{\mathrm{paf}} \iota_{u_0}$.*

Theorem 2. *$\prec_{\mathrm{db}}$ is a preorder.*

Proof. Reflexivity: It follows trivially from Def. 16.

Transitivity: Let $\mathcal{G} = (S, \{S_1, S_2\}, A, \Delta, s_0)$, $\mathcal{G}' = (S', \{S'_1, S'_2\}, A, \Delta', s'_0)$ and $\mathcal{G}'' = (S'', \{S''_1, S''_2\}, A, \Delta'', s''_0)$ be PGA. Let $\mathcal{G} \prec_{\text{db}} \mathcal{G}'$ and $\mathcal{G}' \prec_{\text{db}} \mathcal{G}''$. We prove that $\mathcal{G} \prec_{\text{db}} \mathcal{G}''$ holds. Let R_1 be a DBS relation between $\mathcal{G}$ and $\mathcal{G}'$, and R_2 between $\mathcal{G}'$ and $\mathcal{G}''$. We define the relation $R \subseteq (\text{Dist}(S_1) \times \text{Dist}(S''_1)) \cup (\text{Dist}(S_2) \times \text{Dist}(S''_2))$ as:

$$R = \{(\mu, \mu'') \mid \exists \mu' \in S' : \mu R_1 \mu' \text{ and } \mu' R_2 \mu''\}$$

and show that it fulfils the conditions of Def. 16.

Assume $\mu R \mu''$ where $\mu R_1 \mu'$ and $\mu' R_2 \mu''$ for some $\mu' \in \text{Dist}(S')$.

1. As R_2 is a DBS relation and $\mu' R_2 \mu''$, μ' can be split into sub-distributions according to the support of μ'', i.e., $\mu' = \bigoplus_{s'' \in \text{Supp}(\mu'')} \mu'_{s''}$ such that $|\mu'_{s''}| = \mu''(s'')$ and $\mu'_{s''\downarrow} R_2 \iota_{s''}$ for all $s'' \in \text{Supp}(\mu'')$. Similarly, as R_1 is a DBS relation and $\mu R_1 \mu'$, we have $\mu = \bigoplus_{s' \in \text{Supp}(\mu')} \mu_{s'}$ such that $|\mu_{s'}| = \mu'(s')$ and $\mu_{s'\downarrow} R_1 \iota_{s'}$ for all $s' \in \text{Supp}(\mu')$.

   Let $s'' \in \text{Supp}(\mu'')$ such that $\mu'_{s''}$ is its corresponding sub-distribution of μ'. As each state in the support of $\mu'_{s''}$ has a corresponding sub-distribution in μ, we can create a sub-distribution of μ that corresponds to $\mu'_{s''}$ and subsequently to s'', i.e., $\mu_{s''} = \bigoplus_{u' \in \text{Supp}(\mu'_{s''})} \mu'_{s''}(u') \cdot (\mu_{u'\downarrow})$. This shows that μ can be split into sub-distributions according to the support of μ''.

   Let $s'' \in \text{Supp}(\mu'')$, then $\mu'_{s''}$ is a sub-distribution for s'' in μ' which have a corresponding sub-distribution $\mu_{s''}$ in μ. Thus, $\mu''(s'') = |\mu'_{s''}| = |\mu_{s''}|$. Now as $\mu' R_2 \mu''$, by Def. 16 $\mu'_{s''\downarrow} R_2 \iota_{s''}$; and as $\mu R_1 \mu'$ and by Def. 16 $\mu_{u'\downarrow} R_1 \iota_{u'}$ for every $u' \in \text{Supp}(\mu'_{s''}) \subseteq \text{Supp}(\mu')$, therefore $\mu_{s''\downarrow} R_1 \mu'_{s''\downarrow}$. Thus, $\mu_{s''\downarrow} R \iota_{s''}$.

2. Let $\mu \xrightarrow{a} \nu$. As $\mu R_1 \mu'$, by Def. 16, $\mu' \xrightarrow{a}_{\text{c}} \nu'$ such that $|\nu'| \geq |\nu|$ and $\nu_\downarrow R_1 \nu'_\downarrow$. Similarly, as $\mu' \xrightarrow{a}_{\text{c}} \nu'$ and $\mu' R_2 \mu''$, it implies by Def. 16 that $\mu'' \xrightarrow{a}_{\text{c}} \nu''$ with $|\nu''| \geq |\nu'|$ and $\nu'_\downarrow R_2 \nu''_\downarrow$. As $|\nu| \leq |\nu'|$ and $|\nu'| \leq |\nu''|$, thus $|\nu| \leq |\nu''|$; and as $\nu_\downarrow R_1 \nu'_\downarrow$ and $\nu'_\downarrow R_2 \nu''_\downarrow$, thus $\nu_\downarrow R \nu''_\downarrow$.

□

3.1.3 State-based vs. Distribution-based Simulation Relations

Although state-based simulation relations can be lifted from states to distributions over states (by Def. 12), $\prec_{\text{sb}}$ (lifted to distributions over states) and $\prec_{\text{db}}$ are not comparable in general. However, for closed PGA, $\prec_{\text{sb}}$ is a subset of $\prec_{\text{db}}$.

Proposition 5. *$\prec_{\text{sb}}$ and $\prec_{\text{db}}$ are incomparable for PGA; and $\prec_{\text{sb}} \subseteq \prec_{\text{db}}$ for closed PGA.*

Example 9 illustrates that DBS relations do not imply SBS relations. Similarly, SBS relations do not imply DBS relations, as illustrated by the following example.

Example 13. *$R = \bigcup_{i=0\ldots2}\{(t_i, t'_i)\} \cup \bigcup_{i=0\ldots3}\{(v_i, v'_i)\}$ is an SBS relation between PGA $\mathcal{G}$ and $\mathcal{G}'$ in Fig. 3.4. Note that a DBS relation cannot be defined between $\mathcal{G}$ and $\mathcal{G}'$. This is because the a-transition*

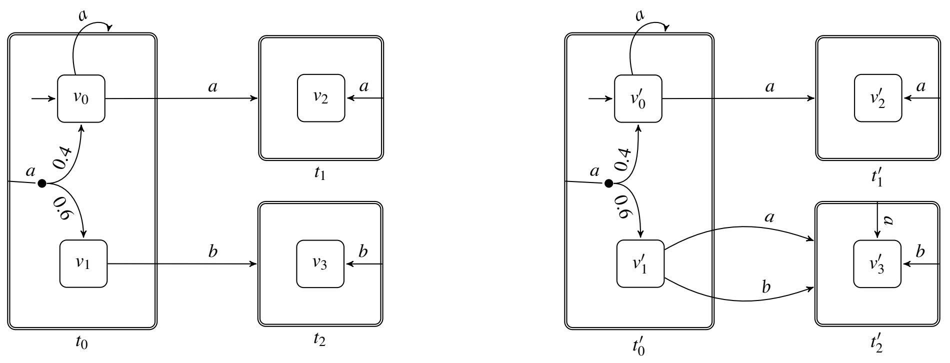


Figure 3.4: For PGA $\mathscr{G}$ (left) and $\mathscr{G}'$ (right), $\mathscr{G} \prec_{\text{sb}} \mathscr{G}'$ but $\mathscr{G} \not\prec_{\text{db}} \mathscr{G}'$.

from distribution $[\![0.4v_0, 0.6v_1]\!]$ *to* $[\![0.4t_1]\!]$ *is not simulated by the a-transition from* $[\![0.4v'_0, 0.6v'_1]\!]$ *to* $[\![0.4t'_1, 0.6t'_2]\!]$.

3.2 Alternating Simulation Relations

To compare two-player stochastic games with competitive objectives (e.g., if player one maximises the probability to reach a certain goal state, her opponent (player two) will try to minimize this quantity), we use *alternating* simulation relations.

3.2.1 State-based Alternating Simulation Relation

Like state-based simulation relations, state-based *alternating* simulation relations are intended to be preorders on a state space requiring that whenever a state s' simulates a state s, then if $s' \in S_2$, s' can mimic at least the step-wise behaviour of s; otherwise, if $s' \in S_1$, s can mimic at least the step-wise behaviour of s'. Our state-based alternating simulation relations are inspired by the notions of alternating simulation of [AHKV98] and strong probabilistic game simulation of [Kat10].

Definition 17. (State-based Alternating Simulation). $R \subseteq \bigcup_{j\in\{1,2\}} S_j \times S_j$ *is a* state-based alternating simulation (SBAS) *relation for a PGA* $\mathscr{G}$ *iff for every* sRs' *the following holds:*

1. *if* $s, s' \in S_1$, *then* $s' \xrightarrow{a} \mu'$ *implies* $s \xrightarrow{a}_{\text{c}} \mu$ *such that* $\mu R \mu'$,
2. *if* $s, s' \in S_2$, *then* $s \xrightarrow{a} \mu$ *implies* $s' \xrightarrow{a}_{\text{c}} \mu'$ *such that* $\mu R \mu'$.

Let $\preccurlyeq_{\text{sb}}$ *be the largest SBAS relation. We write "*s' A-simulates s*" iff* $s \preccurlyeq_{\text{sb}} s'$.

Intuitively, in case of player-one states, the behaviour of s' is mimicked by that of s; whereas in case of player-two states, it is the other way round. The condition (1) asserts that if $s, s' \in S_1$, then an a-transition from s' implies a combined a-transition from s and the resulting distributions are related (by Def. 12) w.r.t. R. The condition (2) asserts that if $s, s' \in S_2$, the similar conditions as in (1) hold for every transition from s.

Proposition 6. *For PGA $\mathcal{G}$ and $\mathcal{G}'$, $\mathcal{G} \preccurlyeq_{\mathrm{sb}} \mathcal{G}'$ implies $\tau(\mathcal{G}) \preccurlyeq_{\mathrm{sb}} \tau(\mathcal{G}')$.*

Theorem 3. *$\preccurlyeq_{\mathrm{sb}}$ is a preorder.*

Proof. The proof of the above theorem follows the similar lines as that of Th. 1. □

Remark 1. *The strong probabilistic game simulation relation in [Kat10, Def. 6.10] is obtained by lifting Def. 15 and 17 to player-two states (by considering hyper-transitions from player-one states) and merging their conditions.*

Note that for PA, SBAS relations coincide with SBS relations — if $S_1 = \emptyset$, then $R \subseteq S_2 \times S_2$ and condition (1) in Def. 17 becomes irrelevant. The following proposition follows directly from the definitions of $\prec_{\mathrm{sb}}$ and $\preccurlyeq_{\mathrm{sb}}$.

Proposition 7. *$\prec_{\mathrm{sb}} = \preccurlyeq_{\mathrm{sb}}$ for PA.*

3.2.2 Distribution-based Alternating Simulation Relation

Like distribution-based simulation relations, distribution-based alternating simulation relations are intended to be preorders on distributions over a state space requiring that whenever a distribution μ' simulates a distribution μ, then if $\mu' \in \mathrm{Dist}(S_2)$, μ' can mimic at least the step-wise behaviour of μ; otherwise, if $\mu' \in \mathrm{Dist}(S_1)$, μ can mimic at least the step-wise behaviour of μ'. Formally,

Definition 18. (Distribution-based Alternating Simulation). *$R \subseteq \bigcup_{j\in\{1,2\}} \mathrm{Dist}(S_j) \times \mathrm{Dist}(S_j)$ is a* distribution-based alternating simulation (DBAS) *relation for a PGA $\mathcal{G}$ iff for every $\mu R \mu'$:*

1. *$\mu = \bigoplus_{s' \in \mathrm{Supp}(\mu')} \mu_{s'}$ and $\forall s' \in \mathrm{Supp}(\mu') : (\mu'(s') = |\mu_{s'}|$ and $\mu_{s'\downarrow} R \iota_{s'})$,*

2. *if $\mu, \mu' \in \mathrm{Dist}(S_1)$, $\mu' \xrightarrow{a} \rho'$ implies $\mu \xrightarrow{a}_{\mathrm{c}} \rho$ such that $|\rho| \geq |\rho'|$ and $\rho_{\downarrow} R \rho'_{\downarrow}$,*

3. *if $\mu, \mu' \in \mathrm{Dist}(S_2)$, $\mu \xrightarrow{a} \rho$ implies $\mu' \xrightarrow{a}_{\mathrm{c}} \rho'$ such that $|\rho| \leq |\rho'|$ and $\rho_{\downarrow} R \rho'_{\downarrow}$.*

Let $\preccurlyeq_{\mathrm{db}}$ be the largest DBAS relation. We write “μ' A-simulates *μ” iff $\mu \preccurlyeq_{\mathrm{db}} \mu'$.*

The condition (1) is the same as in Def. 16. By condition (2), if $\mu, \mu' \in \mathrm{Dist}(S_1)$, then an a-transition from μ' to *some* ρ' implies a combined a-transition from μ to ρ such that the mass of ρ is at least that of ρ' and their conditional distributions are related. And by condition (3), if $\mu, \mu' \in \mathrm{Dist}(S_2)$, the similar conditions as in (2) hold for every transition from μ.

Theorem 4. *$\preccurlyeq_{\mathrm{db}}$ is a preorder.*

Proof. The proof of the above theorem follows the similar lines as that of Th. 2. □

Proposition 8. *For PGA $\mathcal{G}$ and $\mathcal{G}'$, $\mathcal{G} \preccurlyeq_{\mathrm{db}} \mathcal{G}'$ does not imply $\tau(\mathcal{G}) \preccurlyeq_{\mathrm{db}} \tau(\mathcal{G}')$.*

Moreover, like state-based relations (see Proposition 7), DBS relations coincide with DBAS relations for PA. The following proposition follows directly from the definitions of $\prec_{\mathrm{db}}$ and $\preccurlyeq_{\mathrm{db}}$.

Proposition 9. *$\prec_{\mathrm{db}} = \preccurlyeq_{\mathrm{db}}$ for PA.*

3.2.3 State-based vs. Distribution-based Alternating Relations

Like simulation relations, alternating simulation relations are not comparable in general. However, for closed PGA, $\preccurlyeq_{\mathrm{sb}}$ is a subset of $\preccurlyeq_{\mathrm{db}}$.

Proposition 10. *$\preccurlyeq_{\mathrm{sb}}$ and $\preccurlyeq_{\mathrm{db}}$ are incomparable in general; and $\preccurlyeq_{\mathrm{sb}} \subseteq \preccurlyeq_{\mathrm{db}}$ for closed PGA.*

3.3 Reachability Probabilities and (Alternating) Simulation Relations

In this section, we discuss that simulation/alternating simulation relations between closed PGA provide bounds on their reachability probabilities when players collaborate/compete with each other. In fact, simulation relations between PGA bound their $\max^{\blacktriangle}$ and $\min^{\blacktriangledown}$ values; whereas alternating simulation relations bound their $\max^{\blacktriangledown}$ and $\min^{\blacktriangle}$ values.

Theorem 5. *For $x \in \{\mathrm{sb}, \mathrm{db}\}$, and PGA $\mathcal{G}$ and $\mathcal{G}'$:*

- *Let* $\mathcal{G} \prec_x \mathcal{G}'$, $T \subseteq S$ *with* $T' = \{s' \in S' \mid \exists \mu \in \mathrm{Dist}(S) : \mu(T) > 0$ *and* $\mu \prec_x s'\}$, *then* $\min^{\blacktriangledown}(T') \leq \min^{\blacktriangledown}(T)$ *and* $\max^{\blacktriangle}(T) \leq \max^{\blacktriangle}(T')$.
- *Let* $\mathcal{G} \preccurlyeq_x \mathcal{G}'$, $T \subseteq S$ *with* $T'' = \{s' \in S' \mid \exists \mu \in \mathrm{Dist}(S) : \mu(T) > 0$ *and* $\mu \preccurlyeq_x s'\}$, *then* $\min^{\blacktriangle}(T) \leq \min^{\blacktriangle}(T'')$ *and* $\max^{\blacktriangledown}(T'') \leq \max^{\blacktriangledown}(T)$.

Proof. We only prove these two claims for distribution-based relations, as the proof for state-based relations is similar. Moreover, we give a proof for $\min^{\blacktriangle}(T) \leq \min^{\blacktriangle}(T'')$, as the proof for the other cases (e.g., $\max^{\blacktriangledown}(T'') \leq \max^{\blacktriangledown}(T)$) is similar to this.

For PGA $\mathcal{G}$ and $\mathcal{G}'$, let $\mathcal{G} \preccurlyeq_{\mathrm{db}} \mathcal{G}'$. Let $T \subseteq S$ be a set of goal states such that $T'' = \{s' \in S' \mid \exists \mu \in \mathrm{Dist}(S) : \mu(T) > 0$ and $\mu \preccurlyeq_{\mathrm{db}} s'\}$. Let $\tilde{\mathcal{G}}$ be the closed version of the disjoint union of $\mathcal{G}$ and $\mathcal{G}'$ such that $\tilde{T} = T \cup T''$.

Let $\mathbf{1} = \min$ and $\mathbf{2} = \max$ be the objectives of players one and two respectively, such that $\mathrm{Prt}^{\min}_{\max} : W \to W$ (see Def. 9) is a probability valuation transformer function for $\tilde{\mathcal{G}}$.

We prove that $\iota_{s_0} \preccurlyeq_{\mathrm{db}} \iota_{s'_0}$ implies $\min^{\blacktriangle}(T) \leq \min^{\blacktriangle}(T'')$. The proof is by induction on ordered functions in W.

Base case: For $w_0 = \mathrm{Prt}^{\min}_{\max}(\bot)$, we have $w_0(u) = 1$ if $u \in \tilde{T}$ and $w_0(u) = 0$ otherwise.

For $\mu, \eta \in \mathrm{Dist}(\tilde{S})$ such that $\mu \preccurlyeq_{\mathrm{db}} \eta$, we show $w_0(\mu) \leq w_0(\eta)$. Let $s \in \mathrm{Supp}(\eta)$, then μ_s is a sub-distribution of μ for s; and $\mu_{s\downarrow} \preccurlyeq_{\mathrm{db}} \iota_s$ (Def. 18). Let $\mu_{s\downarrow}(T) > 0$, then $s \in T''$, $w_0(\mu_{s\downarrow}) = \sum_{v \in T} \mu_{s\downarrow}(v) \cdot w_0(v) \leq 1$ and $w_0(s) = 1$. Thus, $\sum_{s \in \mathrm{Supp}(\eta)} \eta(s) \cdot w_0(\mu_{s\downarrow}) = w_0(\mu) \leq w_0(\eta) = \sum_{s \in \mathrm{Supp}(\eta)} w_0(s)$.

Induction hypothesis: For $w_n = (\mathrm{Prt}^{\min}_{\max})^n(\bot)$, let $w_n(\nu) \leq w_n(\rho)$ for $n \geq 0$ if $\nu \preccurlyeq_{\mathrm{db}} \rho$ for all $\nu, \rho \in \mathrm{Dist}(\tilde{S})$. We show that it also holds for $n+1$, i.e., $w_{n+1}(\nu) \leq w_{n+1}(\rho)$.

For $\mu, \eta \in \mathrm{Dist}(\tilde{S})$ such that $\mu \preccurlyeq_{\mathrm{db}} \eta$, there are two cases:

1. $\mu, \eta \in \mathrm{Dist}(\tilde{S}_1)$: There are two cases:
   - Let $\eta \to \eta'$, then by Def. 18, $\mu \to_{\mathrm{c}} \mu'$ such that $\mu' \preccurlyeq_{\mathrm{db}} \eta'$. By induction hypothesis $w_n(\mu') \leq w_n(\eta')$. As $\mathbf{1} = \min$, therefore $w_{n+1}(\mu) = \min_{\mu \to_{\mathrm{c}} \mu'} w_n(\mu') \leq \min_{\eta \to \eta'} w_n(\eta') = w_{n+1}(\eta)$.
   - Let $\mathrm{Act}(s) = \emptyset$ for all $s \in \mathrm{Supp}(\eta)$. By Def. 9 $w_{n+1}(s) = 1$. Thus, $w_{n+1}(\mu) \leq w_{n+1}(\eta)$.
2. $\mu, \eta \in \mathrm{Dist}(\tilde{S}_2)$: There are two cases:
   - Let $\mu \to \mu'$, then by Def. 18, $\eta \to_{\mathrm{c}} \eta'$ such that $\mu' \preccurlyeq_{\mathrm{db}} \eta'$. By induction hypothesis $w_n(\mu') \leq w_n(\eta')$. As $\mathbf{2} = \max$, therefore $w_{n+1}(\mu) = \max_{\mu \to \mu'} w_n(\mu') \leq \max_{\eta \to_{\mathrm{c}} \eta'} w_n(\eta') = w_{n+1}(\eta)$.
   - Let $\mathrm{Act}(s) = \emptyset$ for all $s \in \mathrm{Supp}(\mu)$. By Def. 9 $w_{n+1}(s) = 0$. Thus, $w_{n+1}(\mu) \leq w_{n+1}(\eta)$.

As $\mathrm{Prt}^{\mathbf{1}}_{\mathbf{2}}$ is a monotonic function over W, by Tarski's theorem [T+55],it has a least and a greatest fixpoint, i.e., $w = \mathbf{Fix}\ \mathrm{Prt}^{\min}_{\max}(\bot)$. Moreover, as $\mathcal{G} \preccurlyeq_{\mathrm{db}} \mathcal{G}'$, therefore $\iota_{s_0} \preccurlyeq_{\mathrm{db}} \iota_{s'_0}$ and from above $w(\iota_{s_0}) \leq w(\iota_{s'_0})$. Hence, $\min^{\blacktriangle}(T) \leq \min^{\blacktriangle}(T'')$ holds.

□

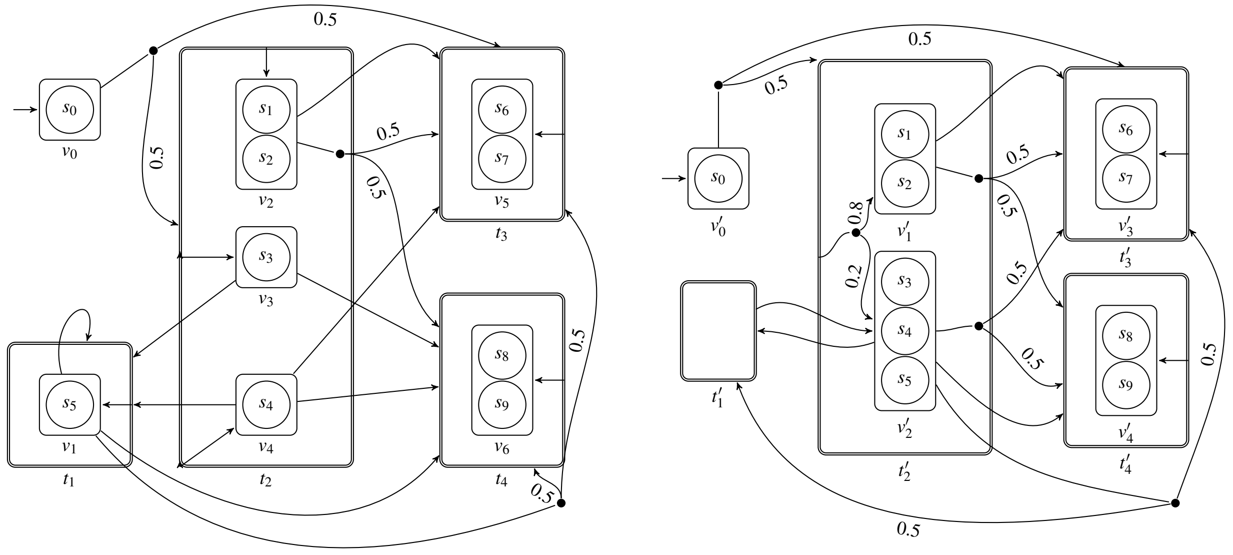


Figure 3.5: For PGA $\mathscr{G}$ (left) and $\mathscr{G}'$ (right). $\mathscr{G} \prec_{\text{db}} \mathscr{G}'$ and $\mathscr{G} \preccurlyeq_{\text{db}} \mathscr{G}'$.

Example 14. *Consider PGA $\mathscr{G}$ and $\mathscr{G}'$ in Fig. 3.5, where $\mathscr{G} \prec_{\text{db}} \mathscr{G}'$ and $\mathscr{G} \preccurlyeq_{\text{db}} \mathscr{G}'$. Let $T = \{t_4\}$ such that $T' = T'' = \{t'_4\}$ for Th. 5. Then the maximum probability in PGA $\mathscr{G}$ to T lies in $[0.25, 0.5]$, whereas in $\mathscr{G}'$ it lies in $[0.3, 0.3]$ for T'.*

3.4 Summary and Discussion

In this chapter, we defined two (alternating) simulation preorders for PGA: a state-based as well as a distribution-based one. Simulation relations are of interest when both players have identical objectives, whereas alternating simulation relations are useful for competitive objectives; moreover, simulation and alternating simulation relations coincide for models — like Segala's PA — in which state spaces are not partitioned. We showed that state-based and distribution-based relations are incomparable in general but for closed PGA the former implies the latter. Moreover, state-based relations between PGA imply state-based relations between their closed versions. A similar result for distribution-based relations does not hold. Finally, both state-based and distribution-based relations preserve reachability probabilities. An overview of the results of this chapter is given in Table 3.1.

Related work: The concept of *distribution-based* relation (probabilistic forward simulation) was first introduced in [Seg95, LSV07] for PA. In [EHZ10], distribution-based weak bisimulation has been defined for *Markov automata* — PA with exponentially distributed delays in states. In [DHR08], distribution-based bisimulation was studied in the context of language equivalence of Rabin's deterministic PA; this was extended to the non-deterministic case in [FZ14]. More recently, in [HKK14] bisimulation for probabilistic systems with uncountable state and action spaces has been defined, which extends [FZ14].

The notion of state-based probabilistic alternating simulation has been discussed in [ZP10, Kat10], which

Simulation relations

	State-based ($\prec_{\text{sb}}$)	Distribution-based ($\prec_{\text{db}}$)
preorder	+	+
monotonicity of closing (τ)	+	-
For Segala's PA	$\prec_{\text{sb}} = \prec_{\text{pa}}$	$\prec_{\text{db}} \neq \prec_{\text{paf}}$
For PGA	$\mathscr{G} \prec_{\text{sb}} \mathscr{G}' \Rightarrow \tau(\mathscr{G}) \prec_{\text{sb}} \tau(\mathscr{G}')$	$\mathscr{G} \prec_{\text{db}} \mathscr{G}' \not\Rightarrow \tau(\mathscr{G}) \prec_{\text{db}} \tau(\mathscr{G}')$
For PGA	$\prec_{\text{sb}} \neq \prec_{\text{db}}$	
For closed PGA	$\prec_{\text{sb}} \subseteq \prec_{\text{db}}$	

Alternating simulation relations

	State-based ($\preccurlyeq_{\text{sb}}$)	Distribution-based ($\preccurlyeq_{\text{db}}$)
preorder	+	+
monotonicity of closing (τ)	+	-
For Segala's PA	$\preccurlyeq_{\text{sb}} = \prec_{\text{sb}} = \prec_{\text{pa}}$	$\preccurlyeq_{\text{sb}} = \preccurlyeq_{\text{db}}$
For PGA	$\mathscr{G} \preccurlyeq_{\text{sb}} \mathscr{G}' \Rightarrow \tau(\mathscr{G}) \preccurlyeq_{\text{sb}} \tau(\mathscr{G}')$	$\mathscr{G} \preccurlyeq_{\text{db}} \mathscr{G}' \not\Rightarrow \tau(\mathscr{G}) \preccurlyeq_{\text{db}} \tau(\mathscr{G}')$
For PGA	$\preccurlyeq_{\text{sb}} \neq \preccurlyeq_{\text{db}}$	
For closed PGA	$\preccurlyeq_{\text{sb}} \subseteq \preccurlyeq_{\text{db}}$	

Preservation of reachability probabilities

$\prec_{\text{sb}}$ and $\prec_{\text{db}}$ yield	$\min^{\blacktriangledown}$ and $\max^{\blacktriangle}$
$\preccurlyeq_{\text{sb}}$ and $\preccurlyeq_{\text{db}}$ yield	$\min^{\blacktriangle}$ and $\max^{\blacktriangledown}$

Table 3.1: Summary of (alternating) simulation relations.

can be obtained by merging our Def. 15 and 17 and lifting them to player-two states. In [ZP10], probabilistic forward simulation ([Seg95, LSV07]) was extended to give alternating simulation relations for games. This notion, however, is not comparable to our notion of distribution-based alternating simulation relation.

Future extensions: In the literature, many algorithms have been proposed to check state-based relations for probabilistic systems; however, very little has been done for distribution-based relations. To build upon this work, one can consider:

- adapting the algorithm in [HKK14] — that checks distribution-based bisimulation relation for probabilistic systems — for (alternating) simulation relations between stochastic games,
- defining weak variants of (alternating) simulation relations with and without preserving branching structures of models, and
- logical characterization of (alternating) simulation relations.

In the next chapter, we introduce a new modeling formalism for stochastic games that extend PGA with *required* and *possible* modalities, called abstract PGA (APGA). We equip APGA with notions of refinements; define a *composition* operator for it; and show that refinement relations are pre-congruences w.r.t. composition preserving reachability probabilities.

4 Stochastic Games with Modalites

Many researchers have introduced modeling formalisms that over- and under-approximate the behaviour of probabilistic systems with different transition functions. In [KKLW12, KKN09], extensions of Markov decision processes (MDPs) and interactive Markov chains (IMCs) are given that annotate transitions with intervals of probabilities instead of single values, thus over- and under-approximating the probabilities to target states. In contrast, abstract probabilistic automata (APA) [DKL$^+$13] group transitions into *possible* and *required* sets — for over- and under-approximating the behaviour respectively — as in *modal transition systems* (MTS) [LT88a], and model distributions by *constraint functions* (mathematical constraints that describe sets of probability distributions (see page 9)) as in constraint Markov chains (CMC) [CDL$^+$11]. To the best of our knowledge, no formalism, that over- and under-approximate the behaviour, has been proposed for probabilistic systems with competing/collaborating stakeholders — such systems are usually modelled as games with probabilistic transitions.

Two-player *stochastic games* (SGs) have been introduced in [Sha53, Con92] as mathematical models for the modeling and analysis of non-deterministic probabilistic systems with competing/collaborating players. In SGs, moves of one of the players yield distributions over states, while moves of the other player just yield states. SGs are generalized to *probabilistic game automata* (PGA) [CL88] that allow both players to make non-deterministic and probabilistic choices at their turns — in SGs one of the players have only non-deterministic choices at her turn.

Inspired by our earlier work on probabilistic automata (PA) [DKL$^+$13], we present a *three-valued* extension of PGA by extending their transitions in two ways. We annotate transitions with *required* and *possible* modalities, and give their targets as sets of probability distributions represented by constraint functions. This yields *abstract probabilistic game automata* (APGA). APGA have as semantics sets of PGA, namely all PGA that have at least all *required* transitions along with their target distributions and zero or more *possible* transitions with at least one of their target distributions. We call these games *implementations* of APGA.

For comparing APGA, we provide a *refinement relation* that implies the inclusion of sets of implementations of APGA. Refinement relations allow for the comparison of APGA at different levels of abstraction, and are important for the step-wise design of system models. Similarly, we present a *satisfaction relation* — a special instance of a refinement relation — for deciding whether PGA are implementations of APGA. As in [DHR08, EHZ10, HKK14, SK14], we treat probability distributions rather than states as first-class citizens and relax *state-based* refinement to *distribution-based* refinement. We show that state-based and distribution-based refinement relations are incomparable in general, but for closed APGA the earlier implies the latter.

We also show that refinement of APGA implies (*alternating*) *simulation* relations [SK14] between their implementations — alternating relations compare the behaviour of PGA in case of competing players, whereas simulation relations are relevant for collaborating players.

Moreover, as APGA may have infinitely many possible implementations, we extend the technique in [SK12] for approximating them such that a finite number of their implementations, called *extreme probabilistic game automata* (EPGA), are sufficient for their extremal reachability analysis, i.e., the extremal reachability probabilities of implementations of APGA are bounded by that of their EPGA.

Finally, we define a *composition* operator for a class of APGA that act as abstract models of PA, and show that our refinement relations are pre-congruences w.r.t. it, thus facilitating APGA-based compositional abstraction of PA (see Ch. 5).

Put in a nutshell, the major contributions of this chapter are:

- a three-valued extension of PGA (called APGA),
- a state-/distribution-based refinement relation for APGA that implies (alternating) simulation between their sets of implementations,
- a finite approximation of implementation sets of APGA,
- results showing that the extremal reachability probabilities of implementations of APGA are bounded by that of their extremal implementations, and
- a *composition* operator for APGA showing refinement relations are pre-congruences w.r.t. it.

Another motivation for proposing a new formalism is the *theory of abstraction* of probabilistic systems. In [SK14], PGA-based abstractions of PA are proposed that yield tighter bounds on probabilistic reachability than SG-based abstractions [KKNP10]; whereas in [SK12], *three-valued* abstractions of PA are proposed as APA that also yield bounds on probabilities. It would be of interest to propose APGA-based abstractions of PA by combining the techniques of [SK14] and [SK12] and do reachability analysis.

4.1 Modal Transition Systems (MTS)

We start with a formalism that first introduced the notion of modalities, i.e., modal transition systems (MTS) [LT88a]. MTS are proposed as *modal* abstractions (i.e. *three-valued* abstractions) of labelled-transition systems (LTS). In the literature, MTS are given as pairs of LTS: one LTS over-approximates the behaviour, whereas the other under-approximates the behaviour of a system. Alternatively, the transitions of MTS are categorized into *required* and *possible* sets, where the *required* transitions must be present whereas the *possible* transitions may be present in every implementation of MTS. (For details about MTS, we refer to [HJS01, LT88b]). Formally,

Definition 19. (Modal Transition Systems). *A* Modal Transition System (MTS) *is a tuple* $\mathscr{Q} = (S, A, \Delta_r, \Delta_p, s_0)$ *where S is a non-empty countable set of states with initial state* $s_0 \in S$*,* $A \subseteq \text{UAct}$*,* $\Delta_r \subseteq S \times A \times S$ *is a set of* required *transitions and* $\Delta_p \subseteq S \times A \times S$ *is a set of* possible *transitions with* $\Delta_r \subseteq \Delta_p$.

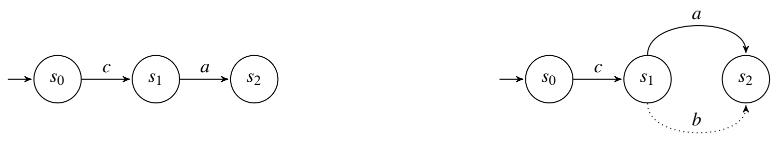


Figure 4.1: The LTS $\mathscr{T}$ (left) is an implementations of the MTS $\mathscr{Q}$ (right).

In the sequel, $\mathscr{Q} = (S, A, \Delta_\mathrm{r}, \Delta_\mathrm{p}, s_0)$ is an MTS. Note that $\mathscr{Q}$ is an LTS when $\Delta_r = \Delta_p$. For simplicity we depict *required* transitions by solid lines, and *possible* transitions by dotted lines, and adopt this convention for all models having the notion of modalities.

Example 15. *Consider the MTS $\mathscr{Q}$ in Fig. 4.1 (right) in which transitions labelled with c and a are* required *transitions whereas the transition labelled with b is a* possible *transition. The LTS $\mathscr{T}$ in Fig. 4.1 (left) is an implementation of $\mathscr{Q}$ in which the* possible *b-transition is not implemented.*

4.2 Abstract Probabilistic Game Automata (APGA)

In this section, we introduce *abstract probabilistic game automata* (APGA) that can be used to abstractly specify PGA. Informally, APGA extend PGA with the notion of modalities and constraint functions. Therefore, in APGA, the transitions are classified into *required* and *possible* sets as in MTS, and the targets of transitions are given as *constraint functions* as in constraint Markov chains (CMC) [CDL$^+$11] — DTMC with sets of probability distributions, represented by constraint functions, for each state — representing (possibly infinite) sets of distributions over states. Formally,

Definition 20. (Abstract Probabilistic Game Automata). *An* Abstract Probabilistic Game Automaton (APGA) *is a tuple $\mathscr{H} = (S, \{S_1, S_2\}, A, \Delta_\mathrm{r}, \Delta_\mathrm{p}, s_0)$ with S, S_1, S_2, A, and s_0 as in PGA, $\Delta_\mathrm{p} \subseteq S_{1+x} \times A \times \mathrm{CFunc}(S_{2-x})$ is a set of* possible *transitions and $\Delta_\mathrm{r} \subseteq S_{1+x} \times A \times \mathrm{CFunc}(S_{2-x})$ is a set of* required *transitions with $\Delta_\mathrm{r} \subseteq \Delta_\mathrm{p}$, where x is a bit.*

We denote $(s, a, \varphi) \in \Delta_\mathrm{p}$ by $s \xrightarrow{a}_\mathrm{p} \varphi$, and $(s, a, \varphi) \in \Delta_\mathrm{r}$ by $s \xrightarrow{a}_\mathrm{r} \varphi$. Note that $s \xrightarrow{a}_\mathrm{p} \varphi$ represents a (possibly infinite) set of a-transitions from s whose target distributions are in $sat(\varphi)$ (that may be an uncountably large set). Thus, the implementations of an APGA can be infinitely branching. Like MTS, the *required* transitions of APGA must be present (in some way) in their implementations, whereas *possible* transitions may or may not be present. However, the implementations must derive their transitions from APGA. Note that PGA are APGA with $\Delta_\mathrm{r} = \Delta_\mathrm{p}$ and $\forall s \in S, a \in A : s \xrightarrow{a} \varphi$ implies $|sat(\varphi)| = 1$. Thus, games, stochastic games, (simple) probabilistic game automata and three-valued abstract games [dAGJ04] are all sub-models of APGA.

Example 16. *Fig. 4.2 represents an APGA. Note that the state s'_1 has one* required *a-transition $s'_1 \xrightarrow{a}_\mathrm{r} \iota_{t'_3}$ and one* possible *a-transition $s'_1 \xrightarrow{a}_\mathrm{p} \varphi_x$ with $sat(\varphi_x) = \{[\![0.5t'_2, 0.5t'_3]\!], [\![0.8t'_2, 0.2t'_3]\!]\}$. In fact, the* possible *a-transition represents two a-transitions with $[\![0.5t'_2, 0.5t'_3]\!]$ and $[\![0.8t'_2, 0.2t'_3]\!]$ as target distributions.*

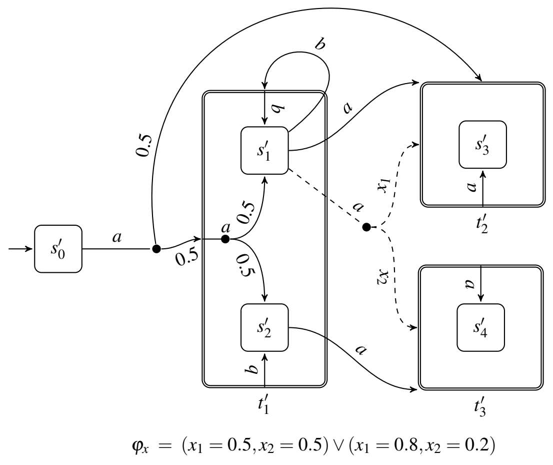


Figure 4.2: An APGA $\mathscr{H}'$

Abstract Probabilistic Automata (APA): We observe that APA [DKL$^+$13, DKL$^+$11], that extend PA with the notion of modalities and constraint functions, can be represented as APGA. Formally, APA are defined as:

Definition 21. (Abstract Probabilistic Automata). *An* Abstract Probabilistic Automaton (APA) *is a tuple* $\mathscr{N} = (S, A, \Delta_\mathrm{r}, \Delta_\mathrm{p}, s_0)$ *with S, A, and* s_0 *as before in PA,* $\Delta_\mathrm{p} \subseteq S \times A \times \mathrm{CFunc}(S)$ *is a set of* possible *transitions and* $\Delta_\mathrm{r} \subseteq S \times A \times \mathrm{CFunc}(S)$ *is a set of* required *transitions with* $\Delta_\mathrm{r} \subseteq \Delta_\mathrm{p}$.

Note that the class of APA, in which $\Delta_p = \Delta_r$ and $\forall s \in S, a \in A, x \in \{\mathrm{p}, \mathrm{r}\}$: $s \xrightarrow{a}_x \varphi$ implies $|sat(\varphi)| = 1$, coincides with PA; whereas, the class of APGA in which $\Delta_\mathrm{p}(S_2) = \Delta_\mathrm{r}(S_2)$ and $\forall s \in S_2, a, b \in A$: $(s \xrightarrow{a}_\mathrm{r} \varphi \wedge s \xrightarrow{b}_\mathrm{r} \varphi')$ implies $\varphi = \varphi'$, $|sat(\varphi)| = 1$ and $\mu \in sat(\varphi)$ is a Dirac distribution, coincides with APA (Fig. 4.3 represents an APA (left) and its equivalent APGA (right)). Thus, APGA can also model systems that can be modelled with APA.

The notions of combined and hyper-transitions (Def. 11) are extended to *possible* and *required* transitions in the obvious manner. Also the notion of closedness (Def. 8) is easily adapted to APGA.

In the sequel, we assume that for each state s in APGA, $|\Delta_\mathrm{p}(s)|$ is finite and by definition so is the case with $|\Delta_\mathrm{r}(s)|$. Moreover, $\forall s \in S, a \in \mathrm{Act}, x \in \{\mathrm{r}, \mathrm{p}\}$: $s \xrightarrow{a}_x \varphi$ and $s \xrightarrow{a}_x \phi$ implies $\varphi = \phi$. Note that an APGA can easily be converted into another APGA that satisfies the above assumptions but have the same set of implementations. Furthermore, for simplicity in definitions, we write $s \xrightarrow{a}_\mathrm{p} \mu$ iff there exists $\varphi \in \mathrm{CFunc}(S)$ with $s \xrightarrow{a}_\mathrm{p} \varphi$ and $\mu \in \varphi$. Similarly, we do this for *required* transitions in definitions. Moreover, in figures we do not indicate the constraint functions but one of their distributions. In the sequel, $\mathscr{H} = (S, \{S_1, S_2\}, A, \Delta_\mathrm{r}, \Delta_\mathrm{p}, s_0)$ is a finitely branching APGA.

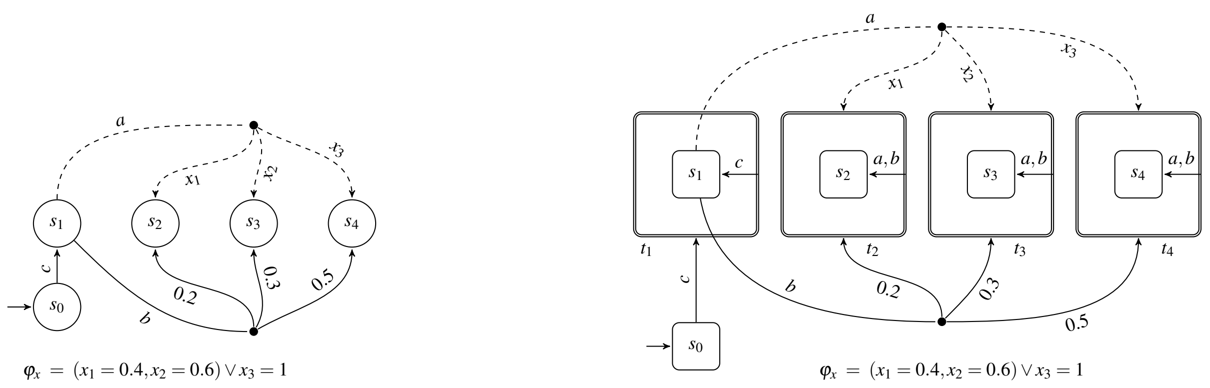


Figure 4.3: An APA $\mathscr{N}$ (left) and its equivalent APGA $\mathscr{H}$ (right).

4.3 Refinement Relations for APGA

APGA at different abstraction levels are compared using *refinement relations*. Intuitively, a refinement relation is intended to imply inclusion of sets of implementations of related APGA. A special class of refinement relations, called *satisfaction relations*, relates implementations (concrete models, i.e., PGA) with APGA (specifications). We propose state-based and distribution-based refinement relations for APGA. We prove them to be *preorders* — a property that is necessary to compare successive refinements of specifications — and study their relationship.

4.3.1 State-based Refinement Relation

Satisfaction relation: We start by giving a state-based definition of satisfaction relations that relate PGA with their specifications, i.e., APGA.

Definition 22. (State-based Satisfaction). *$R \subseteq \bigcup_{j\in\{1,2\}} S_j \times S'_j$ is a* state-based satisfaction (SBSA) *relation between PGA $\mathscr{G}$ and APGA $\mathscr{H}'$ iff for sRs',*

1. *$s \xrightarrow{a} \mu$ implies $s' \xrightarrow{a}_{\mathrm{pc}} \mu'$ such that $\mu R \mu'$, and*
2. *$s' \xrightarrow{a}_{\mathrm{r}} \mu'$ implies $s \xrightarrow{a}_{\mathrm{c}} \mu$ such that $\mu R \mu'$.*

Let $\models_{\mathrm{sb}}$ be the largest SBSA relation.

The condition (1) is the same as in Def. 15 except that it deals with *possible* (combined) transitions from s'. The condition (2) asserts that for every *required* transition from s', there is a combined transition from s and the resulting distributions are related (by Def. 12) w.r.t. R. The set of state-based implementations of APGA $\mathscr{H}'$ is $\llparenthesis \mathscr{H}' \rrparenthesis_{\mathrm{sb}} = \{\mathscr{G} \mid \mathscr{G} \models_{\mathrm{sb}} \mathscr{H}'\}$.

Example 17. *The relation $R = \bigcup_{i=1\dots3}(t_i, t'_i) \cup \bigcup_{i=0\dots4}(s_i, s'_i)$ is an SBSA relation between PGA $\mathscr{G}$ (Fig. 2.4) and APGA $\mathscr{H}'$ (Fig. 4.2). Let us consider $(s_1, s'_1) \in R$ and check whether it fulfils the conditions of Def. 22. For the* required *a-transition from s'_1 to t'_2, there is an a-transition from s_1 to t_2 and $(t_2, t'_2) \in R$. Same is the case with the* required *b-transition from s'_1. For the a-transition from s_1 to $[\![0.5t_2, 0.5t_3]\!]$,*

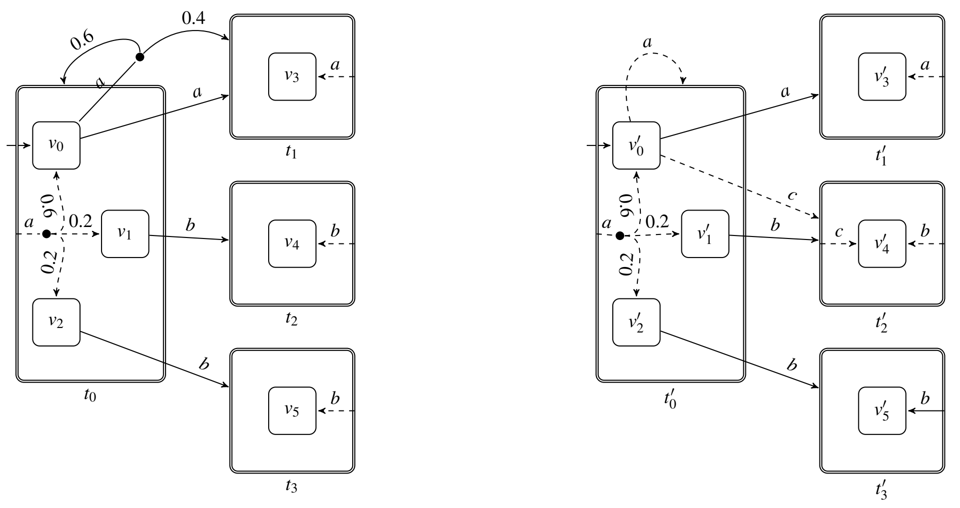


Figure 4.4: $\mathscr{H}$ (left) $\preceq_{\text{sb}}$ $\mathscr{H}'$ (right).

there is a possible *a-transition from* s'_1 *to* $[\![0.5t'_2, 0.5t'_3]\!]$ *and* $[\![0.5t_2, 0.5t_3]\!]R[\![0.5t'_2, 0.5t'_3]\!]$*. Similarly, the conditions hold for the other pairs in R.*

Refinement relation: State-based refinement relations are preorders on a state space requiring that whenever a state s refines a state s', then s' mimics at least the step-wise *possible* behaviour of s, whereas s mimics at least the step-wise *required* behaviour of s'. Formally,

Definition 23. (State-based Refinement). $R \subseteq \bigcup_{j\in\{1,2\}} S_j \times S_j$ *is a* state-based refinement (SBR) *relation on APGA* $\mathscr{H}$ *iff for* sRs',

1. $s \xrightarrow{a}_{\text{p}} \mu$ *implies* $s' \xrightarrow{a}_{\text{pc}} \mu'$ *such that* $\mu R \mu'$, *and*
2. $s' \xrightarrow{a}_{\text{r}} \mu'$ *implies* $s \xrightarrow{a}_{\text{rc}} \mu$ *such that* $\mu R \mu'$.

Let $\preceq_{\text{sb}}$ *be the largest SBR relation.*

The conditions (1) and (2) are the same as in Def. 22 except that in (1) the transition from s is a *possible* transition, whereas in (2) the combined transition from s should be a *required* transition.

Example 18. *APGA* $\mathscr{H}$ *refines* $\mathscr{H}'$ *(Fig. 4.4) as* $R = \bigcup_{i=0\ldots5}\{(v_i, v'_i)\} \cup \bigcup_{i=0\ldots3}\{(t_i, t'_i)\}$ *is an SBR relation. Let us consider* $(v_0, v'_0) \in R$ *and check whether it fulfils the conditions of Def. 23. For the* required *a-transition from* v'_0 *to* t'_1*, there is a* required *a-transition from* v_0 *to* t_1 *and* $(t_1, t'_1) \in R$*. Although, there is a c-transition from* v'_0*, it is a* possible *transition and, therefore, it is not necessary for* v_0 *to implement it. However, for the a-transition from* v_0 *to* $[\![0.6t_0, 0.4t_1]\!]$*, there is a combined a-transition from* v'_0 *to* $[\![0.6t'_0, 0.4t'_1]\!]$ *and* $[\![0.6t_0, 0.4t_1]\!]R[\![0.6t'_0, 0.4t'_1]\!]$*. Similarly, the conditions hold for the other pairs in R.*

Proposition 11. *For APGA $\mathscr{H}$ and $\mathscr{H}'$, $\mathscr{H} \preceq_{\mathrm{sb}} \mathscr{H}'$ implies $\tau(\mathscr{H}) \preceq_{\mathrm{sb}} \tau(\mathscr{H}')$.*

Theorem 6. $\preceq_{\mathrm{sb}}$ *is a preorder.*

Proof. Reflexivity: follows trivially from Def. 23.

Transitivity: Let $\mathscr{H} = (S, \{S_1, S_2\}, A, \Delta_{\mathrm{r}}, \Delta_{\mathrm{p}}, s_0)$, $\mathscr{H}' = (S', \{S'_1, S'_2\}, A, \Delta'_{\mathrm{r}}, \Delta'_{\mathrm{p}}, s'_0)$ and $\mathscr{H}'' = (S'', \{S''_1, S''_2\}, A, \Delta''_{\mathrm{r}}, \Delta''_{\mathrm{p}}, s''_0)$ be APGA. Let $\mathscr{H} \preceq_{\mathrm{sb}} \mathscr{H}'$ and $\mathscr{H}' \preceq_{\mathrm{sb}} \mathscr{H}''$, then we prove that $\mathscr{H} \prec_{\mathrm{sb}} \mathscr{H}''$ holds. Let R_1 be an SBR relation between $\mathscr{H}$ and $\mathscr{H}'$, and R_2 between $\mathscr{H}'$ and $\mathscr{H}''$. We define the relation $R \subseteq (S_1 \times S''_1) \cup (S_2 \times S''_2)$ as:

$$R = \{(s, s'') \mid sR_1s', s'R_2s'' \text{ for } s' \in S'\}$$

and show that it fulfils the conditions of Def. 23.

Assume sRs'' such that sR_1s', $s'R_2s''$ for $s' \in S'$.

1. Let $s \xrightarrow{a}_{\mathrm{p}} \mu$. As sR_1s', by Def. 23, $s' \xrightarrow{a}_{\mathrm{pc}} \mu'$ such that $\mu R_1 \mu'$. As $s' \xrightarrow{a}_{\mathrm{pc}} \mu'$ and $s'R_2s''$, it implies by Def. 23 that $s'' \xrightarrow{a}_{\mathrm{pc}} \mu''$ with $\mu' R_2 \mu''$. As $\mu R_1 \mu'$ and $\mu' R_2 \mu''$, this implies by Lem. 1 that $\mu R \mu''$.

2. Let $s'' \xrightarrow{a}_{\mathrm{r}} \mu''$. As $s'R_2s''$, by Def. 23, $s' \xrightarrow{a}_{\mathrm{rc}} \mu'$ such that $\mu' R_2 \mu''$. As $s' \xrightarrow{a}_{\mathrm{rc}} \mu'$ and sR_1s', it implies by Def. 23 that $s \xrightarrow{a}_{\mathrm{rc}} \mu$ with $\mu R_1 \mu'$. As $\mu R_1 \mu'$ and $\mu' R_2 \mu''$, this implies by Lem. 1 that $\mu R \mu''$.

□

Corollary 1. *$\mathscr{H} \preceq_{\mathrm{sb}} \mathscr{H}'$ implies $\wr \mathscr{H} \wr_{\mathrm{sb}} \subseteq \wr \mathscr{H}' \wr_{\mathrm{sb}}$.*

$\preceq_{\mathrm{sb}}$ coincides with the refinement relation for APA [SK12, Def. 9]; and its kernel coincides with *Segala's probabilistic bisimulation* relation (see Def. 13 on page 18) for PA.

Proposition 12. $\preceq_{\mathrm{sb}} \cap \preceq_{\mathrm{sb}}^{-1} = \sim_{\mathrm{pa}}$ *for PA.*

4.3.2 Distribution-based Refinement Relation

Satisfaction relation: We start by giving a distribution-based definition of satisfaction relations that relate PGA with their specifications, i.e., APGA.

Definition 24. (Distribution-based Satisfaction). *$R \subseteq \bigcup_{j\in\{1,2\}} \mathrm{Dist}(S_j) \times \mathrm{Dist}(S_j)$ is a* distribution-based satisfaction (DBSA) *relation between PGA $\mathscr{G}$ and APGA $\mathscr{H}'$ iff for every $\mu R \mu'$,*

1. *$\mu = \bigoplus_{s'\in\mathrm{Supp}(\mu')} \mu_{s'}$ and $\forall s' \in \mathrm{Supp}(\mu') : (\mu'(s') = |\mu_{s'}|$ and $\mu_{s'\downarrow} R \iota_{s'})$,*
2. *$\mu \xrightarrow{a} \rho$ implies $\mu' \xrightarrow{a}_{\mathrm{pc}} \rho'$ such that $|\rho| \leq |\rho'|$ and $\rho_\downarrow R \rho'_\downarrow$, and*
3. *$\mu' \xrightarrow{a}_{\mathrm{r}} \rho'$ implies $\mu \xrightarrow{a}_{\mathrm{c}} \rho$ such that $|\rho| \geq |\rho'|$ and $\rho_\downarrow R \rho'_\downarrow$.*

Let $\models_{\mathrm{db}}$ be the largest DBSA relation.

The conditions (1) and (2) are the same as in Def. 16 except that the condition (2) deals with *possible* (combined) transitions from s'. The condition (3) asserts that for every *required* transition from μ' to *some* ρ', there is a combined transition from μ to some ρ, the mass of ρ is at least that of ρ' and their conditional distributions are related. The set of distribution-based implementations of APGA $\mathscr{H}'$ is $\wr \mathscr{H}' \wr_{\mathrm{db}} = \{\mathscr{G} \mid \mathscr{G} \models_{\mathrm{db}} \mathscr{H}'\}$.

Refinement relation: Distribution-based refinement relations are preorders on distributions over a state space requiring that whenever a distribution μ refines a distribution μ', then μ' mimics at least the step-wise *possible* behaviour of μ whereas μ mimics at least the step-wise *required* behaviour of μ'. Formally,

Definition 25. (Distribution-based Refinement). *$R \subseteq \bigcup_{j\in\{1,2\}} \mathrm{Dist}(S_j) \times \mathrm{Dist}(S_j)$ is a* distribution-based refinement (DBR) *relation on an APGA $\mathscr{H}$ iff for every $\mu R \mu'$,*

1. *$\mu = \bigoplus_{s'\in\mathrm{Supp}(\mu')} \mu_{s'}$ and $\forall s' \in \mathrm{Supp}(\mu') : (\mu'(s') = |\mu_{s'}|$ and $\mu_{s'\downarrow} R \iota_{s'})$,*
2. *$\mu \xrightarrow{a}_{\mathrm{p}} \rho$ implies $\mu' \xrightarrow{a}_{\mathrm{pc}} \rho'$ such that $|\rho| \leq |\rho'|$ and $\rho_\downarrow R \rho'_\downarrow$, and*
3. *$\mu' \xrightarrow{a}_{\mathrm{r}} \rho'$ implies $\mu \xrightarrow{a}_{\mathrm{rc}} \rho$ such that $|\rho| \geq |\rho'|$ and $\rho_\downarrow R \rho'_\downarrow$.*

Let $\preceq_{\mathrm{db}}$ be the largest DBR relation.

The conditions (1), (2) and (3) are the same as in Def. 24 except that in (2) the transition from μ is a *possible* transition, whereas in (3) the combined transition from μ should be a *required* transition. The following two examples show the role of conditions (2) and (3) in Def. 25 in finding out DBR relations between APGA.

Example 19. *APGA $\mathscr{H}$ refines $\mathscr{H}'$ (Fig. 4.5) as $R = \{(\iota_{v_3}, \iota_{v'_2}), (\iota_{v_2}, \iota_{v'_1}), (\iota_{v_5}, \iota_{v'_4}), (\iota_{v_4}, \iota_{v'_3})\} \cup \{([\![0.2v_0, 0.2v_1, 0.6v_2]\!], [\![0.4v'_0, 0.6v'_1]\!]), ([\![0.2v_0, 0.2v_1]\!]_\downarrow, \iota_{v'_0}), ([\![0.25t_1, 0.75t_2]\!], [\![0.25t'_1, 0.75t'_2]\!]), ([\![0.25v_3, 0.75v_4]\!], [\![0.25v'_2, 0.75v'_3]\!])\} \cup \bigcup_{i=0\ldots3}\{(t_i, t'_i)\}$ is a DBR relation. Let us consider the distributions $\mu = [\![0.2v_0, 0.2v_1, 0.6v_2]\!]$ and $\mu' = [\![0.4v'_0, 0.6v'_1]\!]$, and check the conditions of Def. 25. In Example 10, we have already*

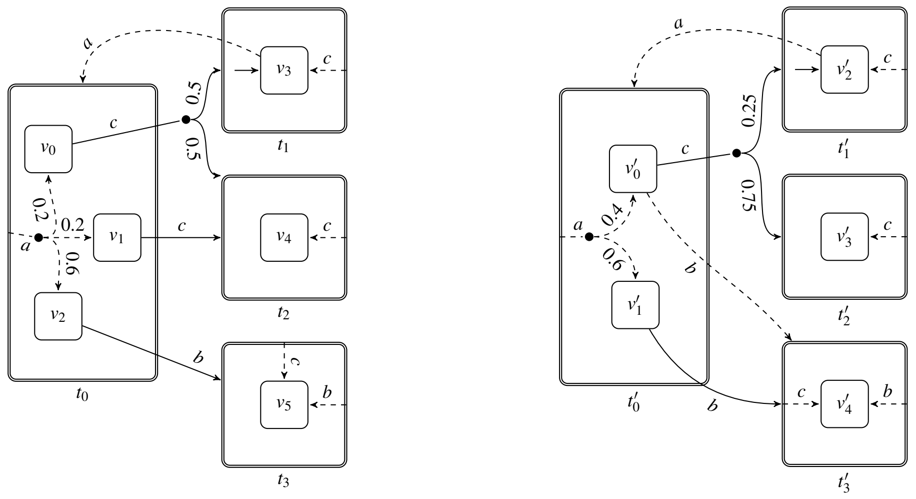


Figure 4.5: $\mathscr{H}$ (left) $\preceq_{\text{db}}$ $\mathscr{H}'$ (right).

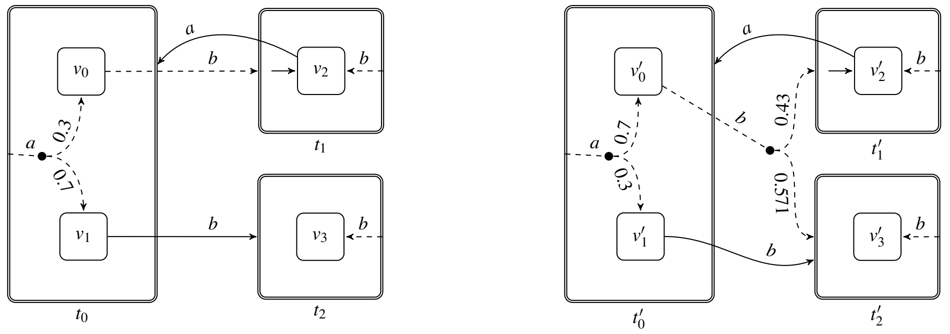


Figure 4.6: $\mathscr{H}$ (left) $\preceq_{\text{db}}$ $\mathscr{H}'$ (right).

checked the conditions (1) and (2), let us check condition (3). For the required *c-transition from μ' to $\rho' = [\![0.1t'_1, 0.3t'_2]\!]$, there is a* required *c-transition from μ to $\rho = [\![0.1t_1, 0.3t_2]\!]$ such that $|\rho'| \leq |\rho|$ and $\rho_{\downarrow} R \rho'_{\downarrow}$ hold. Note that for $t'_1, t'_2 \in \text{Supp}(\rho')$, $[\![0.1t_1]\!]$ and $[\![0.3t_2]\!]$ are the relevant sub-distributions of ρ respectively. Same is the case with b-transitions. Note that no SBR relation exists between $\mathscr{H}$ and $\mathscr{H}'$ as v_0 and v_1 do not refine any state in $\mathscr{H}'$.*

Example 20. *APGA $\mathscr{H}$ refines $\mathscr{H}'$ (Fig. 4.6) as $R = \{([\![0.3v_0, 0.7v_1]\!], [\![0.7v'_0, 0.3v'_1]\!]), ([\![0.3v_0, 0.4v_1]\!]_{\downarrow}, \iota_{v'_0})\} \cup \bigcup_{i=0\ldots2}\{(t_i, t'_i))\} \cup \bigcup_{i=1\ldots3}\{(v_i, v'_i))\}$ is a DBR relation. Let us consider the distributions $\mu = [\![0.3v_0, 0.7v_1]\!]$ and $\mu' = [\![0.7v'_0, 0.3v'_1]\!]$, and check the conditions of Def. 25. For $v'_0 \in supp(\mu')$, $[\![0.3v_0, 0.4v_1]\!]$ is the relevant sub-distribution of μ and $[\![0.3v_0, 0.4v_1]\!]_{\downarrow} R [\![0.7v'_0]\!]_{\downarrow}$ holds. Similarly, for $v'_1 \in \text{Supp}(\mu')$, we have $[\![0.3v_1]\!]$ as a relevant sub-distribution of μ and $[\![0.3v_1]\!]_{\downarrow} R [\![0.3v'_1]\!]_{\downarrow}$ holds, thus fulfilling condition (1). Now for the* required *b-transition from μ' to $[\![0.3t'_2]\!]$, there is a* required *b-transition from μ to $[\![0.7t_2]\!]$ such that $|[\![0.3t'_2]\!]| \leq |[\![0.7t_2]\!]|$ and $[\![0.7t_2]\!]_{\downarrow} R [\![0.3t'_2]\!]_{\downarrow}$ hold. Similarly, for the* possible *b-transition from μ to $[\![0.3t_1, 0.7t_2]\!]$, there is a* possible *b-transition from μ' to $[\![0.3t'_1, 0.7t'_2]\!]$ and $[\![0.3t_1, 0.7t_2]\!] R [\![0.3t'_1, 0.7t'_2]\!]$*

hold; thus, fulfilling conditions (2) and (3) of Def. 25. Note that no SBR relation exists between $\mathscr{H}$ and $\mathscr{H}'$.

Like DBS relations (see Proposition 3 on page 25), a DBR relation between two APGA does not imply a DBR relation between their closed versions.

Proposition 13. *For APGA $\mathscr{H}$ and $\mathscr{H}'$, $\mathscr{H} \preceq_{\mathrm{db}} \mathscr{H}'$ does not imply $\tau(\mathscr{H}) \preceq_{\mathrm{db}} \tau(\mathscr{H}')$.*

Theorem 7. *$\preceq_{\mathrm{db}}$ is a preorder.*

Proof. Reflexivity: follows trivially from Def. 25.

Transitivity: Let $\mathscr{H} = (S, \{S_1, S_2\}, A, \Delta_{\mathrm{r}}, \Delta_{\mathrm{p}}, s_0)$, $\mathscr{H}' = (S', \{S'_1, S'_2\}, A, \Delta'_{\mathrm{r}}, \Delta'_{\mathrm{p}}, s'_0)$ and $\mathscr{H}'' = (S'', \{S''_1, S''_2\}, A, \Delta''_{\mathrm{r}}, \Delta''_{\mathrm{p}}, s''_0)$ be APGA. Let $\mathscr{H} \preceq_{\mathrm{db}} \mathscr{H}'$ and $\mathscr{H}' \preceq_{\mathrm{db}} \mathscr{H}''$, then we prove that $\mathscr{H} \preceq_{\mathrm{db}} \mathscr{H}''$. Let R_1 be the DBR relation between $\mathscr{H}$ and $\mathscr{H}'$, and R_2 between $\mathscr{H}'$ and $\mathscr{H}''$. We define the relation $R \subseteq (\mathrm{Dist}(S_1) \times \mathrm{Dist}(S''_1)) \cup (\mathrm{Dist}(S_2) \times \mathrm{Dist}(S''_2))$ as:

$$R = \{(\mu, \mu'') \mid \mu R_1 \mu', \mu' R_2 \mu'' \text{ for } \mu' \in \mathrm{Dist}(S')\}$$

and show that it fulfils the conditions of Def. 25.

Assume $\mu R \mu''$ where $\mu R_1 \mu'$ and $\mu' R_2 \mu''$ for some $\mu' \in \mathrm{Dist}(S')$.

1. The proof of condition (1) is the same as in that of Th. 2.

2. Let $\mu \xrightarrow{a}_{\mathrm{p}} \nu$. As $\mu R_1 \mu'$, by condition (2) of Def. 25, $\mu' \xrightarrow{a}_{\mathrm{pc}} \nu'$ such that $|\nu'| \geq |\nu|$ and $\nu_{\downarrow} R_1 \nu'_{\downarrow}$. Similarly, as $\mu' \xrightarrow{a}_{\mathrm{pc}} \nu'$ and $\mu' R_2 \mu''$, it implies by Def. 25 that $\mu'' \xrightarrow{a}_{\mathrm{pc}} \nu''$ with $|\nu''| \geq |\nu'|$ and $\nu'_{\downarrow} R_2 \nu''_{\downarrow}$. As $|\nu| \geq |\nu'|$ and $|\nu'| \geq |\nu''|$, thus $|\nu| \geq |\nu''|$; and as $\nu_{\downarrow} R_1 \nu'_{\downarrow}$ and $\nu'_{\downarrow} R_2 \nu''_{\downarrow}$, thus $\nu_{\downarrow} R \nu''_{\downarrow}$.

3. Let $\mu'' \xrightarrow{a}_{\mathrm{r}} \nu''$. As $\mu' R_2 \mu''$, by condition (3) of Def. 25, $\mu' \xrightarrow{a}_{\mathrm{rc}} \nu'$ such that $|\nu''| \leq |\nu'|$ and $\nu'_{\downarrow} R_2 \nu''_{\downarrow}$. Similarly, as $\mu' \xrightarrow{a}_{\mathrm{rc}} \nu'$ and $\mu R_1 \mu'$, it implies by Def. 25 that $\mu \xrightarrow{a}_{\mathrm{rc}} \nu$ with $|\nu'| \leq |\nu|$ and $\nu_{\downarrow} R_1 \nu'_{\downarrow}$. As $|\nu'| \leq |\nu|$ and $|\nu''| \leq |\nu'|$, thus $|\nu''| \leq |\nu|$; and as $\nu_{\downarrow} R_1 \nu'_{\downarrow}$ and $\nu'_{\downarrow} R_2 \nu''_{\downarrow}$, thus $\nu_{\downarrow} R \nu''_{\downarrow}$.

□

Corollary 2. *$\mathscr{H} \preceq_{\mathrm{db}} \mathscr{H}'$ implies $\wr \mathscr{H} \wr_{\mathrm{db}} \subseteq \wr \mathscr{H}' \wr_{\mathrm{db}}$.*

State- vs. distribution-based refinements: $\preceq_{\text{db}}$ is not comparable with $\preceq_{\text{sb}}$ (lifted to distributions over states). One can see that for $\mu \preceq_{\text{db}} \iota_s$, the condition (3) of Def. 25 enforces that if $\mu \xrightarrow{a}_{\text{p}} \nu$, then $|\nu| = 1$; however, if $\mu \preceq_{\text{sb}} \iota_s$, then $0 \leq |\nu| \leq 1$ by Def. 23. For closed APGA, the two refinement relations are comparable.

Proposition 14. *$\preceq_{\text{sb}}$ and $\preceq_{\text{db}}$ are incomparable for APGA; and $\preceq_{\text{sb}} \subseteq \preceq_{\text{db}}$ for closed APGA.*

4.4 Approximation of APGA

In APGA, a transition with as target a constraint function φ represents a (possibly infinite) set of transitions whose target distributions satisfy φ. Stated differently, APGA can be considered as finite symbolic representations of infinitely branching PGA. On the other hand, the infinite cardinality of satisfaction sets of constraint functions gives rise to infinitely-many implementations that even have the same state space as that of APGA, thus, making the extremal reachability analysis of APGA impossible in general.

We, therefore, approximate the implementations of an APGA by a finite set; and (in Section 4.5) show that two of these implementations are sufficient for extremal reachability analysis of APGA. We focus our technique on APGA having polynomial constraints.

To have finitely-many implementations of an APGA, we approximate its constraint functions with the ones having finite satisfaction sets. We adopt the strategy proposed in [SK12]. We over(under)-approximate polynomial constraint functions by linear ones whose satisfaction sets can be represented by their extreme elements (distributions). In the following, we discuss how a polynomial constraint function is over- and under-approximated. Let φ^{c} be a constraint function derived from φ such that $sat(\varphi^{\text{c}}) = \{\mu = \bigoplus_{i\in I} c_i \cdot \mu_i \mid \exists I \subseteq \mathbb{N}^+ : (\forall i \in I : c_i \in \mathbb{R}_{\geq 0}, \mu_i \in sat(\varphi)) \wedge \sum_{i\in I} c_i = 1\}$.

For $i \in \mathbb{N}$, let θ_i and φ_i be a *linear constraint* and a *linear constraint function* respectively in variables denoting probabilities over S. Let $\varphi_\iota = \{\iota_s \mid s \in S\}$ be a linear constraint function characterizing Dirac distributions over S, and φ_μ be a linear constraint function characterizing only one distribution, i.e., μ.

Consider a polynomial constraint function ϕ representing a set of distributions over S. It at least contains a linear constraint $\sum_{s\in S} x_s = 1$ which implies that $\phi^C \subseteq \varphi_\iota^C$. We can now generate a series of linear constraint functions $\varphi_0 = \varphi_\iota$, $\varphi_1 = \varphi_0 \wedge \theta_0$, $\varphi_2 = \varphi_1 \wedge \theta_1$ and so on such that $\phi^C \subseteq \varphi_{i+1}^C \subseteq \varphi_i^C$ for all $i \geq 0$ and $\phi^C = \lim_{i\to\infty} \varphi_i^C$. Every φ_i in the above series over-approximates ϕ, i.e., every μ in ϕ^C also exists in φ_i^C.

Now we under-approximate ϕ by a linear constraint function. Let $\mu \in \phi$ such that $\varphi_\mu = \mu$ is a linear constraint function. Like in the above case, we can generate a series of constraint functions $\varphi_0 = \varphi_\mu$, $\varphi_1 = \varphi_0 \vee \theta_0$, $\varphi_2 = \varphi_1 \vee \theta_1$ and so on such that $\phi^C \supseteq \varphi_{i+1}^C \supseteq \varphi_i^C$ for all $i \geq 0$ and $\phi^C = \lim_{i\to\infty} \varphi_i^C$. Every φ_i in the above series under-approximates ϕ, i.e., every μ in φ_i^C also exists in ϕ^C.

Definition 26. (Constraint Approximation). [SK12] *A function $\varsigma : \text{CFunc}(S) \to \text{CFunc}(S)$ over-approximates a polynomial constraint function by a linear one iff the following holds for polynomial constraint functions $\phi, \phi_1, \phi_2 \in \text{CFunc}(S)$:*

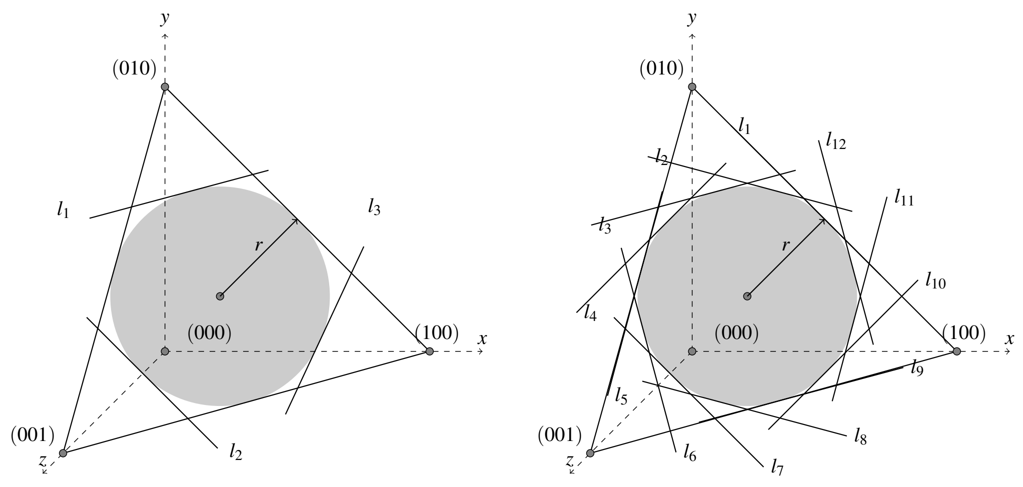


Figure 4.7: An example polynomial constraint function (left) and a linear over-approximation (right).

1. *$\varsigma(\phi)$ is a linear constraint function with $\phi^c \subseteq \varsigma(\phi)^c$, and*

2. $\phi_1^c \subseteq \phi_2^c \Rightarrow \varsigma(\phi_1)^c \subseteq \varsigma(\phi_2)^c$.

Similarly, the function ς^{-1} under-approximates a polynomial constraint function by a linear one, i.e., $\phi^c \supseteq \varsigma^{-1}(\phi)^c$ and $\phi_1^c \supseteq \phi_2^c \Rightarrow \varsigma^{-1}(\phi_1)^c \supseteq \varsigma^{-1}(\phi_2)^c$.

Moreover, we consider *constraint approximating function* ς_1 to be more precise than ς_2 iff $\varsigma_1(\varphi) \subseteq \varsigma_2(\varphi)$ and $\varsigma_1^{-1}(\varphi) \supseteq \varsigma_2^{-1}(\varphi)$ for $\varphi \in \mathrm{CFunc}(S)$.

Example 21. *Consider a polynomial constraint function $\phi = (x^2+y^2+z^2 \leq r^2 \wedge x+y+z=1)$ representing a set of distributions by a shaded-circular region of radius r within each triangle of Fig. 4.7 and 4.8. Let ς_1 and ς_2 be the constraint-approximating functions such that $\varsigma_1(\phi)$ represents the region enclosed by the lines l_1, l_2, l_3 and the sides of the left triangle in Fig. 4.7; and $\varsigma_2(\phi)$ represents the region enclosed by the lines $l_1, \ldots, l_{12}$ in the right triangle. Let $\varsigma_2^{-1}(\phi)$ represent the region enclosed by the lines $l_1, \ldots l_6$ in Fig.4.8. It is clear that $\varsigma_2(\phi)$ is an over-approximation and $\varsigma_2^{-1}(\phi)$ is an under-approximation of ϕ, i.e., $\varsigma_2^{-1}(\phi) \subseteq \phi \subseteq \varsigma_2(\phi)$. Moreover, $\varsigma_2(\phi)$ gives a better over-approximation of ϕ than $\varsigma_1(\phi)$, i.e., $\phi \subseteq \varsigma_2(\phi) \subseteq \varsigma_1(\phi)$.*

The notion of constraint-approximation can be lifted to APGA as:

Definition 27. *Let $\mathscr{H}$ be an APGA with polynomial constraints, then the constraint-approximating function $\varsigma : \mathrm{CFunc}(S) \rightarrow \mathrm{CFunc}(S)$ induces the APGA $\mathscr{H}' = \varsigma(\mathscr{H})$ where S, A and s_0 are the same as in $\mathscr{H}$; and for all $s \in S$,*

1. *$(s,a,\phi) \in \Delta_r$ iff $(s,a,\varsigma^{-1}(\phi)) \in \Delta_r'$, and*

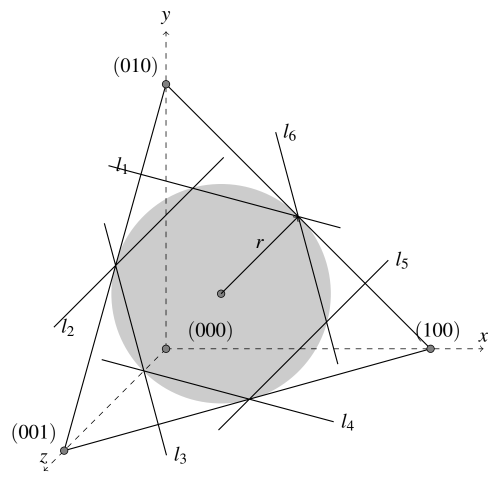


Figure 4.8: Linear under-approximation of a polynomial constraint function.

2. $(s,a,\phi) \in \Delta_p$ *iff* $(s,a,\varsigma(\phi)) \in \Delta'_p$,

Note that by Def. 26, $\varsigma^{-1}(\phi) \subseteq \phi$ and $\phi \subseteq \varsigma(\phi)$. This implies that every *required* transition of s in $\mathscr{H}'$ is also a *required* transition in $\mathscr{H}$. However, the case of *possible* transitions is the other way around. This leads to the fact that $\mathscr{H}$ refines $\mathscr{H}'$ as given by the following theorem:

Proposition 15. $\mathscr{H} \preceq_x \varsigma(\mathscr{H})$ *for* $x \in \{\text{sb}, \text{db}\}$.

Proof. The proof of the above proposition follows directly from Def. 26. □

After over(under)-approximating polynomial constraint functions by linear ones, the second issue is how to deal with their satisfaction sets that may be countably infinite. We approximate them by finite sets by considering their extreme distributions. The concept of extreme distributions is explained in [KKN09] for *interval constraints* that can easily be extended for linear constraints. In the following, we define the notion of *extreme linear function* and then use it to define *extreme transitions* for APGA:

Definition 28. [SK12] *Let $\mathscr{H}$ be an APGA with linear constraints, and φ be a linear constraint function, then:*

1. φ_{extr} *is an* extreme linear function *of φ iff $sat(\varphi_{extr})$ is the smallest finite subset of $sat(\varphi)$ and $sat(\varphi_{extr})^c = sat(\varphi)^c$.*
2. $s \xrightarrow{a}_p \varphi_{extr}$ *is an* extreme possible transition *iff $s \xrightarrow{a}_p \varphi$. Similarly, we have an* extreme required transition. *Moreover, $\mathscr{H}_{extr}$ represents a game with only extreme transitions.*

Based on Def. 27 and 28, we are now in a position to approximate an APGA, say $\mathscr{H}$, by an APGA, say $\mathscr{H}'$, such that $\mathscr{H}'$ only has a finite set of implementations that have the same state space as that of $\mathscr{H}$. In the sequel, we assume that every APGA admitting infinite implementations is approximated in this way.

4.5 Extreme Games and Reachability Analysis

In this section, we show that two implementations of an APGA, called *extreme probabilistic game automata* (EPGA), bound the extremal reachability probabilities of its implementations. In fact, one EPGA bounds the probabilities in case of competing players; whereas the other EPGA does so in case of collaborating players.

Let the two implementations of APGA $\mathscr{H}$ be the PGA $\mathscr{G}^{\circ}_{\circ}$ and $\mathscr{G}^{\bullet}_{\circ}$, where $\mathscr{G}^{\circ}_{\circ}$ inherits its player-one and player-two transitions, denoted by the upper and the lower circles in $\mathscr{G}^{\circ}_{\circ}$ respectively, from the *possible* transitions in $\mathscr{H}$, whereas $\mathscr{G}^{\bullet}_{\circ}$ inherits its player-one transitions from the *required* transitions and the player-two transitions from the *possible* transitions in $\mathscr{H}$. Formally,

Definition 29. (Extremal Games). *For $\star \in \{\bullet, \circ\}$, $\mathscr{G}^{\star}_{\circ}$ is an EPGA of $\mathscr{H}$ iff S, A and s_0 in $\mathscr{G}^{\star}_{\circ}$ are the same as in $\mathscr{H}$,*
$\Delta(S_2) = \{(s,a,\mu) \mid \exists s \in S_2, a \in \mathrm{Act} : (s,a,\varphi) \in \Delta_{\mathrm{p}} \wedge \mu \in sat(\varphi)\}$,
and if $\star = \circ$, then
$\Delta(S_1) = \{(s,a,\mu) \mid \exists s \in S_1, a \in \mathrm{Act} : (s,a,\varphi) \in \Delta_{\mathrm{p}} \wedge \mu \in sat(\varphi)\}$, *else*
$\Delta(S_1) = \{(s,a,\mu) \mid \exists s \in S_1, a \in \mathrm{Act} : (s,a,\varphi) \in \Delta_{\mathrm{r}} \wedge \mu \in sat(\varphi)\}$.

In the sequel, $\mathscr{G}^{\star}_{\circ} = (S, \{S_1, S_2\}, A, \Delta, s_0)$ is an EPGA of $\mathscr{H}$ for $\star \in \{\bullet, \circ\}$.

The following lemma establishes a relationship between the (state-based and the distribution-based) implementations of an APGA $\mathscr{H}$ and its EPGA.

Lemma 2. *Let $\mathscr{G} \in \wr \mathscr{H} \wr_x$, then $\mathscr{G} \prec_x \mathscr{G}^{\circ}_{\circ}$ and $\mathscr{G} \preccurlyeq_x \mathscr{G}^{\bullet}_{\circ}$ for $x \in \{\mathrm{sb}, \mathrm{db}\}$.*

Proof. We only give proof for distribution-based relations.

Let $R \subseteq (\mathrm{Dist}(S'_1) \times \mathrm{Dist}(S_1)) \cup (\mathrm{Dist}(S'_2) \times \mathrm{Dist}(S_2))$ be a distribution-based satisfaction relation between $\mathscr{G}' \in \wr \mathscr{H} \wr_{\mathrm{db}}$ and $\mathscr{H}$.

- As per Def. 29, the state space of $\mathscr{H}$ and $\mathscr{G}^{\circ}_{\circ}$ is the same. We show that R is also a distribution-based simulation relation between $\mathscr{G}'$ and EPGA $\mathscr{G}^{\circ}_{\circ}$ of $\mathscr{H}$.

  Let $\mu' R \mu$. We show that it fulfils the conditions of Def. 16.

  1. As R is a distribution-based satisfaction relation, the condition of splitting μ' into sub-distributions as per the support of μ trivially holds, and

2. Let $\mu' \xrightarrow{a} \eta'$. By condition (2) of Def. 24, $\mu \xrightarrow{a}_{\text{pc}} \eta$ with $|\eta'| \leq |\eta|$ and $\eta'_{\downarrow} R \eta_{\downarrow}$. By Def. 29, every *possible* transition of $\mathscr{H}$ is also a transition in $\mathscr{G}^{\circ}_{\circ}$, $\mu \xrightarrow{a}_{\text{c}} \eta$ exists in $\mathscr{G}^{\circ}_{\circ}$ as well.

- As per Def. 29, the state space of $\mathscr{H}$ and $\mathscr{G}^{\bullet}_{\circ}$ is the same. We show that R is a distribution-based alternating simulation relation between $\mathscr{G}'$ and EPGA $\mathscr{G}^{\bullet}_{\circ}$ of $\mathscr{H}$.

  Let $\mu' R \mu$. We show that it fulfils the conditions of Def. 18.

  1. As R is a distribution-based satisfaction relation, the condition of splitting μ' into sub-distributions as per the support of μ trivially holds,
  2. Let $\mu' \in \text{Dist}(S'_2)$ and $\mu' \xrightarrow{a} \eta'$. By condition (2) of Def. 24, $\mu \xrightarrow{a}_{\text{pc}} \eta$ with $|\eta'| \leq |\eta|$ and $\eta'_{\downarrow} R \eta_{\downarrow}$. By Def. 29, every *possible* transition from a player-two state in $\mathscr{H}$ is a transition in $\mathscr{G}^{\bullet}_{\circ}$, $\mu \xrightarrow{a}_{\text{c}} \eta$ exists in $\mathscr{G}^{\bullet}_{\circ}$ as well.
  3. Let $\mu \in \text{Dist}(S_1)$ and $\mu \xrightarrow{a} \eta$. By Def. 29, every transition from a player-one state in $\mathscr{G}^{\bullet}_{\circ}$ is a *required* transition in $\mathscr{H}$, $\mu \xrightarrow{a}_{\text{r}} \eta$ exists in $\mathscr{H}$. By condition (3) of Def. 24, $\mu' \xrightarrow{a}_{\text{c}} \eta'$ with $|\eta'| \geq |\eta|$ and $\eta'_{\downarrow} R \eta_{\downarrow}$.

□

Note that state-based implementations of $\mathscr{H}$ are related to its EPGA through state-based relations, whereas distribution-based implementations are related through distribution-based relations. Moreover, as $\mathscr{G}^{\circ}_{\circ}$ simulates and $\mathscr{G}^{\bullet}_{\circ}$ A-simulates every (state-based and distribution-based) implementation of $\mathscr{H}$, therefore, they suffice for the extremal reachability analysis of $\mathscr{H}$. (Note that the other two extreme implementations, i.e., $\mathscr{G}^{\bullet}_{\bullet}$ and $\mathscr{G}^{\circ}_{\bullet}$, are simulated and A-simulated by $\mathscr{G}^{\circ}_{\circ}$ and $\mathscr{G}^{\bullet}_{\circ}$ respectively.)

Theorem 8. *$\mathscr{H} \preceq_x \mathscr{H}'$ implies $\mathscr{G}^{\circ}_{\circ} \prec_x \mathscr{G}^{\circ'}_{\circ}$ and $\mathscr{G}^{\bullet}_{\circ} \preccurlyeq_x \mathscr{G}^{\bullet'}_{\circ}$ for $x \in \{\text{sb}, \text{db}\}$.*

Proof. The proof of Th. 8 follows directly from Def. 23 and 25. □

Moreover, it implies that $\mathscr{G}^{\circ'}_{\circ}$ simulates every implementation of $\mathscr{H}$ and $\mathscr{H}'$; whereas $\mathscr{G}^{\bullet'}_{\circ}$ A-simulates them. Thus, $\mathscr{H}'$ bounds the extremal reachability probabilities of $\mathscr{H}$ in case of competing/collaborating players.

We show that EPGA of an APGA bound extremal reachability probabilities of its implementations with competing/collaborating players. In fact, $\mathscr{G}^{\bullet}_{\circ}$ bounds extremal reachability probabilities of implementations of $\mathscr{H}$ in case of competing players, whereas $\mathscr{G}^{\circ}_{\circ}$ does so otherwise. This is based on the fact that EPGA simulate/A-simulate each (state-based and distribution-based) implementation of an APGA as given by Lem. 2.

Theorem 9. *Let $\mathscr{G}^{\circ'}_{\circ}$ and $\mathscr{G}^{\bullet''}_{\circ}$ be the EPGA of $\mathscr{H}$. Let $\mathscr{G} \in \llparenthesis \mathscr{H} \rrparenthesis_x$ for $x \in \{\text{sb}, \text{db}\}$ such that $T \subseteq S$ is a set of goal states in $\mathscr{G}$, then*

1. *for goal states* $T' = \{s' \in S' \mid \exists \mu \in \mathrm{Dist}(S) : \mu(T) > 0$ *and* $\mu \prec_x s'\}$ *in* $\mathscr{G}_{\circ}^{\circ\prime}$*,*
   $\min^{\blacktriangledown}(T') \leq \min^{\blacktriangledown}(T)$ *and* $\max^{\blacktriangle}(T) \leq \max^{\blacktriangle}(T')$*, and*

2. *for goal states* $T'' = \{s'' \in S'' \mid \exists \mu \in \mathrm{Dist}(S) : \mu(T) > 0$ *and* $\mu \preccurlyeq_x s''\}$ *in* $\mathscr{G}_{\circ}^{\bullet\prime\prime}$*,*
   $\min^{\blacktriangle}(T) \leq \min^{\blacktriangle}(T'')$ *and* $\max^{\blacktriangledown}(T'') \leq \max^{\blacktriangledown}(T)$*.*

Proof. The proof of the above theorem follows from Lem. 2 and Th. 5. □

We now extend the above results for APGA that are in a refinement relation with each other. As discussed before, for APGA that refine each other, their EPGA are in a simulation/alternating simulation relation with each other. Therefore, their extremal reachability probabilities are comparable. In fact, when $\mathscr{H}$ refines $\mathscr{H}'$, then the extremal reachability probabilities of the EPGA of $\mathscr{H}'$ bound that of all implementations of $\mathscr{H}$. Formally,

Theorem 10. *Let* $\mathscr{H} \preceq_x \mathscr{H}'$ *for* $x \in \{\mathrm{sb}, \mathrm{db}\}$*, and let* $T \subseteq S$ *such that* $T' = \{s' \in S' \mid \exists \mu \in \mathrm{Dist}(S) : \mu(T) = 1$ *and* $\mu \preceq_x s'\}$*, then:*

1. $\min^{\blacktriangledown}(T') \leq \min^{\blacktriangledown}(T)$ *and* $\max^{\blacktriangle}(T) \leq \max^{\blacktriangle}(T')$*,*
2. $\min^{\blacktriangle}(T) \leq \min^{\blacktriangle}(T')$ *and* $\max^{\blacktriangledown}(T') \leq \max^{\blacktriangledown}(T)$*.*

Proof. The proof of the above theorem follows from Th. 5 and 8. □

Example 22. *Consider APGA* $\mathscr{H}$ *and* $\mathscr{H}'$ *in Fig. 4.6, where* $\mathscr{H} \preceq_{\mathrm{db}} \mathscr{H}'$*. Let* $T = \{t_2\}$ *such that* $T' = \{t_2'\}$ *(in Th. 10). Then the maximum probability in (each implementation of) APGA* $\mathscr{H}$ *to T lies in* $[0.7, 1]$*, whereas in* $\mathscr{H}'$ *it lies in* $[0.3, 1]$ *for* T'*. (Recall that we only consider* required *transitions from player one states to calculate the lower bound of the maximum probability to target state.)*

4.6 Composition of Stochastic Games

We define a composition operator for the class of APGA that satisfies the following property:

Property 1. *For APGA* $\mathscr{H}$*,* $\forall s \in S_2 : a \in \mathrm{Act}(s)$ *iff* $\exists u \in S_1 : u \xrightarrow{a}_{\mathrm{p}} \nu$ *and* $\nu(s) > 0$*.*

The class of APGA, satisfying the above property, can represent abstractions of PA (that we discuss in Ch. 5). In this way our operator generalizes composition operators for LTS, MTS, PA and APA. Composition operator is defined in a TCSP-like manner, i.e., it is parametrized by a set of actions that need to be performed simultaneously by both games; other actions occur autonomously. Formally,

Definition 30. *For APGA $\mathscr{H}$ and $\mathscr{H}'$, the composition w.r.t. synchronization set $\bar{A} \subseteq (A \cap A') \backslash \{\tau\}$ is given as: $\mathscr{H} \|_{\bar{A}} \mathscr{H}' = (S \times S', \{S_1 \times S'_1, S \times S' \backslash S_1 \times S'_1\}, A \cup A', \Delta'_{\mathrm{r}}, \Delta'_{\mathrm{p}}, (s_0, s'_0))$, where for all $a \in A \cup A'$, $(s,s') \in S \times S'$ and $x \in \{\mathrm{rc}, \mathrm{pc}\}$, $(s,s') \xrightarrow{a}_x (\mu \| \mu')$ iff one of the following holds:*

1. *if $(s,s') \in S_1 \times S'_1$, then:*
   - *(a) $a \in \bar{A}$, $s \xrightarrow{a}_x \mu$ and $s' \xrightarrow{a}_x \mu'$, or*
   - *(b) $a \in A$, $s \xrightarrow{a}_x \mu$ and $\iota_{s'} = \mu'$, or*
   - *(c) $a \in A'$, $s' \xrightarrow{a}_x \mu'$ and $\iota_s = \mu$,*
2. *if $(s,s') \in S_2 \times S'_2$, then $a \in \bar{A}$, $s \xrightarrow{a}_x \mu$ and $s' \xrightarrow{a}_x \mu'$,*
3. *otherwise,*
   - *(a) $s \in S_2$, $s \xrightarrow{a}_x \mu$ and $\iota_{s'} = \mu'$, or*
   - *(b) $s' \in S'_2$, $s' \xrightarrow{a}_x \mu'$ and $\iota_s = \mu$.*

As every *required* transition is a *possible* transition, the above definition does not restrict the synchronization of only *required* transitions; it also allows synchronization of *required* transitions with *possible* transitions. Note that the state space of our composite game is disjointly dividable based on the actions which are enabled. Although, we allow composition of $S_1(S_2)$ states with that of $S'_2(S'_1)$ states, but only a player two can make a move in such a state. The conditions (1) to (3) apply to *required* as well as to *possible* transitions. (1) asserts that states in $S_1 \times S'_1$ can either synchronize with each other or act independently. Note that a state in $S_2 \times S'_2$ is only reached by a synchronizing action performed by players of type one in some $S_1 \times S'_1$ state; and (2) asserts that the next state is reached only by some synchronizing action. (3) tells that for a state in $S_{(1+x)} \times S'_{(2-x)}$, where x is a bit, no synchronization occurs and only a player two can make a move independently. Note that such a state can only be reached by a non-synchronizing action.

The above definition asserts that the composition of two APGA whose constraints are systems of linear inequalities (or polynomial constraints) leads to an APGA with polynomial constraints because composition of linear inequalities results in polynomial constraints (which are closed under composition) [DKL⁺11]. Thus, like APA [DKL⁺11], the class of APGA having polynomial constraints is closed under composition.

Theorem 11. *For any set $\bar{A}$ and $x \in \{\mathrm{sb}, \mathrm{db}\}$, $\preceq_x$ is a pre-congruence w.r.t. $\|_{\bar{A}}$.*

We only prove this for the distribution-based refinement relation. The proof for the other case is similar.

Proof. Let $\mathscr{H} = (S, \{S_1, S_2\}, A, \Delta_{\mathrm{r}}, \Delta_{\mathrm{p}}, s_0)$, $\mathscr{H}' = (S', \{S'_1, S'_2\}, A, \Delta'_{\mathrm{r}}, \Delta'_{\mathrm{p}}, s'_0)$ and $\hat{\mathscr{H}} = (\hat{S}, \{\hat{S}_1, \hat{S}_2\}, \hat{A}, \hat{\Delta}_{\mathrm{r}}, \hat{\Delta}_{\mathrm{p}}, \hat{s}_0)$, $\hat{\mathscr{H}}' = (\hat{S}', \{\hat{S}'_1, \hat{S}'_2\}, \hat{A}, \hat{\Delta}'_{\mathrm{r}}, \hat{\Delta}'_{\mathrm{p}}, \hat{s}'_0)$ be APGA. Let $\bar{A} \subseteq A \cap \hat{A}$ such that $\mathscr{H} \|_{\bar{A}} \hat{\mathscr{H}} = (S \times \hat{S}, \{S_1 \times \hat{S}_1, S \times \hat{S} \backslash S_1 \times \hat{S}_1\}, A \cup \hat{A}, \tilde{\Delta}_{\mathrm{r}}, \tilde{\Delta}_{\mathrm{p}}, (s_0, \hat{s}_0))$ and $\mathscr{H}' \|_{\bar{A}} \hat{\mathscr{H}}' = (S' \times \hat{S}', \{S'_1 \times \hat{S}'_1, S' \times \hat{S}' \backslash S'_1 \times \hat{S}'_1\}, A \cup \hat{A}, \tilde{\Delta}'_{\mathrm{r}}, \tilde{\Delta}'_{\mathrm{p}}, (s'_0, \hat{s}'_0))$. Let $\mathscr{H} \preceq_{\mathrm{db}} \mathscr{H}'$ and $\hat{\mathscr{H}} \preceq_{\mathrm{db}} \hat{\mathscr{H}}'$, then we prove that $\mathscr{H} \|_{\bar{A}} \hat{\mathscr{H}} \preceq_{\mathrm{db}} \mathscr{H}' \|_{\bar{A}} \hat{\mathscr{H}}'$ holds. Let R_1

be a DBR relation between $\mathscr{H}$ and $\mathscr{H}'$, and R_2 between $\hat{\mathscr{H}}$ and $\hat{\mathscr{H}}'$. We define the relation $R \subseteq (\mathrm{Dist}(S_1 \times \hat{S}_1) \times \mathrm{Dist}(S_1' \times \hat{S}_1')) \cup (\mathrm{Dist}(S_2 \times \hat{S}_2) \times \mathrm{Dist}(S_2' \times \hat{S}_2'))$ as:

$$R = \{(\mu \| \hat{\mu}, \mu' \| \hat{\mu}') \mid \mu R_1 \mu', \hat{\mu} R_2 \hat{\mu}'\}$$

and show that it fulfils the conditions of Def. 25 ($\preceq_{\mathrm{db}}$).

Let $(\mu \| \hat{\mu})\ R\ (\mu' \| \hat{\mu}')$.

1. Let $(s', \hat{s}') \in \mathrm{Supp}(\mu' \| \hat{\mu}')$. As $\mu R_1 \mu'$ and $s' \in \mathrm{Supp}(\mu')$, by Def. 25, we have a sub-distribution $\mu_{s'}$ of μ with $|\mu_{s'}| = \mu'(s')$ and $\mu_{s'\downarrow} R_1 \iota_{s'}$. Similarly, as $\hat{\mu} R_2 \hat{\mu}'$, we have a sub-distribution $\hat{\mu}_{\hat{s}'}$ of $\hat{\mu}$ with $|\hat{\mu}_{\hat{s}'}| = \hat{\mu}'(\hat{s}')$ and $\hat{\mu}_{\hat{s}'\downarrow} R_2 \iota_{\hat{s}'}$. Therefore, for $s' \| \hat{s}'$ we have a sub-distribution $(\mu \| \hat{\mu})_{(s',\hat{s}')} = \mu_{s'} \| \hat{\mu}_{\hat{s}'}$ of $\mu \| \hat{\mu}$ with $|\mu_{s'} \| \hat{\mu}_{\hat{s}'}| = \mu' \| \hat{\mu}'((s', \hat{s}'))$ and $(\mu_{s'} \| \hat{\mu}_{\hat{s}'})_{\downarrow} R \iota_{(s',\hat{s}')}$.

2. Let $\mu \| \hat{\mu} \xrightarrow{a}_{\mathrm{p}} \eta \| \hat{\eta}$. There are four possible cases by Def. 30:
   (a) Let $\mu \| \hat{\mu} \in \mathrm{Dist}(S_1) \| \mathrm{Dist}(\hat{S}_1)$. There are three possible cases:
      i. if $a \in \bar{A}$, then $\mu \xrightarrow{a}_{\mathrm{p}} \eta$ and $\hat{\mu} \xrightarrow{a}_{\mathrm{p}} \hat{\eta}$. As $\mu R_1 \mu'$, by condition (2) of Def. 25, we have $\mu' \xrightarrow{a}_{\mathrm{pc}} \eta'$ with $|\eta| \le |\eta'|$ and $\eta_{\downarrow} R_1 \eta'_{\downarrow}$. Similarly, as $\hat{\mu} R_2 \hat{\mu}'$, we have $\hat{\mu}' \xrightarrow{a}_{\mathrm{pc}} \hat{\eta}'$ with $|\hat{\eta}| \le |\hat{\eta}'|$ and $\hat{\eta}_{\downarrow} R_2 \hat{\eta}'_{\downarrow}$. Therefore, $\mu' \| \hat{\mu}' \xrightarrow{a}_{\mathrm{pc}} \eta' \| \hat{\eta}'$ and $|\eta \| \hat{\eta}| \le |\eta' \| \hat{\eta}'|$. As $\eta_{\downarrow} R_1 \eta'_{\downarrow}$ and $\hat{\eta}_{\downarrow} R_2 \hat{\eta}'_{\downarrow}$, we have $(\eta \| \hat{\eta})_{\downarrow} R (\eta' \| \hat{\eta}')_{\downarrow}$.
      ii. if $a \in A \backslash \bar{A}$, then $\mu \xrightarrow{a}_{\mathrm{p}} \eta$ and $\hat{\mu} = \hat{\eta}$. As $\mu R_1 \mu'$, by condition (2) of Def. 25, we have $\mu' \xrightarrow{a}_{\mathrm{pc}} \eta'$ with $|\eta| \le |\eta'|$ and $\eta_{\downarrow} R_1 \eta'_{\downarrow}$. Therefore, $\mu' \| \hat{\mu}' \xrightarrow{a}_{\mathrm{pc}} \eta' \| \hat{\mu}'$ and $|\eta \| \hat{\mu}| \le |\eta' \| \hat{\mu}'|$. As $\eta_{\downarrow} R_1 \eta'_{\downarrow}$ and $\hat{\mu} R_2 \hat{\mu}'$, we have $(\eta \| \hat{\mu})_{\downarrow} R (\eta' \| \hat{\mu}')_{\downarrow}$.
      iii. if $a \in \hat{A} \backslash \bar{A}$, then proof is the same as in the previous case.
   (b) Let $\mu \| \hat{\mu} \in \mathrm{Dist}(S_2) \| \mathrm{Dist}(\hat{S}_2)$. The proof is similar to the previous case.
   (c) Let $\mu \| \hat{\mu} \in \mathrm{Dist}(S_2) \| \mathrm{Dist}(\hat{S}_1)$. The proof is similar to the previous case.
   (d) Let $\mu \| \hat{\mu} \in \mathrm{Dist}(S_1) \| \mathrm{Dist}(\hat{S}_2)$. The proof is similar to the previous case.

3. Let $\mu' \| \hat{\mu}' \xrightarrow{a}_{\mathrm{r}} \eta' \| \hat{\eta}'$. There are four possible cases by Def. 30:
   (a) Let $\mu \| \hat{\mu} \in \mathrm{Dist}(S_1) \| \mathrm{Dist}(\hat{S}_1)$. There are three possible cases:
      i. if $a \in \bar{A}$, then $\mu' \xrightarrow{a}_{\mathrm{r}} \eta'$ and $\hat{\mu}' \xrightarrow{a} \hat{\eta}'$. As $\mu R_1 \mu'$, by condition (3) of Def. 25, we have $\mu \xrightarrow{a}_{\mathrm{rc}} \eta$ with $|\eta| \ge |\eta'|$ and $\eta_{\downarrow} R_1 \eta'_{\downarrow}$. Similarly, as $\hat{\mu} R_2 \hat{\mu}'$, we have $\hat{\mu} \xrightarrow{a}_{\mathrm{rc}} \hat{\eta}$ with $|\hat{\eta}| \ge |\hat{\eta}'|$ and $\hat{\eta}_{\downarrow} R_2 \hat{\eta}'_{\downarrow}$. Therefore, $\mu \| \hat{\mu} \xrightarrow{a}_{\mathrm{rc}} \eta \| \hat{\eta}$ and $|\eta \| \hat{\eta}| \le |\eta' \| \hat{\eta}'|$. As $\eta_{\downarrow} R_1 \eta'_{\downarrow}$ and $\hat{\eta}_{\downarrow} R_2 \hat{\eta}'_{\downarrow}$, we have $(\eta \| \hat{\eta})_{\downarrow} R (\eta' \| \hat{\eta}')_{\downarrow}$.
      ii. if $a \in A \backslash \bar{A}$, then $\mu' \xrightarrow{a}_{\mathrm{r}} \eta'$ and $\hat{\mu}' = \hat{\eta}'$. As $\mu R_1 \mu'$, by condition (3) of Def. 25, we have $\mu \xrightarrow{a}_{\mathrm{rc}} \eta$ with $|\eta| \ge |\eta'|$ and $\eta_{\downarrow} R_1 \eta'_{\downarrow}$. Therefore, $\mu \| \hat{\mu} \xrightarrow{a}_{\mathrm{rc}} \eta \| \hat{\mu}$ and $|\eta \| \hat{\mu}| \ge |\eta' \| \hat{\mu}'|$. As $\eta_{\downarrow} R_1 \eta'_{\downarrow}$ and $\hat{\mu} R_2 \hat{\mu}'$, we have $(\eta \| \hat{\mu})_{\downarrow} R (\eta' \| \hat{\mu}')_{\downarrow}$.
      iii. if $a \in \hat{A} \backslash \bar{A}$, then proof is the same as in the previous case.
   (b) Let $\mu \| \hat{\mu} \in \mathrm{Dist}(S_2) \| \mathrm{Dist}(\hat{S}_2)$. The proof is similar to the previous case.
   (c) Let $\mu \| \hat{\mu} \in \mathrm{Dist}(S_2) \| \mathrm{Dist}(\hat{S}_1)$. The proof is similar to the previous case.
   (d) Let $\mu \| \hat{\mu} \in \mathrm{Dist}(S_1) \| \mathrm{Dist}(\hat{S}_2)$. The proof is similar to the previous case.

$\square$

4.7 Summary and Discussion

In this chapter, we considered an extension of probabilistic game automata (PGA) with modalities (i.e., *required* and *possible* transitions) and *constraint functions* as for PA in [DKL$^+$13]. Abstract PGA (APGA), therefore, over- and under-approximate the behaviour of PGA. We equipped APGA with the notion of refinement that is state-based as well as distribution-based, showing that refinement relations between APGA imply (alternating) simulation relations between their implementations. As APGA may have infinite sets of implementations, we proposed their approximations to have finite sets of implementations. We also showed that maximal and minimal reachability probabilities in APGA can be bound by considering extremal games – those PGA that besides all *required* transitions contain all *possible* transitions, and those that contain only *required* transitions for one set and all *possible* transitions for the other set of states. Finally, we defined a composition operator for the class of APGA that can be abstractions of PA, and showed that our refinement relations are pre-congruences w.r.t. it, thus facilitating APGA-based compositional abstraction of PA. An overview of the results of this chapter is given in Table 4.1.

Related work: Several works have introduced *modal* abstractions of probabilistic systems [KKLW12, KKN09, DKL$^+$13]. In [KKLW12], interval-based abstraction of MDPs has been introduced by giving probabilities of transitions as intervals instead of single values, yielding *interval Markov chains*. Thus, (the upper and the lower bound of) each interval over- and under-approximates the probability to the target state. In [KKN09], this technique has been extended to *interactive Markov chains* (IMC) — a combination of PA and continuous-time Markov chains (CTMC) — yielding *abstract interactive Markov chains* that allow for compositional modeling of systems. In the setting of games, the abstract models most closely related to APGA are *three-valued abstract games* [dAGJ04] in which transitions are grouped into *possible* and *required* sets.

Many researchers have worked on parallel *composition* of probabilistic systems; the earliest work in this direction is in [Seg95] which defines a composition operator for PA; and in [KKN09] for interactive Markov chains. To the best of our knowledge, no parallel compositional operator has been defined for game-based modal abstractions of probabilistic systems.

Furthermore, apart from parallel composition, researchers are also dealing with logical composition of probabilistic systems, i.e., defining specifications of systems in a compositional way [CDL$^+$11, DKL$^+$13]. Intuitively, by conjunction of a set of sub-specifications (given as APA), a composed specification is obtained that only contains the common parts of sub-specifications. Technically speaking, the set of implementations of a composed APA is the intersection of that of the composing APA.

Future extensions: One can consider:

- giving efficient algorithms for finding state-based and distribution-based refinement relations between APGA, and approximating APGA to have finite set of implementations,
- logical characterization of refinement relations for APGA, and
- defining a complete *specification theory* for APGA following the lines in [Lar90].

In the next chapter, we propose state-based and distribution-based compositional abstraction techniques of APGA, thus presenting APGA as abstract models of P(G)A. We show that APGA are related with their

Refinement relations

	State-based ($\preceq_{\mathrm{sb}}$)	Distribution-based ($\preceq_{\mathrm{db}}$)
preorder	+	+
monotonicity of closing	+	-
pre-congruence w.r.t. $\|\|_{\bar{A}}$	+	+
For Segala's PA	$\preceq_{\mathrm{sb}} \cap \preceq_{\mathrm{sb}}^{-1} = \sim_{\mathrm{pa}}$	
For APGA	$\mathscr{H} \preceq_{\mathrm{sb}} \mathscr{H}' \Rightarrow \llparenthesis \mathscr{H} \rrparenthesis_{\mathrm{sb}} \subseteq \llparenthesis \mathscr{H}' \rrparenthesis_{\mathrm{sb}}$	$\mathscr{H} \preceq_{\mathrm{db}} \mathscr{H}' \Rightarrow \llparenthesis \mathscr{H} \rrparenthesis_{\mathrm{db}} \subseteq \llparenthesis \mathscr{H}' \rrparenthesis_{\mathrm{db}}$
For $x \in \{\mathrm{sb}, \mathrm{db}\}$: $\mathscr{H} \preceq_x \mathscr{H}'$	$\mathscr{G}_\circ^\circ \prec_{\mathrm{sb}} \mathscr{G}_\circ^{\circ\prime}$ and $\mathscr{G}_\circ^\bullet \preccurlyeq_{\mathrm{sb}} \mathscr{G}_\circ^{\bullet\prime}$	$\mathscr{G}_\circ^\circ \prec_{\mathrm{db}} \mathscr{G}_\circ^{\circ\prime}$ and $\mathscr{G}_\circ^\bullet \preccurlyeq_{\mathrm{db}} \mathscr{G}_\circ^{\bullet\prime}$
For APGA	$\preceq_{\mathrm{sb}} \neq \preceq_{\mathrm{db}}$	
For closed APGA	$\preceq_{\mathrm{sb}} \subseteq \preceq_{\mathrm{db}}$	

Preservation of reachability probabilities

$\preceq_{\mathrm{sb}}$ and $\preceq_{\mathrm{db}}$ yield	$\min^{\blacktriangledown}$ and $\min^{\blacktriangle}$ as well as $\max^{\blacktriangledown}$ and $\max^{\blacktriangle}$

Table 4.1: Summary of refinement relations.

concrete models through refinement relations, thus, showing that abstract models preserve the reachability probabilities of concrete models.

5 Modal Abstraction of Stochastic Games

Informally speaking, abstraction is a technique that allows omitting unnecessary details from system models for the verification of properties. The models induced as a result are aimed to be finite and smaller in size as compared to the concrete models, moreover, they have at least the behaviour of the concrete models.

In the literature, a popular abstraction mechanism is to merge states of systems' models in different ways. In the most basic technique, states are merged together by simply collecting their behaviour, that as a result induces additional non-deterministic behaviour in the abstract states. Due to this, abstract models have at least the behaviour of concrete models and are, thus, comparable using *simulation relations*. When such abstract models are analysed, say for extremal reachability probabilities, they provide safe over-approximations for concrete models.

In [SK12], we have extended the above technique for probabilistic automata (PA); we categorize the behaviour of abstract states into *required* and *possible* behaviour as in modal transition systems (MTS) [LT88a] — the behaviour common among all concrete states of an abstract state becomes its *required* behaviour, whereas their complete behaviour becomes its *possible* behaviour. The abstract models induced as a result are abstract PA (APA) [DKL$^+$13] — PA with *required* and *possible* modalities; and are comparable with concrete models using *refinement relations*. Moreover, this approach allows for bounding the extremal reachability probabilities of concrete models from above and below. Note that because of the non-deterministic behaviour from abstraction, these bounds do not coincide with that of concrete models.

Another interesting and fruitful direction is given in [KKNP10] aggregating states in such a way that the non-deterministic behaviour in concrete systems is handled by one set of states while the non-determinism from abstraction is handled by another set of states. This naturally yields turn-based stochastic two-player games (SGs) [Sha53, Con92], where one player controls the non-determinism in the concrete models, whereas the other is in charge of the non-determinism from the abstraction. SG-based abstraction also yields upper and lower bounds on extremal reachability probabilities, and significantly improves these bounds as evidently shown by several case studies [KKNP10]. Besides, SG-based abstraction is proven to be the optimal in the sense of abstract interpretation [WZ10], i.e., with the given partition of the state space of a system, no other abstraction technique can induce more precise model than the SG-based abstraction.

In [SK14], we extend the technique of [KKNP10] from states to distributions over states for PA; we consider the non-deterministic behaviour of concrete systems at the level of distributions over states rather than at the level of states themselves. The abstract models are then *probabilistic game automata* (PGA)

— stochastic games in which both the players have non-deterministic and probabilistic behaviour. In PGA-based abstract models, one set of states represents the non-deterministic behaviour of distributions over states in concrete systems, whereas the other set represents the (probabilistic) behaviour from abstraction. Our PGA-based abstract models yield tighter upper and lower bounds on extremal reachability probabilities than SG-based models [KKNP10]. Moreover, they (PGA-based abstract models) are comparable with concrete models using our *distribution-based* (*alternating*) *simulation* relations (see Def. 16 and 18 on pages 23 and 28 respectively).

In this chapter, we combine the techniques of [KKNP10], [SK12] and [SK14] in two ways. In the first way, we combine the techniques of [KKNP10] and [SK12]; the induced models are then a class of *abstract probabilistic game automata* (APGA) — PGA with *required* and *possible* modalities — in which one of the players have only non-deterministic behaviour. This is called *state-based abstraction* of APGA. State-based abstraction differs from [KKNP10] in a sense that the non-deterministic behaviour in concrete systems is not completely handled by one set of states: in state-based abstraction, concrete states are merged together *if* they have the same step-wise behaviour after abstraction; whereas in [KKNP10], concrete states are merged together *iff* they have the same step-wise behaviour. Because of this, the bounds on extremal reachability probabilities in state-based APGA-based models are at most as tight as in SG-based models, however, they are at most the size of SG-based models. Our abstract models are comparable with concrete models using *state-based refinement* relations (see Def. 23 on page 40). We show that game-based abstraction [KKNP10] is a special case of our state-based abstraction.

In the second way, we combine the techniques of [SK12] and [SK14]; the induced models are APGA with both players having non-deterministic and probabilistic behaviour. This is called *distribution-based abstraction* of APGA. The difference between distribution-based abstraction and [SK14] is the same as between state-based abstraction and [KKNP10], i.e., in distribution-based abstraction, (support sets of) concrete distributions are merged together *if* they have the same step-wise behaviour after abstraction; whereas in [SK14], they are merged together *iff* they have the same step-wise behaviour. Thus, the bounds on extremal reachability probabilities in distribution-based APGA-based models are at most as tight as in PGA-based models; and they are at most the size of PGA-based models. Moreover, they are comparable with concrete models using *distribution-based refinement* relation (see Def. 25 on page 42). We show that our PGA-based abstraction in [SK14] is a special case of our distribution-based abstraction. Furthermore, we illustrate with examples that our distribution-based abstraction may induce more precise as well as concise models than our state-based abstraction.

Moreover, we show that our state-based abstractions are defined in a sense that for all partitions of a concrete state space, it is possible to induce abstract models. But, this is not the case with our distribution-based abstractions, where it might be the case that for some partition of a state space, the abstract model is not defined. However, for closed systems – which do not interact with outside environment —, distribution-based abstractions are also defined.

Put in a nutshell, the major contributions of this chapter are:

- a *state-based abstraction* of APGA showing concrete models refine abstractions using *state-based refinement* relation,
- a *distribution-based abstraction* of APGA showing concrete models refine abstractions using *distribution-based refinement* relation,

- results showing that APGA-based abstract models bound the extremal reachability probabilities of concrete models,
- results showing that the distribution-based abstract models are more precise (as well as concise in some cases) than the state-based abstract models, and
- APGA-based abstraction techniques are compositional.

Finally, for simplicity in figures we only provide examples of abstraction of PA — a subclass of APGA. Moreover, to compare the sizes of concrete models with their abstractions (in terms of number of states and transitions), we take the sizes of probabilistic transitions equal to the cardinality of the support sets of their target distributions, e.g., the size of a transition $s \xrightarrow{a} \mu$ is equal to $|\mathrm{Supp}(\mu)|$.

5.1 Abstraction of APGA

In this section, we consider abstractions of APGA that can mimic their step-wise behaviour. Let $\mathscr{H}$ be an APGA with $S = S_1 \cup S_2$. Intuitively, the sub-state space S_1/S_2 of $\mathscr{H}$ is partitioned and each partition is represented by a single state in the abstract sub-state space S'_1/S'_2 — $S' = S'_1 \cup S'_2$. In fact, at first a suitable partition of S_2 is decided, that then constrains the partition of S_1. For a given partition of S_2, we propose two different ways for the partition of S_1 and for defining the transitions of abstract states. In the first way, the conditions for the partition of S_1 are defined at the level of states, and S' states derive their transitions from that of S, called *state-based abstraction*. Whereas in the second way, the conditions are defined at the level of distributions over S_1 states, and S' states derive their transitions from that of distributions over S, called *distribution-based abstraction*. In the sequel, we show that the latter technique induces more precise models than the former one.

Let (α, γ) be an abstraction-concretization pair such that $\alpha : S \to S'$ is a surjection and $\gamma : S' \to 2^S$ is the corresponding concretization function. That is, $\alpha(s)$ is the abstract state of s whereas $\gamma(s')$ is the set of concrete states abstracted by s'. The abstraction of distribution μ is given as $\alpha(\mu)(s') = \mu(\gamma(s'))$. The functions α and γ are lifted to sets of states or sets of distributions in a point-wise manner.

5.1.1 State-based Abstraction of APGA

In state-based abstraction, we assert that S_1 states that have the same set of transitions (under S'_2) must be assigned to the same abstract state. For defining transitions of abstract states, we adopt the way used for APA in [SK12]; for $s' \in S'_1$, the *required* transitions that are similar among the concrete states of s' become the *required* transitions (after abstraction) of s', and every transition of a concrete state becomes a *possible* transition (after abstraction) of s'; whereas for S'_2 states, only *possible* transitions are defined for them. This is intuitive as the *required* transitions for S'_2 states play no role in the analysis of extremal reachability probabilities (see Th. 9 on page 49).

Definition 31. (State-based Game Abstraction). *For APGA $\mathscr{H}$, the abstraction function $\alpha : S \to S'$ induces the APGA $\mathscr{H}' = \alpha(\mathscr{H})$ where*

- *$A' = A$;*
- *$S'_i = \alpha(S_i)$ for $i \in \{1,2\}$;*

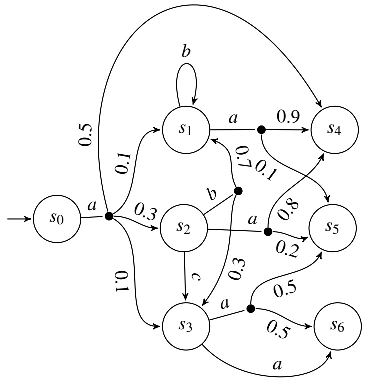
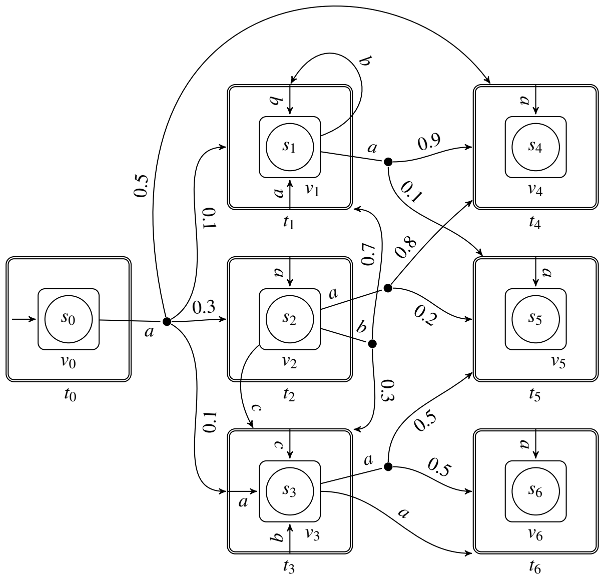



Figure 5.1: A PA $\mathscr{M}$ (left) and its embedding $\mathscr{H} = \alpha_{\mathrm{PA}}(\mathscr{M})$ (right)

- *$\forall s, u \in S_1 : \alpha(\Delta_x(s)) = \alpha(\Delta_x(u))$ for $x \in \{\mathrm{p}, \mathrm{r}\}$ implies $\alpha(s) = \alpha(u)$;*

and for every $s' \in S'$:

1. *if $s' \in S'_1$, then:*

   (a) *$s' \xrightarrow{a}_{\mathrm{r}} \mu'$ iff $\forall s \in \gamma(s') : s \xrightarrow{a}_{\mathrm{rc}} \mu$ such that $\alpha(\mu) = \mu'$,*

   (b) *$\exists s \in \gamma(s') : s \xrightarrow{a}_{\mathrm{p}} \mu$ implies $s' \xrightarrow{a}_{\mathrm{pc}} \mu'$ such that $\alpha(\mu) = \mu'$,*

   (c) *$s' \xrightarrow{a}_{\mathrm{p}} \mu'$ implies $\exists s \in \gamma(s') : s \xrightarrow{a}_{\mathrm{p}} \mu$ such that $\alpha(\mu) = \mu'$,*

2. *if $s' \in S'_2$, then:*

   (a) *$\exists s \in \gamma(s') : s \xrightarrow{a}_{\mathrm{p}} \mu$ implies $s' \xrightarrow{a}_{\mathrm{pc}} \mu'$ such that $\alpha(\mu) = \mu'$.*

   (b) *$s' \xrightarrow{a}_{\mathrm{p}} \mu'$ implies $\exists s \in \gamma(s') : s \xrightarrow{a}_{\mathrm{p}} \mu$ such that $\alpha(\mu) = \mu'$.*

In the sequel, $(\alpha_{\mathrm{sb}}, \gamma_{\mathrm{sb}})$ denotes a pair of state-based abstraction-concretization *functions, and let Abst$_{\mathrm{sb}}$ be the set of state-based abstraction functions for APGA.*

The above definition asserts that every two states in S_1 having the same transitions (after abstraction) are grouped together. The condition (1) deals with player-one states in the abstract model, and by (1a) for every *required* a-transition from an abstract state s' to μ', there is a *required* combined transition from every concrete state s of s' to *some* μ such that μ is abstracted by μ'. By (1b) the behaviour of each concrete state (after abstraction) is present in its corresponding abstract state, i.e., if a concrete state s can perform a *possible* a-transition to *some* μ, then its abstract state s' performs a *possible* combined

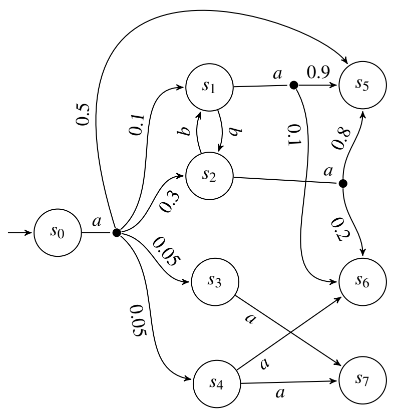


Figure 5.2: A PA $\mathscr{M}$

a-transition to *some* μ' such that μ is abstracted by μ'. By (1c) the *possible* behaviour of an abstract state is derived from its corresponding concrete states. The condition (2) deals with player-two states and (2a) and (2b) are identical to (1b) and (1c) respectively.

Example 23. *Let $\mathscr{H}' = \alpha_{\text{sb}}(\mathscr{H})$ (Fig. 5.3 left) be the induced state-based abstract model of APGA $\mathscr{H}$ (Fig. 5.1) with $\gamma_{\text{sb}}(t'_0) = \{t_0\}$, $\gamma_{\text{sb}}(t'_1) = \{t_1,t_2,t_3\}$, $\gamma_{\text{sb}}(t'_2) = \{t_4,t_5\}$ and $\gamma_{\text{sb}}(t'_3) = \{t_6\}$ as well as $\gamma_{\text{sb}}(v'_0) = \{v_0\}$, $\gamma_{\text{sb}}(v'_1) = \{v_1,v_2\}$, $\gamma_{\text{sb}}(v'_2) = \{v_3\}$, $\gamma_{\text{sb}}(v'_3) = \{v_4,v_5\}$ and $\gamma_{\text{sb}}(v'_4) = \{v_6\}$. Let us consider the abstract state v'_1, it has a* required *a-transition to t'_2 because both of its concrete states (v_1 and v_2) have* required *a-transitions with target distributions over t_4 and t_5 (the concrete states of t'_3). By a similar reason there exists a* required *b-transition from v'_1 to t'_1. However, only v_2 has a* required *c-transition to t_3, therefore, v'_1 has a* possible *c-transition to t'_1 (the abstract state of t_3). The rest of the example is trivial.*

Lemma 3. *For a state-based abstraction function $\alpha \in Abst_{\text{sb}}$, let $\eta \in \text{Dist}(S)$ and $\eta' \in \text{Dist}(\alpha(S))$. Let $R \subseteq S \times \alpha(S)$, then:*

$$\eta' = \alpha(\eta) \textit{ implies } \eta R \eta'.$$

Proof. In order to prove that $\eta R \eta'$, we define a weight function for distributions η and $\alpha(\eta)$ w.r.t. relation R such that for all $s \in S$ and $s' \in \alpha(S)$:

$$\delta(s,s') = \eta(s) \cdot \iota_{s'}(\alpha(s))$$

and prove that it fulfils the three properties of a weight function (Def. 12 on page 17).

1. Let $s' = \alpha(s)$ and $\eta(s) > 0$, it implies that $\delta(s,s') > 0$ and sRs',

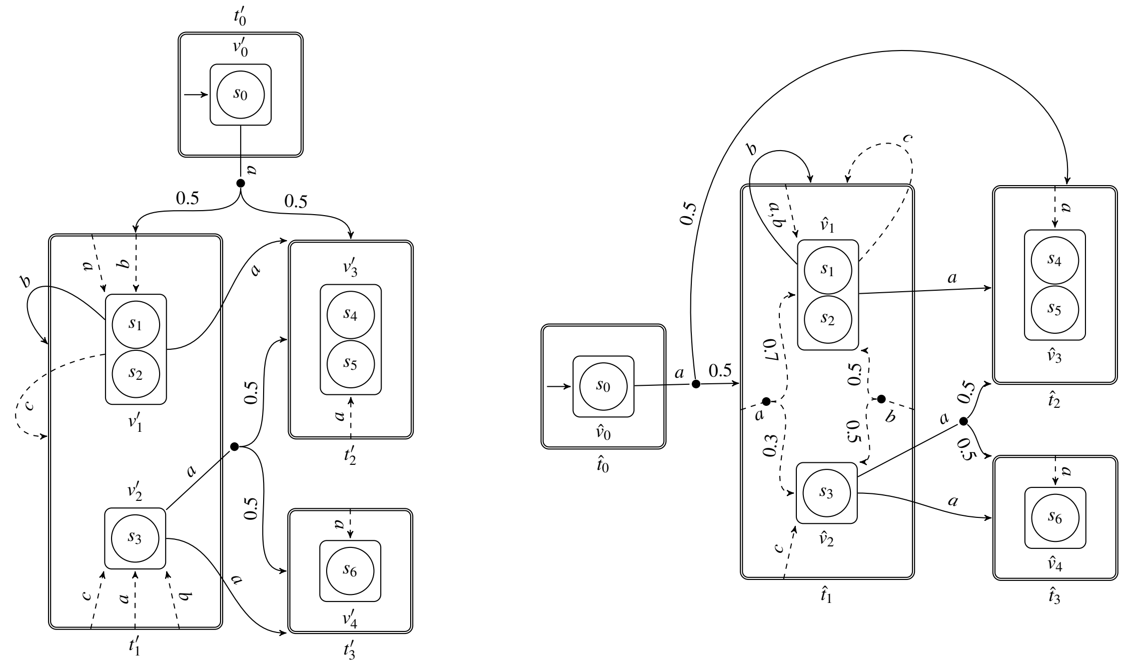


Figure 5.3: For APGA $\mathscr{H}$ (Fig. 5.1 right), $\mathscr{H}' = \alpha_{\text{sb}}(\mathscr{H})$ (left) and $\hat{\mathscr{H}} = \alpha_{\text{db}}(\mathscr{H})$ (right)

2. The proof for $\delta(s, \alpha(S)) = \eta(s)$ goes as:

$$\begin{aligned}\textstyle\sum_{s' \in \alpha(S)} \delta(s, s') &= \textstyle\sum_{s' \in \alpha(S)} \eta(s) \cdot \iota_{s'}(\alpha(s)) \\ \delta(s, \alpha(S)) &= \textstyle\sum_{s' \in \alpha(S): s' = \alpha(s)} \eta(s) \\ \delta(s, \alpha(S)) &= \eta(s)\end{aligned}$$

3. The proof for $\delta(S, s') = \eta'(s')$ goes as:

$$\begin{aligned}\textstyle\sum_{s \in S} \delta(s, s') &= \textstyle\sum_{s \in S} \eta(s) \cdot \iota_{s'}(\alpha(s)) \\ \delta(S, s') &= \textstyle\sum_{s \in S: s' = \alpha(s)} \eta(s) \\ &= \eta'(s')\end{aligned}$$

□

The following theorem establishes that concrete models refine their state-based abstractions.

Theorem 12. $\mathscr{H} \preceq_{\text{sb}} \alpha_{\text{sb}}(\mathscr{H})$.

Proof. Let $\mathscr{H} = (S, \{S_1, S_2\}, A, \Delta_\mathrm{r}, \Delta_\mathrm{p}, s_0)$ be an APGA and let $\alpha_\mathrm{sb} : S \to S'$ such that $\alpha_\mathrm{sb}(\mathscr{H}) = \mathscr{H}' = (S', \{S'_1, S'_2\}, A, \Delta'_\mathrm{r}, \Delta'_\mathrm{p}, s'_0)$ is the induced APGA. We define the relation $R \subseteq (S_1 \times S'_1) \cup (S_2 \times S'_2)$ as:

$$R = \{(s, \alpha_\mathrm{sb}(s)) \mid s \in S\}$$

and show that it fulfils the conditions of Def. 23.

Let sRs', then $s \in \gamma_\mathrm{sb}(s')$.

1. Let $s' \xrightarrow{a}_\mathrm{r} \mu'$. By condition (1a) of Def. 31, $s \xrightarrow{a}_\mathrm{rc} \mu$ such that $\alpha_\mathrm{sb}(\mu) = \mu'$. Thus, by Lem. 3 $\mu R \mu'$ holds.
2. Let $s \xrightarrow{a}_\mathrm{p} \mu$ and $s \in S_1$. By condition (1b) of Def. 31, $s' \xrightarrow{a}_\mathrm{pc} \mu'$ such that $\alpha_\mathrm{sb}(\mu) = \mu'$ hold. Thus, by Lem. 3 $\mu R \mu'$ holds.
3. Let $s \xrightarrow{a}_\mathrm{p} \mu$ and $s \in S_2$. By condition (2a) of Def. 31, $s' \xrightarrow{a}_\mathrm{pc} \mu'$ such that $\alpha_\mathrm{sb}(\mu) = \mu'$ hold. Thus, by Lem. 3 $\mu R \mu'$ holds.

$\square$

State-based abstraction of APGA vs. closed APGA. Observe that the abstract models obtained by first closing APGA and then abstracting them may be different from the ones obtained by first abstracting and then closing, i.e., $\tau(\alpha_\mathrm{sb}(\mathscr{H}))$ may not be the same as $\alpha_\mathrm{sb}(\tau(\mathscr{H}))$ for APGA $\mathscr{H}$. For example, if every concrete state in a partition has a transition with the same distribution, say μ, as its target, then the corresponding abstract state would have a *required* transition with target distribution $\alpha_\mathrm{sb}(\mu)$ in case of a closed concrete game; otherwise it might have a *possible* transition if action labels of concrete transitions are different. However, $\tau(\alpha_\mathrm{sb}(\mathscr{H}))$ and $\alpha_\mathrm{sb}(\tau(\mathscr{H}))$ are in a refinement relation as given by the following proposition.

Proposition 16. *For APGA $\mathscr{H}$, $\alpha_\mathrm{sb}(\tau(\mathscr{H})) \preceq_\mathrm{sb} \tau(\alpha_\mathrm{sb}(\mathscr{H}))$.*

Thus, by Th. 10 (page 50) abstractions of closed APGA give more precise models than abstractions of open APGA.

State-based APGA-based vs. game-based abstraction [KKNP10] of PA. State-based APGA-based abstraction differs from [KKNP10] in a sense that the non-deterministic behaviour in a PA is not completely handled by one set of states: APGA-based abstraction merges concrete states if they have the same step-wise behaviour after abstraction; whereas SG-based abstraction does so iff they have the same step-wise behaviour. Consequently, the bounds on reachability probabilities in APGA-based abstractions are at most as tight as in SG-based abstractions, however, they are at most the sizes of SG-based abstractions.

Proposition 17. α_sb *coincides with game-based abstraction [KKNP10] of PA iff all player-one transitions are* required *transitions in the abstraction.*

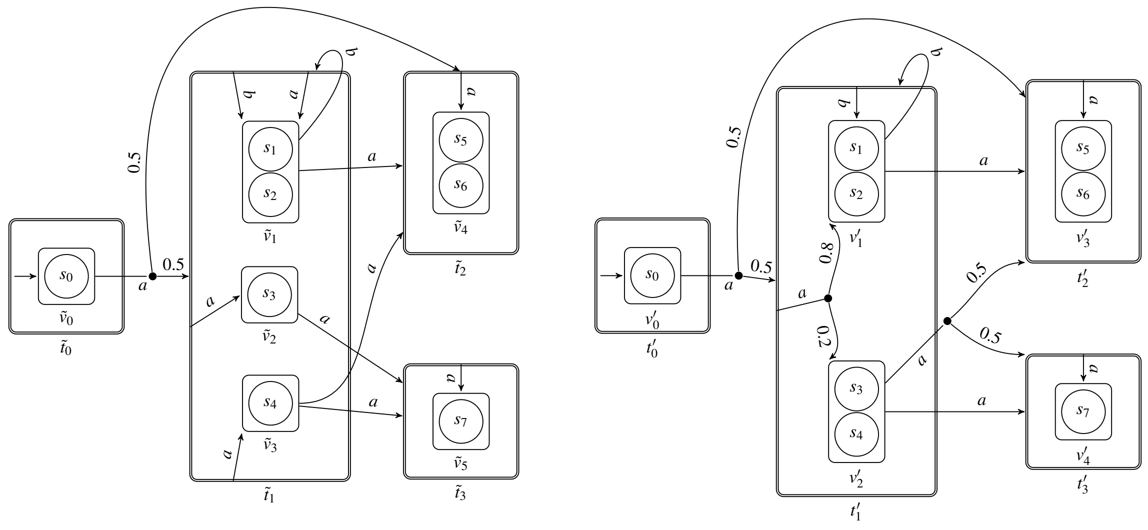


Figure 5.4: For PA $\mathscr{M}$ (Fig. 5.2), $\mathscr{H} = \alpha_{\text{PA}}(\mathscr{M})$ with $\tilde{\mathscr{H}} = \alpha_{\text{sb}}(\mathscr{H})$ (left) and $\mathscr{H}' = \alpha_{\text{db}}(\mathscr{H})$ (right).

Example 24. *Let $\mathscr{H} = \alpha_{\text{PA}}(\mathscr{M})$ for PA $\mathscr{M}$ (Fig. 5.2) and $\tilde{\mathscr{H}} = \alpha_{\text{sb}}(\mathscr{H})$ (Fig. 5.4 left). Note that in $\tilde{\mathscr{H}}$ only those player one states of $\mathscr{H}$ are merged that have the same behaviour under $\tilde{S}_2$. Therefore, α_{sb} induces a game-based abstraction of PA $\mathscr{M}$, thus showing game-based abstraction is a special case of state-based abstraction.*

5.1.2 Distribution-based Abstraction of APGA

We start this section with the definition of *maximal singular-distributions*, which is then used in defining the conditions for the partition of S_1 states as well as for defining the transitions of abstract states. Intuitively, the support sets of maximal singular-distributions are unique in an APGA; and we require that the support sets of such distributions be abstracted by a single state. Moreover, maximal singular-distributions whose hyper-transitions are the same after abstraction must be abstracted by the same state. To formally define the notion of a maximal singular-distribution, we need to define the notion of a *maximal* sub-distribution of a distribution.

Definition 32. *A sub-distribution $\eta \in \text{SDist}(S)$ is a* maximal *sub-distribution of a distribution μ iff $\forall s \in \text{Supp}(\eta) : \eta(s) = \mu(s)$. Let* $\text{MaxSDist}(\mu)$ *denote the set of maximal sub-distributions of μ.*

Informally, a distribution μ is a singular-distribution in an APGA $\mathscr{H}$ if for every other distribution ρ with $\text{Supp}(\mu) \cap \text{Supp}(\rho) \neq \emptyset$, there exists a maximal sub-distribution η of ρ such that $\eta_{\downarrow} = \mu$. Formally,

Definition 33. *In an APGA $\mathscr{H}$, a distribution $\mu \in \text{Dist}(S)$ is a* singular-distribution *iff*

$$\forall \nu \in \text{Dist}(S) : \text{Supp}(\mu) \cap \text{Supp}(\nu) \neq \emptyset \textit{ implies } \exists \nu' \in \text{MaxSDist}(\nu) \textit{ with } \nu'_{\downarrow} = \mu.$$

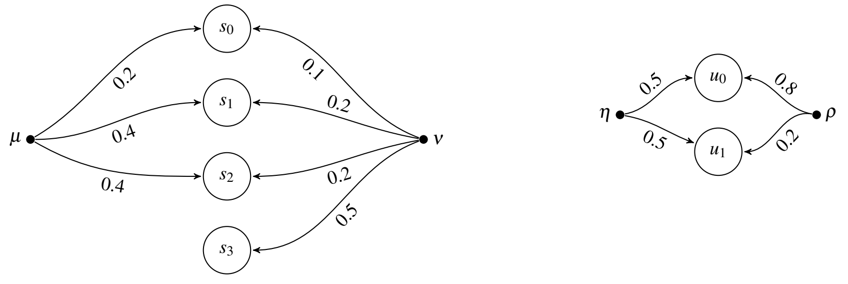


Figure 5.5: Distribution μ is a maximal singular-distribution, whereas ν, η and ρ are not singular-distributions. Note that every maximal sub-distribution ν' of ν is a singular-distribution if $\mathrm{Supp}(\nu') \subseteq \mathrm{Supp}(\mu)$ or $\mathrm{Supp}(\nu') \cap \mathrm{Supp}(\mu) = \emptyset$.

Let $\mathrm{SingDist}(S) \subseteq \mathrm{Dist}(S)$ *denote the set of singular-distributions over S in an APGA* $\mathscr{H}$*, and let* $\mathrm{MaxSingDist}(S) = \{\mu \in \mathrm{SingDist}(S) \mid \forall \rho \in \mathrm{SingDist}(S) : \mathrm{Supp}(\rho) \subseteq \mathrm{Supp}(\mu)$ *or* $\mathrm{Supp}(\rho) \cap \mathrm{Supp}(\mu) = \emptyset\}$ *denote the set of* maximal singular-distributions.

Thus, a maximal singular-distribution μ is a distribution such that every other singular-distribution is either a maximal sub-distribution of μ or does not share any state with μ. Note that every Dirac distribution over S is a singular-distribution, i.e., $\iota_s \in \mathrm{SingDist}(S)$ for $s \in S$.

Example 25. *In Fig. 5.5, the distribution μ is a maximal singular-distribution:* $\mathrm{Supp}(\mu) \cap \mathrm{Supp}(\nu) = \mathrm{Supp}(\mu) = \{s_0, s_1, s_2\}$ *and* $\mu = [\![0.1s_0, 0.2s_2, 0.2s_3]\!]_\downarrow$ *where* $[\![0.1s_0, 0.2s_2, 0.2s_3]\!]$ *is a maximal sub-distribution of ν; and for every singular-distribution μ' over $\{s_0, s_1, s_2, s_3\}$, either* $\mathrm{Supp}(\mu') \subseteq \mathrm{Supp}(\mu)$ *or* $\mathrm{Supp}(\mu') \cap \mathrm{Supp}(\mu) = \emptyset$*. For example,* $[\![0.2s_0, 0.4s_1]\!]_\downarrow$ *and* ι_{s_3} *are singular-distributions, and* $\{s_0, s_1\} \subseteq \mathrm{Supp}(\mu)$ *and* $\{s_3\} \cap \mathrm{Supp}(\mu) = \emptyset$ *respectively. Moreover, neither η nor ρ is a singular-distribution whereas ι_{u_0} and ι_{u_1} are.*

In case of distribution-based abstraction, we require that every maximal singular-distribution over S_1 is abstracted by a Dirac distribution, and two maximal singular-distributions that have the same behaviour (under S'_2) are abstracted by the same Dirac distribution; thus restricting the partition of S_1 states for a given S'_2. Moreover, we derive the transitions of S' states from their concrete distributions. For $s' \in S'_1$, the *required* transitions that are the same among distributions over concrete states of s' become the *required* transitions (after abstraction) of s', and every transition of a concrete distribution becomes a *possible* transition (after abstraction) of s'. Like in state-based abstraction, for S'_2 states only *possible* transitions are defined.

Definition 34. (Distribution-based Game Abstraction). *For APGA $\mathscr{H}$, the abstraction function $\alpha : S \to S'$ induces the APGA $\mathscr{H}' = \alpha(\mathscr{H})$ where*

- $A' = A$;
- $S'_i = \alpha(S_i)$ *for* $i \in \{1, 2\}$;
- $\forall \rho \in \mathrm{MaxSingDist}(S_1) : |\mathrm{Supp}(\alpha(\rho))| = 1$;

- $\forall \nu, \eta \in \text{MaxSingDist}(S_1)$: $\alpha(\Delta_x(\nu)) = \alpha(\Delta_x(\eta))$ *for all* $x \in \{\text{p}, \text{r}\}$ *implies* $\alpha(\nu) = \alpha(\eta)$*;*

and for all $\mu' \in \text{Dist}(S')$*:*

1. $\forall s' \in \text{Supp}(\mu')$, $\mu \in \gamma(\mu') : \exists \mu_{s'} \in \text{MaxSDist}(\mu) : \mu'(s') = |\mu_{s'}| \wedge \alpha(\mu_{s'})_{\downarrow} = \iota_{s'}$,
2. *if* $\mu' \in \text{Dist}(S'_1)$*, then:*
   (a) $\mu' \xrightarrow{a}_{\text{r}} \rho'$ *iff* $\forall \mu \in \gamma(\mu') : \mu \xrightarrow{a}_{\text{rc}} \rho$ *such that* $|\rho| \geq |\rho'|$ *and* $\alpha(\rho)_{\downarrow} = \rho'_{\downarrow}$,
   (b) $\exists \mu \in \gamma(\mu') : \mu \xrightarrow{a}_{\text{p}} \rho$ *implies* $\mu' \xrightarrow{a}_{\text{pc}} \rho'$ *such that* $|\rho| \leq |\rho'|$ *and* $\alpha(\rho)_{\downarrow} = \rho'_{\downarrow}$,
   (c) $\mu' \xrightarrow{a}_{\text{p}} \rho'$ *implies* $\exists \mu \in \gamma(\mu') : \mu \xrightarrow{a}_{\text{p}} \rho$ *such that* $|\rho| \leq |\rho'|$ *and* $\alpha(\rho)_{\downarrow} = \rho'_{\downarrow}$,
3. *if* $\mu' \in \text{Dist}(S'_2)$*, then:*
   (a) $\exists \mu \in \gamma(\mu') : \mu \xrightarrow{a}_{\text{p}} \rho$ *implies* $\mu' \xrightarrow{a}_{\text{pc}} \rho'$ *such that* $|\rho| \leq |\rho'|$ *and* $\alpha(\rho)_{\downarrow} = \rho'_{\downarrow}$,
   (b) $\mu' \xrightarrow{a}_{\text{p}} \rho'$ *implies* $\exists \mu \in \gamma(\mu') : \mu \xrightarrow{a}_{\text{p}} \rho$ *such that* $|\rho| \leq |\rho'|$ *and* $\alpha(\rho)_{\downarrow} = \rho'_{\downarrow}$.

In the sequel, $(\alpha_{\text{db}}, \gamma_{\text{db}})$ *denotes a pair of* distribution-based abstraction-concretization *functions, and let* Abst_{db} *be a set of distribution-based abstraction functions for APGA.*

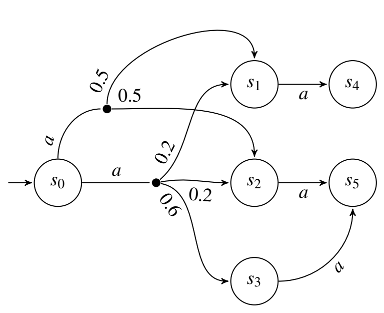
By Def. 34 every maximal singular-distribution over S_1 is abstracted by some Dirac distribution, say $\iota_{s'}$; this then also implies that $\gamma(\iota_{s'})$ contains all maximal singular-distributions that are defined over concrete states of s'. Thus, every distribution that is abstracted by $\iota_{s'}$ is a convex combination of maximal singular-distributions in $\gamma_{\text{db}}(\iota_{s'})$. Moreover, it also implies that all transitions of s' are derived from that of maximal singular-distributions in $\gamma_{\text{db}}(\iota_{s'})$.

Furthermore, by Def. 34 every two maximal singular-distributions $\nu, \eta \in \text{MaxSingDist}(S_1)$ having the same *required* and *possible* transitions under S'_2 are abstracted by the same Dirac distribution.

The condition (1) guarantees the splitting of every concrete distribution $\mu \in \gamma_{\text{db}}(\mu')$ into maximal sub-distributions as per the support of the abstract distribution μ' in such a way that the (conditional) distribution $\mu_{s'\downarrow}$ is also abstracted by its corresponding abstract distribution, i.e., $\iota_{s'}$. The condition (2) deals with player one distributions, and (2a) asserts that for every *required* a-transition from an abstract distribution μ' to ρ', there is a *required* combined a-transition from every corresponding concrete distribution μ to *some* ρ such that the mass of ρ' is at most that of ρ and (the conditional distribution of) ρ is abstracted by (that of) ρ'; (2b) asserts that for every *possible* a-transition from a concrete distribution μ to ρ, there is a *possible* combined a-transition from the abstract distribution μ' to *some* ρ' such that the mass of ρ' is at least that of ρ and (the conditional distribution of) ρ is abstracted by (that of) ρ'; whereas (2c) asserts that the behaviour of abstract distribution is derived from its corresponding concrete distributions. The condition (3) deals with player two distributions and (3a) and (3b) are similar to (2b) and (2c) respectively.

The following example shows that a maximal singular-distribution is always abstracted by a Dirac distribution in distribution-based abstraction.

Example 26. *Let* $\mathscr{H}' = \alpha_{\text{db}}(\alpha_{\text{PA}}(\mathscr{M}))$ *(right) for PA* $\mathscr{M}$ *(left) in Fig. 5.6 with* $\gamma_{\text{db}}(t'_0) = \{t_0\}$, $\gamma_{\text{db}}(t'_1) = \{t_1, t_2, t_3\}$, $\gamma_{\text{db}}(t'_2) = \{t_4, t_5\}$ *as well as* $\gamma_{\text{db}}(v'_0) = \{v_0\}$, $\gamma_{\text{db}}(v'_1) = \{v_1, v_2\}$, $\gamma_{\text{db}}(v'_2) = \{v_3\}$ *and* $\gamma_{\text{db}}(v'_3) =$



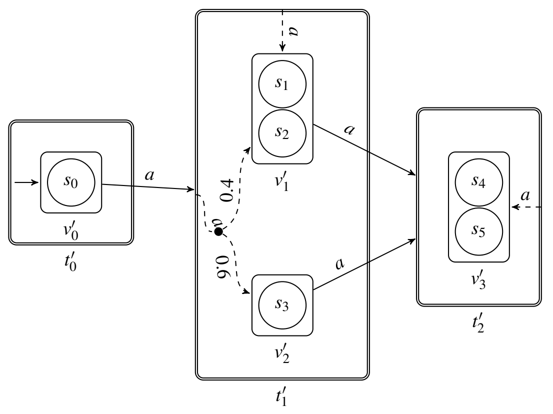


Figure 5.6: For PA $\mathscr{M}$ (left), $\mathscr{H}' = \alpha_{\mathrm{db}}(\alpha_{\mathrm{PA}}(\mathscr{M}))$ (right).

$\{v_4, v_5\}$. *Note that* $[\![0.5v_1, 0.5v_2]\!]$ *is a maximal singular-distribution (which is also a (conditional) maximal sub-distribution of* $[\![0.2v_1, 0.2v_2, 0.6v_3]\!]$*) and therefore its support set* $\{v_1, v_2\}$ *is abstracted by* v'_1*. Had we not merged* v_1 *and* v_2*, this would not be a distribution-based abstraction — which asserts that a maximal singular-distribution is abstracted by a Dirac distribution.*

Note that if $\mathrm{MaxSingDist}(S_1) = \mathrm{Dirac}(S_1)$ for APGA $\mathscr{H}$, then a state-based abstraction coincides with a distribution-based abstraction provided they have the same partition of player two state space.

Proposition 18. *For APGA* $\mathscr{H}$*, if* $\mathrm{MaxSingDist}(S_1) = \mathrm{Dirac}(S_1)$ *and* $\alpha_{\mathrm{sb}}(S_2) = \alpha_{\mathrm{db}}(S_2)$*, then* $\alpha_{\mathrm{sb}}(\mathscr{H})$ *coincides with* $\alpha_{\mathrm{db}}(\mathscr{H})$ *ignoring transitions from player two states.*

The following example illustrates how the transitions of abstract distributions are derived from their corresponding concrete distributions.

Example 27. *Let* $\hat{\mathscr{H}} = \alpha_{\mathrm{db}}(\mathscr{H})$ *(right Fig. 5.3 on page 60) for APGA* $\mathscr{H}$ *(Fig.5.1 on page 58) where* $\gamma_{\mathrm{db}}(\hat{t}_0) = \{t_0\}$*,* $\gamma_{\mathrm{db}}(\hat{t}_1) = \{t_1, t_2, t_3\}$*,* $\gamma_{\mathrm{db}}(\hat{t}_2) = \{t_4, t_5\}$ *and* $\gamma_{\mathrm{db}}(\hat{t}_3) = \{t_6\}$ *as well as* $\gamma_{\mathrm{db}}(\hat{v}_0) = \{v_0\}$*,* $\gamma_{\mathrm{db}}(\hat{v}_1) = \{v_1, v_2\}$*,* $\gamma_{\mathrm{db}}(\hat{v}_2) = \{v_3\}$*,* $\gamma_{\mathrm{db}}(\hat{v}_3) = \{v_4, v_5\}$ *and* $\gamma_{\mathrm{db}}(\hat{v}_4) = \{v_6\}$*. Consider* $\iota_{\hat{t}_1}$ *with* $\gamma_{\mathrm{db}}(\iota_{\hat{t}_1}) = \{[\![0.2t_1, 0.6t_2, 0.2t_3]\!], \iota_{t_1}, \iota_{t_3}\}$*. Let us check whether the transitions from* $[\![0.2t_1, 0.6t_2, 0.2t_3]\!]$ *are present in* $\iota_{\hat{t}_1}$*. For* $[\![0.2t_1, 0.6t_2, 0.2t_3]\!] \xrightarrow{a} [\![0.2v_1, 0.6v_2, 0.2v_3]\!]$*, there is* $\iota_{\hat{t}_1} \xrightarrow{a}_{\mathrm{pc}} [\![0.8\hat{v}_1, 0.2\hat{v}_2]\!]$ *with* $[\![0.2v_1, 0.6v_2, 0.2v_3]\!]_{\downarrow} \in \gamma_{\mathrm{db}}([\![0.8\hat{v}_1, 0.2\hat{v}_2]\!])$*. Similarly, for* $[\![0.2t_1, 0.6t_2, 0.2t_3]\!] \xrightarrow{b} [\![0.2v_1, 0.2v_3]\!]$*, there is* $\iota_{\hat{t}_1} \xrightarrow{b}_{\mathrm{p}} [\![0.5\hat{v}_1, 0.5\hat{v}_2]\!]$ *with* $[\![0.2v_1, 0.2v_3]\!]_{\downarrow} \in \gamma_{\mathrm{db}}([\![0.5\hat{v}_1, 0.5\hat{v}_2]\!])$*. Same is the case for the c-transition from* $[\![0.2v_1, 0.6v_2, 0.2v_3]\!]$ *and* $\iota_{\hat{t}_1}$*.*

Now consider $\iota_{\hat{v}_1}$ *with* $\gamma_{\mathrm{db}}(\iota_{\hat{v}_1}) = \{\iota_{v_1}, \iota_{v_2}, [\![0.25v_1, 0.75v_2]\!]\}$ *—* $\iota_{v_1}, \iota_{v_2} \in \gamma_{\mathrm{db}}(\iota_{\hat{v}_1})$ *as* $\iota_{v_1}, \iota_{v_2} \in \mathrm{MaxSingDist}$ (S_1)*. As* $\iota_{v_1} \xrightarrow{a}_{\mathrm{r}} [\![0.9t_4, 0.1t_5]\!]$*,* $\iota_{v_2} \xrightarrow{a}_{\mathrm{r}} [\![0.8t_4, 0.2t_5]\!]$ *and* $[\![0.25v_1, 0.75v_2]\!] \xrightarrow{a}_{\mathrm{r}} [\![0.825t_4, 0.175t_5]\!]$ *with* $[\![0.9t_4,$

$0.1t_5]\!]$, $[\![0.8t_4, 0.2t_5]\!]$ and $[\![0.825t_4, 0.175t_5]\!] \in \gamma_{\mathrm{db}}(\iota_{\hat{t}_2})$, therefore, $\iota_{\hat{v}_1} \xrightarrow{a}_{\mathrm{r}} \iota_{\hat{t}_2}$. This also applies to b-transitions. Now consider $\iota_{v_2} \xrightarrow{c}_{\mathrm{r}} \iota_{t_3}$. As there is no c-transition from ι_{v_1}, we have $\iota_{\hat{v}_1} \xrightarrow{c}_{\mathrm{p}} \iota_{\hat{t}_1}$.

In the following example, we illustrate that maximal singular-distributions that have the same behaviour are abstracted by the same Dirac distribution.

Example 28. *For PA $\mathscr{M}$ (Fig. 5.8 on page 70), let $\mathscr{H} = \alpha_{\mathrm{PA}}(\mathscr{M})$ be its induced game. Let $\hat{\mathscr{H}} = \alpha_{\mathrm{db}}(\mathscr{H})$ (Fig. 5.9 right) be the distribution-based abstract model of $\mathscr{H}$ with $\gamma_{\mathrm{db}}(\hat{t}_0) = \{t_0\}$, $\gamma_{\mathrm{db}}(\hat{t}_1) = \{t_5\}$, $\gamma_{\mathrm{db}}(\hat{t}_2) = \{t_1, t_2, t_3, t_4\}$, $\gamma_{\mathrm{db}}(\hat{t}_3) = \{t_6, t_7\}$, $\gamma_{\mathrm{db}}(\hat{t}_4) = \{t_8, t_9\}$ and $\gamma_{\mathrm{db}}(\hat{v}_0) = \{v_0\})$, $\gamma_{\mathrm{db}}(\hat{v}_1) = \{v_1, v_2\})$, $\gamma_{\mathrm{db}}(\hat{v}_2) = \{v_3, v_4, v_5\})$, $\gamma_{\mathrm{db}}(\hat{v}_3) = \{v_6, v_7\})$, and $\gamma_{\mathrm{db}}(\hat{v}_4) = \{v_8, v_9\})$. Note that as the behaviour of concrete maximal singular-distributions $[\![0.5v_3, 0.5v_4]\!]$ and ι_{v_5} are the same (under partition of player two state space), therefore, they are abstracted by the same distribution, i.e., $\iota_{\hat{v}_2}$.*

Distribution-based abstraction of APGA vs. closed APGA. It is observed that not for every partition of S_2, a distribution-based abstract model can be given, i.e., there exists no α_{db} fulfilling the conditions of Def. 34. However, for closed versions of APGA, this is possible. Remember that the partition of S_1 is conditioned on the partition of S_2 in both state-based and distribution-based abstractions.

Proposition 19. *$\alpha_{\mathrm{db}}(\tau(\mathscr{H}))$ is defined for* APGA *$\mathscr{H}$.*

By this proposition, for every partition of S_2 of a closed APGA, we can construct a distribution-based abstract model. However, for some APGA (not closed) we can have partitions of state spaces that do not define distribution-based abstract models. Moreover, although we do not aggregate any states when the partition is trivially S, α_{db} is defined for this partition. In the sequel, we assume that $\alpha_{\mathrm{db}}(\mathscr{H})$ is defined for APGA $\mathscr{H}$.

Moreover, like state-based abstraction of APGA (see page 61), distribution-based abstraction may also induce different models for closed and open APGA; however, unlike state-based abstractions (see Proposition 16 on page 61), they might not be related with each other. However, for a class of APGA (that represents the embeddings of PA (see Property 1 on page 50)), the extreme probabilistic game automata (see Def. 29 on page 48) of the abstractions of the closed and open models are related. (Recall that EPGA $\mathscr{G}^{\bullet}_{\circ}$ of APGA $\mathscr{H}$ inherits its player-one transitions from the *required* transitions of that of $\mathscr{H}$, whereas player-two transitions from the *possible* transitions of that of $\mathscr{H}$.)

Theorem 13. *For PA $\mathscr{M}$, let $\mathscr{H} = \alpha_{\mathrm{PA}}(\mathscr{M})$ with $\mathscr{H}' = \alpha'_{\mathrm{db}}(\tau(\mathscr{H}))$ and $\mathscr{H}'' = \tau(\alpha''_{\mathrm{db}}(\mathscr{H}))$. Then $\alpha'_{\mathrm{db}}(S) = \alpha''_{\mathrm{db}}(S)$ implies $\mathscr{G}^{\bullet\prime}_{\circ} \preccurlyeq_{\mathrm{db}} \mathscr{G}^{\bullet\prime\prime}_{\circ}$.*

Proof. Let $\mathscr{H} = (S, \{S_1, S_2\}, A, \Delta_{\mathrm{r}}, \Delta_{\mathrm{p}}, s_0)$ be an embedding of a PA. Let $\alpha'_{\mathrm{db}} : S \to S'$ and $\alpha''_{\mathrm{db}} : S \to S''$ such that $\alpha'_{\mathrm{db}}(\tau(\mathscr{H})) = \mathscr{H}' = (S', \{S'_1, S'_2\}, \{\tau\}, \Delta'_{\mathrm{r}}, \Delta'_{\mathrm{p}}, s'_0)$ and $\tau(\alpha''_{\mathrm{db}}(\mathscr{H})) = \mathscr{H}'' = (S'', \{S''_1, S''_2\}, \{\tau\},$

$$\begin{tikzcd}
\text{PA } \mathscr{M} \arrow[r, "\alpha_{\mathrm{PA}}"] & \mathscr{H} \arrow[r, "\alpha'_{\mathrm{db}} \circ \tau"] \arrow[dd, "\tau \circ \alpha''_{\mathrm{db}}"] & \mathscr{H}' \arrow[r] & \mathscr{G}^{\bullet\prime}_{\circ} & \\
 & & & \preceq_{\mathrm{db}} & \alpha'_{\mathrm{db}}(S) = \alpha''_{\mathrm{db}}(S) \arrow[l, "\text{implies}"'] \\
 & \mathscr{H}'' \arrow[rr] & & \mathscr{G}^{\bullet\prime\prime}_{\circ} &
\end{tikzcd}$$

Figure 5.7: Graphical representation of Th. 13.

$\Delta''_{\mathrm{r}}, \Delta''_{\mathrm{p}}, s''_0)$ are the induced APGA with $S' = S''$. Let $\mathscr{G}^{\bullet\prime}_{\circ}$ and $\mathscr{G}^{\bullet\prime\prime}_{\circ}$ be the EPGA of $\mathscr{H}'$ and $\mathscr{H}''$ respectively. We define the relation $R \subseteq (\mathrm{Dist}(S'_1) \times \mathrm{Dist}(S''_1)) \cup (\mathrm{Dist}(S'_2) \times \mathrm{Dist}(S''_2))$ on the state spaces of $\mathscr{G}^{\bullet\prime}_{\circ}$ and $\mathscr{G}^{\bullet\prime\prime}_{\circ}$ as:

$$R = \{(\alpha'_{\mathrm{db}}(\mu), \alpha''_{\mathrm{db}}(\mu)) \mid \mu \in \mathrm{Dist}(S)\}$$

and show that it fulfils the conditions of Def. 18 (page 28).

As $S' = S''$, the condition (1) of Def. 18 trivially holds for every pair of distributions in R. This also allows us to check the remaining conditions of Def. 18 only for those pairs in R that relate Dirac distributions. Moreover, for $s' = s''$, $\gamma'_{\mathrm{db}}(\iota_{s'}) = \gamma''_{\mathrm{db}}(\iota_{s''})$. Let $\mu' R \mu''$ where μ' and μ'' are Dirac distributions.

1. Let μ' and μ'' be player one distributions, and $\mu'' \xrightarrow{\tau} \nu''$ in $\mathscr{G}^{\bullet\prime\prime}_{\circ}$. Then, by Def. 29, $\mu'' \xrightarrow{\tau}_{\mathrm{r}} \nu''$ in $\tau(\alpha''_{\mathrm{db}}(\mathscr{H}))$; and by Def. 8 for some $a \in A$, $\mu'' \xrightarrow{a}_{\mathrm{r}} \nu''$ in $\alpha''_{\mathrm{db}}(\mathscr{H})$. Let $\mu \in \gamma''_{\mathrm{db}}(\mu'')$, then by condition (2a) of Def. 34, $\mu \xrightarrow{a}_{\mathrm{rc}} \nu$ such that $|\nu''| = |\nu|$ and $\alpha''_{\mathrm{db}}(\nu) = \nu''$. Therefore, by Def. 8 $\mu \xrightarrow{\tau}_{\mathrm{rc}} \nu$ also holds in $\tau(\mathscr{H})$. As $\mu \in \gamma'_{\mathrm{db}}(\mu')$ and $\gamma'_{\mathrm{db}}(\mu') = \gamma''_{\mathrm{db}}(\mu'')$, therefore, by condition (2a) of Def. 34, $\mu' \xrightarrow{\tau}_{\mathrm{r}} \nu'$ in $\alpha'_{\mathrm{db}}(\tau(\mathscr{H}))$ such that $|\nu'| = |\nu|$ and $\alpha'_{\mathrm{db}}(\nu) = \nu'$; and by Def. 29, $\mu' \xrightarrow{\tau} \nu'$ in $\mathscr{G}^{\bullet\prime}_{\circ}$. As $\nu' = \nu''$ as well as $S' = S''$, therefore, $\nu' R \nu''$ holds.

2. Let μ' and μ'' be player two distributions, and $\mu' \xrightarrow{\tau} \nu'$ in $\mathscr{G}^{\bullet\prime}_{\circ}$. Then, by Def. 29, $\mu' \xrightarrow{\tau}_{\mathrm{p}} \nu'$ in $\alpha'_{\mathrm{db}}(\tau(\mathscr{H}))$. Then by condition (3b) of Def. 34, there exists $\mu \in \gamma'_{\mathrm{db}}(\mu')$ and $\mu \xrightarrow{\tau} \nu$ such that $|\nu'| = |\nu|$ and $\alpha'_{\mathrm{db}}(\nu) = \nu'$. As $\forall s \in \mathrm{Supp}(\mu) : |\mathrm{Succ}(s)| = 1$, therefore for some $a \in A$, $\mu \xrightarrow{a} \nu$ holds (Note that if for some $u \in \mathrm{Supp}(\mu) : |\mathrm{Succ}(u)| > 1$, this would not be a valid argument). Also $\mu \in \gamma''_{\mathrm{db}}(\mu'')$ and $\gamma'_{\mathrm{db}}(\mu') = \gamma''_{\mathrm{db}}(\mu'')$, therefore, by condition (3a) of Def. 34, $\mu'' \xrightarrow{a}_{\mathrm{pc}} \nu''$ in $\alpha''_{\mathrm{db}}(\mathscr{H})$ such that $|\nu''| = |\nu|$ and $\alpha''_{\mathrm{db}}(\nu) = \nu''$; and $\mu'' \xrightarrow{\tau}_{\mathrm{pc}} \nu''$ in $\tau(\alpha''_{\mathrm{db}}(\mathscr{H}))$, and by Def. 29 $\mu'' \xrightarrow{\tau}_{\mathrm{c}} \nu''$ in $\mathscr{G}^{\bullet\prime\prime}_{\circ}$. As $\nu'' = \nu'$ as well as $S' = S''$, therefore, $\nu' R \nu''$ holds.

□

The graphical representation of Th. 13 is given in Fig. 5.7.

Theorem 14. $\mathscr{H} \preceq_{\mathrm{db}} \alpha_{\mathrm{db}}(\mathscr{H})$.

Proof. Let $\mathscr{H} = (S, \{S_1, S_2\}, A, \Delta_r, \Delta_p, s_0)$ be an APGA and let $\alpha_{db} : S \to S'$ such that $\alpha_{db}(\mathscr{H}) = \mathscr{H}' = (S', \{S'_1, S'_2\}, A, \Delta'_r, \Delta'_p, s'_0)$ is the induced APGA. We define the relation $R \subseteq (\mathrm{Dist}(S_1) \times \mathrm{Dist}(S'_1)) \cup (\mathrm{Dist}(S_2) \times \mathrm{Dist}(S'_2))$ as:

$$R = \{(\mu, \alpha_{db}(\mu)) \mid \mu \in \mathrm{Dist}(S)\}$$

and show that it fulfils the conditions of Def. 25 (on page 42).

Let $\mu R \mu'$, then $\mu \in \gamma_{db}(\mu')$.

1. Let $s' \in \mathrm{Supp}(\mu')$. By condition (1) of Def. 34, for s' there exists a corresponding maximal sub-distribution in $\mathrm{MaxSDist}(\mu)$, i.e., $\mu_{s'}$ such that $|\mu_{s'}| = \mu'(s')$ and $\alpha_{db}(\mu_{s'})_{\downarrow} = \iota_{s'}$. Thus, $\mu_{s'\downarrow} R \iota_{s'}$ holds.

2. Let $\mu' \xrightarrow{a}_r \mu'$. By condition (2a) of Def. 34, $\mu \xrightarrow{a}_{rc} \mu$ such that $|\mu'| \leq |\mu|$ and $\alpha_{db}(\mu)_{\downarrow} = \mu'$. Thus, $\mu_{\downarrow} R \mu'$ holds.

3. Let $\mu \xrightarrow{a}_p \mu$ and $\mu \in \mathrm{Dist}(S_1)$. By condition (2b) of Def. 34, $\mu' \xrightarrow{a}_{pc} \mu'$ such that $|\mu'| \geq |\mu|$ and $\alpha_{db}(\mu)_{\downarrow} = \mu'$ hold. Thus, $\mu_{\downarrow} R \mu'$ holds.

4. Let $\mu \xrightarrow{a}_p \mu$ and $\mu \in \mathrm{Dist}(S_2)$. By condition (3a) of Def. 34, $\mu' \xrightarrow{a}_{pc} \mu'$ such that $|\mu'| \geq |\mu|$ and $\alpha_{db}(\mu)_{\downarrow} = \mu'$ hold. Thus, $\mu_{\downarrow} R \mu'$ holds.

□

In the following example, we observe that the distribution-based abstractions of an APGA obtained by first abstracting and then closing is smaller in size than the one obtained by first closing and then abstracting. By Th. 13 the EPGA of both the abstract models are related, that allows for computing the extremal reachability probabilities (in case of competing players by Th. 9) on the smaller model.

Example 29. *For PA $\mathscr{M}$ (Fig. 5.8), let $\mathscr{H} = \alpha_{PA}(\mathscr{M})$ be its induced game. Let $\hat{\mathscr{H}} = \alpha_{db}(\mathscr{H})$ (Fig. 5.9 right) be the distribution-based abstract model of $\mathscr{H}$ with $\gamma_{db}(\hat{t}_0) = \{t_0\}$, $\gamma_{db}(\hat{t}_1) = \{t_5\}$, $\gamma_{db}(\hat{t}_2) = \{t_1, t_2, t_3, t_4\}$, $\gamma_{db}(\hat{t}_3) = \{t_6, t_7\}$, $\gamma_{db}(\hat{t}_4) = \{t_8, t_9\}$ and $\gamma_{db}(\hat{v}_0) = \{v_0\}$), $\gamma_{db}(\hat{v}_1) = \{v_1, v_2\}$), $\gamma_{db}(\hat{v}_2) = \{v_3, v_4, v_5\}$), $\gamma_{db}(\hat{v}_3) = \{v_6, v_7\}$), and $\gamma_{db}(\hat{v}_4) = \{v_8, v_9\}$). Let $\mathscr{H}' = \alpha_{db}(\tau(\mathscr{H}))$ (Fig.5.8 right) be the distribution-based abstract model of closed $\mathscr{H}$ with the same partition as above. Let $\hat{\mathscr{G}}^{\bullet}_{\circ}$ and $\mathscr{G}^{\bullet\prime}_{\circ}$ be the EPGA of $\tau(\hat{\mathscr{H}})$ and $\mathscr{H}'$, then $\mathscr{G}^{\bullet\prime}_{\circ} \preccurlyeq_{db} \hat{\mathscr{G}}^{\bullet}_{\circ}$ as $R = \bigcup_{i=0\ldots4}\{(t'_i, \hat{t}_i), (v'_i, \hat{v}_i)\}$ is a DBAS relation.*

Distribution-based APGA-based vs. PGA-based abstraction [SK14] of PA. The difference between distribution-based APGA-based and PGA-based abstractions [SK14] are the same as between state-based APGA-based and SG-based abstractions [KKNP10], i.e., APGA-based abstractions merge (support sets of) maximal singular-distributions over player-one states if they have the same step-wise behaviour after abstraction; whereas PGA-based abstractions do so iff they have the same step-wise behaviour. Consequently, the bounds on reachability probabilities in distribution-based APGA-based abstractions are at most as tight as in PGA-based abstractions; and they are at most the sizes of PGA-based abstractions.

Proposition 20. *α_{db} coincides with PGA-based abstraction [SK14] of PA iff all player-one transitions are* required *transitions in the abstraction.*

Example 30. *Let $\mathscr{H} = \alpha_{\mathrm{PA}}(\mathscr{M})$ for PA $\mathscr{M}$ (Fig. 5.2) with $\mathscr{H}' = \alpha_{\mathrm{db}}(\mathscr{H})$ (Fig. 5.4 right). Note that in $\mathscr{H}'$ only those player one maximal singular-distributions in $\mathscr{H}$ are merged together that have the same behaviour under S'_2. Therefore, α_{db} induces a PGA-based abstract model of $\mathscr{M}$, thus showing distribution-based abstraction of PGA is a special case of that of APGA.*

5.1.3 State- vs. Distribution-based Abstractions of APGA

State-based abstraction is not a special case of distribution-based abstraction of APGA. In fact, for every partition of S_2, we can have a state-based abstract model of APGA, but not a distribution-based abstract model (which is only possible for closed versions of APGA)

Now we prove that the distribution-based abstraction of APGA is more precise than the state-based abstraction. In fact, when two abstract models, obtained by a state-based and a distribution-based abstraction, have the same state space; then the latter one is at least as precise as the former one (see Section 5.3).

Theorem 15. *For* APGA $\mathscr{H}$ *and* $\prec \in \{\prec_{\mathrm{sb}}, \preccurlyeq_{\mathrm{sb}}\}$, $\alpha_{\mathrm{sb}}(S) = \alpha_{\mathrm{db}}(S)$ *implies* $\alpha_{\mathrm{db}}(\mathscr{H}) \prec \alpha_{\mathrm{sb}}(\mathscr{H})$.

Proof. Let $\mathscr{H} = (S, \{S_1, S_2\}, A, \Delta_{\mathrm{r}}, \Delta_{\mathrm{p}}, s_0)$ be an APGA. Let $\alpha'_{\mathrm{db}} : S \to S'$ and $\alpha''_{\mathrm{sb}} : S \to S''$ such that $\mathscr{H}' = \alpha'_{\mathrm{db}}(\mathscr{H}) = (S', \{S'_1, S'_2\}, A, \Delta'_{\mathrm{r}}, \Delta'_{\mathrm{p}}, s'_0)$ and $\mathscr{H}'' = \alpha''_{\mathrm{sb}}(\mathscr{H}) = (S'', \{S''_1, S''_2\}, A, \Delta''_{\mathrm{r}}, \Delta''_{\mathrm{p}}, s''_0)$ are the induced APGA with $S' = S''$. We define the relation $R \subseteq (S'_1 \times S''_1) \cup (S'_2 \times S''_2)$ as:

$$R = \{(\alpha'_{\mathrm{db}}(s), \alpha''_{\mathrm{sb}}(s)) \mid s \in S\}$$

and show that it fulfils the conditions of Def. 23 (page 40).

Let $s' R s''$, then:

1. Let $s' \in S'_1$ and $s' \xrightarrow{a}_{\mathrm{p}} \mu'$. By condition (2c) of Def. 34, there exists $\mu \in \gamma'_{\mathrm{db}}(\iota_{s'})$: $\mu \xrightarrow{a}_{\mathrm{p}} \eta$ such that $|\mu'| \geq |\eta|$ and $\alpha'_{\mathrm{db}}(\eta)_{\downarrow} = \mu'$. As $\gamma_{\mathrm{db}}(s') = \gamma_{\mathrm{sb}}(s'')$, this implies by condition (1b) of Def. 31 that $s'' \xrightarrow{a}_{\mathrm{pc}} \mu''$ such that $\alpha''_{\mathrm{sb}}(\mu)_{\downarrow} = \mu''$. As $\alpha'_{\mathrm{db}}(\mu)_{\downarrow} = \mu'$ and $\alpha''_{\mathrm{sb}}(\mu)_{\downarrow} = \mu''$, thus $\mu' R \mu''$ as $S' = S''$.

2. Let $s' \in S'_2$ and $s' \xrightarrow{a}_{\mathrm{p}} \mu'$. By condition (3b) of Def. 34, there exists $\mu \in \gamma'_{\mathrm{db}}(\iota_{s'})$: $\mu \xrightarrow{a}_{\mathrm{p}} \mu$ such that $|\mu'| \geq |\mu|$ and $\alpha'_{\mathrm{db}}(\mu)_{\downarrow} = \mu'$. As $\gamma_{\mathrm{db}}(s') = \gamma_{\mathrm{sb}}(s'')$, this implies by condition (2a) of Def. 31 that $s'' \xrightarrow{a}_{\mathrm{pc}} \mu''$ such that $\alpha''_{\mathrm{sb}}(\mu)_{\downarrow} = \mu''$. As $\alpha'_{\mathrm{db}}(\mu)_{\downarrow} = \mu'$ and $\alpha''_{\mathrm{sb}}(\mu)_{\downarrow} = \mu''$, thus $\mu' R \mu''$ as $S' = S''$.

3. Let $s'' \in S''_1$ and $s'' \xrightarrow{a}_{\mathrm{r}} \mu''$. By condition (1a) of Def. 31, for every $s \in \gamma''_{\mathrm{sb}}(s'')$: $s \xrightarrow{a}_{\mathrm{rc}} \mu$ such that $\alpha''_{\mathrm{sb}}(\mu) = \mu''$. As $\gamma_{\mathrm{db}}(s') = \gamma_{\mathrm{sb}}(s'')$, this implies by condition (2a) of Def. 34 that $s' \xrightarrow{a}_{\mathrm{rc}} \mu'$ such that $\alpha'_{\mathrm{db}}(\mu) = \mu'$. As $\alpha'_{\mathrm{db}}(\mu) = \mu'$ and $\alpha''_{\mathrm{sb}}(\mu) = \mu''$, thus $\mu' R \mu''$ as $S' = S''$.

□

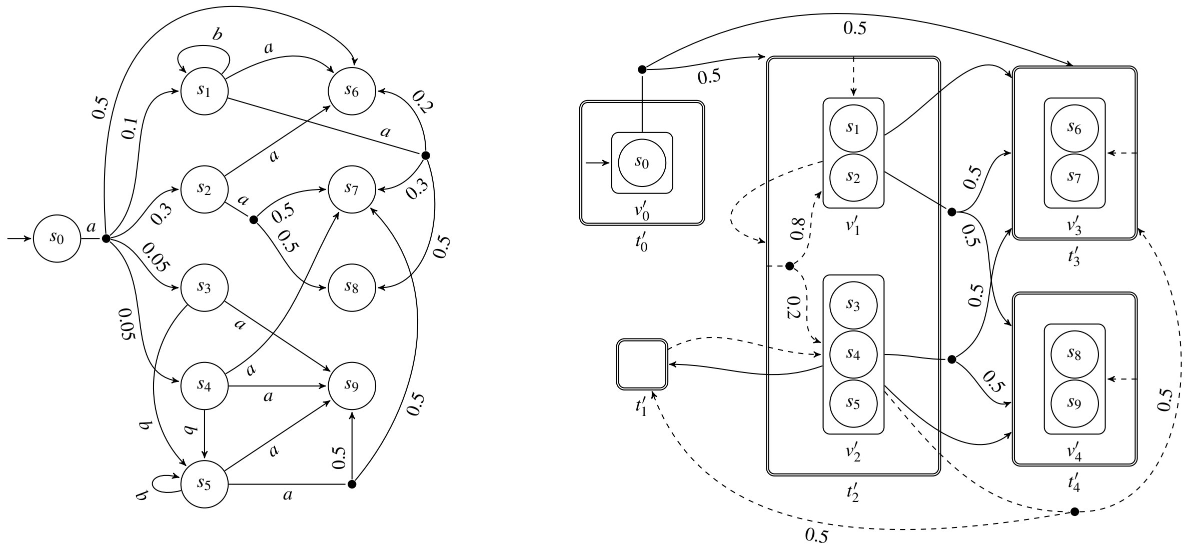


Figure 5.8: PA $\mathscr{M}$ (left) and distribution-based abstraction of $\tau(\mathscr{M})$ — $\mathscr{H} = \alpha_{\mathrm{db}}(\tau(\mathscr{M}))$ (right). Note that $|\Delta| = 19$ and $|S_1| = 5$ in $\mathscr{H}$.

5.2 Compositional Abstraction

Like for APA [SK12], our abstraction techniques for APGA are compositional. Intuitively, the composite APGA may be exponentially larger in size as compared to the composing ones. This problem could be avoided by applying abstraction prior to composition.

Theorem 16. *For* APGA $\mathscr{H}$ *and* $\mathscr{H}'$, *synchronization set* $\bar{A}$ *and abstraction functions* α_x, α'_x; $\alpha(\mathscr{H}) \,||_{\bar{A}}\, \alpha'_x(\mathscr{H}') = (\alpha \times \alpha'_x)(\mathscr{H} ||_{\bar{A}} \mathscr{H}')$ *up to isomorphism, where* $x \in \{\mathrm{sb}, \mathrm{db}\}$ *and* $\alpha_x \times \alpha'_x$ *is defined as* $(\alpha_x \times \alpha'_x)(s, s') = (\alpha_x(s), \alpha'_x(s'))$.

We only prove for distribution-based abstraction as the proof for the other case is similar.

Proof. Let $\mathscr{H} = (S, \{S_1, S_2\}, A, \Delta_{\mathrm{r}}, \Delta_{\mathrm{p}}, s_0)$ and $\hat{\mathscr{H}} = (\hat{S}, \{\hat{S}_1, \hat{S}_2\}, \hat{A}, \hat{\Delta}_{\mathrm{r}}, \hat{\Delta}_{\mathrm{p}}, \hat{s}_0)$ be APGA and let $\bar{A} \subseteq A \cap \hat{A}$ such that $\mathscr{H} ||_{\bar{A}} \hat{\mathscr{H}} = (S \times \hat{S}, \{S_1 \times \hat{S}_1, S \times \hat{S} \backslash S_1 \times \hat{S}_1\}, A \cup \hat{A}, \tilde{\Delta}_{\mathrm{r}}, \tilde{\Delta}_{\mathrm{p}}, (s_0, \hat{s}_0))$. Let $\alpha_{\mathrm{db}} : S \to S'$ and $\hat{\alpha}_{\mathrm{db}} : \hat{S} \to \hat{S}'$ be the distribution-based abstraction functions: $\alpha_{\mathrm{db}}(\mathscr{H}) = \mathscr{H}' = (S', \{S'_1, S'_2\}, A, \Delta'_{\mathrm{r}}, \Delta'_{\mathrm{p}}, s'_0)$, $\hat{\alpha}_{\mathrm{db}}(\hat{\mathscr{H}}) = \hat{\mathscr{H}}' = (\hat{S}', \{\hat{S}'_1, \hat{S}'_2\}, \hat{A}, \hat{\Delta}'_{\mathrm{r}}, \hat{\Delta}'_{\mathrm{p}}, \hat{s}'_0)$ and $(\alpha_{\mathrm{db}} \times \hat{\alpha}_{\mathrm{db}})(\mathscr{H} ||_{\bar{A}} \hat{\mathscr{H}}) = (S' \times \hat{S}', \{S'_1 \times \hat{S}'_1, S' \times \hat{S}' \backslash S'_1 \times \hat{S}'_1\}, A \cup \hat{A}, \tilde{\Delta}'_{\mathrm{r}}, \tilde{\Delta}'_{\mathrm{p}}, (s'_0, \hat{s}'_0))$. Let $\alpha_{\mathrm{db}}(\mathscr{H}) ||_{\bar{A}} \hat{\alpha}_{\mathrm{db}}(\hat{\mathscr{H}}) = (S' \times \hat{S}', \{S'_1 \times \hat{S}'_1, S' \times \hat{S}' \backslash S'_1 \times \hat{S}'_1\}, A \cup \hat{A}, \tilde{\Delta}''_{\mathrm{r}}, \tilde{\Delta}''_{\mathrm{p}}, (s'_0, \hat{s}'_0))$.

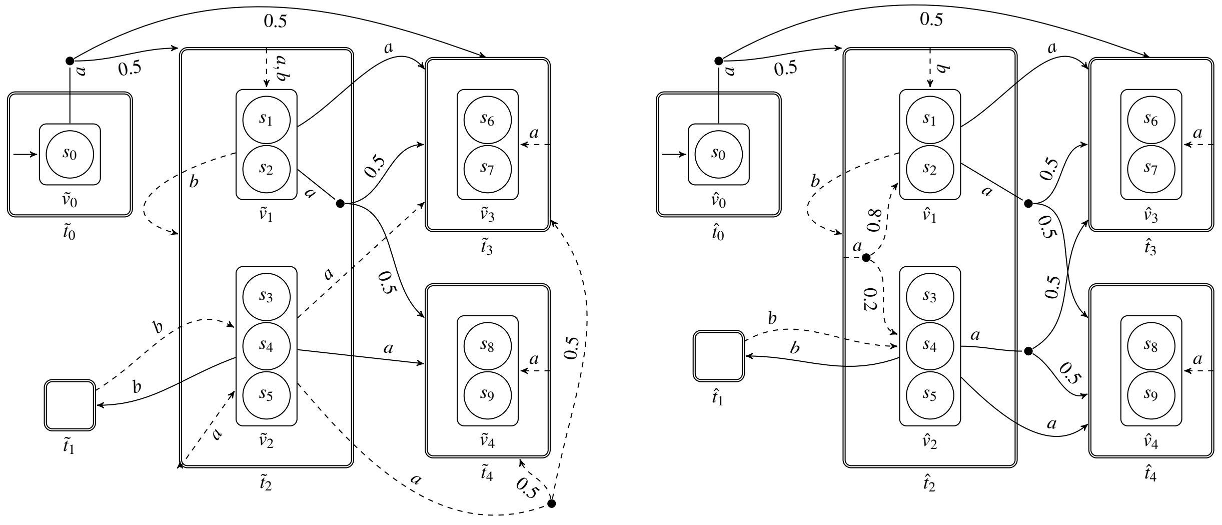


Figure 5.9: For PA $\mathscr{M}$ (Fig. 5.8), $\tilde{\mathscr{H}} = \alpha_{\text{sb}}(\alpha_{\text{PA}}(\mathscr{M}))$ (left) and $\hat{\mathscr{H}} = \alpha_{\text{db}}(\alpha_{\text{PA}}(\mathscr{M}))$ (right) with $|\tilde{\Delta}| = |\hat{\Delta}| = 17$ and $|\tilde{S}_1| = |\hat{S}_1| = 5$. Note that $\hat{S}_1$ states have only *required* transitions.

Note that the signature of $(\alpha_{\text{db}} \times \hat{\alpha}_{\text{db}})(\mathscr{H}\|_{\bar{A}}\hat{\mathscr{H}})$ and $\alpha_{\text{db}}(\mathscr{H})\|_{\bar{A}}\hat{\alpha}_{\text{db}}(\hat{\mathscr{H}})$ only differs on the sets of transitions. We establish the result by proving that $\tilde{\Delta}'_{\text{r}} = \tilde{\Delta}''_{\text{r}}$ and $\tilde{\Delta}'_{\text{p}} = \tilde{\Delta}''_{\text{p}}$.

Let us prove $\tilde{\Delta}'_{\text{p}} = \tilde{\Delta}''_{\text{p}}$. We prove this by showing that for every transition in $\tilde{\Delta}'_{\text{p}}$, there is a corresponding combined transition in $\tilde{\Delta}''_{\text{p}}$ and vice versa. Let $\tilde{\Delta}'_{\text{pc}}$ and $\tilde{\Delta}''_{\text{pc}}$ denote the sets of combined transitions in $\tilde{\Delta}'_{\text{p}}$ and $\tilde{\Delta}''_{\text{p}}$ respectively.

- $\tilde{\Delta}'_{\text{p}} \subseteq \tilde{\Delta}''_{\text{pc}}$: For some $a \in A \cup \hat{A}$, let $(s',\hat{s}') \xrightarrow{a}_{\text{p}} \mu'\|\hat{\mu}'$. We prove that this transition is also possible from $\tilde{\Delta}''_{\text{pc}}$.

   1. Let $(s',\hat{s}') \in S'_1 \times \hat{S}'_1$. As $(s',\hat{s}') \xrightarrow{a}_{\text{p}} \mu'\|\hat{\mu}'$, by (2c) of Def. 34, there exists $\mu\|\hat{\mu} \in (\gamma_{\text{db}} \times \hat{\gamma}_{\text{db}})(\iota_{s'}\|\iota_{\hat{s}'})$, $\mu\|\hat{\mu} \xrightarrow{a}_{\text{p}} \eta\|\hat{\eta}$ such that $|\eta\|\hat{\eta}| \leq |\mu'\|\hat{\mu}'|$ and $(\mu'\|\hat{\mu}')_{\downarrow} = (\alpha_{\text{db}} \times \hat{\alpha}_{\text{db}})(\eta\|\hat{\eta})_{\downarrow}$. There are three possible cases by Def. 30:
      (a) if $a \in \bar{A}$, then $\mu \xrightarrow{a}_{\text{p}} \eta$ and $\hat{\mu} \xrightarrow{a}_{\text{p}} \hat{\eta}$. As $|\eta| \leq |\alpha_{\text{db}}(\eta)|$ and $\mu \xrightarrow{a}_{\text{p}} \eta$, by (2b) of Def. 34 $s' \xrightarrow{a}_{\text{pc}} \alpha_{\text{db}}(\eta)_{\downarrow}$. Similarly, as $|\hat{\eta}| \leq |\hat{\alpha}_{\text{db}}(\hat{\eta})|$ and $\hat{\mu} \xrightarrow{a}_{\text{p}} \hat{\eta}$, by (2b) of Def. 34 $\hat{s}' \xrightarrow{a}_{\text{pc}} \hat{\alpha}_{\text{db}}(\hat{\eta})_{\downarrow}$. As a is a synchronizing action, by Def. 30, $(s',\hat{s}') \xrightarrow{a}_{\text{pc}} (\alpha_{\text{db}}(\eta)\|\hat{\alpha}_{\text{db}}(\hat{\eta}))_{\downarrow}$.
      (b) if $a \in A\backslash\bar{A}$, then $\mu \xrightarrow{a}_{\text{p}} \eta$ and $\hat{\mu} = \hat{\eta}$. As $|\eta| \leq |\alpha_{\text{db}}(\eta)|$ and $\mu \xrightarrow{a}_{\text{p}} \eta$, by (2b) of Def. 34 $s' \xrightarrow{a}_{\text{pc}} \alpha_{\text{db}}(\eta)_{\downarrow}$ and $\iota_{\hat{s}'} = \hat{\alpha}_{\text{db}}(\hat{\mu})_{\downarrow}$. As $a \in A\backslash\bar{A}_1$, by Def. 30, $(s',\hat{s}') \xrightarrow{a}_{\text{pc}} \alpha_{\text{db}}(\eta)_{\downarrow}\|\iota_{\hat{s}'}$.
      (c) if $a \in \hat{A}\backslash\bar{A}$, then the proof is the same as in the previous case.
   2. Let $s'\|\hat{s}' \in S'_2\|\hat{S}'_2$. The proof goes on the same lines as the previous case.
   3. Let $s'\|\hat{s}' \in S'_1\|\hat{S}'_2$. The proof goes on the same lines as the previous case.
   4. Let $s'\|\hat{s}' \in S'_2\|\hat{S}'_1$. The proof goes on the same lines as the previous case.

- $\tilde{\Delta}''_{\text{p}} \subseteq \tilde{\Delta}'_{\text{pc}}$: The proof goes on the same lines as the previous case.

Let us prove $\tilde{\Delta}'_{\mathrm{r}} = \tilde{\Delta}''_{\mathrm{r}}$.

- $\tilde{\Delta}'_{\mathrm{r}} \subseteq \tilde{\Delta}''_{\mathrm{r}}$: For some $a \in A \cup \hat{A}$, let $(s', \hat{s}') \xrightarrow{a}_{\mathrm{r}} \mu' \| \hat{\mu}'$. We prove that this transition is also possible from $\tilde{\Delta}''_{\mathrm{r}}$.

  As $(s', \hat{s}') \xrightarrow{a}_{\mathrm{r}} \mu' \| \hat{\mu}'$, by (2a) of Def. 34 for every $\mu \| \hat{\mu} \in (\gamma_{\mathrm{db}} \times \hat{\gamma}_{\mathrm{db}})(\iota_{s'} \| \iota_{\hat{s}'})$, $\mu \| \hat{\mu} \xrightarrow{a}_{\mathrm{rc}} \eta \| \hat{\eta}$ such that $|\eta \| \hat{\eta}| \geq |\mu' \| \hat{\mu}'|$ and $(\mu' \| \hat{\mu}')_{\downarrow} = (\alpha_{\mathrm{db}} \times \hat{\alpha}_{\mathrm{db}})(\eta \| \hat{\eta})_{\downarrow}$. There are three possible cases by Def. 30:

  1. if $a \in \bar{A}$, then $\mu \xrightarrow{a}_{\mathrm{rc}} \eta$ and $\hat{\mu} \xrightarrow{a}_{\mathrm{rc}} \hat{\eta}$. As $|\eta| \geq |\alpha_{\mathrm{db}}(\eta)|$ and $\mu \xrightarrow{a}_{\mathrm{rc}} \eta$; and this holds for every distribution in $\gamma_{\mathrm{db}}(\iota_{s'})$, by (2a) of Def. 34 $s' \xrightarrow{a}_{\mathrm{r}} \alpha_{\mathrm{db}}(\eta)_{\downarrow}$. Similarly, as $|\hat{\eta}| \geq |\hat{\alpha}_{\mathrm{db}}(\hat{\eta})|$ and $\hat{\mu} \xrightarrow{a}_{\mathrm{rc}} \hat{\eta}$, and this holds for every distribution in $\hat{\gamma}_{\mathrm{db}}(\iota_{\hat{s}'})$, by (2a) of Def. 34 $\hat{s}' \xrightarrow{a}_{\mathrm{r}} \hat{\alpha}_{\mathrm{db}}(\hat{\eta})_{\downarrow}$. As a is a synchronizing action, by Def. 30 $(s', \hat{s}') \xrightarrow{a}_{\mathrm{r}} (\alpha_{\mathrm{db}}(\eta) \| \hat{\alpha}_{\mathrm{db}}(\hat{\eta}))_{\downarrow}$.
  2. if $a \in A \backslash \bar{A}$, then $\mu \xrightarrow{a}_{\mathrm{rc}} \eta$ and $\hat{\mu} = \hat{\eta}$. As $|\eta| \geq |\alpha_{\mathrm{db}}(\eta)|$ and $\mu \xrightarrow{a}_{\mathrm{rc}} \eta$; and this holds for every distribution in $\gamma_{\mathrm{db}}(\iota_{s'})$, by Def. 34 $s' \xrightarrow{a}_{\mathrm{r}} \alpha_{\mathrm{db}}(\eta)_{\downarrow}$ and $\iota_{\hat{s}'} = \hat{\alpha}_{\mathrm{db}}(\hat{\mu})_{\downarrow}$. As $a \in A \backslash \bar{A}_1$, by (2a) of Def. 30 $(s', \hat{s}') \xrightarrow{a}_{\mathrm{r}} \alpha_{\mathrm{db}}(\eta)_{\downarrow} \| \iota_{\hat{s}'}$.
  3. if $a \in \hat{A} \backslash \bar{A}$, then the proof goes on the same lines as the previous case.

- $\tilde{\Delta}''_{\mathrm{r}} \subseteq \tilde{\Delta}'_{\mathrm{r}}$: The proof goes on the same lines as the previous case.

$\square$

5.3 Preservation of Reachability Probabilities

This section discusses that the extremal (i.e., maximal and minimal) reachability probabilities of closed APGA are preserved under (state-based and distribution-based) abstraction. This can be shown by analysing only EPGA of abstract models as they simulate/A-simulate every implementation of abstract models (by Lem. 2).

Corollary 3. *For APGA $\mathscr{H}$ and $x \in \{\mathrm{sb}, \mathrm{db}\}$ with $\mathscr{H}' = \alpha_x(\mathscr{H})$. Let $T \subseteq S$ such that $T' = \alpha_x(T)$. Then* $\min^{\blacktriangledown}(T') \leq \min^{\blacktriangledown}(T) \leq \min^{\blacktriangle}(T) \leq \min^{\blacktriangle}(T')$ *and* $\max^{\blacktriangledown}(T') \leq \max^{\blacktriangledown}(T) \leq \max^{\blacktriangle}(T) \leq \max^{\blacktriangle}(T')$.

The above corollary follows from Th. 10. Next, we show that distribution-based abstraction of APGA is more precise than the state-based abstraction.

Corollary 4. *For APGA $\mathscr{H}$, $\mathscr{H}_{\mathrm{sb}} = \alpha_{\mathrm{sb}}(\mathscr{H})$ and $\mathscr{H}_{\mathrm{db}} = \alpha_{\mathrm{db}}(\mathscr{H})$ with $S_{\mathrm{sb}} = S_{\mathrm{db}}$. Let $T \subseteq S$ such that $T_{\mathrm{sb}} = \alpha_{\mathrm{sb}}(T)$ and $T_{\mathrm{db}} = \alpha_{\mathrm{db}}(T)$. Then* $\min^{\blacktriangledown}(T_{\mathrm{sb}}) \leq \min^{\blacktriangledown}(T_{\mathrm{db}}) \leq \min^{\blacktriangle}(T_{\mathrm{db}}) \leq \min^{\blacktriangle}(T_{\mathrm{sb}})$ *and* $\max^{\blacktriangledown}(T_{\mathrm{sb}}) \leq \max^{\blacktriangledown}(T_{\mathrm{db}}) \leq \max^{\blacktriangle}(T_{\mathrm{db}}) \leq \max^{\blacktriangle}(T_{\mathrm{sb}})$.

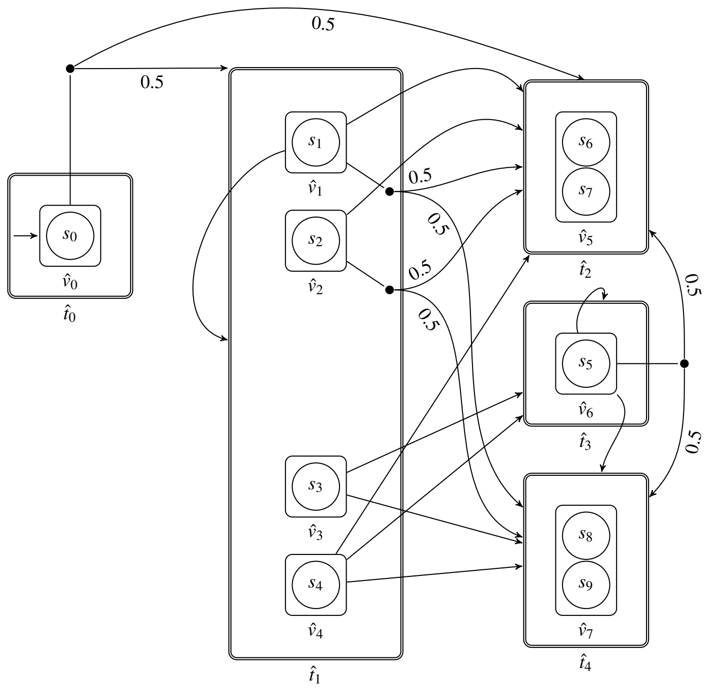


Figure 5.10: For PA $\mathscr{M}$ (Fig. 5.8), $\hat{\mathscr{H}} = \alpha_{\text{sb}}(\alpha_{\text{PA}}(\mathscr{M}))$ with $|\hat{\Delta}| = 26$, $|\hat{S}_1| = 8$ and $|\hat{S}_2| = 5$.

The above corollary is a direct consequence of Th. 15 and Corollary 3.

Example 31. *The maximum probability in PA $\mathscr{M}$ (Fig. 5.8) to reach states $\{s_6, s_7\}$ equals 0.975. By Corollary 3, this probability lies in $[0.5, 1]$ for the state-based abstraction $\tau(\tilde{\mathscr{H}})$ (Fig. 5.9 left). Instead, distribution-based abstraction $\mathscr{H}'$ (Fig. 5.8 right) yields $[0.95, 1]$ — recall that to calculate the lower bound of the maximum probability only* required *transitions of player one are considered (see Th. 15). Moreover, by Corollary 4, $\mathscr{H}'$ has tighter bounds than $\tau(\tilde{\mathscr{H}})$. Although the number of transitions in $\mathscr{H}'$ is slightly more than in $\tilde{\mathscr{H}}$ ($|\tilde{\Delta}| = 17$ versus $|\Delta'| = 19$), but $\mathscr{H}'$ is substantially more precise.*

By Th. 13 and 9, distribution-based abstraction $\tau(\hat{\mathscr{H}})$ (Fig. 5.9 right) also bounds the maximum probability from below at 0.95. Note that the sizes of $\tau(\hat{\mathscr{H}})$ and $\tau(\tilde{\mathscr{H}})$ coincide.

The following example shows that the bounds on reachability probabilities in state-based APGA-based abstractions of PA are at most as tight as in SG-based abstractions, however, they are at most the sizes of SG-based abstractions. A similar example can also be given for distribution-based abstractions.

Example 32. *Consider the game-based abstraction $\hat{\mathscr{H}}$ (Fig. 5.10) of PA $\mathscr{M}$ (Fig. 5.8). Note that the maximum probability to reach states $\{s_6, s_7\}$ lies in $[0.75, 1]$ in $\hat{\mathscr{H}}$ whereas in state-based APGA-based abstraction $\tau(\tilde{\mathscr{H}})$ (Fig. 5.9 left), it lies in $[0.5, 1]$. Note that both $\tilde{S}_2$ and $\hat{S}_2$ represent the same partitioning of the concrete state-space, but the reachability probability bounds of $\tilde{S}_2$ contains that of $\hat{S}_2$. Moreover, $\tau(\tilde{\mathscr{H}})$ is considerably smaller than $\hat{\mathscr{H}}$ in terms of numbers of states and transitions.*

5.4 Distribution-based Game Abstraction of MDPs

In [KKNP10], abstract models of Markov decision processes (MDPs) are given as stochastic games (SGs). These abstractions coincide with our (special case of) state-based abstractions (see Proposition 5.1.1 on page 62). When applying this to MDPs, the abstract models induced by our (special case of) *distribution-based abstraction* are PGA (see Proposition 20 on page 68). By Th. 15, our distribution-based abstraction induces more precise abstract models than state-based abstraction (on the same abstract state space). This shows the superiority of our distribution-based abstraction technique over [KKNP10]. The following corollary follows from Def. 8 and Th. 15. It asserts that our distribution-based abstraction induces more precise abstractions than [KKNP10].

Corollary 5. *For PA $\mathscr{M}'$, let $\mathscr{G} = \alpha_{\text{PA}}(\mathscr{M}')$. If $\alpha_{\text{sb}}(S) = \alpha_{\text{db}}(S)$, then: $\alpha_{\text{db}}(\tau(\mathscr{G})) \prec_{\text{sb}} \alpha_{\text{sb}}(\tau(\mathscr{G}))$ and $\alpha_{\text{db}}(\tau(\mathscr{G})) \preccurlyeq_{\text{sb}} \alpha_{\text{sb}}(\tau(\mathscr{G}))$.*

One may argue that although PGA-based abstract models of MDP are at least as precise as SG-based ones this comes at the expense of larger games, — e.g. more space is required to store the target distributions of player two transitions. The following example shows that for abstracting $\mathscr{G}$ — an embedding on an MDP — with $\alpha_{\text{db}}(S_2) = \alpha_{\text{sb}}(S_2)$, $\alpha_{\text{db}}(\mathscr{G})$ is at least as precise as $\alpha_{\text{sb}}(\mathscr{G})$ and $|\alpha_{\text{db}}(S)| \leq |\alpha_{\text{sb}}(S)|$.

Example 33. *Consider PA $\mathscr{M}$ (Fig. 5.2 right) with its state-based abstraction $\tilde{\mathscr{H}}$ (Fig. 5.4 left) and distribution-based abstraction $\mathscr{H}'$ (Fig. 5.4 right). The maximum probability to $\{s_5, s_6\}$ is 0.95 in $\mathscr{M}$. By Corollary 3, this probability lies in $[0.5, 1]$ for the state-based abstraction $\tau(\tilde{\mathscr{H}})$. Instead, distribution-based abstraction $\tau(\mathscr{H}')$ yields $[0.95, 1]$. Note that $|\tilde{\Delta}| = |\Delta'| = 13$, but $\tilde{S}_1 = 6$ and $S'_1 = 5$. Furthermore, by ignoring player-two transitions — such that the successor states from player-two states are decided non-deterministically — yields $[0.5, 1]$ in $\tilde{\mathscr{H}}$ and $[0.75, 1]$ in $\mathscr{H}'$ whereas $\Delta' = 11$.*

Example 34. *In Example 31, ignoring player-two transitions yields $[0.5, 1]$ in $\tilde{\mathscr{H}}$ and $[0.75, 1]$ in $\mathscr{H}'$.*

As a side result of our achievement, we put the result of [WZ10, Th. 2] in perspective: game-based abstraction [KKNP10] is the optimal abstraction defined on states, but not the optimal abstraction preserving reachability probabilities. Technically speaking, the pair of abstraction-concretization functions, in Wachter's abstract interpretation framework for MDPs [WZ10], is defined at the level of states rather then at the level of distributions. As the abstract valuation transformer is defined in terms of the abstraction-concretization pair and the concrete valuation transformer, therefore the abstract valuation transformer is the most precise transformer in case of *state-based* abstractions.

Remark 2. *The game-based abstraction of MDPs [KKNP10] is the optimal* state-based *abstraction preserving reachability probabilities, but not the optimal* distribution-based *abstraction.*

5.5 Summary and Discussion

In this chapter, we proposed two compositional abstraction techniques of APGA: a state-based and a distribution-based abstraction. In state-based abstraction, we combined the techniques of [KKNP10] and [SK12] yielding a class of APGA in which one of the players has only non-deterministic behaviour as in SGs. Whereas in distribution-based abstraction, we combined the techniques of [SK12] and [SK14] yielding APGA with both players having non-deterministic and probabilistic behaviour as in PGA. Our state-based (distribution-based) abstraction differs from [KKNP10] ([SK14]) in a sense that the non-deterministic behaviour in concrete systems is not completely handled by one set of states: in state-based (distribution-based) abstraction, concrete states (support sets of concrete distributions) are merged if they have the same step-wise behaviour after abstraction; whereas in [KKNP10] ([SK14]), they are merged iff they have the same step-wise behaviour. Because of this, the bounds on extremal reachability probabilities in state-based (distribution-based) APGA-based models are at most as tight as in SG(PGA)-based models, however, they are at most the size of SG(PGA)-based models.

Both state-based and distribution-based abstractions are comparable with concrete models using state-based and distribution-based refinement relations respectively, thus, showing that abstract models preserve the reachability probabilities of concrete models. Moreover, we showed that game-based abstraction [KKNP10] is not the optimal abstraction preserving reachability probabilities. Furthermore, we illustrated with examples that our distribution-based abstraction may induce more precise as well as more concise models than our state-based abstraction. An overview of the results of this chapter is given in Table 5.1.

Related work: In the literature, many *modal* abstraction techniques have been discussed for probabilistic systems. In [KKLW12], abstractions of MDPs have been given as *interval Markov chains* in which probabilities of transitions are given as intervals instead of single values. This interval-based technique has then been extended to define abstract models of *interactive Markov chains* (IMC) in [KKN09]. In another direction [KKLW08], interval abstraction is generalized for CTMC by considering *sequences of transitions* of length, say k, as *abstract transitions*. The abstract models induced as a result k-stepwise simulate the concrete models. In the setting of games, most notable abstraction techniques are *game-based abstraction* [KKNP10] and *menu-based abstraction* [WZ10] (discussed in Ch. 1). In [Kat10], game-based abstraction technique of MDPs has been generalized: the abstract models are still stochastic games but they over- and under-approximate the behaviour of concrete models; however, in such games there is no direct correspondence between concrete states of MDPs (that have the same transitions after abstraction) and abstract states. More recently in [BFH$^+$14], game-based and menu-based abstraction techniques have been combined to abstract *Markov automata* in which *game-based* technique is employed to abstract *Markovian states* — states with exponentially distributed delay transitions — and *menu-based* technique for *probabilistic states*.

Future extensions: One can extend this work by:

- determining a class of PA for which distribution-based abstraction is defined, i.e., it exists for every possible partition of the state space,
- designing efficient algorithms to construct APGA-based state-based (distribution-based) abstract models of PA from their high-level descriptions, e.g., using predicate-abstraction techniques [CKSY05, KKNP08].

Modal game-based abstraction

For $\alpha \in \{\alpha_{\text{sb}}, \alpha_{\text{db}}\}$ and APGA $\mathscr{H}$	State-based (α_{sb})	Distribution-based (α_{db})
$\alpha(\mathscr{H})$ is defined	+	for closed APGA
$\alpha(\mathscr{H})$ refines $\mathscr{H}$	+	+ (provided exists)
$\alpha(\tau(\mathscr{H}))$ refines $\tau(\mathscr{H})$	+	+
compositional	+	+ (provided exists)
For $x \in \{\text{sb}, \text{db}\}$, $\alpha(\tau(\mathscr{H})) \preceq_x \tau(\alpha(\mathscr{H}))$	+	-
$\mathscr{H}' = \alpha(\mathscr{H})$ with $\Delta'_{\text{p}}(S'_1) = \Delta'_{\text{r}}(S'_1)$	α is SG-based abstraction	α is PGA-based abstraction
$\mathscr{H}' = \alpha'(\tau(\mathscr{H})) \wedge \mathscr{H}'' = \tau(\alpha''(\mathscr{H}))$,	$\mathscr{G}_{\circ}^{\bullet\prime} \preccurlyeq_{\text{sb}} \mathscr{G}_{\circ}^{\bullet\prime\prime}$, and	$\mathscr{G}_{\circ}^{\bullet\prime} \preccurlyeq_{\text{db}} \mathscr{G}_{\circ}^{\bullet\prime\prime}$
then $\alpha'(S) = \alpha''(S)$ implies	$\mathscr{G}_{\circ}^{\circ\prime} \prec_{\text{sb}} \mathscr{G}_{\circ}^{\circ\prime\prime}$	-
	$\text{MaxSingDist}(S_1) = \text{Dirac}(S_1) \wedge \alpha_{\text{sb}}(S_2) = \alpha_{\text{db}}(S_2) \Rightarrow$ $\alpha_{\text{sb}}(\mathscr{H}) = \alpha_{\text{db}}(\mathscr{H})$, ignoring player two transitions.	
For $\prec \in \{\prec_{\text{sb}}, \preccurlyeq_{\text{sb}}\}$	$\alpha_{\text{sb}}(S) = \alpha_{\text{db}}(S) \Rightarrow \alpha_{\text{db}}(\mathscr{H}) \prec \alpha_{\text{sb}}(\mathscr{H})$	

Preservation of reachability probabilities

α_{sb} and α_{db} yield	$\min^{\blacktriangledown}$ and $\min^{\blacktriangle}$ as well as $\max^{\blacktriangledown}$ and $\max^{\blacktriangle}$

Table 5.1: Summary of modal game-based abstraction.

- exploring whether our PGA-based distribution-based abstraction of MDPs is optimal w.r.t. reachability probabilities using the framework of *abstract interpretation* like [WZ10] did for game-based abstraction [KKNP10] of MDPs, and

- repeating the above exercise for our APGA-based abstractions of MDPs, and showing whether it is optimal w.r.t. properties in probabilistic extension of modal μ-calculus [Mio12]. For this, one need to consider an extension of the work done for non-probabilistic models in [SG06], where the optimality of non-probabilistic *modal* abstractions was proved w.r.t. properties in modal μ-calculus.

In the next chapter, we propose a *state-based* and a *distribution-based* abstraction-refinement framework for PA, which are inspired by the frameworks of [KKLW12] and [KKNP10]. Intuitively, they eliminate from abstract models the effect of non-deterministic behaviour induced by abstraction in such a way that the extremal reachability probabilities of one set of abstract states totally depend on the non-deterministic behaviour from concrete (distributions over) states, as is the case with (PGA-)SG-based abstractions.

6 A Modal Abstraction-Refinement Framework

In this chapter, we present a *state-based* and a *distribution-based* framework to automatically generate APGA-based abstractions of closed PA aimed at the verification of reachability properties, i.e., *maximum/minimum* probabilities to reach some goal states.

In the previous chapters, we showed that the analysis of APGA-based abstractions of PA yields *lower* and *upper* bounds on the reachability probabilities of the PA; we analysed two extremal implementations of such abstractions to obtain these bounds (see Section 4.5 on page 48, and Section 5.3 on page 72). Moreover, we also observed that these bounds may not be optimal for a given partitioning of the state space (see Example 32 on page 73). In this chapter, we show that it is possible to automatically obtain *a modal game-based abstraction* for a given state-space partitioning of a PA that *optimally* bounds the reachability probabilities of the PA with an a priori given tolerance.

We start our abstraction-refinement process by abstracting a closed PA for a given state-space partitioning (see Def. 31 and 34 on pages 57 and 63 respectively). We then verify the abstract model yielding lower and upper bounds on the reachability probabilities for the goal states in the PA. As in APGA-based abstractions the non-deterministic behaviour from abstraction is present both in player-one states and in player-two states, we first incrementally refine only player-one states until the behaviour from abstraction in player-one states has no impact anymore on the reachability probabilities of the corresponding player-two states. This is guaranteed by the fact that any further refinement of player-one states has no impact on the reachability probabilities of player-two states.

After this step, we check whether the difference between upper- and lower-bounds on reachability probabilities is within an allowed range, say $\varepsilon \in \mathbb{R}_{[0,1]}$, for a set of player-two states (that are of interest). If not, then these states are refined. The first step is then repeated for the new abstraction. The above two steps form the inner and the outer loops of our abstraction-refinement framework: the inner-loop refines player-one states whereas the outer-loop refines player-two states. Fig. 6.1 shows our abstraction-refinement framework.

Our *refinement strategy* for both player-one and player-two states is based on optimal probability valuation functions that map states to reachability probabilities in case of competing/collaborating players. For player-two states, it is almost the same as *strategy-based refinement* of Kattenbelt et al. [KKNP10]. Our refinement strategy induces a strictly finer partition in each iteration. Moreover, it guarantees that if an abstract state is partitioned, its concrete states that have the same behaviour (i.e., their sets of transitions are identical after abstraction) remain together in one of the new partition blocks. This is true for both player-one and player-two abstract states. Furthermore, for a given error bound ε, our abstraction-refinement (outer-)loop eventually terminates for finite-state closed PA. This is also true in case of an infinite PA with a finite bisimulation quotient provided bisimilar concrete states are grouped together in the same partition blocks (player-two state).

We show that the model obtained after each iteration of the inner-loop is in a refinement relation (see Def. 23 and 25 on pages 40 and 42 respectively) with the abstract model; and, thus, each refined model is comparatively more precise by Th. 10 (page 50). Similarly, after each iteration of the outer-loop, the player-two states in the refined model have comparatively tighter bounds on the reachability probabilities (of the PA).

The abstraction-refinement framework for MDPs [KKNP10] is a special case of our state-based framework in which the inner loop does nothing. This is because in SG-based abstractions of MDPs the player-one states have no non-deterministic behaviour from abstraction (all player-one transitions are *required* transitions). We show that the reachability probabilities of player-two states in an APGA-based state-based abstraction of an MDP (after refinement) are the same as in the SG-based abstraction generated with the same state-space partitioning. However, the size of the player-one state space in our abstraction is at most that of in SG-based abstraction. The same is also true for our APGA-based distribution-based abstractions and PGA-based abstractions [SK14] (see page 68).

Put in a nutshell, the major contributions of this chapter are:

- a *state-based* (and a *distribution-based*) abstraction-refinement framework for (closed) PA,
- results showing that our state-based abstractions (after refinement) and SG-based abstractions of [KKNP10] have the same reachability probabilities for player-two states, but our models are smaller in size,
- examples showing that our distribution-based abstractions (after refinement) and PGA-based abstractions have the same reachability probabilities for player-two states, but our former models are smaller in size, and
- examples showing that the abstract models resulting from the distribution-based abstraction-refinement framework are more precise than the ones resulting from the state-based framework, and their sizes are comparable.

6.1 State-based Abstraction-Refinement Framework

In this section, we discuss the refinement process for state-based abstractions (see Def. 31 on page 57) of PA aimed at the verification of a given reachability property.

Let $\mathscr{M}$ be a closed finite PA with $\mathscr{H}' = \alpha_{\mathrm{PA}}(\mathscr{M})$ and $T' \subseteq S'_2$ a set of goal states. Let $\Pr^x(T')$ be the reachability probability for T', where $x \in \{\min, \max\}$. Let $Abst_{\mathrm{sb}}(\mathscr{H}')$ be the set of state-based abstraction functions defined on $\mathscr{H}'$ such that $\gamma(\alpha(T')) = T'$ for all $\alpha \in Abst_{\mathrm{sb}}(\mathscr{H}')$, i.e., α does not merge T' states with $S'_2 \backslash T'$ states. Let $\mathscr{H} = \alpha_{\mathrm{sb}}(\mathscr{H}')$ and $T = \alpha_{\mathrm{sb}}(T')$ for $\alpha_{\mathrm{sb}} \in Abst_{\mathrm{sb}}(\mathscr{H}')$.

Depending on the probability $\Pr^x(T')$, let $\mathbf{1}, \mathbf{2} \in \{\min, \max\}$ with $\mathbf{1} \neq \mathbf{2}$. Let $w_{\mathbf{11}}, w_{\mathbf{12}} \in W$ be the probability valuation transformers (see page 15) such that $w_{\mathbf{12}} = \mathbf{Fix}\,\mathrm{Prt}^{\mathbf{1}}_{\mathbf{2}}(\bot)$ (player-one and player-two have different objectives, i.e., $\mathbf{1}$ and $\mathbf{2}$ respectively) and $w_{\mathbf{11}} = \mathbf{Fix}\,\mathrm{Prt}^{\mathbf{1}}_{\mathbf{1}}(\bot)$ (player-one and player-two have the same objectives, i.e., $\mathbf{1}$) are defined on EPGA $\mathscr{G}^{\circ}_{\circ}$ of $\mathscr{H}$ — an implementation of $\mathscr{H}$ having the same state space and the transitions as that of $\mathscr{H}$ (see Def. 29 on page 48). Recall that $w_{\mathbf{12}}$ and $w_{\mathbf{11}}$ map a

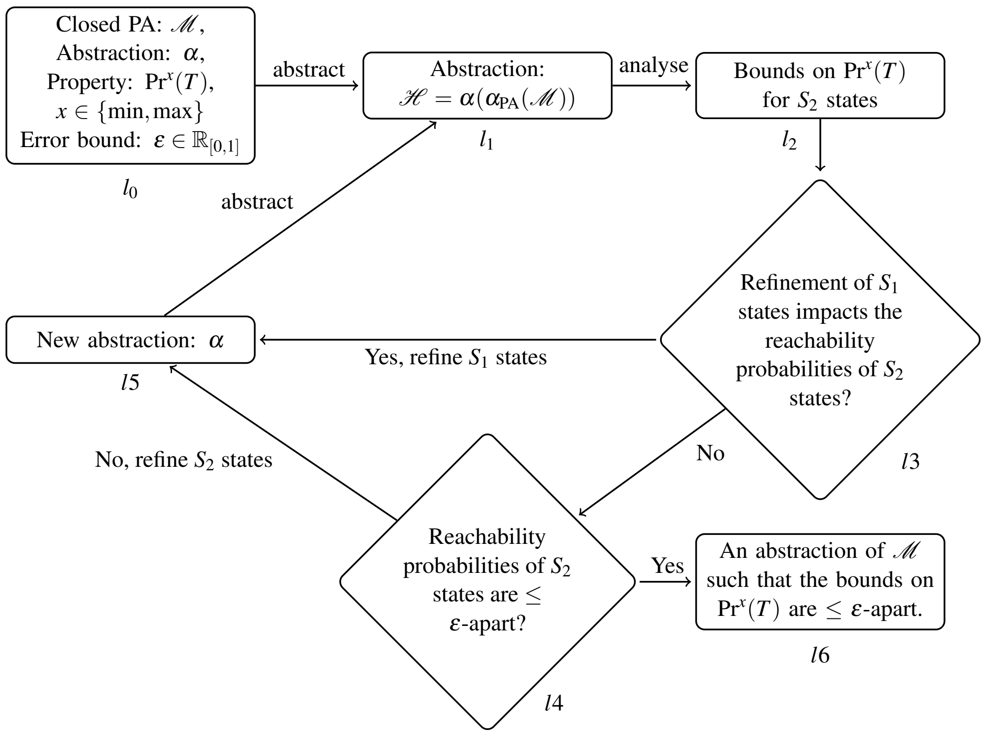

Closed PA: $\mathscr{M}$, Abstraction: α, Property: $\Pr^x(T)$, $x \in \{\min, \max\}$ Error bound: $\varepsilon \in \mathbb{R}_{[0,1]}$
l_0
abstract
Abstraction: $\mathscr{H} = \alpha(\alpha_{\mathrm{PA}}(\mathscr{M}))$
l_1
analyse
Bounds on $\Pr^x(T)$ for S_2 states
l_2
abstract
Refinement of S_1 states impacts the reachability probabilities of S_2 states?
$l3$
New abstraction: α
$l5$
Yes, refine S_1 states
No
No, refine S_2 states
Reachability probabilities of S_2 states are $\leq$ ε-apart?
$l4$
Yes
An abstraction of $\mathscr{M}$ such that the bounds on $\Pr^x(T)$ are $\leq$ ε-apart.
$l6$

Figure 6.1: State(Distribution)-based abstraction-refinement framework for PA, where the inner-loop ($l1, l2, l3, l5, l1$) refines the player-one states while the outer-loop ($l1, l2, l3, l4, l5, l1$) refines the player-two states.

state $s \in S$ to reachability probabilities of goal states $T = \alpha_{\mathrm{sb}}(T')$ in case of competing and collaborating players respectively (see Def. 9 on page 15), and thus define bounds on $\Pr^x(T)$ for state s.

In the sequel, we assume that $(\alpha_{\mathrm{sb}}, \gamma_{\mathrm{sb}})$, $\mathscr{H}'$, $\mathscr{H}$, $\mathbf{1}$, $\mathbf{2}$, $w_{\mathbf{11}}$ and $w_{\mathbf{12}}$ are given; unless otherwise specified. Moreover, we assume that the optimal valuation functions are defined on all APGA in the same way as we defined $w_{\mathbf{11}}$ and $w_{\mathbf{12}}$ on $\mathscr{H}$.

In the following, we step-by-step explain our framework given in Fig. 6.1. We first check whether the reachability probabilities of player-two states in $\mathscr{H}$ are affected by the non-deterministic behaviour induced by the abstraction process in their corresponding player-one states. Alternatively, we check whether the refinement of player-one states alone affect the reachability probabilities of player-two states.

6.1.1 Stable Abstractions

Recall that the player-one states of an abstraction represent behaviour of concrete states, i.e. they inherit their transitions from their corresponding concrete states. If all concrete states of a player-one state have the same transitions (after abstraction), then the abstraction process has not induced any behaviour in that state; and the player-one state has only *required* transitions. Otherwise, it has *possible* transitions and *required* transitions depending on whether the concrete states have some common transitions (after abstraction) (see Def. 31 on page 57). We consider the *required* transitions of player-one states as the

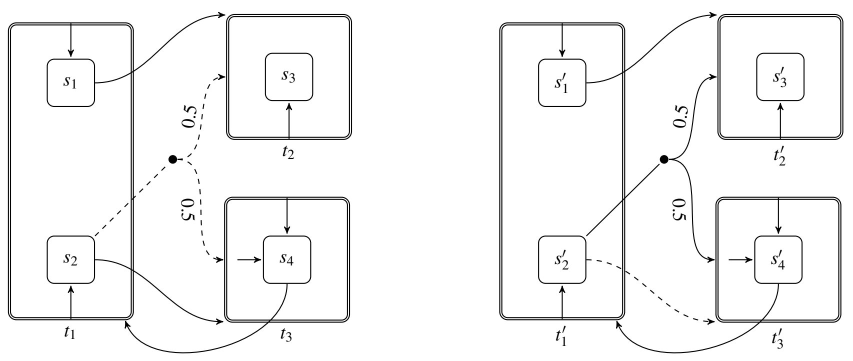


Figure 6.2: APGA $\mathscr{H}$ (left) and $\mathscr{H}'$ (right).

behaviour derived from concrete states, and the *possible* transitions as the behaviour induced by the abstraction process.

State $s \in S_2$ in APGA $\mathscr{H}$ is called *stable* whenever the reachability probability of s w.r.t. $w_{\mathbf{12}}$:

a) coincides with that of one of its direct successors that obtains it via a *required* transition, and

b) remains unchanged by any refinement of only its direct successor states.

Example 35. *Consider the APGA $\mathscr{H}$ in Fig. 6.2 (left) with $\mathbf{1} = \max$, $\mathbf{2} = \min$ and goal states $T = \{t_2\}$. The minimal reachability probability of t_1 is 0.5 and it depends on the* possible *transition of s_2. Hence, t_1 is an unstable state as condition (a) is violated.*

Now consider the APGA $\mathscr{H}'$ Fig. 6.2 (right) with the same objectives as above and goal states $T' = \{t'_2\}$. The state t'_1 is stable as its reachability probability is 0.5 that is the same as that of the required *transition of s'_2. Moreover, any refinement of only s'_2 will not impact the reachability probability of t'_1.*

To formalize the notion of a *stable* player-two state, we first define the notion of a *stable* player-one state. A player-one state is *stable* w.r.t. $w_{\mathbf{12}}$ and player-two states if its reachability probability depends on one of its *required* transitions. Otherwise, it is *unstable*. Recall that $w(\mu) = \sum_{s \in S} \mu(s) \cdot w(s)$ for $\mu \in \mathrm{Dist}(S)$.

Definition 35. (Stable Player-one States) *State $s \in S_1$ in APGA $\mathscr{H}$ is* stable *w.r.t. $w_{\mathbf{12}}$ and S_2 iff $w_{\mathbf{12}}(s) = w_{\mathbf{12}}(\mu)$ for some $\mu \in \Delta_{\mathrm{r}}(s)$. Distribution $\eta \in \mathrm{Dist}(S_1)$ is* stable *iff $\forall u \in \mathrm{Supp}(\eta)$: u is stable.*

Example 36. *APGA $\mathscr{H}$ (Fig. 6.4 left) is an abstraction of PA $\mathscr{M}$ (Fig. 6.3). Let $\mathbf{1} = \min$, $\mathbf{2} = \max$ and $T = \{t_3\}$ with $w = \mathbf{Fix}\,\mathrm{Prt}^{\mathbf{1}}_{\mathbf{2}}(\bot)$ where $w(v_0) = 0.25$, $w(v_1) = 0$, $w(v_2) = 0.5$, $w(v_3) = 0$, $w(v_4) = 0$, $w(t_0) = 0$, $w(t_1) = 0.5$, $w(t_2) = 0$ and $w(t_3) = 1$. Note that v_1 has two transitions: a* required *and a* possible *transition. As the reachability probability of v_1 depends on its* required *transition, v_1 is stable.*

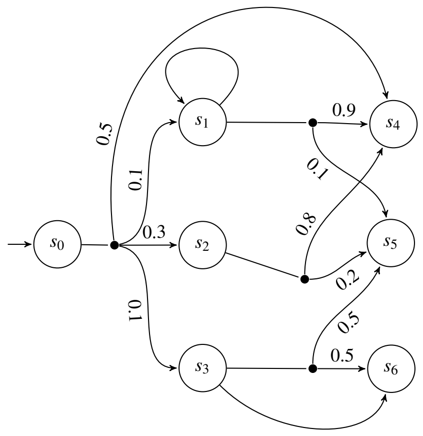


Figure 6.3: A PA $\mathscr{M}$.

Intuitively speaking, if the reachability probability of a player-two state depends on a stable player-one state, then it will remain unchanged by a refinement of just any of its stable direct successor states. This is because refinement preserves *required* transitions. Therefore, the reachability probability of a player-two state may only be affected by refining an unstable direct successor state.

Lemma 4. *The refinement of stable player-one states alone preserves the reachability probabilities.*

Proof. Let APGA $\mathscr{H}$ with $w = w_{\mathbf{12}}$. We show that the partitioning of stable player-one states alone in a refinement of $\mathscr{H}$ preserves the reachability probabilities. We give a proof for the case in which only one player-one state is split.

Let $\hat{\alpha} \in Abst_{\text{sb}}(\mathscr{H}') : \hat{\mathscr{H}} = \hat{\alpha}(\mathscr{H}')$ with $\hat{S}_2 = S_2$. Let $\{\hat{v}_1, \hat{v}_2\}$ be the partition blocks of a stable state $v \in S_1$ such that $\hat{S}_1 \setminus \{\hat{v}_1, \hat{v}_2\} = S_1 \setminus \{v\}$.

As v is stable, by Def. 35 $w(v) = w(\mu)$ for some $\mu \in \Delta_{\text{r}}(v)$. As $\hat{\gamma}(\hat{v}_i) \subseteq \gamma(v)$ for $i \in \{1,2\}$, $\Delta_{\text{r}}(v) \subseteq \hat{\Delta}_{\text{r}}(v_i)$ and $\hat{\Delta}_{\text{p}}(v_i) \subseteq \Delta_{\text{p}}(v)$ (by Def. 31). Therefore, it holds that $w(\mu) = \mathbf{1}\{w(\mu), x_i\}$ where $x_i = \mathbf{1}\{w(\rho) \mid \rho \in \hat{\Delta}_{\text{p}}(\hat{v}_i)\}$ for $i \in \{1,2\}$.

This allows us to define a probability valuation function $\hat{w}$ on $\hat{\mathscr{H}}$ w.r.t. $\mathbf{1}$, $\mathbf{2}$ and $\hat{T} = T$ as: $\hat{w}(s) = w(s)$ for all $s \in S_2 \cup S_1 \setminus \{v\}$ and $\hat{w}(\hat{v}_i) = w(\mu)$ for $i \in \{1,2\}$. Moreover, as $w(v) = \hat{w}(\hat{v}_i)$, for $i \in \{1,2\}$, it trivially follows that $\hat{w}$ is a fixpoint of $\text{Prt}^{\mathbf{1}}_{\mathbf{2}}$.

□

Remark 3. *The refinement of unstable direct successors of a player-two state may impact its reachability probability.*

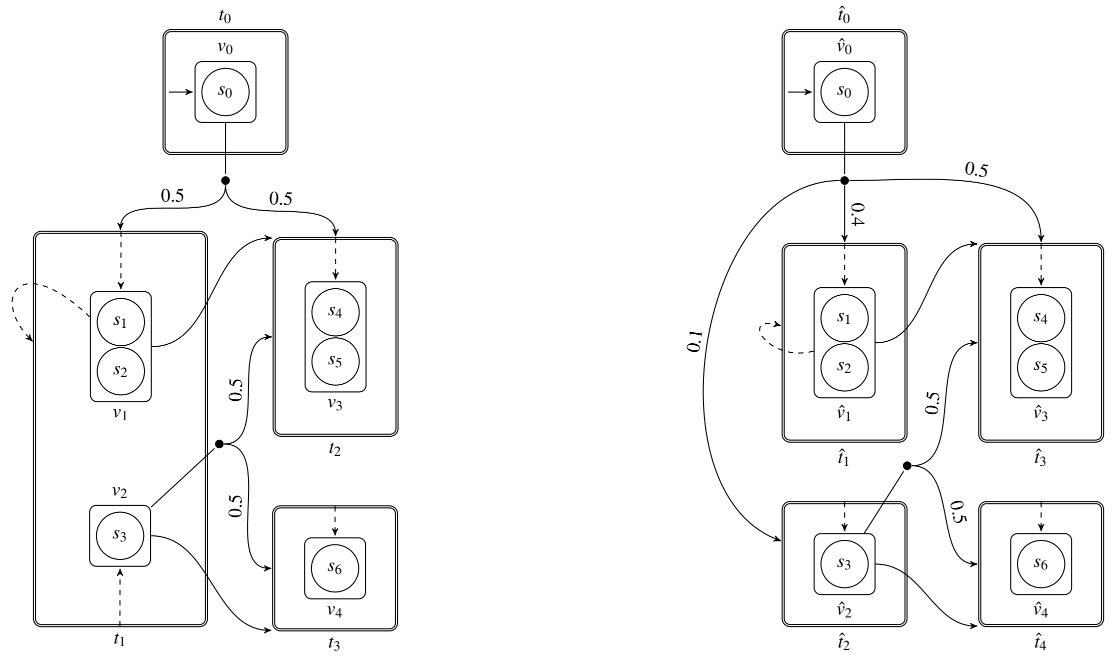


Figure 6.4: For $\mathbf{1} = \min$ and $\mathbf{2} = \max$, $\mathscr{H}$ (left) is a stable abstraction of PA $\mathscr{M}$ (Fig. 6.3) w.r.t. $T = \{t_3\}$; whereas $\hat{\mathscr{H}}$ (right) is a bounded abstraction w.r.t. $T = \{\hat{t}_4\}$ and $\varepsilon = 0.4$.

Now, we formalize the notion of a *stable* player-two state.

Definition 36. (Stable Player-two State) *State $s \in S_2$ in APGA $\mathscr{H}$ is* stable *w.r.t. $w_{\mathbf{12}}$ iff there exists some stable $v \in \mathrm{Succ}(s)$ satisfying:*

1. *$w_{\mathbf{12}}(s) = w_{\mathbf{12}}(v)$, and*
2. *for all unstable $u \in \mathrm{Succ}(s)$: $w_{\mathbf{12}}(v) = \mathbf{2}\{w_{\mathbf{12}}(v), w_{\mathbf{12}}(\eta)\}$ for every $\eta \in \Delta_{\mathrm{p}}(u)$.*

Intuitively speaking, if condition (2) holds none of the transitions from any successor of s can change the reachability probability of s in any refinement. Therefore, the refinement of direct successors, whether stable or unstable, of a stable player-two state alone does not change the bounds on its reachability probabilities.

Example 37. *Let $\mathbf{1} = \min$ and $\mathbf{2} = \max$ for APGA $\mathscr{H}$ (left in Fig. 6.6) with $w = \mathbf{Fix}\,\mathrm{Prt}_{\mathbf{2}}^{\mathbf{1}}(\bot)$ for $T = \{t_3\}$, where $w(v_0) = 0.25$, $w(v_1) = 0$, $w(v_2) = 0.5$, $w(v_3) = 0$, $w(v_4) = 0$, $w(t_0) = 0$, $w(t_1) = 0.5$, $w(t_2) = 0$ and $w(t_3) = 1$. (Note that this APGA is a copy of Fig. 6.4 left, except that $v_1 \to t_2$ is now a* possible *transition.) The reachability probability of t_1 depends on a stable successor v_2, i.e., $w(t_1) = 0.5 = \max\{w(v_1) = 0, w(v_2) = 0.5\}$ (fulfilling condition (1) of Def. 36). Moreover, as $w(v_2) = 0.5 = \max\{w(\iota_{t_1}) = 0.5, w(\iota_{t_2}) = 0, w(v_2) = 0.5\}$ (fulfilling condition (2) of Def. 36), therefore the* possible

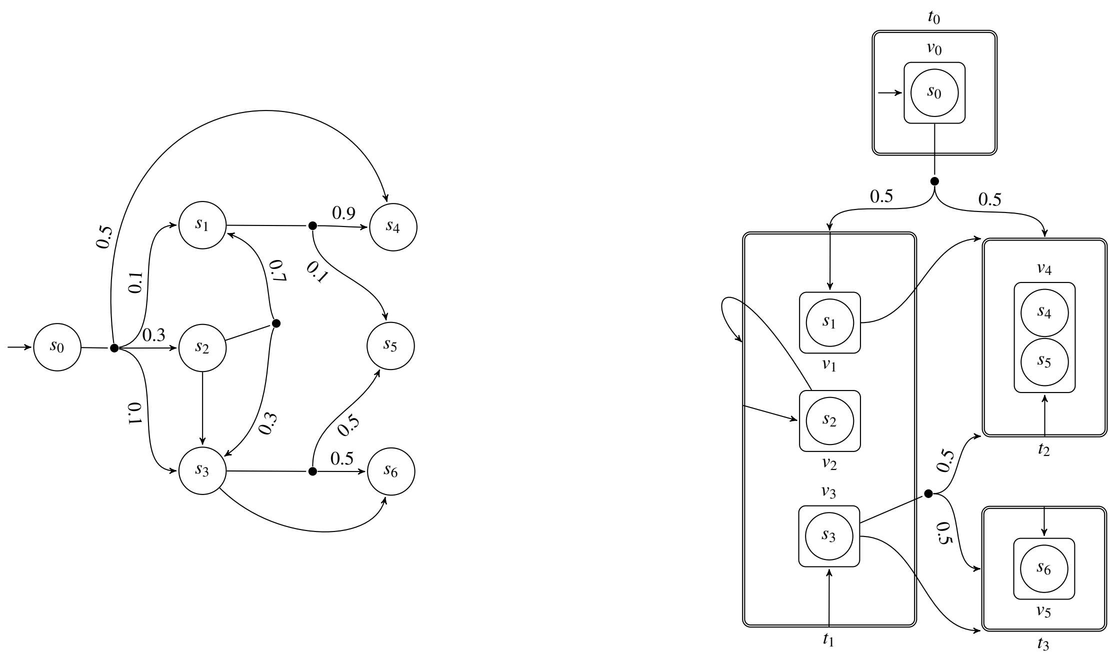


Figure 6.5: PA $\mathscr{M}$ (left) with its game-based abstraction ([KKNP10]) $\mathscr{H}$ (right).

transitions of unstable state v_1 have no impact on the reachability probability of t_1 in any refinement of v_1. Thus, t_1 is stable. Note that in APGA $\hat{\mathscr{H}}$ (right in Fig. 6.6), the state $\hat{t}_1$ is not stable w.r.t. the above objectives as the reachability probability of $\hat{t}_1$ does not depend on a stable successor.

Theorem 17. *The reachability probabilities of stable player-two states are invariant to the refinement of their direct successors alone.*

Proof. Let APGA $\mathscr{H}$ with $w = w_{\mathbf{12}}$. We show that the splitting of direct successors of stable player-two states alone in a refinement of $\mathscr{H}$ preserves the reachability probabilities of player-two states. We give a proof for the case in which only one player-one state is split.

Let $\hat{\alpha} \in Abst_{\text{sb}}(\mathscr{H}') : \hat{\mathscr{H}} = \hat{\alpha}(\mathscr{H}')$ with $\hat{S}_2 = S_2$. Let $\{\hat{v}_1, \hat{v}_2\}$ be the partition blocks of a state $v \in S_1$ such that $\hat{S}_1 \setminus \{\hat{v}_1, \hat{v}_2\} = S_1 \setminus \{v\}$. There are two possible cases:

1. v is a stable player-one state. Then by Lemma 4, $\hat{\mathscr{H}}$ preserves the reachability probabilities.

2. v is an unstable direct successor of a stable state $s \in S_2$. As s is stable, by Def. 36 $w(s) = w(t)$ for some stable $t \in \text{Succ}(s)$; and $w(t) = \mathbf{2}\{w(t), w(\eta)\}$ for every $\eta \in \Delta_{\text{p}}(v)$. As $\Delta_{\text{p}}(v) = \hat{\Delta}_{\text{p}}(\hat{v}_1) \cup \hat{\Delta}_{\text{p}}(\hat{v}_2)$, we can write $w(t) = \mathbf{2}\{w(t), x_i\}$ where $x_i = \mathbf{1}\{w(\rho) \mid \rho \in \hat{\Delta}_{\text{p}}(\hat{v}_i)\}$ for $i \in \{1,2\}$.

   This allows us to define a probability valuation function $\hat{w}$ on $\hat{\mathscr{H}}$ w.r.t. $\mathbf{1}$, $\mathbf{2}$ and $\hat{T} = T$ as: $\hat{w}(s) =$

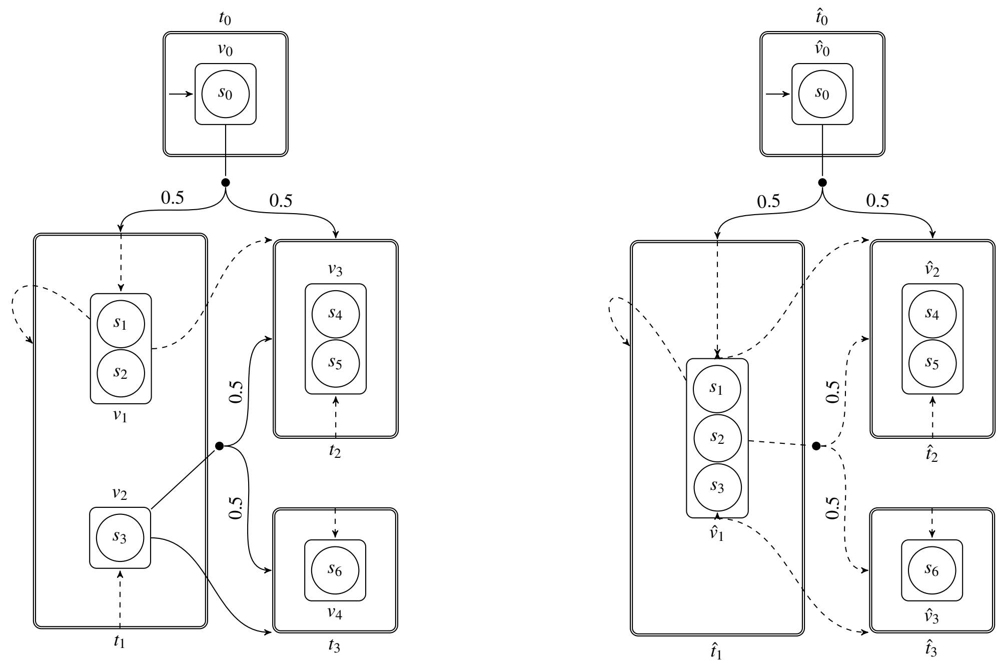


Figure 6.6: $\mathscr{H}$ (left) and $\hat{\mathscr{H}}$ (right) are state-based abstract models of PA $\mathscr{M}$ in Fig. 6.5 with $\mathscr{H} \preceq_{\text{sb}} \hat{\mathscr{H}}$.

$w(s)$ for all $s \in S_2 \cup S_1 \setminus \{v\}$ and $\hat{w}(\hat{v}_i) = x_i$ for $i \in \{1,2\}$. Moreover, as $\hat{w}(s) = w(t)$ does not depend on $\hat{w}(\hat{v}_i)$ for $i \in \{1,2\}$ — $\hat{w}(s) = \hat{w}(t) = \mathbf{2}\{\hat{w}(t), \hat{w}(v_1), \hat{w}(v_2)\}$ —, it trivially follows that $\hat{w}$ is a fixpoint of $\text{Prt}^{\mathbf{1}}_{\mathbf{2}}$.

□

We now lift the notion of stability from states to APGA, and call an APGA $\mathscr{H}$ a *stable abstraction* if all its player-two states are stable. Formally,

Definition 37. *APGA $\mathscr{H}$ is* stable *iff every S_2 state is stable.*

If $\mathscr{H}$ is a stable abstraction, we call the abstraction function α_{sb} generating $\mathscr{H}$ ($\mathscr{H} = \alpha_{\text{sb}}(\mathscr{H}'))$ a *stable abstraction function*. Any implementation of a stable abstraction, having the same player-two state space, preserve the reachability probabilities of player-two states. The following proposition follows from Def. 37.

Proposition 21. *If $\Delta_{\text{r}}(S_1) = \Delta_{\text{p}}(S_1)$ in APGA $\mathscr{H}$, then $\mathscr{H}$ is stable.*

We are now in a position to check whether the refinement of player-one states impacts the reachability probabilities of player-two states in an abstraction (see location l_3 in Fig. 6.1). This can be verified by checking whether every player-two state is stable or not (see Def. 36). If the answer is yes, we consider refining player-two states if further tightening of the reachability probability bounds is required. Otherwise, we refine some player-one states as given in the next section.

6.1.2 Refinement of Player-one States

In this subsection, we discuss the refinement of (unstable) player-one states. As the refinement of direct successors — whether stable or unstable — of a stable player-two state has no impact on its reachability probability (see Th. 17), we only consider the unstable direct successors of unstable player-two states for refinement. We call them *effectively unstable* player-one states.

Definition 38. *State $s \in S_1$ in APGA $\mathscr{H}$ is* effectively unstable *iff s is unstable and there exists an unstable $u \in S_2 : s \in \mathrm{Succ}(u)$.*

Let $\mathrm{EUS}(\mathscr{H}) = \{s \in S_1 \mid \exists u \in S_2 : u \text{ is unstable and } s \in \mathrm{Succ}(u) \text{ is unstable}\}$ be the set of effectively unstable states in $\mathscr{H}$.

Remark 4. *An unstable direct successor s_1 of a stable player-two state u_1 differs from an unstable direct successor s_2 of an unstable player-two state u_2 in such a way that the refinement of s_1 does not impact the reachability probability of u_1. However, the refinement of s_2* may *impact the reachability probability of u_2. Note that it is not guaranteed that the refinement of s_2 will impact the reachability probability of u_2.*

We split each effectively unstable state into two blocks.

Definition 39. *For state $s \in \mathrm{EUS}(\mathscr{H})$, let $\mu \in \Delta_{\mathrm{p}}(s) : w_{\mathbf{12}}(s) = w_{\mathbf{12}}(\mu)$. Then,*

- $\mathrm{B}_1(s) = \{s' \in \gamma_{\mathrm{sb}}(s) \mid \exists \rho' \in \Delta'(s') : \alpha_{\mathrm{sb}}(\rho') = \mu\}$
- $\mathrm{B}_2(s) = \gamma_{\mathrm{sb}}(s) \backslash \mathrm{B}_1(s)$

Theorem 18. *For state $s \in \mathrm{EUS}(\mathscr{H})$, $\mathrm{B}_1(s)$ is a proper subset of $\gamma_{\mathrm{sb}}(s)$.*

Proof. The proof is by contradiction. Let $s \in \mathrm{EUS}(\mathscr{H})$. Assume $\mathrm{B}_1(s) = \gamma_{\mathrm{sb}}(s)$, then for every $s' \in \gamma_{\mathrm{sb}}(s)$, $\exists \rho' \in \Delta'(s') : \alpha_{\mathrm{sb}}(\rho') = \mu$, where $\mu \in \Delta_{\mathrm{p}}(s) : w_{\mathbf{12}}(s) = w_{\mathbf{12}}(\mu)$. This implies that all concrete states of s have a transition to *some* ρ' with $\alpha_{\mathrm{sb}}(\rho') = \alpha_{\mathrm{sb}}(\mu')$. Then $\mu \in \Delta_{\mathrm{r}}(s)$ and s is not an unstable state. Hence, contradiction. □

The following function splits effectively unstable states in an abstraction, and yields a new partitioning of the state space of $\mathscr{H}'$ (i.e., the embedding of PA $\mathscr{M}$). In fact, this function describes the functionality of the inner-loop of our framework (see Fig. 6.1).

Definition 40. (Inner Abstraction Transformer). *The* abstraction transformer *function* IAT : $Abst_{\mathrm{sb}}(\mathscr{H}') \to Abst_{\mathrm{sb}}(\mathscr{H}')$ *is defined for* $\alpha \in Abst_{\mathrm{sb}}(\mathscr{H}')$*:* $\mathscr{H} = \alpha(\mathscr{H}')$ *and* $s' \in S'$ *as:*

$$\mathrm{IAT}(\alpha)(s') = \begin{cases} \alpha(s') & \text{if } \alpha(s') \in S_2, \text{ or } \alpha(s') \in S_1 \backslash \mathrm{EUS}(\mathscr{H}). \\ \mathrm{B}_1(\alpha(s')) & \text{if } \alpha(s') \in \mathrm{EUS}(\mathscr{H})\text{: } s' \in \mathrm{B}_1(\alpha(s')). \\ \mathrm{B}_2(\alpha(s')) & \text{if } \alpha(s') \in \mathrm{EUS}(\mathscr{H})\text{: } s' \in \mathrm{B}_2(\alpha(s')). \end{cases}$$

Note that the function IAT maps s' to the same partition block as α does if $\alpha(s')$ is an S_2 state or not an effectively unstable S_1 state. If $\alpha(s') = s$ is an effectively unstable S_1 state, then s' is either mapped to partition block $\mathrm{B}_1(s)$ or to $\mathrm{B}_2(s)$. In case there is no effectively unstable S_1 state, there is no partition of the state space. In other words, then $\mathrm{IAT}(\alpha) = \alpha$. Note that in this way, all concrete states of s that have the same transitions (after abstraction) are assigned to the same new partition block.

Example 38. *APGA $\hat{\mathscr{H}}$ (right Fig. 6.6) is a state-based abstraction of PA $\mathscr{M}$ (left in Fig. 6.5). Let* $\mathbf{1} = \min$ *and* $\mathbf{2} = \max$ *for $\hat{\mathscr{H}}$ with* $\hat{w} = \mathbf{Fix}\,\mathrm{Prt}^{\mathbf{1}}_{\mathbf{2}}(\bot)$ *for* $\hat{T} = \{\hat{t}_2\}$*, where* $\hat{w}(\hat{v}_0) = 0.5$*,* $\hat{w}(\hat{v}_1) = 0$*,* $\hat{w}(\hat{v}_2) = 0$*,* $\hat{w}(\hat{v}_3) = 0$*,* $\hat{w}(\hat{t}_0) = 0$*,* $\hat{w}(\hat{t}_1) = 0$*,* $\hat{w}(\hat{t}_2) = 1$ *and* $\hat{w}(\hat{t}_3) = 0$*. Note that $\hat{t}_1$ has only one successor, i.e. $\hat{v}_1$, having only* possible *transitions. Therefore, $\hat{\mathscr{H}}$ is not a stable abstraction of PA $\mathscr{M}$.*

Let us discuss how to refine $\hat{\mathscr{H}}$. Let $\mathscr{H}' = \alpha_{\mathrm{PA}}(\mathscr{M})$. For the successor state $\hat{v}_1$ of $\hat{t}_1$, we have $\hat{v}_1 \to \iota_{\hat{t}_3}$ with $\hat{w}(\iota_{\hat{t}_3}) = \hat{w}(\hat{v}_1) = 0$. We separate the concrete states of $\hat{v}_1$ that have a transition (after abstraction) to $\iota_{\hat{t}_3}$, and the only concrete state is v'_3. Therefore, $\hat{v}_1$ is partitioned into two blocks $v_1 = \{v'_1, v'_2\}$ and $v_2 = \{v'_3\}$; and $\mathscr{H}$ (left in Fig. 6.6) is the APGA induced by the new partitioning of the state space of $\mathscr{H}'$. Note that $\mathscr{H}$ is a stable abstraction w.r.t. $\mathbf{1}$, $\mathbf{2}$ *and* $T = \{t_2\}$*; and moreover* $\mathscr{H} \preceq_{\mathrm{sb}} \hat{\mathscr{H}}$*.*

It is possible to refine all effectively unstable player-one states in $\mathscr{H}$ in a single iteration. This will result in a faster convergence of the abstract-refine loop towards a stable abstraction of $\mathscr{M}$. However, at the expense of a larger state space. Instead of refining every effectively unstable state in a single iteration, one may consider refining a selection of them. In this way, unnecessary refinements of some effectively unstable states in the next iteration may be avoided because of splitting of states in the current iteration.

Theorem 19. $\mathrm{IAT}(\alpha) \preceq_{\mathrm{sb}} \alpha$ *for* $\alpha \in Abst_{\mathrm{sb}}(\mathscr{H}')$.

Proof. Let $\mathscr{H}' = (S', \{S'_1, S'_2\}, \{\tau\}, \Delta'_{\mathrm{r}}, \Delta'_{\mathrm{p}}, s'_0)$ be an embedding of PA $\mathscr{M}$ with $\alpha : S' \to S \in Abst_{\mathrm{sb}}(\mathscr{H}')$ such that $\alpha(\mathscr{H}') = \mathscr{H} = (S, \{S_1, S_2\}, \{\tau\}, \Delta_{\mathrm{r}}, \Delta_{\mathrm{p}}, s_0)$ is the induced APGA. Let $\alpha'' = \mathrm{IAT}(\alpha) : S' \to S''$ such that $\alpha''(\mathscr{H}') = \mathscr{H}'' = (S'', \{S''_1, S''_2\}, \{\tau\}, \Delta''_{\mathrm{r}}, \Delta''_{\mathrm{p}}, s''_0)$ is the induced APGA. We define the relation $R \subseteq (S''_1 \times S_1) \cup (S''_2 \times S_2)$ as:

$$R = \{(\alpha''(s), \alpha(s)) \mid s \in S'\}$$

and show that it fulfils the conditions of Def. 23 (page 40).

Let $s''Rs$, then by Def. 40 either $\gamma''(s'') \subset \gamma(s)$ (if s is split) or $\gamma''(s'') = \gamma(s)$:

1. Let $s \xrightarrow{\tau}_{\mathrm{r}} \mu$. By condition (1a) of Def. 31, $\forall s' \in \gamma(s) : s' \xrightarrow{\tau}_{\mathrm{rc}} \mu'_{s'}$ such that $\alpha(\mu'_{s'}) = \mu$. As $\gamma''(s'') \subseteq \gamma(s)$, by condition (1a) of Def. 31 $s'' \xrightarrow{\tau}_{\mathrm{r}} \mu''$ such that $\alpha''(\mu'_{s'}) = \mu''$. As $S''_2 = S_2$, $\mu = \mu''$ and thus $\mu''R\mu$.

2. Let $s'' \xrightarrow{\tau}_{\mathrm{p}} \mu''$ and $s'' \in S''_1$. By condition (1c) of Def. 31, $\exists s' \in \gamma''(s'') : s' \xrightarrow{\tau}_{\mathrm{p}} \mu'_{s'}$ such that $\alpha''(\mu'_{s'}) = \mu''$. As $\gamma''(s'') \subseteq \gamma(s)$, by condition (1b) of Def. 31 $s \xrightarrow{\tau}_{\mathrm{pc}} \mu$ such that $\alpha(\mu'_{s'}) = \mu$. As $S''_2 = S_2$, $\mu'' = \mu$ and thus $\mu''R\mu$.

3. Let $s'' \xrightarrow{\tau}_{\mathrm{p}} \iota_{u''}$ and $s'' \in S''_2$. By condition (2b) of Def. 31, $\exists s' \in \gamma''(s'') : s' \xrightarrow{\tau}_{\mathrm{p}} \iota_{u'}$ such that $\alpha''(\iota_{u'}) = \iota_{u''}$. As $s' \in \gamma(s)$, by condition (2a) of Def. 31 $s \xrightarrow{\tau}_{\mathrm{pc}} \iota_u$ such that $\alpha(\iota_{u'}) = \iota_u$. Thus, $u''Ru$ holds.

□

The fixpoint of the function IAT (Def. 40) is guaranteed to exist for the embeddings of finite PA. This therefore provides the basis to iteratively refine player-one states in an abstraction $\alpha(\mathscr{H}')$, for $\alpha \in Abst_{\mathrm{sb}}(\mathscr{H}')$, such that $\mathbf{Fix}\,\mathrm{IAT}(\alpha)(\mathscr{H}')$ is a stable abstraction of $\mathscr{H}'$.

Theorem 20. *For $\alpha \in Abst_{\mathrm{sb}}(\mathscr{H}')$, $\mathbf{Fix}\,\mathrm{IAT}(\alpha)(\mathscr{H}')$ is a stable abstraction of APGA $\mathscr{H}'$.*

Proof. For PA $\mathscr{M}$ let $\mathscr{H}' = \alpha_{\mathrm{PA}}(\mathscr{M})$ with $T' \subseteq S'_2$. Let $\alpha \in Abst_{\mathrm{sb}}(\mathscr{H}')$ with $w = \mathbf{Fix}\,\mathrm{Prt}^{\mathbf{1}}_{\mathbf{2}}(\perp)$ defined on $\mathscr{H} = \mathbf{Fix}\,\mathrm{IAT}(\alpha)(\mathscr{H}')$ for the objectives $\mathbf{1}, \mathbf{2} \in \{\min, \max\}$ and goal states $\mathbf{Fix}\,\mathrm{IAT}(\alpha)\ (T')$. We need to prove that every $s \in S_2$ is stable, i.e., it satisfies the conditions of Def. 36.

The proof is by contradiction. Assume $\mathscr{H}$ is not a stable abstraction. Let $s \in S_2$ be an unstable state. This implies that there exists at least one unstable player-one state in $\mathrm{Succ}(s)$ — if all states in $\mathrm{Succ}(s)$ are stable, then s is stable. Let $v \in \mathrm{Succ}(s)$ be an unstable state. As s is unstable and v is unstable, then v is effectively unstable (by Def. 38). Therefore, $\gamma(v)$ is split into two partition blocks by Def. 39. This implies that $\mathscr{H}$ is not generated by a fixpoint of IAT. Hence, contradiction.

□

The following corollary follows from Th. 20, and shows that for a given partitioning of states of PA $\mathscr{M}$, the game-based abstraction [KKNP10] is as precise as the state-based APGA-based abstraction (see Def. 31 page 57) when refined to a stable abstraction (see Def. 37). However, the size of the latter is at most the size of the former.

Corollary 6. *For APGA $\mathscr{H}'$, let $\tilde{\mathscr{H}}$ be a game-based abstraction [KKNP10] and $\hat{\alpha}(\mathscr{H}') = \hat{\mathscr{H}}$ be a state-based abstraction of $\mathscr{H}'$ with $\tilde{S}_2 = \hat{S}_2$. Let $\tilde{w}_{\mathbf{12}}$ and $w_{\mathbf{12}}$ be defined on $\tilde{\mathscr{H}}$ and $\mathscr{H} =$* $\mathbf{Fix}\,\mathrm{IAT}(\hat{\alpha})(\mathscr{H}')$ *respectively. Then,* *1.* $\forall \tilde{s} \in \tilde{S}_2, s \in S_2 : \tilde{w}(\tilde{s}) = w(s)$, *and*

2. $|\tilde{S}_1| \geq |S_1| \geq |\hat{S}_1|$.

Example 39. *APGA $\mathscr{H}$ (right) is a game-based abstraction [KKNP10] of PA $\mathscr{M}$ (left) in Fig. 6.5; whereas the left APGA in Fig. 6.6, say $\mathscr{H}''$, is a stable abstraction of $\mathscr{M}$ w.r.t. the objectives $\mathbf{1} = \min$, $\mathbf{2} = \max$ and $T = \{t_3\}$. Note that both models have the same reachability probabilities to t_3 (i.e., 0.25) from the initial states. Moreover, $|S_1| = 6$ and $|S''_1| = 5$, and $|\Delta| = 12$ and $|\Delta''| = 11$.*

In the following example, we show that APGA-based abstraction can be significantly smaller than game-based abstraction having the same player-two state space.

Example 40. *For PA $\mathscr{M}$ (Fig. 6.8 page 94), in Fig. 6.10 (page 96) $\tilde{\mathscr{H}}$ (left) is a game-based-abstraction [KKNP10] with $|\tilde{\Delta}| = 33$ and $|\tilde{S}_1| = 11$, and $\hat{\mathscr{H}}$ (right) is a state-based APGA-based abstraction with $|\hat{\Delta}| = 23$ and $|\hat{S}_1| = 7$. Note that $\hat{\mathscr{H}}$ is a stable abstraction w.r.t. $\mathbf{1} = \max$, $\mathbf{2} = \min$ and $T = \{\hat{t}_3\}$.*

6.1.3 Refinement of Player-two States

In this subsection we describe the functionality of the outer-loop of our framework (see Fig. 6.1). Let $\mathscr{H}$ be a *stable* abstraction of APGA $\mathscr{H}'$ w.r.t. $w_{\mathbf{12}}$. We will determine whether the bounds on reachability probabilities of S_2 states, calculated by $w_{\mathbf{12}}$ and $w_{\mathbf{11}}$ on $\mathscr{H}$, are at most ε-apart. If they are so, we are done; otherwise, we refine some (player-two) states whose bounds on reachability probabilities are not ε-apart.

Definition 41. (Bounded Abstraction) *State $s \in S$ in APGA $\mathscr{H}$ is* ε-bounded *iff* $|w_{\mathbf{12}}(s) - w_{\mathbf{11}}(s)| \leq \varepsilon$ *for a given $\varepsilon \in \mathbb{R}_{[0,1]}$. Distribution $\eta \in \mathrm{Dist}(S)$ is* ε-bounded *iff* $|w_{\mathbf{12}}(\mu) - w_{\mathbf{11}}(\mu)| \leq \varepsilon$. *$\mathscr{H}$ is* ε-bounded *iff $\forall s \in S : s$ is ε-bounded.*

Lemma 5. *If all successors of a state s are ε-bounded, then s is ε-bounded.*

Proof. We prove only for player-one states with $\mathbf{1} = \min$ and $\mathbf{2} = \max$. The proof for the other case is similar.

Let $\mathscr{H}$ be a stable abstraction of $\mathscr{H}'$ with $w_{\mathbf{11}}$ and $w_{\mathbf{12}}$ defined on it. Let $s \in S_1$ with ε-bounded direct successors $\mathrm{Succ}(s)$. Let $\mu \in \Delta_{\mathrm{p}}(s)$ such that $w_{\mathbf{11}}(s) = w_{\mathbf{11}}(\mu)$, i.e., the lower-bound of the minimum probability of s is $w_{\mathbf{11}}(\mu)$. First we show that $w_{\mathbf{11}}(\mu)$ and $w_{\mathbf{12}}(\mu)$ are ε-bounded.

$$\begin{aligned}
|w_{\mathbf{12}}(\mu) - w_{\mathbf{11}}(\mu)| &= |\sum_{t \in \mathrm{Supp}(\mu)} \mu(t) \cdot w_{\mathbf{12}}(t) - \sum_{t \in \mathrm{Supp}(\mu)} \mu(t) \cdot w_{\mathbf{11}}(t)| \\
&= |\sum_{t \in \mathrm{Supp}(\mu)} \mu(t) \cdot (w_{\mathbf{12}}(t) - w_{\mathbf{11}}(t))| \\
&\leq |\sum_{t \in \mathrm{Supp}(\mu)} \mu(t) \cdot \varepsilon| && \text{as } t \text{ is } \varepsilon\text{-bounded} \\
&= \varepsilon
\end{aligned}$$

Now we show that $w_{\mathbf{12}}(s) \leq w_{\mathbf{12}}(\mu)$, i.e., the upper-bound of the minimum probability of s is at most $w_{\mathbf{12}}(\mu)$. Let $\eta \in \Delta_{\mathrm{p}}(s)$: $w_{\mathbf{12}}(\mu) \leq w_{\mathbf{12}}(\eta)$. Then as per the objective of player-one (i.e., $\mathbf{1} = \min$), $w_{\mathbf{12}}(\mu) = \min\{w_{\mathbf{12}}(\mu), w_{\mathbf{12}}(\eta)\}$. This implies that $w_{\mathbf{12}}(s) \leq w_{\mathbf{12}}(\mu)$. As $w_{\mathbf{12}}(s) \leq w_{\mathbf{12}}(\mu)$ and $w_{\mathbf{11}}(s) = w_{\mathbf{11}}(\mu)$, $|w_{\mathbf{12}}(s) - w_{\mathbf{11}}(s)| \leq |w_{\mathbf{12}}(\mu) - w_{\mathbf{11}}(\mu)| \leq \varepsilon$.

□

The following corollary follows from the above lemma.

Corollary 7. *In APGA $\mathscr{H}$, a state s, with $w(s) = w(\mu)$ for $w \in \{w_{\mathbf{11}}, w_{\mathbf{12}}\}$ and $\mu \in \Delta_{\mathrm{p}}(s)$, is ε-bounded iff μ is ε-bounded.*

The following proposition asserts that for embeddings of PA the reachability probability bounds coincide. This is because of the fact that each player-two state has only one direct successor state in embeddings of PA.

Proposition 22. *$\mathscr{H}'$ (the embedding of PA $\mathscr{M}$) is ε-bounded with $\varepsilon = 0$.*

The following corollary follows from the above proposition. It expresses that APGA in which reachability probability bounds of player-two states depend only on one of their direct successors are ε-bounded. It is trivial to reduce such APGA (by removing player-one states and their associated transitions on which player-two states are not dependent for their reachability probabilities) without changing the reachability probabilities of the player-two states.

Corollary 8. *If $\forall s \in S_2$, $w(s) = w(u)$ for $w \in \{w_{\mathbf{11}}, w_{\mathbf{12}}\}$ and some $u \in \mathrm{Succ}(s)$, then APGA $\mathscr{H}$ is ε-bounded.*

Let $\mathrm{UB}(\mathscr{H})_\varepsilon = \{s \in S_2 \text{ is } \varepsilon\text{-unbounded} \mid \exists u, v \in \mathrm{Succ}(s) : u \neq v \wedge w_{\mathbf{12}}(s) = w_{\mathbf{12}}(u) \wedge w_{\mathbf{11}}(s) = w_{\mathbf{11}}(v)\}$ be the set of *ε-unbounded* player-two states in $\mathscr{H}$ whose reachability probability bounds depend on two different successors. Note that for an unbounded APGA $\mathscr{H}$, $\mathrm{UB}(\mathscr{H})_\varepsilon \neq \emptyset$ — if for all $s \in S_2$: $v \in \mathrm{Succ}(s)$ it holds that $w_{\mathbf{12}}(s) = w_{\mathbf{12}}(v)$ and $w_{\mathbf{11}}(s) = w_{\mathbf{11}}(v)$, then by Corollary 8 $\mathscr{H}$ is ε-bounded. We split each state in $\mathrm{UB}(\mathscr{H})_\varepsilon$ into two blocks.

Definition 42. *State $s \in \mathrm{UB}(\mathscr{H})_\varepsilon$ can be split into two partition blocks as:*

- $\mathrm{P}_1(s) = \{s' \in \gamma_{\mathrm{sb}}(s) \mid \exists\iota_{u'} \in \Delta'(s') : w_{\mathbf{12}}(s) = w_{\mathbf{12}}(\alpha_{\mathrm{sb}}(u'))\}$
- $\mathrm{P}_2(s) = \gamma_{\mathrm{sb}}(s) \backslash \mathrm{P}_1(s)$

The above definition separates concrete states of an unbounded state s, whose (player-one) abstract states' reachability probability bounds coincide with $w_{\mathbf{12}}(s)$, from other concrete states. Note that for any $s \in \mathrm{UB}(\mathscr{H})_\varepsilon$ it holds that neither $\mathrm{P}_1(s) = \emptyset$ nor $\mathrm{P}_2(s) = \emptyset$. This is because the reachability probability bounds of s depend on two different successors.

Theorem 21. *For state $s \in \mathrm{UB}(\mathscr{H})_\varepsilon$, $\mathrm{P}_1(s)$ is a proper subset of $\gamma_{\mathrm{sb}}(s)$.*

Proof. The proof is by contradiction. For $s \in \mathrm{UB}(\mathscr{H})_\varepsilon$, assume $\mathrm{P}_1(s) = \gamma_{\mathrm{sb}}(s)$. Then by Def. 42 for every $s' \in \gamma_{\mathrm{sb}}(s)$, $\exists\iota_{u'} \in \Delta'(s') : w_{\mathbf{12}}(s) = w_{\mathbf{12}}(\alpha_{\mathrm{sb}}(u'))$. This implies that $w_{\mathbf{12}}(s) = w_{\mathbf{12}}(v)$ and $w_{\mathbf{11}}(s) = w_{\mathbf{11}}(v)$ for some $v \in \mathrm{Succ}(s)$. But then $s \notin \mathrm{UB}(\mathscr{H})_\varepsilon$. Hence, contradiction. □

The following function splits unbounded states in $\mathrm{UB}(\mathscr{H})_\varepsilon$ and yields a new partitioning of the state space of $\mathscr{H}'$ (i.e., the embedding of PA $\mathscr{M}$). In fact, this function describes the functionality of the outer-loop of our framework (see Fig. 6.1).

Definition 43. (Outer Abstraction Transformer). *The* abstraction transformer *function* OAT : $Abst_{\mathrm{sb}}(\mathscr{H}') \to Abst_{\mathrm{sb}}(\mathscr{H}')$ *is defined for $\hat{\alpha} \in Abst_{\mathrm{sb}}(\mathscr{H}')$: $\mathscr{H} = \mathbf{Fix}\,\mathrm{IAT}(\hat{\alpha})(\mathscr{H}')$ and $s' \in S'$ as:*

$$\mathrm{OAT}(\alpha = \mathbf{Fix}\,\mathrm{IAT}(\hat{\alpha}))(s') = \begin{cases} \alpha(s') & \text{if } \alpha(s') \in S_1 \text{ or } \alpha(s') \in S_2 \backslash \mathrm{UB}(\mathscr{H})_\varepsilon. \\ \mathrm{P}_1(\alpha(s')) & \text{if } \alpha(s') \in \mathrm{UB}(\mathscr{H})_\varepsilon : s' \in \mathrm{P}_1(\alpha(s')). \\ \mathrm{P}_2(\alpha(s')) & \text{if } \alpha(s') \in \mathrm{UB}(\mathscr{H})_\varepsilon : s' \in \mathrm{P}_2(\alpha(s')). \end{cases}$$

Note that the function OAT maps s' to the same partition block as α does if $\alpha(s')$ is a player-one state or a player-two state that is not an element of $\mathrm{UB}(\mathscr{H})_\varepsilon$. If $\alpha(s') = s$ is an element of $\mathrm{UB}(\mathscr{H})_\varepsilon$, then s' is mapped to $\mathrm{P}_1(s)$ if there exists $\iota_{u'} \in \Delta'(s') : w_{\mathbf{12}}(s) = w_{\mathbf{12}}(\alpha_{\mathrm{sb}}(u'))$. Otherwise, it is mapped onto the other partition block, i.e., $\mathrm{P}_2(s)$. In case all player-two states are bounded, there is no partitioning of the state space. In other words, then $\mathrm{OAT}(\alpha) = \alpha$.

Example 41. *For $\varepsilon = 0.4$, $\mathbf{1} = \min$, $\mathbf{2} = \max$ and $T = \{t_3\}$, the APGA $\mathscr{H}$ (left) in Fig. 6.4 is not an ε-bounded abstraction of PA $\mathscr{M}$ in Fig. 6.3, as the difference between reachability probability bounds of t_1 is $0.5 = |0.5 - 0| > \varepsilon$ (0 is the probability with $\mathbf{1} = \min$ and $\mathbf{2} = \min$). It is possible to refine*

$\mathscr{H}$ in order to have reachability probability bounds of t_1 at most ε-apart. $\hat{\mathscr{H}}$ (Fig. 6.4 right) is an ε-bounded abstraction of $\mathscr{M}$ obtained by partitioning the concrete states of t_1 in $\mathscr{H}$ into two blocks, i.e., $\mathrm{P}_1 = \{s_3\} = v'_2$ and $\mathrm{P}_2 = \{s_1, s_2\} = v'_1$. Note that $0 = |0-0| < \varepsilon$ and $0 = |0.5-0.5| < \varepsilon$ for $\hat{t}_1$ and $\hat{t}_2$ respectively.

The following theorem asserts that for $\tilde{\alpha} \in Abst_{\mathrm{sb}}(\mathscr{H}')$ with $\alpha = \mathbf{Fix}\,\mathrm{IAT}(\tilde{\alpha})$, the model induced by $\mathbf{Fix}\,\mathrm{IAT}(\mathrm{OAT}(\alpha))$ has at least as tight bounds on the reachability probabilities of player two states as the model induced by α. Note that the bounds on reachability probabilities are compared only for stable abstractions.

Theorem 22. *For $\tilde{\alpha} \in Abst_{\mathrm{sb}}(\mathscr{H}')$ with $\alpha = \mathbf{Fix}\,\mathrm{IAT}(\tilde{\alpha})$, $\mathbf{Fix}\,\mathrm{IAT}(\mathrm{OAT}(\alpha))(\mathscr{H}')$ has at least as tight bounds on the reachability probabilities of player-two states as $\alpha(\mathscr{H}')$.*

Proof. Let $\mathscr{H}' = \alpha_{\mathrm{PA}}(\mathscr{M})$ for PA $\mathscr{M}$, and let $\tilde{\alpha} \in Abst_{\mathrm{sb}}(\mathscr{H}')$ with $\alpha = \mathbf{Fix}\,\mathrm{IAT}(\tilde{\alpha})$.

Let $\mathscr{H} = \alpha(\mathscr{H}')$ be a stable abstraction (by Th. 20), and let $\mathscr{H}'' = \alpha''_{\mathrm{sb}}(\mathscr{H}')$ be a state-based abstraction with $\Delta''_{\mathrm{r}}(S''_1) = \Delta''_{\mathrm{p}}(S''_1)$ and $S_2 = S''_2$. Then, by Corollary 6 $\forall s \in S_2, s'' \in S''_2$, $s = s''$ implies $w_{\mathbf{12}}(s) = w''_{\mathbf{12}}(s'')$, where $w_{\mathbf{12}}$ and $w''_{\mathbf{12}}$ are defined on $\mathscr{H}$ and $\mathscr{H}''$ respectively.

Similarly, let $\hat{\mathscr{H}} = \mathbf{Fix}\,\mathrm{IAT}(\mathrm{OAT}(\alpha))(\mathscr{H}')$ be a stable abstraction (by Th. 20), and let $\hat{\mathscr{H}}' = \hat{\alpha}'_{\mathrm{sb}}(\tilde{\mathscr{H}})$ be a state-based abstraction with $\hat{\Delta}'_{\mathrm{r}}(\hat{S}'_1) = \hat{\Delta}'_{\mathrm{p}}(\hat{S}'_1)$ and $\hat{S}_2 = \hat{S}'_2$. Then, by Corollary 6 $\forall \hat{s} \in \hat{S}_2, \hat{s}' \in \hat{S}'_2$, $\hat{s} = \hat{s}'$ implies $\hat{w}_{\mathbf{12}}(\hat{s}) = \hat{w}'_{\mathbf{12}}(\hat{s}')$, where $\hat{w}_{\mathbf{12}}$ and $\hat{w}'_{\mathbf{12}}$ are defined on $\hat{\mathscr{H}}$ and $\hat{\mathscr{H}}'$ respectively.

If we prove $\hat{\mathscr{H}}' \preceq_{\mathrm{sb}} \mathscr{H}''$, then by Th. 10, $\hat{\mathscr{H}}'$ has tighter bounds than $\mathscr{H}''$; and subsequently $\hat{\mathscr{H}}$ has tighter bounds than $\mathscr{H}$. Let relation $R \subseteq \hat{S}' \times S''$ be:

$$R = \{(\hat{\alpha}'_{\mathrm{sb}}(s), \alpha''_{\mathrm{sb}}(s)) \mid s \in S'\}$$

We show that R fulfils the conditions of Def. 23 (page 40).

Let $\hat{s}' R s''$,

1. Let $s'' \xrightarrow{\tau}_{\mathrm{r}} \mu''$ and $s'' \in S''_1$. Let $s \in \gamma''_{\mathrm{sb}}(s'')$, by condition (1a) of Def. 31 $s \xrightarrow{\tau}_{\mathrm{rc}} \mu$ such that $\alpha''_{\mathrm{sb}}(\mu) = \mu''$. As all player one transitions are *required* transitions in $\hat{\mathscr{H}}'$, $\hat{s}' = \hat{\alpha}'_{\mathrm{sb}}(s) \xrightarrow{\tau}_{\mathrm{r}} \hat{\alpha}'_{\mathrm{sb}}(\mu) = \hat{\mu}'$. Now by Def. 42 $\forall \hat{u}' \in \hat{S}'_2, \exists! u'' \in S''_2 : \hat{\gamma}'_{\mathrm{sb}}(\hat{u}') \subseteq \gamma''_{\mathrm{sb}}(u'')$, i.e., S''_2 is an abstract state space of $\hat{S}'_2$. Then by Lemma 3, $\hat{\mu}' R \mu''$.
2. Let $\hat{s}' \xrightarrow{\tau}_{\mathrm{p}} \iota_{\hat{v}'}$ and $\hat{s}' \in \hat{S}'_2$. By condition (2b) of Def. 31, $\exists s \in \hat{\gamma}'_{\mathrm{sb}}(\hat{s}') : s \xrightarrow{\tau}_{\mathrm{p}} \iota_v$ such that $\hat{\alpha}'_{\mathrm{sb}}(v) = \hat{v}'$. By condition (2a) of Def. 31, $s'' = \alpha''_{\mathrm{sb}}(s) \xrightarrow{\tau}_{\mathrm{pc}} \alpha''_{\mathrm{sb}}(\iota_v) = v''$. Thus, $\hat{v}' R v''$.

$\square$

The fixpoint of the function OAT (Def. 43) is guaranteed to exist for the embeddings of finite PA (see Proposition 22). This therefore provides the basis to iteratively compute the partitioning of the state space of the model induced by $\alpha \in Abst_{\mathrm{sb}}(\mathscr{H}')$ such that the model induced by $\mathbf{Fix}\,\mathrm{OAT}(\alpha)$ is an ε-bounded abstraction. Note that an ε-bounded abstraction is also a stable abstraction.

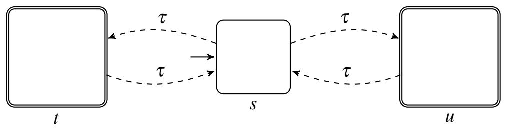


Figure 6.7: APGA $\alpha_{\top}(\mathscr{H}')$ for any APGA $\mathscr{H}'$.

Theorem 23. *For $\alpha \in Abst_{\mathrm{sb}}(\mathscr{H}')$,* **Fix** $\mathrm{OAT}(\alpha)(\mathscr{H}')$ *is an ε-bounded abstraction of $\mathscr{H}'$.*

Proof. The proof is by contradiction. Let $\mathscr{H}' = \alpha_{\mathrm{PA}}(\mathscr{M})$ for PA $\mathscr{M}$, and let $\alpha \in Abst_{\mathrm{sb}}(\mathscr{H}')$. Assume $\mathscr{H} =$ **Fix** $\mathrm{OAT}(\alpha)(\mathscr{H}')$ is an ε-unbounded abstraction. Then, there exists at least one state in S_2 that is not ε-bounded and can be split by Def. 42. Then, $\mathscr{H}$ is not obtained by a fixpoint of OAT (by Def. 43). Hence, contradiction. □

In order to have more concise abstractions than game-based abstractions with ε-bounded player-two states, one can abstract $\mathscr{H}'$ as $\alpha_{\top}(\mathscr{H}') = (\{s = \alpha(S'_1), t = \alpha_{\top}(T'), u = \alpha_{\top}(S'_2 \backslash T')\}, \{\{s\}, \{t, u\}\}, \{\tau\}, \emptyset,$ $\{s \rightarrow_{\mathrm{p}} t, s \rightarrow_{\mathrm{p}} u, t \rightarrow_{\mathrm{p}} s, u \rightarrow_{\mathrm{p}} s\}, s)$ — recall that T' is a set of goal states in $\mathscr{H}'$ — (see Fig. 6.7), and then refine it iteratively by Def. 43. The following corollary follows from Th. 23.

Corollary 9. **Fix** $\mathrm{OAT}(\alpha_{\top})(\mathscr{H}')$ *is at least as concise as game-based abstraction [KKNP10] of $\mathscr{H}'$ with the same set of ε-bounded player-two states.*

6.2 Distribution-based Abstraction-Refinement Framework

The basic steps of the state-based and distribution-based abstraction-refinement frameworks are the same (see Fig. 6.1). In this section, we claim that the state-based framework can be extended to the distribution-based setting. In distribution-based setting, the notion of stability for player-two states is lifted from states to distributions over states. Moreover, the splitting of player-one and player-two states is defined at the level of distributions rather than states. Therefore, instead of repeating everything in the previous section, we discuss only those definitions that need to be lifted from states to distributions over states. Let $\mathscr{H}$ be a distribution-based abstraction of $\mathscr{H}'$, i.e., $\mathscr{H} = \alpha_{\mathrm{db}}(\mathscr{H}')$.

The notion of stability for player-one states is as before (see Def. 35). The same applies to distributions over player-one states. Let $\eth(s) = 1$ if $s \in S_1$ is stable; otherwise, $\eth(s) = 0$. Now, we define stable player-two states in distribution-based abstraction.

Definition 44. (Stable Player-two State) *State $s \in S_2$ in APGA $\mathscr{H}$ is* stable *w.r.t. $w_{\mathbf{12}}$ iff there exists some stable $\mu \in \Delta_{\mathrm{p}}(s)$:*

1. $w_{\mathbf{12}}(s) = w_{\mathbf{12}}(\mu)$, and

2. *$\forall$ unstable $\eta \in \Delta_{\mathrm{p}}(s)$: $w_{\mathbf{12}}(\mu) = \mathbf{2}\{w_{\mathbf{12}}(\mu), \sum_{u \in S_1} \eta(u) \cdot \big(\eth(u) ? w_{\mathbf{12}}(u) : \mathbf{2}\{w_{\mathbf{12}}(\nu) \mid \nu \in \Delta_{\mathrm{p}}(u)\}\big)\}$.*

Condition (1) asserts that $w_{\mathbf{12}}(s)$ depends on some stable distribution $\mu \in \Delta_{\mathrm{p}}(s)$. Condition (2) asserts that for every unstable distribution $\eta \in \Delta_{\mathrm{p}}(s)$, $w_{\mathbf{12}}(\mu)$ is the optimal value between $w_{\mathbf{12}}(\mu)$ and $x = \sum_{u \in S_1} \eta(u) \cdot \big(\eth(u) ? w_{\mathbf{12}}(u) : \mathbf{2}\{w_{\mathbf{12}}(\nu) \mid \nu \in \Delta_{\mathrm{p}}(u)\}\big)$ w.r.t. the objective $\mathbf{2}$. Note that to calculate x, we change the *objective* only for unstable (player-one) states in the support of ν as their non-deterministic behaviour from the abstraction process may affect the reachability probability of s in a refinement (having the same player-two state space) of $\mathscr{H}$.

Intuitively speaking, if condition (2) holds none of the transitions from any direct distribution of s can change the reachability probability of s in any refinement. Therefore, the refinement of direct successors, whether stable or unstable, of a stable player-two state alone does not change the bounds on its reachability probabilities.

Example 42. *Consider the distribution-based abstraction $\mathscr{H}'$ (right) of PA $\mathscr{M}$ (left) in Fig. 6.8. Let $\mathbf{1} = \max$ and $\mathbf{2} = \min$ with $w = \mathbf{Fix}\, \mathrm{Prt}^{\mathbf{1}}_{\mathbf{2}}(\perp)$ for $T = \{t'_3\}$, where $w(v'_0) = 0.5$, $w(v'_1) = 0.5$, $w(v'_2) = 1$, $w(v'_3) = 1$, $w(v'_4) = 0$, $w(v'_5) = 0.5$, $w(t'_0) = 0$, $w(t'_1) = 0.5$, $w(t'_2) = 0$ and $w(t'_3) = 1$. Let us check whether t'_1 is stable. As $w(t'_1) = w(\iota_{v'_1}) = 0.5$ and $\iota_{v'_1} \in \hat{\Delta}_{\mathrm{p}}(t'_1) = \{\iota_{v'_1}, \iota_{v'_2}, \mu = [\![0.4v'_2, 0.6v'_3]\!]\}$ is stable, condition (1) of Def. 44 holds. Let us consider the only unstable distribution μ in $\hat{\Delta}_{\mathrm{p}}(t'_1)$. As $w(\iota_{v'_1}) = 0.5 = \min\{w(\iota_{v'_1}) = 0.5, \mu(v'_2) \cdot w(v'_2) + \mu(v'_3) \cdot 0.4 = 0.4 + 0.6 * 0.4 = 0.64\}$, condition (2) of Def. 44 holds. Note that as v'_3 is unstable we consider the minimum instead of the maximum probability from v'_3 to T. Thus, t'_1 is stable. Note that as all player-two states are stable, $\mathscr{H}'$ is stable.*

Now assume a state-based APGA-based abstraction $\hat{\mathscr{H}}'$ of PA $\mathscr{M}$ with the same state space partitioning as for $\mathscr{H}'$. APGA $\hat{\mathscr{H}}'$ can be obtained by merging the states $\hat{v}_3$ and $\hat{v}_4$ in the right APGA of Fig. 6.10. Note that APGA $\hat{\mathscr{H}}'$ will not be a stable abstraction. This is because the abstract state, say $\hat{v}'_3$, representing s_{10} and s_{11}, will be unstable as it will have only possible *transitions. $\hat{v}'_3$ therefore may affect the reachability probability of its corresponding player-two state, say $\hat{t}'_1$. In order to check the stability of $\hat{t}'_1$, we therefore will consider minimum (instead of maximum) reachability probability from $\hat{v}'_3$ to T which is* 0.4. *This violates condition (2) of Def. 36.*

6.2.1 Refinement of Player-one States

We now discuss how effectively unstable player-one states in $\mathrm{EUS}(\mathscr{H})$ (see Def. 38) are split. We split each effectively unstable state into two blocks.

Definition 45. *For state $s \in \mathrm{EUS}(\mathscr{H})$, let $\mu \in \Delta_{\mathrm{p}}(s) : w_{\mathbf{12}}(s) = w_{\mathbf{12}}(\mu)$. Then,*

- $\mathrm{B}_1(s) = \bigcup\{\mathrm{Supp}(\eta') \mid \eta' \in \gamma_{\mathrm{db}}(\iota_s) : \eta' \to \nu' \wedge \alpha_{\mathrm{db}}(\nu') = \mu\}$
- $\mathrm{B}_2(s) = \gamma_{\mathrm{db}}(s) \backslash \mathrm{B}_1(s)$

Note that for any $s \in \mathrm{EUS}(\mathscr{H})$ it holds that neither $\mathrm{B}_1(s) = \emptyset$ nor $\mathrm{B}_2(s) = \emptyset$. This is because if all distributions over concrete states of s have a transition to *some* ν' with $\alpha_{\mathrm{db}}(\nu') = \mu$, then s will not be

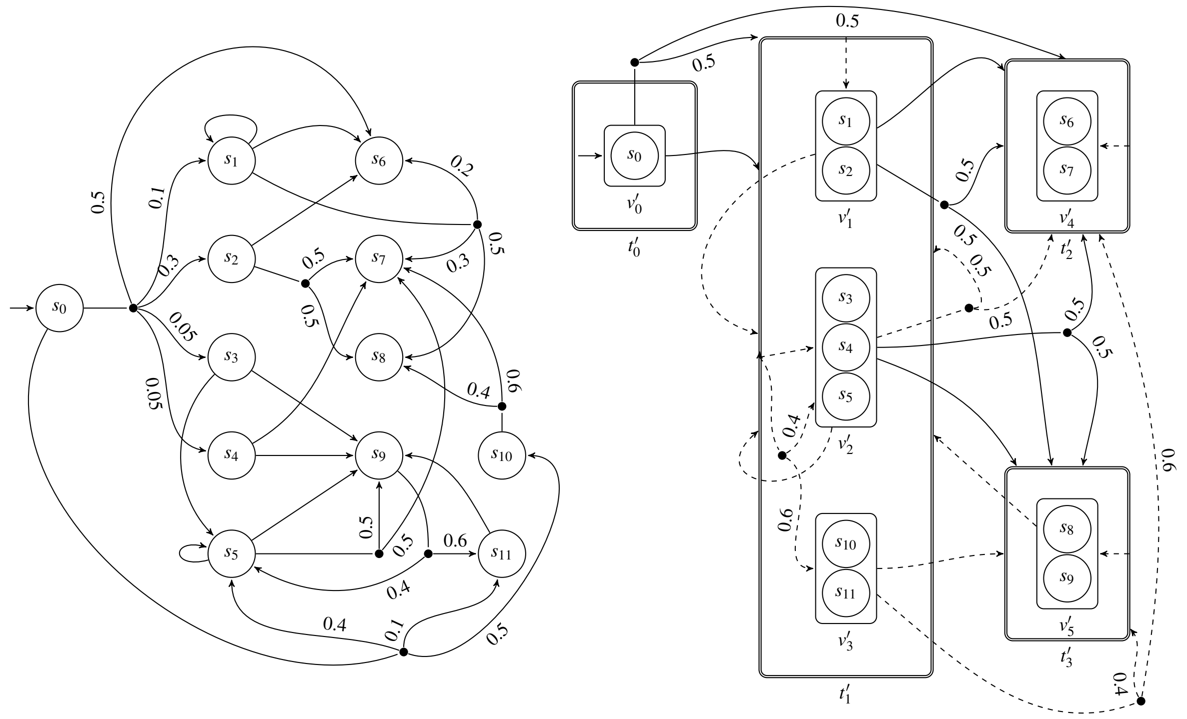


Figure 6.8: PA $\mathscr{M}$ (left) with $\mathscr{H}' = \alpha_{\text{db}}(\alpha_{\text{PA}}(\mathscr{M}))$ (right). $\mathscr{H}'$ is a stable abstraction w.r.t. $\mathbf{1} = \max$, $\mathbf{2} = \min$ and $T = \{t'_3\}$, with $|\Delta'| = 24$, $|S'_1| = 6$ and $|S'_2| = 4$.

an unstable state. Recall that all distributions in $\gamma_{\text{db}}(\iota_s)$ are convex combinations of maximal singular-distributions in $\gamma_{\text{db}}(\iota_s)$, and therefore it suffices to consider only maximal singular-distributions in $\gamma_{\text{db}}(\iota_s)$ (see Def. 34 on page 63) to split s.

Moreover, by this way the maximal singular-distributions (over the concrete states of s) that have the same transitions after abstraction are assigned to the same new partition block.

We can now use Def. 40 for refining player-one states, thus giving the functionality of the inner-loop of our distribution-based framework (see Fig. 6.1).

Example 43. *Consider the distribution-based abstraction $\hat{\mathscr{H}}$ (Fig. 6.9 right) of PA $\mathscr{M}$ (Fig. 6.8 left). Let $\mathbf{1} = \max$ and $\mathbf{2} = \min$ with $w = \mathbf{Fix}\,\text{Prt}_{\mathbf{2}}^{\mathbf{1}}(\perp)$ for $T = \{\hat{t}_4\}$ where $w(\hat{v}_0) = 0.7$, $w(\hat{v}_1) = 0.5$, $w(\hat{v}_2) = 1$, $w(\hat{v}_3) = 0$, $w(\hat{v}_4) = 0.7$, $w(\hat{t}_0) = 0$, $w(\hat{t}_1) = 1$, $w(\hat{t}_2) = 0.5$, $w(\hat{t}_3) = 0$ and $w(\hat{t}_4) = 1$. Let us check whether $\hat{t}_2$ is stable. As $w(\hat{t}_2) = w(\iota_{\hat{v}_1}) = 0.5$ and $\iota_{\hat{v}_1} \in \hat{\Delta}(\hat{t}_2) = \{\iota_{\hat{v}_1}, \iota_{\hat{v}_2}\}$ is stable, condition (1) of Def. 44 holds. As $\iota_{\hat{v}_2} \in \hat{\Delta}(\hat{t}_2)$ is unstable and $w(\iota_{\hat{v}_1}) \neq \min\{w(\iota_{\hat{v}_1}) = 0.5, 0\}$, condition (2) of Def. 44 does not hold (note that as $\hat{v}_2$ is unstable we consider the minimum instead of the maximum probability from $\hat{v}_2$ to T). Thus, $\hat{t}_2$ is unstable.*

Now let us refine $\hat{v}_2$. As $w(\hat{v}_2) = 1$ is because of a possible *transition from $\hat{v}_2$ to T, we find out which concrete distributions in $\gamma_{\text{db}}(\iota_{\hat{v}_2}) = \{[\![0.5v_3, 0.5v_4]\!], \iota_{v_5}, \iota_{v_{10}}, \iota_{v_{11}}\}$ have a transition to concrete states of T*

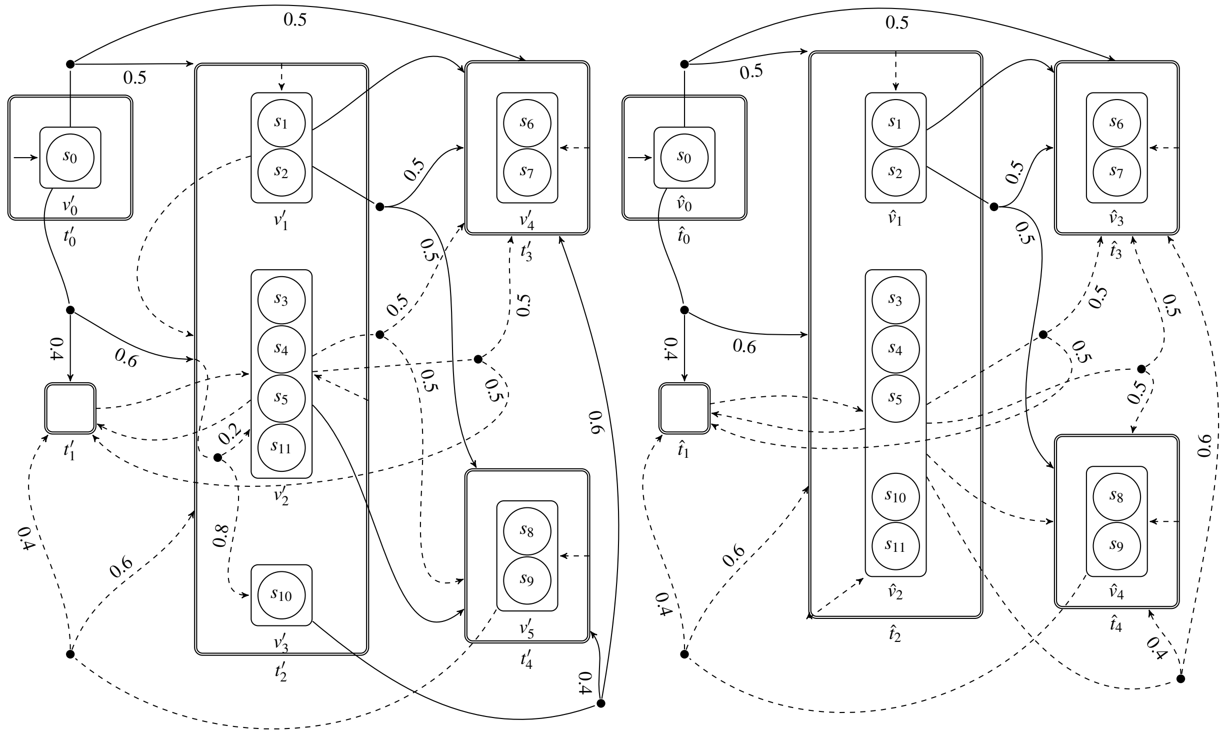


Figure 6.9: For PA $\mathscr{M}$ (Fig. 6.8 left), $\hat{\mathscr{H}} = \alpha_{\text{db}}(\alpha_{\text{PA}}(\mathscr{M}))$ (right) and $\mathscr{H}' = \text{IAT}(\alpha_{\text{db}})(\alpha_{\text{PA}}(\mathscr{M}))$ (left). $\mathscr{H}'$ is a stable abstraction w.r.t. $\mathbf{1} = \max$, $\mathbf{2} = \min$ and $T = \{t'_4\}$, with $|\Delta'| = 26$ and $|S'_1| = 6$.

with probability 1; and split $\gamma_{\text{db}}(\iota_{\hat{v}_2})$ based on this information. We have $\text{B}_1(v'_2) = \{v_3, v_4, v_5, v_{11}\}$ and $\text{B}_2(v'_2) = \{v_{10}\}$ as all accept $\iota_{v_{10}}$ in $\gamma_{\text{db}}(\iota_{\hat{v}_2})$ have a transition to concrete states of T with probability 1. The abstraction obtained after partitioning the concrete states of $\hat{v}_2$ is the APGA $\mathscr{H}'$ (Fig. 6.9 left). Note that t'_2 is stable in $\mathscr{H}'$.

6.2.2 Refinement of Player-two States

Let $\text{UB}(\mathscr{H})_\varepsilon = \{s \in S_2 \text{ is } \varepsilon\text{-unbounded} \mid \exists \mu, \nu \in \Delta_{\text{p}}(s) : \mu \neq \nu \wedge w_{\mathbf{12}}(s) = w_{\mathbf{12}}(\mu) \wedge w_{\mathbf{11}}(s) = w_{\mathbf{11}}(\nu)\}$ be the set of ε-*unbounded* player-two states in $\mathscr{H}$ whose reachability probability bounds depend on two different direct successor distributions. For an unbounded APGA $\mathscr{H}$, we can show that $\text{UB}(\mathscr{H})_\varepsilon \neq \emptyset$ — if for all $s \in S_2$: $\nu \in \Delta_{\text{p}}(s)$ it holds that $w_{\mathbf{12}}(s) = w_{\mathbf{12}}(\nu)$ and $w_{\mathbf{11}}(s) = w_{\mathbf{11}}(\nu)$, then it can be proved that $\mathscr{H}$ is ε-bounded (see Corollary 8). Moreover, based on the definition of $\text{UB}(\mathscr{H})_\varepsilon$, we see that each state in $\text{UB}(\mathscr{H})_\varepsilon$ has at least two direct successor states. Therefore, we can split each state in $\text{UB}(\mathscr{H})_\varepsilon$ into two blocks.

Definition 46. *For state $s \in \text{UB}(\mathscr{H})_\varepsilon$, let $v \in \text{Succ}(s) : w_{\mathbf{11}}(v) \neq w_{\mathbf{12}}(v)$. Then,*

- $\text{P}_1(s) = \{s' \in \gamma_{\text{db}}(s) \mid \text{Succ}(s') \subseteq \gamma_{\text{db}}(v)\}$

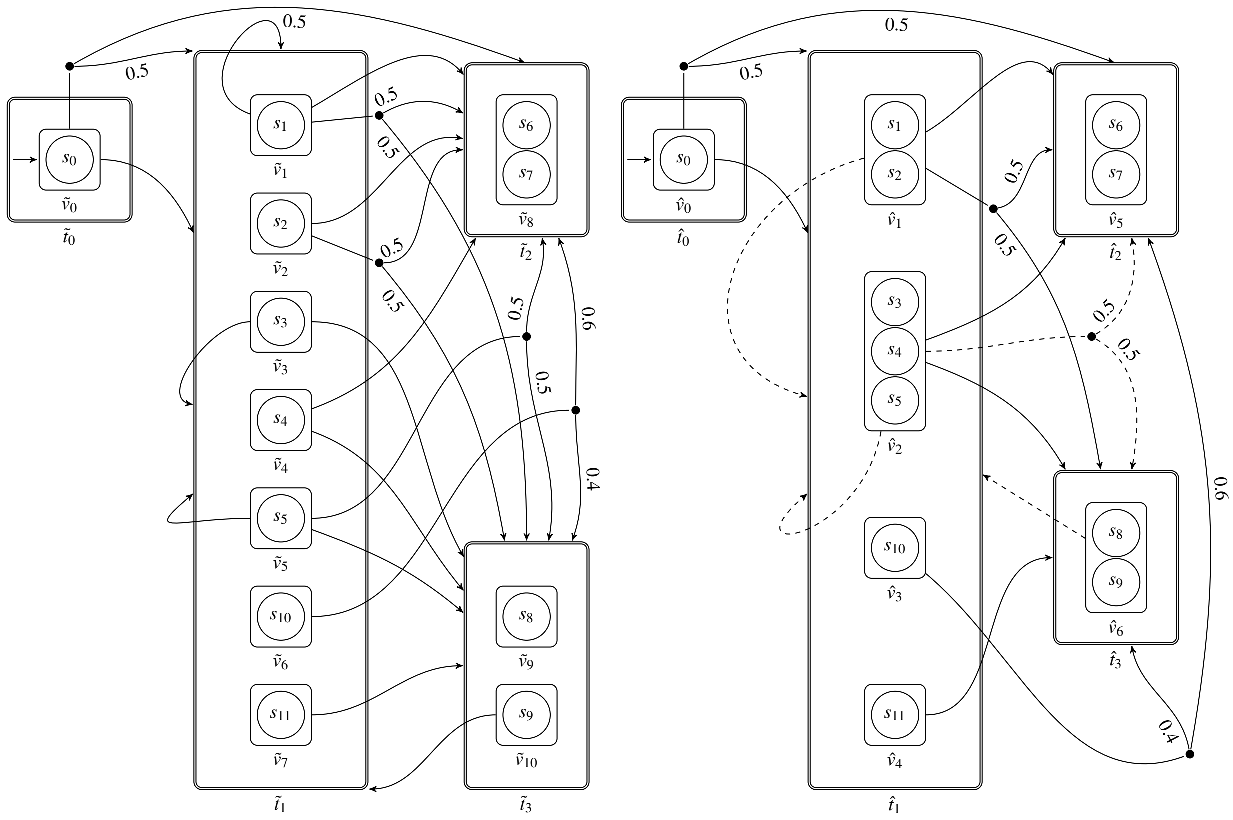


Figure 6.10: For PA $\mathscr{M}$ (Fig. 6.8), $\tilde{\mathscr{H}}$ (left) is a game-based-abstraction [KKNP10] with $|\tilde{\Delta}| = 33$ and $|\tilde{S}_1| = 11$, and $\hat{\mathscr{H}}$ (right) is a state-based abstraction with $|\hat{\Delta}| = 23$ and $|\hat{S}_1| = 7$. Note that $\hat{\mathscr{H}}$ is stable w.r.t. $\mathbf{1} = \max$, $\mathbf{2} = \min$ and $T = \{\hat{t}_3\}$.

- $\mathrm{P}_2(s) = \gamma_{\mathrm{db}}(s) \backslash \mathrm{P}_1(s)$

The above definition separates concrete states, whose direct successors are the concrete states of $v \in \mathrm{Succ}(s) : w_{\mathbf{11}}(v) \neq w_{\mathbf{12}}(v)$, of an unbounded state s from other concrete states. Note that for any $s \in \mathrm{UB}(\mathscr{H})_\varepsilon$ it holds that neither $\mathrm{P}_1(s) = \emptyset$ nor $\mathrm{P}_2(s) = \emptyset$. This is because in case all successors of s have the same reachability probability bounds, s will not be an unbounded state. The above mentioned way of splitting player-two states can also be used in state-based framework.

We can now use Def. 43 for refining player-one as well as player-two states, thus giving the functionality of the outer-loop of our distribution-based framework (see Fig. 6.1).

Distribution-based APGA-based vs. PGA-based abstractions. In the following example, we show that our distribution-based APGA-based abstraction (after refinement) is as precise as PGA-based abstraction [SK14] for the same player-two state spaces, but APGA-based abstraction is comparatively smaller in size.

Example 44. *Consider the distribution-based APGA-based abstraction $\mathscr{H}'$ (right) of PA $\mathscr{M}$ (left) in Fig. 6.8. Assume $\mathscr{H}$ is a PGA-based abstraction of $\mathscr{M}$ with $S_2 = S'_2$. In $\mathscr{H}$, the concrete states s_1, s_2, s_5, s_8, s_9, s_{10} and s_{11} will be abstracted by different player-one states. $\mathscr{H}$ is therefore larger in size than $\mathscr{H}'$, but both of them bound the maximum probability to states $\{s_8, s_9\}$ in $\mathscr{M}$, which is 0.7, by* $[0.5, 1]$.

APGA-based state-based vs. distribution-based abstractions. Finally, we present an example that compares relative precision and concision of state-based and distribution-based APGA-based abstractions of a PA.

Example 45. *The maximum probability in PA $\mathscr{M}$ (Fig. 6.8) to reach states $\{s_8, s_9\}$ equals* 0.7. *By Corollary 3, this probability lies in* $[0.4, 1]$ *for the game-based [KKNP10] and state-based APGA-based abstractions $\tilde{\mathscr{H}}$ and $\hat{\mathscr{H}}$ respectively in Fig. 6.10. Instead, distribution-based abstraction $\mathscr{H}'$ (Fig. 6.8 right) yields* $[0.5, 1]$. *Note that $\hat{\mathscr{H}}$ and $\mathscr{H}'$ are stable abstractions w.r.t.* $\mathbf{1} = \max$ *and* $\mathbf{2} = \min$.

Moreover, although $\tilde{\mathscr{H}}$ and $\hat{\mathscr{H}}$ yield the same bounds, i.e., $[0.4, 1]$, *but (in terms of number of transitions and states) $\hat{\mathscr{H}}$ is smaller than $\tilde{\mathscr{H}}$. Furthermore, the sizes of $\mathscr{H}'$ and $\hat{\mathscr{H}}$ are comparable but $\mathscr{H}'$ yields tighter bounds, i.e.,* $[0.5, 1]$.

We conjecture that the theorems given for state-based abstraction-refinement framework can be extended to the distribution-based abstraction-refinement framework.

6.3 Summary and Discussion

In this chapter, we proposed a *state-based* and a *distribution-based* abstraction-refinement framework that automatically generate APGA-based abstractions of closed PA having finitely many states. The abstractions are aimed at the verification of reachability properties. A main characteristic of these frameworks that distinguishes them from the abstraction-refinement framework for MDPs [KKNP10] is an *additional refinement loop* that is embedded inside the main loop. The inner loop refines player-one states until the non-deterministic behaviour from the abstraction process in player-one states has no impact on the reachability probabilities of player-two states. The outer loop works in the same way as in [KKNP10], i.e., it refines player-two states whose reachability probabilities are not at most ε-apart. We showed that our APGA-based state-based and distribution-based abstractions (after refinement) are as precise as game-based [KKNP10] and PGA-based abstractions [SK14] respectively for the same partitioning of the concrete state spaces, but APGA-based abstractions may be smaller in size. Moreover, we illustrated with examples that the distribution-based framework may induce more precise and concise abstract models of PA than the state-based framework.

Related work: In the literature, different abstraction-refinement frameworks have been discussed, e.g. the CEGAR framework [CGJ$^+$00, HWZ08] for *existential* abstractions (see details on page 2); *three-valued analysis* based frameworks [SG07, dAR07, KKLW12] for *modal* abstractions (see details on page 3); *game-based* frameworks [KKNP10, WZ10] for *game-based* abstractions (see details on page 3), etc. In fact, the framework of [KKNP10] is a special case of our state-based framework if the process of abstraction does not induce any non-deterministic behaviour in player-one states. More recently, an abstraction-refinement framework (MeGARA) [BFH$^+$14] is given for *Markov automata* that combines the techniques of [KKNP10] and [WZ10] for Markovian and non-Markovian states respectively, i.e., Markovian states are abstracted using game-based abstraction [KKNP10] and non-Markovian states using Menu-based abstraction [WZ10] (see details on page 5). To the best of our knowledge, our abstraction-refinement framework is unique in the sense that it combines *modal* frameworks with *game-based* frameworks.

Future extensions: In our framework some of the definitions assume that player-one states have only non-deterministic choices but not the probabilistic ones as in SGs. Therefore, it works well for PA and not for PGA. Possible future work consists of:

- adopting our framework to PGA rather than PA,
- exploring new methods to decide which states should be refined to have a faster convergence with minimal state space,
- generating modal game-based abstractions using predicate-abstraction techniques [CKSY05, KKNP08], and adopting this framework to refine such models, and
- implementing our framework and conducting some case studies.

7 Conclusion

Among the different formal methods techniques used for the production of trustworthy ICT systems, model checking is quite renowned. The main problem in its widespread application is the tremendous (or even infinite) sizes of the state spaces of even small high-level models given as Petri nets, programs in guarded command languages, etc, — known as the *state space explosion* problem. In the literature, different methods have been proposed to tackle this problem, *abstraction* is one of the most prominent ones. This thesis proposes some new abstraction techniques for probabilistic systems aiming at preserving the reachability probabilities of concrete models. This aims at extending the existing limits of model checking of probabilistic systems.

The key aspect in our work is that we perceive probabilistic systems not just as stochastic processes but transformers of probabilities as well. Therefore, we treated distributions over states rather than states as first-class citizens and lifted the notion of abstraction from states to *distributions over states*. Moreover, we also defined formal relationships between concrete and abstract models at the level of distributions over states. To be precise, we gave *game-based* abstractions of probabilistic systems over the distributions over states; and *distribution-based simulation* and *alternating simulation* relations to compare concrete and abstract models. We showed that simulation relations preserve reachability probabilities in case of collaborating players; whereas alternating simulation relations do so in case of competing players.

Moreover, we introduced *modal game-based abstraction*, by merging *modal* and *game-based* abstraction techniques. This yielded *modal* stochastic games in which player-two completely handles behaviour induced by abstraction whereas player-one handles behaviour induced by abstraction and from concrete models. Due to this additional non-deterministic behaviour in player-one states, the bounds of reachability probabilities in modal stochastic games are at most as tight as in stochastic games, but modal games are comparatively smaller in size. Modal stochastic games are compared with concrete models using *refinement* relations preserving the reachability probabilities in case of both competing and collaborating players. We also lifted the modal game-based abstraction from states to distributions over states and showed with examples that in some cases this technique may induce more precise as well as concise models than *state-based* modal game-based abstraction.

Furthermore, our abstraction techniques are compositional, i.e., they allow systems to be broken down into components and abstract each component individually, which can then be plugged together to get abstract models of complete systems.

Finally, we introduced a state-based and a distribution-based abstraction-refinement framework for probabilistic automata. Intuitively, states of modal game-based abstractions are refined in two nested loops.

In the inner loop, player-one states are refined until the behaviour from abstraction in player-one states has no impact on the reachability probabilities of player-two states; whereas in the outer loop, player-two states are refined until the difference between probability bounds of player-two states does not exceed certain threshold value.

Our contributions are *theoretical* in nature. It would be interesting to implement our proposed abstraction-refinement framework and check its practical usefulness by performing some industrial case studies.

Bibliography

[ADD00] Robert B. Ash and Cathrine A. Doléans-Dade. *Probability & Measure Theory, 2nd Edition*. Academic Press, 2000.

[AHKV98] Rajeev Alur, Thomas A. Henzinger, Orna Kupferman, and Moshe Y. Vardi. Alternating refinement relations. In *Concurrency Theory*, volume 1466 of *LNCS*, pages 163–178, 1998.

[Alf99] Luca de Alfaro. Computing minimum and maximum reachability times in probabilistic systems. In *Concurrency Theory*, volume 1664 of *LNCS*, pages 66–81, 1999.

[BEMC00] Christel Baier, Bettina Engelen, and Mila E. Majster-Cederbaum. Deciding bisimilarity and similarity for probabilistic processes. *Computer and System Sciences*, 60(1):187–231, 2000.

[BFH$^+$14] Bettina Braitling, Luis María Ferrer Fioriti, Hassan Hatefi, Ralf Wimmer, Bernd Becker, and Holger Hermanns. MeGARA: Menu-based game abstraction and abstraction refinement of Markov automata. In *Quantitative Aspects of Programming Languages and Systems*, volume 154 of *EPTCS*, pages 48–63, 2014.

[BHH$^+$09] Eckard Böde, Marc Herbstritt, Holger Hermanns, Sven Johr, Thomas Peikenkamp, Reza Pulungan, Jan Rakow, Ralf Wimmer, and Bernd Becker. Compositional dependability evaluation for STATEMATE. *IEEE Software Engineering*, 35(2):274–292, 2009.

[BK08] Christel Baier and Joost-Pieter Katoen. *Principles of Model Checking*. The MIT Press, 2008.

[BT91] Dimitri P. Bertsekas and John N. Tsitsiklis. An analysis of stochastic shortest path problems. *Mathematics of Operations Research*, 16:580–595, 1991.

[CC77] Patrick Cousot and Radhia Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Principles of programming languages*, pages 238–252, 1977.

[CC92] Patrick Cousot and Radhia Cousot. Abstract interpretation frameworks. *Logic and Computation*, 2(4):511–547, 1992.

[CCG$^+$02] Alessandro Cimatti, Edmund M. Clarke, Enrico Giunchiglia, Fausto Giunchiglia, Marco Pistore, Marco Roveri, Roberto Sebastiani, and Armando Tacchella. NuSMV version 2: An opensource tool for symbolic model checking. In *Computer-Aided Verification*, volume 2404 of *LNCS*, 2002.

[CDL$^+$11] Benoît Caillaud, Benoît Delahaye, Kim G. Larsen, Axel Legay, Mikkel L. Pedersen, and Andrzej Wasowski. Constraint Markov chains. *Theoretical Computer Science*, 412(34):4373–4404, 2011.

[CGJ$^+$00] Edmund M. Clarke, Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith. Counterexample-guided abstraction refinement. In *Computer-Aided Verification*, volume 1855 of *LNCS*, pages 154–169, 2000.

[CGJ$^+$03] Edmund Clarke, Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith. Counterexample-guided abstraction refinement for symbolic model checking. *Journal of the ACM*, 50(5):752–794, 2003.

[CGL94] Edmund M. Clarke, Orna Grumberg, and David E. Long. Model checking and abstraction. *Programming Languages and Systems*, 16(5):1512–1542, 1994.

[CKSY05] Edmund M. Clarke, Daniel Kroening, Natasha Sharygina, and Karen Yorav. SAT-based predicate abstraction for ANSI-C. In *Tools and Algorithms for the Construction and Analysis of Systems*, volume 3440 of *LNCS*, pages 570–574, 2005.

[CL88] Anne Condon and Richard E. Ladner. Probabilistic game automata. *Computer and System Sciences*, 36(3):452–489, 1988.

[Con92] Anne Condon. The complexity of stochastic games. *Information and Computation*, 96:203–224, 1992.

[dAGJ04] Luca de Alfaro, Patrice Godefroid, and Radha Jagadeesan. Three-valued abstractions of games: Uncertainty, but with precision. In *Logic in Computer Science*, pages 170–179, July 2004.

[dAR07] Luca de Alfaro and Pritam Roy. Solving games via three-valued abstraction refinement. In *Concurrency Theory*, volume 4703 of *LNCS*, pages 74–89, 2007.

[DHR08] Laurent Doyen, Thomas A. Henzinger, and Jean-François Raskin. Equivalence of labeled Markov chains. *Foundations of Computer Science*, 19(3):549–563, 2008.

[Dij75] Edsger W. Dijkstra. Guarded commands, non-determinancy and a calculus for the derivation of programs. In *Language Hierarchies and Interfaces*, pages 111–124, 1975.

[DJL01] Pedro R. D'Argenio, Henrik E. Jensen, and Kim G. Larsen. Reachability analysis of probabilistic systems by successive refinements. In *Process Algebra and Probabilistic Methods, Performance Modeling and Verification*, volume 2165 of *LNCS*, pages 39–56, 2001.

[DKL$^+$11] Benoît Delahaye, Joost-Pieter Katoen, Kim G. Larsen, Axel Legay, Mikkel L. Pedersen, Falak Sher, and Andrzej Wasowski. Abstract probabilistic automata. In *Verification, Model Checking, and Abstract Interpretation*, volume 6538 of *LNCS*, pages 324–339, 2011.

[DKL$^+$13] Benoît Delahaye, Joost-Pieter Katoen, Kim G. Larsen, Axel Legay, Mikkel L. Pedersen, Falak Sher, and Andrzej Wasowski. Abstract probabilistic automata. *Information and Computation*, 232:66–116, 2013.

[EHZ10] Christian Eisentraut, Holger Hermanns, and Lijun Zhang. On probabilistic automata in continuous time. In *Logic in Computer Science*, pages 342–351, 2010.

[FZ14] Yuan Feng and Lijun Zhang. When equivalence and bisimulation join forces in probabilistic automata. In *Formal Methods*, volume 8442 of *LNCS*, pages 247–262, 2014.

[GB06] Marcus Größer and Christel Baier. Partial order reduction for Markov decision processes: A survey. In *Formal Methods for Components and Objects*, volume 4111 of *LNCS*, pages 408–427, 2006.

[Har87] David Harel. Statecharts: A visual formalism for complex systems. *Science of Computer Programming*, 8(3):231–274, 1987.

[Her90] Ted Herman. Probabilistic self-stabilization. *Information Processing Letters*, 35(2):63–67, 1990.

[HHK02] Holger Hermanns, Ulrich Herzog, and Joost-Pieter Katoen. Process algebra for performance evaluation. *Theoretical Computer Science*, 274(1-2):43–87, 2002.

[HJM03] Thomas A. Henzinger, Ranjit Jhala, and Rupak Majumdar. Counterexample-guided control. In *Automata, Languages and Programming*, volume 2719 of *LNCS*, pages 886–902, 2003.

[HJS01] Michael Huth, Radha Jagadeesan, and David Schmidt. Modal transition systems: A foundation for three-valued program analysis. In *Programming Languages and Systems*, volume 2028 of *LNCS*, pages 155–169, 2001.

[HKK14] Holger Hermanns, Jan Krcál, and Jan Kretínský. Probabilistic bisimulation: Naturally on distributions. In *Concurrency Theory*, volume 8704 of *LNCS*, pages 249–265, 2014.

[HSM97] Jifeng He, Karen Seidel, and Annabelle McIver. Probabilistic models for the guarded command language. *Science of Computer Programming*, 28(2-3):171–192, 1997.

[HWZ08] Holger Hermanns, Björn Wachter, and Lijun Zhang. Probabilistic CEGAR. In *Computer-Aided Verification*, volume 5123 of *LNCS*, pages 162–175, 2008.

[JHK02] David N. Jansen, Holger Hermanns, and Joost-Pieter Katoen. A probabilistic extension of UML statecharts: Specification and verification. In *Formal Techniques in Real-Time and Fault-Tolerant Systems*, volume 2469 of *LNCS*, pages 355–374, 2002.

[JL91] Bengt Jonsson and Kim G. Larsen. Specification and refinement of probabilistic processes. In *Logic in Computer Science*, pages 266–277, 1991.

[Kat10] Mark Kattenbelt. *Automated Quantitative Software Verification*. PhD thesis, University of Oxford, 2010.

[KKLW08] Joost-Pieter Katoen, Daniel Klink, Martin Leucker, and Verena Wolf. Abstraction for stochastic systems by Erlang's method of stages. In *Concurrency Theory*, volume 5201 of *LNCS*, pages 279–294, 2008.

[KKLW12] Joost-Pieter Katoen, Daniel Klink, Martin Leucker, and Verena Wolf. Three-valued abstraction for probabilistic systems. *Logic and Algebraic Programming*, 81(4):356–389, 2012.

[KKN09] Joost-Pieter Katoen, Daniel Klink, and Martin R. Neuhäußer. Compositional abstraction for stochastic systems. In *Formal Modeling and Analysis of Timed Systems*, volume 5813 of *LNCS*, pages 195–211, 2009.

[KKNP08] Mark Kattenbelt, Marta Z. Kwiatkowska, Gethin Norman, and David Parker. Game-based probabilistic predicate abstraction in PRISM. volume 220, 2008.

[KKNP10] Mark Kattenbelt, Marta Z. Kwiatkowska, Gethin Norman, and David Parker. A game-based abstraction-refinement framework for Markov decision processes. *Formal Methods in System Design*, 36(3):246–280, 2010.

[KKZJ07] Joost-Pieter Katoen, Tim Kemna, Ivan Zapreev, and David N. Jansen. Bisimulation minimisation mostly speeds up probabilistic model checking. In *Tools and Algorithms for the Construction and Analysis of Systems*, volume 4424 of *LNCS*, pages 87–101, 2007.

[Kli10] Daniel Klink. *Three-Valued Abstraction for Stochastic Systems*. PhD thesis, RWTH Aachen University, 2010.

[KNP09] Marta Z. Kwiatkowska, Gethin Norman, and David Parker. PRISM: probabilistic model checking for performance and reliability analysis. *SIGMETRICS Performance Evaluation Review*, 36(4):40–45, 2009.

[Kud05] Manfred Kudlek. Probability in Petri nets. *Fundamenta Informaticae*, 67(1-3):121–130, 2005.

[KZH$^+$11] Joost-Pieter Katoen, Ivan S. Zapreev, Ernst Moritz Hahn, Holger Hermanns, and David N. Jansen. The ins and outs of the probabilistic model checker MRMC. *Performance Evaluation*, 68(2):90–104, 2011.

[Lar90] Kim Guldstrand Larsen. Modal specifications. In *Automatic Verification Methods for Finite State Systems*, volume 407 of *LNCS*, pages 232–246, 1990.

[LMZL11] Yang Liu, Huaikou Miao, Hongwei Zeng, and Zhuang Li. Probabilistic Petri net and its logical semantics. In *Software Engineering Research, Management and Applications*, pages 73–78, 2011.

[LSV07] Nancy A. Lynch, Roberto Segala, and Frits W. Vaandrager. Observing branching structure through probabilistic contexts. *SIAM Journal on Computing*, 37(4):977–1013, 2007.

[LT88a] Kim G. Larsen and Bent Thomsen. Compositional proofs by partial specification of processes. In *Mathematical Foundations of Computer Science*, volume 324 of *LNCS*, pages 414–423, 1988.

[LT88b] Kim G. Larsen and Bent Thomsen. A modal process logic. In *Logic in Computer Science*, pages 203–210, 1988.

[LV92] Nancy Lynch and Frits Vaandrager. Forward and backward simulations for timing-based systems. In *Real-Time: Theory in Practice*, volume 600 of *LNCS*, pages 397–446, 1992.

[Mil89] Robin Milner. *Communication and Concurrency*. Prentice-Hall, Inc., 1989.

[Mio12] Matteo Mio. Probabilistic modal μ-calculus with independent product. *Logical Methods in Computer Science*, 8(4), 2012.

[Pet62] Carl Adam Petri. *Kommunikation mit Automaten.* PhD thesis, Bonn: Institut für Instrumentelle Mathematik, Schriften des IIM Nr. 2, 1962. Second Edition:, New York: Griffiss Air Force Base, Technical Report RADC-TR-65–377, Vol.1, 1966, Pages: Suppl. 1, English translation.

[Put94] Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 1994.

[Rab82] Michael O. Rabin. N-process mutual exclusion with bounded waiting by $4\log_2 N$-valued shared variable. *Computer and System Sciences*, 25(1):66–75, 1982.

[Seg95] Roberto Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems*. PhD thesis, Massachusetts Institute of Technology, 1995.

[SG06] Sharon Shoham and Orna Grumberg. 3-valued abstraction: More precision at less cost. In *Logic in Computer Science*, pages 399–410, 2006.

[SG07] Sharon Shoham and Orna Grumberg. A game-based framework for CTL counterexamples and 3-valued abstraction-refinement. volume 9, 2007.

[Sha53] L. S. Shapley. Stochastic games. *Proceedings of the National Academy of Sciences of the United States of America*, 39(10):1095–1100, 1953.

[SK12] Falak Sher and Joost-Pieter Katoen. Compositional abstraction techniques for probabilistic automata. In *Theoretical Computer Science*, volume 7604 of *LNCS*, pages 325–341, 2012.

[SK14] Falak Sher and Joost-Pieter Katoen. Tight game abstractions of probabilistic automata. In *Concurrency Theory*, volume 8704 of *LNCS*, pages 576–591, 2014.

[SL95] Roberto Segala and Nancy A. Lynch. Probabilistic simulations for probabilistic processes. *Nordic Journal of Computing*, 2(2):250–273, 1995.

[T$^+$55] Alfred Tarski et al. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5(2):285–309, 1955.

[WZ10] Björn Wachter and Lijun Zhang. Best probabilistic transformers. In *Verification, Model Checking, and Abstract Interpretation*, volume 5944 of *LNCS*, pages 362–379, 2010.

[ZP10] Chenyi Zhang and Jun Pang. On probabilistic alternating simulations. In *Theoretical Computer Science*, volume 323, pages 71–85. Springer, 2010.

Aachener Informatik-Berichte

2012-01	Fachgruppe Informatik: Annual Report 2012
2012-02	Thomas Heer: Controlling Development Processes
2012-03	Arne Haber, Jan Oliver Ringert, Bernhard Rumpe: MontiArc - Architectural Modeling of Interactive Distributed and Cyber-Physical Systems
2012-04	Marcus Gelderie: Strategy Machines and their Complexity
2012-05	Thomas Ströder, Fabian Emmes, Jürgen Giesl, Peter Schneider-Kamp, and Carsten Fuhs: Automated Complexity Analysis for Prolog by Term Rewriting
2012-06	Marc Brockschmidt, Richard Musiol, Carsten Otto, Jürgen Giesl: Automated Termination Proofs for Java Programs with Cyclic Data
2012-07	André Egners, Björn Marschollek, and Ulrike Meyer: Hackers in Your Pocket: A Survey of Smartphone Security Across Platforms
2012-08	Hongfei Fu: Computing Game Metrics on Markov Decision Processes
2012-09	Dennis Guck, Tingting Han, Joost-Pieter Katoen, and Martin R. Neuhäußer: Quantitative Timed Analysis of Interactive Markov Chains
2012-10	Uwe Naumann and Johannes Lotz: Algorithmic Differentiation of Numerical Methods: Tangent-Linear and Adjoint Direct Solvers for Systems of Linear Equations
2012-12	Jürgen Giesl, Thomas Ströder, Peter Schneider-Kamp, Fabian Emmes, and Carsten Fuhs: Symbolic Evaluation Graphs and Term Rewriting — A General Methodology for Analyzing Logic Programs
2012-15	Uwe Naumann, Johannes Lotz, Klaus Leppkes, and Markus Towara: Algorithmic Differentiation of Numerical Methods: Tangent-Linear and Adjoint Solvers for Systems of Nonlinear Equations

2012-16 Georg Neugebauer and Ulrike Meyer: SMC-MuSe: A Framework for Secure Multi-Party Computation on MultiSets

2012-17 Viet Yen Nguyen: Trustworthy Spacecraft Design Using Formal Methods

2013-01 * Fachgruppe Informatik: Annual Report 2013

2013-02 Michael Reke: Modellbasierte Entwicklung automobiler Steuerungssysteme in Klein- und mittelständischen Unternehmen

2013-03 Markus Towara and Uwe Naumann: A Discrete Adjoint Model for OpenFOAM

2013-04 Max Sagebaum, Nicolas R. Gauger, Uwe Naumann, Johannes Lotz, and Klaus Leppkes: Algorithmic Differentiation of a Complex C++ Code with Underlying Libraries

2013-05 Andreas Rausch and Marc Sihling: Software & Systems Engineering Essentials 2013

2013-06 Marc Brockschmidt, Byron Cook, and Carsten Fuhs: Better termination proving through cooperation

2013-07 André Stollenwerk: Ein modellbasiertes Sicherheitskonzept für die extrakorporale Lungenunterstützung

2013-08 Sebastian Junges, Ulrich Loup, Florian Corzilius and Erika Ábrahám: On Gröbner Bases in the Context of Satisfiability-Modulo-Theories Solving over the Real Numbers

2013-10 Joost-Pieter Katoen, Thomas Noll, Thomas Santen, Dirk Seifert, and Hao Wu: Performance Analysis of Computing Servers using Stochastic Petri Nets and Markov Automata

2013-12 Marc Brockschmidt, Fabian Emmes, Stephan Falke, Carsten Fuhs, and Jürgen Giesl: Alternating Runtime and Size Complexity Analysis of Integer Programs

2013-13 Michael Eggert, Roger Häußling, Martin Henze, Lars Hermerschmidt, René Hummen, Daniel Kerpen, Antonio Navarro Pérez, Bernhard Rumpe, Dirk Thißen, and Klaus Wehrle: SensorCloud: Towards the Interdisciplinary Development of a Trustworthy Platform for Globally Interconnected Sensors and Actuators

2013-14 Jörg Brauer: Automatic Abstraction for Bit-Vectors using Decision Procedures

2013-16 Carsten Otto: Java Program Analysis by Symbolic Execution

2013-19 Florian Schmidt, David Orlea, and Klaus Wehrle: Support for error tolerance in the Real-Time Transport Protocol

2013-20 Jacob Palczynski: Time-Continuous Behaviour Comparison Based on Abstract Models

2014-01 * Fachgruppe Informatik: Annual Report 2014

2014-02	Daniel Merschen: Integration und Analyse von Artefakten in der modellbasierten Entwicklung eingebetteter Software
2014-03	Uwe Naumann, Klaus Leppkes, and Johannes Lotz: dco/c++ User Guide
2014-04	Namit Chaturvedi: Languages of Infinite Traces and Deterministic Asynchronous Automata
2014-05	Thomas Ströder, Jürgen Giesl, Marc Brockschmidt, Florian Frohn, Carsten Fuhs, Jera Hensel, and Peter Schneider-Kamp: Automated Termination Analysis for Programs with Pointer Arithmetic
2014-06	Esther Horbert, Germán Martín García, Simone Frintrop, and Bastian Leibe: Sequence Level Salient Object Proposals for Generic Object Detection in Video
2014-07	Niloofar Safiran, Johannes Lotz, and Uwe Naumann: Algorithmic Differentiation of Numerical Methods: Second-Order Tangent and Adjoint Solvers for Systems of Parametrized Nonlinear Equations
2014-08	Christina Jansen, Florian Göbe, and Thomas Noll: Generating Inductive Predicates for Symbolic Execution of Pointer-Manipulating Programs
2014-09	Thomas Ströder and Terrance Swift (Editors): Proceedings of the International Joint Workshop on Implementation of Constraint and Logic Programming Systems and Logic-based Methods in Programming Environments 2014
2014-14	Florian Schmidt, Matteo Ceriotti, Niklas Hauser, and Klaus Wehrle: HotBox: Testing Temperature Effects in Sensor Networks
2014-15	Dominique Gückel: Synthesis of State Space Generators for Model Checking Microcontroller Code
2014-16	Hongfei Fu: Verifying Probabilistic Systems: New Algorithms and Complexity Results
2015-01 *	Fachgruppe Informatik: Annual Report 2015
2015-05	Florian Frohn, Jürgen Giesl, Jera Hensel, Cornelius Aschermann, and Thomas Ströder: Inferring Lower Bounds for Runtime Complexity
2015-06	Thomas Ströder and Wolfgang Thomas (Editors): Proceedings of the Young Researchers' Conference "Frontiers of Formal Methods"
2015-07	Hilal Diab: Experimental Validation and Mathematical Analysis of Cooperative Vehicles in a Platoon
2015-09	Xin Chen: Reachability Analysis of Non-Linear Hybrid Systems Using Taylor Models

* These reports are only available as a printed version.
Please contact biblio@informatik.rwth-aachen.de to obtain copies.