More precise partition abstractions

Harald Fecher[1] and Michael Huth[2]

[1] Christian-Albrechts-University Kiel, Germany
hf@informatik.uni-kiel.de
[2] Imperial College London, United Kingdom
M.Huth@doc.imperial.ac.uk

Abstract. We define, for any partition of a state space and for formulas of the modal μ-calculus, two variants of precision for abstractions that have the partition set as state space. These variants are defined via satisfaction parity games in which the Refuter can replace a concrete state with any state in the same partition before, respectively after, a quantifier move. These games are independent of the kind of abstraction. Our first variant makes the abstraction games of de Alfaro et al. model-independent, captures the definition of precision given by Shoham & Grumberg, and corresponds to generalized Kripke modal transition systems. Our second variant is then shown to render more precise abstractions in the form of μ-automata without fairness. We discuss the relative tradeoffs of both variants in terms of the size of abstractions, the perceived cost of their synthesis via theorem provers, and the preservation of equations that are valid over concrete models. Finally, we sketch a combination of both abstraction methods.

1 Introduction

Model checking [4, 20] provides a framework for verifying properties of systems: a system is represented by a mathematical model M, a property of interest is coded within a formal language as some ϕ, and satisfaction is a formally defined predicate $M \models \phi$. Here we consider the prominent approach where properties are described in terms of the modal μ-calculus [18]. The fact that the size of M is often exponential or even infinite in its description, which could be a computer program, often requires alternatives to the direct computation of $M \models \phi$. Predicate abstraction [13] attempts to address this by constructing an abstract model M^α from a partition of the state space of M such that properties of a certain kind that hold in M^α also hold in M.

Dams [7] develops techniques that, for a fixed kind of abstraction, produce an abstract model M^α of M, with an abstraction relation between states of M and states of M^α, that is as precise as possible: no other abstraction of M with the same abstraction relation as M^α will satisfy more properties that hold for M. This links "precision" to "completeness" as used in abstract interpretation [6]. Such notions of precision have then been studied further in the literature, as, e.g., in [5, 8, 21, 23, 14]. De Alfaro et al. [1] define precision for the (alternating-time) μ-calculus for may- and must-transitions via a parity game in which the Refuter can replace in its first move a concrete state with any state in the same partition. In order to obtain a model-independent definition of preciseness (which cannot speak of must- and may-transitions) we transform their games into a satisfaction game, restricting our attention to the modal μ-calculus and to partition-based abstraction. We call the resulting approach *pre-abstraction*.

Main contribution. In this paper, we present a new variant of precision for abstractions over an underlying state space partition. It is similar to the definition of pre-abstraction except that the Refuter can replace a concrete state with any state in the same partition *after* a quantifier move (and not before, as is the case for pre-abstraction). We show that our variant of precision renders more precise abstractions in the form of μ-automata without fairness, whereas generalized Kripke modal transition systems are obtained from pre-abstraction. Both notions are incremental and sound for abstraction-based verification, and so suitable enhancements for algorithms based on counter-example-guided abstraction-refinement (CEGAR). Our discussion shows that both notions have their relative merits, so tools may benefit from having both notions at their disposal.

Further related work. In [15] a kind of model is developed for which precise, finite-state abstractions can be computed by a generalization of predicate abstraction; indeed, pre-abstractions can be obtained in this framework as a special case. Shoham & Grumberg [22] define precision also independent of the abstract models and in terms of properties of algebraic operators, instead of coinduction in terms of games. Their approach coincides with pre-abstraction for partition based abstractions, a fact we show in this paper, but also considers arbitrary abstraction relations, which render abstract models with may-hypertransitions. An alternative to studying the precision of abstract models is the analysis of precision of model checking algorithms: Bruns & Godefroid develop generalized model checking in [3] as an improvement of the compositional model checking semantics in [2], and Jagadeesan & Godefroid apply this in the context of abstraction refinement in [11]. Semantically self-minimizing formulas [12] enjoy the property that the compositional check of [2] yields the same result as the expensive one based on generalized model checking.

Outline. In Section 2 we review Kripke structures, alternating tree automata, and their standard satisfaction game. For an abstraction function mapping concrete to abstract states, “pre-abstraction”, and our new variant “post-abstraction”, are presented in Section 3, shown to be sound approximations of standard satisfaction games, and our new variant is proved to be more precise than pre-abstraction. In Sections 4 and 5, these variant games — which operate on the concrete state space — are proven to have equivalent versions on abstract models: pre-abstraction corresponds to using generalized Kripke modal transition systems, and post-abstraction to using μ-automata without fairness. In Section 6 we sketch how both kinds of abstract models can be (approximatively) synthesized with the help of a theorem prover. The tradeoffs between both abstraction methods are investigated in Section 7 and the incremental nature of both abstraction techniques is established in Section 8. A combination of both abstraction techniques is discussed in Section 9 and conclusions are stated in Section 10. An appendix contains some proof sketches, at the discretion of referees.

2 Preliminaries

Throughout, $\mathbb{P}(S)$ denotes the power set of a set S. Functional composition is denoted by $\circ$. Given a relation $\rho \subseteq B \times C$ with subset $X \subseteq B$ we write $X.\rho$ for $\{c \in C \mid \exists b \in X\colon (b,c) \in \rho\}$. For a sequence of tuples Φ we write $\Phi[i]$ for the sequence obtained from Φ through projection into the i-th coordinate. Let $\mathrm{map}(f, \Phi)$ be the sequence obtained from Φ by applying function f to all elements of Φ in situ.

Kripke structures. Without loss of generality, we won't consider action labels on models in this paper. Thus the concrete models, e.g. discrete dynamical systems or the semantical models of programs, considered here are Kripke structures over a finite set of propositions AP, the building blocks for properties one wants to model check.

Definition 1 (Kripke structure). *A* Kripke structure K *over* AP *is a tuple* (S, R, L) *such that S is a set of states, $R \subseteq S \times S$ its state transition relation, and $L\colon S \to \mathbb{P}(\mathrm{AP})$ its labeling function.*

A Kripke structure is illustrated in Figure 1.

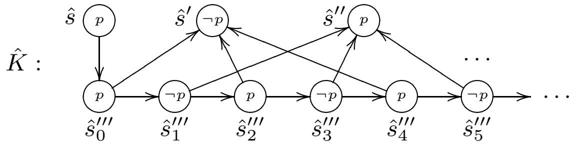


Fig. 1. A Kripke structure. The propositions true (resp. false) at a state are depicted (resp. depicted in negated form) within state borders. Arrows $s \to s'$ denote $(s, s') \in R$. State names are depicted close to the corresponding state

Alternating tree automata. We present the modal μ-calculus in its equivalent form of alternating tree automata [24].

Definition 2 (Tree automata). *An* alternating tree automaton $A = (Q_A, \delta_A, \Theta_A)$ *has*

- *a finite, nonempty set of states* $(q \in)Q_A$
- *a transition relation δ_A mapping automaton states to one of the following forms, where q', q'' are automaton states and p propositions from* AP*:* $p \mid \neg p \mid q' \mid q' \tilde{\wedge} q'' \mid q' \tilde{\vee} q'' \mid \mathsf{EX}\, q' \mid \mathsf{AX} q'$ *and*
- *an acceptance condition $\Theta_A\colon Q_A \to \mathbb{N}$ with finite image, where an infinite sequence of automata states is accepted iff the maximal acceptance number of those that occur infinitely often is even.*

An alternating tree automaton is depicted in Figure 2. For any infinite but bounded sequence $\boldsymbol{n}$ of elements in $\mathbb{N}$ we write $\sup(\boldsymbol{n})$ for the largest m that occurs in $\boldsymbol{n}$ infinitely often. The satisfaction relation of Kripke structures is defined via 2-person

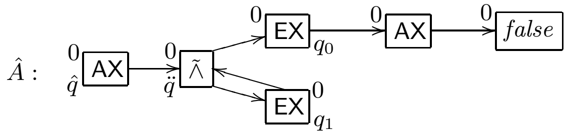


Fig. 2. An alternating tree automaton. Accepting values are depicted next to states. At state $\hat{q}$, it expresses that, after any transition, (i) there is a transition (to q_0) such that no further transition is possible; and (ii) there is a transition (to q_1) such that $\ddot{q}$ holds again

games over configurations. In such games, only one player may move in a given configuration and a strategy for a player is a partial function that, given a finite word of configurations that ends in a configuration of that player, determines the new configuration such that this choice is consistent with the rules of the game.

Definition 3 (Satisfaction game).

- *Finite satisfaction plays for Kripke structure K and alternating tree automaton A have the rules and winning conditions given in Table 1. An infinite play Φ is a* win *for the Verifier iff* $\sup(\mathrm{map}(\Theta, \Phi[2]))$ *is even; otherwise it is won by the Refuter.*
- *Kripke structure K* satisfies *automaton A in $(s, q) \in S \times Q$, written $(K, s) \models (A, q)$, iff the Verifier has a strategy for the corresponding satisfaction game between K and A with which he wins all satisfaction plays started at (s, q).*

p: the Verifier wins if $p \in L(s)$; the Refuter wins if $p \notin L(s)$
$\neg p$: the Verifier wins if $p \notin L(s)$; the Refuter wins if $p \in L(s)$
q': the next configuration is (s, q')
$q_1 \tilde{\wedge} q_2$: the Refuter picks a q' from $\{q_1, q_2\}$; the next configuration is (s, q')
$q_1 \tilde{\vee} q_2$: the Verifier picks a q' from $\{q_1, q_2\}$; the next configuration is (s, q')
$\mathsf{EX}\, q'$: the Verifier picks $s' \in \{s\}.R$; the next configuration is (s', q')
$\mathsf{AX}\, q'$: the Refuter picks $s' \in \{s\}.R$; the next configuration is (s', q').

Table 1. Rules for satisfaction game at configuration $(s, q) \in S \times Q$, specified through a case analysis on the value of $\delta(q)$. Satisfaction plays are sequences of configurations generated thus

Example 1. For the Kripke structure $\hat{K}$ in Figure 1 and the alternating tree automaton $\hat{A}$ in Figure 2 we have $(\hat{K}, \hat{s}) \models (\hat{A}, \hat{q})$: at the EX-state q_0 the Verifier chooses the transition pointing to $\hat{s}'$ or $\hat{s}''$, respectively; at the EX-state q_1 the Verifier chooses the transition along the lower line of states in $\hat{K}$.

3 Partition-induced satisfaction games

We introduce two variants of the satisfaction game between a Kripke structure K with state set S and an alternating tree automaton A: a pre-game and a post-game, referring to the pre- and post-image of the state transition relation of the Kripke structure, respectively. These games are derived from a surjective abstraction function $\alpha\colon S \rightarrow I$ that maps concrete states $s \in S$ into designated abstract ones $\alpha(s) \in I$. States s and s' are compatible iff $\alpha(s) = \alpha(s')$.

Configurations of the pre-game are pairs $(i, q) \in I \times Q_A$, but the Verifier will always be forced to move with respect to a concrete configuration (s, q) with $\alpha(s) = i$ where s is chosen by the Refuter, and these moves for the Verifier are as for the ordinary satisfaction game. Configurations of the post-game are pairs $(s, q) \in S \times Q_A$. The difference between the pre- and the post-game resides in the capabilities of the Refuter. In the pre-game he can switch between compatible states *before* a quantifier or literal move by either player is being made; in the post-game he may switch to a compatible state *after* a quantifier move by either player has been made. We formalize this:

Definition 4 (Pre-games and post-games). *Let $K = (S, R, L)$ be a Kripke structure and $\alpha\colon S \rightarrow I$ a surjective abstraction function.*

- *Pre-games:*
  - *Finite pre-satisfaction plays for K, α, and alternating tree automaton A have the rules and winning conditions given in Table 2. An infinite play Φ is* won *by the Verifier iff* $\sup(\mathrm{map}(\Theta, \Phi[2]))$ *is even; otherwise it is won by the Refuter.*
  - *Model K* pre-satisfies *automaton A for α in $(s, q) \in S \times Q$, written $(K, s) \models^{\alpha}_{-} (A, q)$, iff the Verifier has a strategy for the corresponding pre-satisfaction game between K and A such that the Verifier wins all pre-satisfaction plays started at $(\alpha(s), q)$ with that strategy.*
- *Post-games:*
  - *Finite post-satisfaction plays for K, α, and alternating tree automaton A have the rules and winning conditions given in Table 3. An infinite play Φ is* won *by the Verifier iff* $\sup(\mathrm{map}(\Theta, \Phi[2]))$ *is even; otherwise it is won by the Refuter.*
  - *Model K* post-satisfies *automaton A for α in $(s, q) \in S \times Q$, written $(K, s) \models^{\alpha}_{+} (A, q)$, iff the Verifier has a strategy for the corresponding post-satisfaction game between K and A such that the Verifier wins all post-satisfaction plays started at (s, q) with that strategy.*

Example 2. Consider the Kripke structure $\hat{K}$ from Figure 1 and the alternating tree automaton $\hat{A}$ from Figure 2. Let $\hat{\alpha}$ be the function that maps $\hat{s}$ to $\hat{\imath}$, $\hat{s}'$ to $\hat{\imath}'$, $\hat{s}''$ to $\hat{\imath}''$, and $\hat{s}'''_n$ to $\hat{\imath}'''$ for $n \in \mathbb{N}$. Then $(\hat{K}, \hat{s}) \models^{\hat{\alpha}}_{-} (\hat{A}, \hat{q})$: at the EX-state q_0 the Verifier chooses $\hat{s}'$ or $\hat{s}''$, depending on which one is possible; at the EX-state q_1 the Verifier chooses an element from $\{\hat{s}'''_n \mid n \in \mathbb{N}\}$, depending on which one is possible. Furthermore, $(\hat{K}, \hat{s}) \models^{\hat{\alpha}}_{+} (\hat{A}, \hat{q})$ holds, reasoned similarly as before.

Both pre-satisfaction and post-satisfaction are sound, their satisfaction instances imply satisfaction instances of the underlying Kripke structure. Moreover, post-satisfaction is more precise than pre-satisfaction. Formally:

Theorem 1 (Soundness of abstract satisfaction games). *Let $K = (S, R, L)$ be a Kripke structure and $\alpha\colon S \rightarrow I$ a surjective function. Then pre-satisfaction implies*

p: the Refuter picks $s \in S$ with $\alpha(s) = i$; the Verifier wins if $p \in L(s)$; the Refuter wins if $p \notin L(s)$

$\neg p$: the Refuter picks $s \in S$ with $\alpha(s) = i$; the Verifier wins if $p \notin L(s)$; the Refuter wins if $p \in L(s)$

q': the next configuration is (i, q')

$q_1 \tilde{\wedge} q_2$: the Refuter picks a q' from $\{q_1, q_2\}$; the next configuration is (i, q')

$q_1 \tilde{\vee} q_2$: the Verifier picks a q' from $\{q_1, q_2\}$; the next configuration is (i, q')

$\mathsf{EX}\, q'$: the Refuter picks $s \in S$ with $\alpha(s) = i$; the Verifier picks $s' \in \{s\}.R$; the next configuration is $(\alpha(s'), q')$

$\mathsf{AX}\, q'$: the Refuter picks $s \in S$ with $\alpha(s) = i$ and then picks $s' \in \{s\}.R$; the next configuration is $(\alpha(s'), q')$.

Table 2. Rules for pre-satisfaction game for $\alpha\colon S \to I$ at configuration $(i, q) \in I \times Q$, based on a case analysis of $\delta(q)$. Pre-satisfaction plays are sequences of configurations generated thus

$\mathsf{EX}\, q'$: the Verifier picks $s' \in \{s\}.R$; the Refuter picks $s'' \in S$ with $\alpha(s') = \alpha(s'')$; the next configuration is (s'', q')

$\mathsf{AX}\, q'$: the Refuter picks $s' \in \{s\}.R$ and then some $s'' \in S$ with $\alpha(s') = \alpha(s'')$; the next configuration is (s', q').

Table 3. Rules for post-satisfaction game at configuration $(s, q) \in S \times Q$, specified through a case analysis of $\delta(q)$. Omitted rules are as for the standard satisfaction game in Table 1. Post-satisfaction plays are sequences of configurations generated thus

post-satisfaction, which in turn implies satisfaction: for all states s and all rooted alternating tree automata (A, q):

$$(K, s) \models^{\alpha}_{-} (A, q) \Rightarrow (K, s) \models^{\alpha}_{+} (A, q)$$
$$(K, s) \models^{\alpha}_{+} (A, q) \Rightarrow (K, s) \models (A, q)\,.$$

Moreover, the above implications are strict in general.

Example 3. The strictness of the inclusions in Theorem 1 can be seen as follows: For the Kripke structure $\hat{K}$ from Figure 1 and $\hat{\alpha}$ from Example 2 we have $(\hat{K}, \hat{s}) \models \mathsf{EX}\, p$, but $(\hat{K}, \hat{s}) \not\models^{\hat{\alpha}}_{+} \mathsf{EX}\, p$; and $(\hat{K}, \hat{s}) \models^{\hat{\alpha}}_{+} \mathsf{EX}\,(p \,\tilde{\vee}\, \mathsf{EX}\, p)$, but $(\hat{K}, \hat{s}) \not\models^{\hat{\alpha}}_{-} \mathsf{EX}\,(p \,\tilde{\vee}\, \mathsf{EX}\, p)$.

The notions of pre- and post-satisfaction, and their soundness with respect to satisfaction, make them suitable for abstraction-based model checking. Their operational make-up appeals to information that resides only in the concrete model to begin with. Therefore a method for abstracting concrete models, such that the satisfaction game over abstract models precisely captures the pre- (resp. post-)game for the concrete model, is needed. We carry out this programme in the next two sections and show, as perhaps expected, that the generalized Kripke modal transition systems of [22] and their satisfaction game capture pre-satisfaction. For post-satisfaction, we suggest to work with μ-automata [16, 9] without fairness and show a precise correspondence as well.

4 Precise abstractions for pre-satisfaction

Our models for abstraction of pre-satisfaction are taken from [22], and are a variant of disjunctive modal transition systems [19].

Definition 5 (Generalized Kripke modal transition systems). *A* generalized Kripke modal transition system M *over* AP *is a tuple* (S, R^-, R^+, L) *with* S *as set of states,* $R^- \subseteq S \times \mathbb{P}(S)$ *as set of* must-*transitions,*[1] $R^+ \subseteq S \times S$ *as set of* may-*transitions, and* $L\colon\, S \to \mathbb{P}(\mathrm{AP} \cup \{\neg p \mid p \in \mathrm{AP}\})$ *as labeling function. To highlight that a must-transition* (s, D) *has a non-singleton target set* D *we speak of must-*hypertransitions. *This generalized Kripke modal transition system* M *is* finite *if* S *is finite.*

A generalized Kripke modal transition system is illustrated in Figure 3. We now define satisfaction over generalized Kripke modal transition systems:

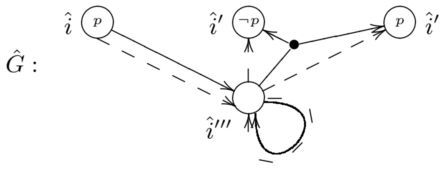


Fig. 3. A generalized Kripke modal transition system. Propositional labelings and state names are used as in Figure 1. Dashed arrows model may-transitions and solid arrows model must-transitions. The self-loop at state $\hat{\imath}'''$ is a must-transition with $\{\hat{\imath}'''\}$ as target. The must-hypertransition from $\hat{\imath}'''$ has set $\{\hat{\imath}', \hat{\imath}''\}$ as target

Definition 6 (Satisfaction for generalized Kripke modal transition systems).

- *Finite satisfaction plays for a generalized Kripke modal transition system* G *and an alternating tree automaton* A *have the rules and winning conditions as stated in Table 4. An infinite play* Φ *is a* win *for the Verifier iff* $\sup(\mathrm{map}(\Theta, \Phi[2]))$ *is even; otherwise it is won by the Refuter.*
- *The generalized Kripke modal transition system* G satisfies *the alternating tree automaton* A *in configuration* $(s, q) \in S \times Q$*, written* $(G, s) \models (A, q)$*, iff the Verifier has a strategy for the corresponding satisfaction game between* G *and* A *such that the Verifier wins all satisfaction plays started at* (s, q) *with that strategy.*

The satisfaction game in Table 4 amounts to playing a parity game, so the decidability of such satisfaction instances is in UP $\cap$ coUP [17].

Example 4. For the generalized Kripke modal transition system $\hat{G}$ from Figure 3 and the alternating tree automaton $\hat{A}$ from Figure 2 we have $(\hat{G}, \hat{\imath}) \models (\hat{A}, \hat{q})$: at the EX-state

[1] We adhere to the convention of using $-$ for must-transitions and $+$ for may-transitions and note that the paper [22] uses $+$ for must-transitions and $-$ for may-transitions.

$\neg p$: the Verifier wins if $\neg p \in L(s)$; the Refuter wins if $\neg p \notin L(S)$
$\mathsf{EX}\, q'$: the Verifier picks $D' \in \{s\}.R^-$; the Refuter picks $s' \in D'$; the next configuration is (s', q')
$\mathsf{AX}\, q'$: the Refuter picks $s' \in \{s\}.R^+$; the next configuration is (s', q')

Table 4. Rules for satisfaction game between a generalized Kripke modal transition system and an alternating tree automaton at configuration $(s, q) \in S \times Q$, based on a case analysis of $\delta(q)$. Omitted rules are as in Table 1. Satisfaction plays are sequences of configurations generated thus

q_0 the Verifier chooses the must-transition $(\hat{i}''', \{\hat{i}', \hat{i}''\})$; at the EX-state q_1 the Verifier chooses the self-loop $(\hat{i}''', \{\hat{i}'''\})$. Note that $(\hat{G}, \hat{i})$ neither satisfies $\mathsf{EX}\,(p \tilde{\vee} \mathsf{EX}\, p)$ nor its 'negation' $\mathsf{AX}\,(\neg p \tilde{\wedge} \mathsf{AX}\, \neg p)$.

Every Kripke structure K has a natural representation as a generalized Kripke modal transition system $G[K]$ where $L_{G[K]}(s) = L_K(s) \cup \{\neg p \mid p \notin L_K(s)\}$, $R^+_{G[K]} = R$, and $R^-_{G[K]}$ is R embedded from $S \times S$ into $S \times \mathbb{P}(S)$. Since $(K, s) \models (A, q)$ iff $(G[K], s) \models (A, q)$ the overloading of "satisfaction" and its symbol $\models$ are justified.

Given a surjective function $\alpha \colon S \to I$ and a Kripke structure $K = (S, R, L)$ we follow [22] and let I be the abstract state set, have a may-transition between two abstract states iff there is a transition in the Kripke structure such that the source and target are abstracted to the corresponding abstract states, and have a must-transition from i to I' iff every Kripke state s abstracted to i has a transition to an element that is abstracted to an element from I'. Abstract labelings have a similar must-interpretation. Formally:

Definition 7 (Pre-abstractions). *Let $K = (S, R, L)$ be a Kripke structure and $\alpha \colon S \to I$ a surjective function. Then the pre-abstraction of K for α is the generalized Kripke modal transition system $G^K_\alpha = (I, R^-_{K,\alpha}, R^+_{K,\alpha}, L_{K,\alpha})$ where*

$$\begin{aligned} R^-_{K,\alpha} &= \{(i, D) \mid \forall s \in S\colon [\alpha(s) = i \Rightarrow \exists s' \in \{s\}.R : \alpha(s') \in D]\} \\ R^+_{K,\alpha} &= \{(i, i') \mid \exists (s, s') \in R\colon [\alpha(s) = i \;\&\; \alpha(s') = i']\} \\ L_{K,\alpha}(i) &= \{p \mid p \in \mathrm{AP} \;\&\; \forall s \in S\colon [\alpha(s) = i \Rightarrow p \in L(s)]\} \cup \\ &\qquad \{\neg p \mid p \in \mathrm{AP} \;\&\; \forall s \in S\colon [\alpha(s) = i \Rightarrow p \notin L(s)]\} \end{aligned}$$

The pre-abstraction is finite whenever I is finite.

Example 5. The pre-abstraction of $\hat{K}$ from Figure 1 for $\hat{\alpha}$ from Example 3 is the generalized Kripke modal transition system from Figure 3, where must-transitions (s, D) are omitted if they have a must-transition (s, D') with $D' \subseteq D$ — those omissions won't impact the satisfaction relation and can speed up the synthesis of abstractions.

Possible occurrences of must-hypertransitions make the complexity of the abstraction exponential in I in the worst case. Pre-abstraction is precise and sound for pre-satisfaction, as already illustrated via Examples 2, 3, 4 and 5:

Theorem 2 (Correspondence of pre-game and pre-abstraction). *Let $K = (S, R, L)$ be a Kripke structure, A an alternating tree automaton, $q \in Q_A$, $s \in S$, and $\alpha \colon S \to I$*

a surjective function. Then

$$(K, s) \models^{\alpha}_{-} (A, q) \iff (G^K_{\alpha}, \alpha(s)) \models (A, q)$$

5 Precise abstractions for post-satisfaction

The models we propose for post-abstractions are μ-automata [16] without fairness. We use the notation of [9]. All μ-automata in this paper are without fairness.

Definition 8 (μ-automata). *A* μ-automaton M *over* AP *is a tuple* $(O, B, \rightrightarrows, \rightarrow, L)$ *such that* $(o \in)O$ *is the set of OR-states,* $(b \in)B$ *the set of BRANCH-states (disjoint from O),* $\rightrightarrows\ \subseteq O \times B$ *the OR-transition relation,* $\rightarrow\ \subseteq B \times O$ *the BRANCH-transition relation, and* $L\colon B \rightarrow \mathbb{P}(\mathrm{AP})$ *the labeling function.* M *is* finite *if both O and B are.*

A μ-automaton is given in Figure 4. We define satisfaction over μ-automata next:

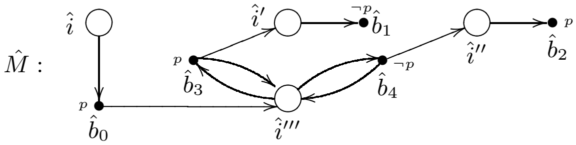


Fig. 4. A μ-automaton. OR-states are depicted as unfilled circles and BRANCH-states as filled circles. Literals (shown in smaller font) that are true at a BRANCH-state are depicted next to it. The name of a state is also shown close to it

Definition 9 (Satisfaction for μ-automata).

- *Finite satisfaction plays for a μ-automaton M and alternating tree automaton A have the rules and winning conditions stated in Table 5. An infinite play Φ is a* win *for the Verifier iff* $\sup(\mathrm{map}(\Theta, \Phi[2]))$ *is even; otherwise it is won by the Refuter.*
- *The μ-automaton M* satisfies *the alternating tree automaton A in $(\beta, q) \in (O \cup B) \times Q$, written $(M, \beta) \models (A, q)$, iff the Verifier has a strategy for the corresponding satisfaction game between M and A such that the Verifier wins all satisfaction plays started at (β, q) with her strategy.*

Similar to the satisfaction game for generalized Kripke modal transition systems, the satisfaction game for μ-automata corresponds to a parity game. So deciding such satisfaction instances is in UP $\cap$ coUP.

Example 6. For the μ-automaton $\hat{M}$ in Figure 4 and the alternating tree automaton $\hat{A}$ in Figure 2 we have $(\hat{M}, \hat{i}) \models (\hat{A}, \hat{q})$: at $(\hat{b}_3, q_1)$ and at $(\hat{b}_4, q_1)$ the Verifier chooses the transition to $\hat{i}'''$; at $(\hat{b}_3, q_0)$ and at $(\hat{b}_4, q_0)$ she chooses the transition to $\hat{i}'$, resp. $\hat{i}''$. Furthermore, it is easily seen that $\hat{G}$ satisfies neither $\mathsf{EX}\, p$ nor its ‘negation’ $\mathsf{AX}\, \neg p$ at $\hat{q}$.

$\beta \in O$: the Refuter picks $b \in \{\beta\}.\rightrightarrows$; the next configuration is (b, q)
$\beta \in B$ and $q = p$: the Verifier wins if $p \in L(\beta)$; the Refuter wins if $p \notin L(\beta)$
$\beta \in B$ and $q = \neg p$: the Verifier wins if $p \notin L(\beta)$; the Refuter wins if $p \in L(\beta)$
$\beta \in B$ and $q = q'$: the next configuration is (β, q')
$\beta \in B$ and $q = q_1 \tilde{\wedge} q_2$: the Refuter picks a q' from $\{q_1, q_2\}$; the next configuration is (β, q')
$\beta \in B$ and $q = q_1 \tilde{\vee} q_2$: the Verifier picks a q' from $\{q_1, q_2\}$; the next configuration is (β, q')
$\beta \in B$ and $q = \mathsf{EX}\, q'$: the Verifier picks $o \in \{\beta\}.\rightarrow$; the next configuration is (o, q')
$\beta \in B$ and $q = \mathsf{AX}\, q'$: the Refuter picks $o \in \{\beta\}.\rightarrow$; the next configuration is (o, q').

Table 5. Rules for satisfaction game between a μ-automaton M and an alternating tree automaton A at configuration $(\beta, q) \in (O \cup B) \times Q$, based on the given case analysis. Satisfaction plays are sequences of configurations generated thus

Given a surjective function $\alpha \colon S \to I$ and a Kripke structure $K = (S, R, L)$ we now show that μ-automata yield precise abstractions for post-satisfaction. We consider two states of K to be post-equivalent iff they satisfies the same propositions and if the same elements of the partition induced by α are reachable by their one-step transitions. The equivalence classes obtained by the post-equivalences, called post-equivalent classes, are encoded as elements from $\mathbb{P}(I \cup \mathrm{AP})$, where (i) a proposition is valid at $b \in \mathbb{P}(I \cup \mathrm{AP})$ iff $p \in b$; and (ii) exactly those elements of the partition induced by α that are contained in b are reachable. The expression $\mathsf{BR}^K_\alpha(s)$, formally defined below, determines the post-equivalence class of s. Post-equivalence classes become BRANCH-states. The ability of the Refuter to switch to an element compatible with the target of a transition is modeled in the abstraction by OR-states, elements of I. The OR-state $i \in I$ has a transition to a post-equivalence class E iff a concrete state s exists that is abstracted to i by α and yields the post-equivalence class E. Formally:

Definition 10 (Post-abstractions). *For Kripke structure $K = (S, R, L)$ and surjective function $\alpha \colon S \to I$, its* post-abstraction *is the μ-automaton M^K_α over $\mathbb{P}(I \cup \mathrm{AP})$ where*

$$\begin{aligned} O^K_\alpha &= I \qquad\qquad B^K_\alpha = \mathbb{P}(I \cup \mathrm{AP}) \\ \rightrightarrows^K_\alpha &= \{(i, \mathsf{BR}^K_\alpha(s)) \mid \alpha(s) = i\} \qquad \textit{with } \mathsf{BR}^K_\alpha(s) = L(s) \cup \alpha(\{s\}.R) \\ \rightarrow^K_\alpha &= \{(b, i) \mid i \in b\} \\ L^K_\alpha(b) &= b \cap \mathrm{AP} \end{aligned}$$

The post-abstraction is finite whenever I is finite.

Example 7. The five post-equivalence classes of $\hat{K}$ from Figure 1 for $\hat{\alpha}$ from Example 3 are $\{\hat{s}\}$, $\{\hat{s}'\}$, $\{\hat{s}''\}$, $\{\hat{s}'''_{2n} \mid n \in \mathbb{N}\}$, and $\{\hat{s}'''_{2n+1} \mid n \in \mathbb{N}\}$, having the representatives (via function BR^K_α) $\hat{b}_0 = \{p, \hat{\imath}'''\}$, $\hat{b}_1 = \{\}$, $\hat{b}_2 = \{p\}$, $\hat{b}_3 = \{p, \hat{\imath}', \hat{\imath}'''\}$, and $\hat{b}_4 = \{\hat{\imath}'', \hat{\imath}'''\}$. The post-abstraction of $\hat{K}$ from Figure 1 for $\hat{\alpha}$ from Example 3 is the μ-automaton from Figure 4, where non-reachable BRANCH-states from $\hat{\imath}$ are omitted.

The size of this abstraction can be exponential in I and in AP. The post-abstraction is precise and sound for post-satisfaction, illustrated via Examples 2, 3, 6 and 7:

Theorem 3 (Correspondence of post-game and post-abstraction). *Let $K = (S, R, L)$ be a Kripke structure, A an alternating tree automaton, $q \in Q_A$, $s \in S$, and $\alpha\colon S \to I$ a surjective function. Then*

$$(K, s) \models_{+}^{\alpha} (A, q) \iff (M_{\alpha}^{K}, \mathsf{BR}_{\alpha}^{K}(s)) \models (A, q)$$

6 Automated synthesis of precise abstractions

We discuss how the pre- and post-abstraction of a program can be automatically synthesized with the use of theorem provers for a Kripke structure $K = (S, R, L)$ and surjective abstraction function $\alpha\colon S \to I$, along the lines of [13, 10]. Suppose $\mathcal{L}$ is a logic that contains at least all operators of propositional logic and all $p \in \mathrm{AP}$ as predicates with a closed interpretation $[\![p]\!] \subseteq S$ such that:

(i) The interpretation of atomic propositions matches that implicit in the labeling function; for all $p \in \mathrm{AP}$ we have $[\![p]\!] = \{s \in S \mid p \in L(s)\}$.
(ii) Satisfiability and validity of $\mathcal{L}$ are decidable.
(iii) The logic $\mathcal{L}$ is effectively closed under exact successor and predecessor operations in Kripke structures; that is, for every formula $\psi \in \mathcal{L}$ and every $R \subseteq S \times S$ we can compute $\mathrm{pre}(\psi), \mathrm{post}(\psi) \in \mathcal{L}$ such that

$$[\![\mathrm{pre}(\psi)]\!] = \{s' \in S \mid \exists s \in [\![\psi]\!] : (s', s) \in R\}$$
$$[\![\mathrm{post}(\psi)]\!] = \{s' \in S \mid \exists s \in [\![\psi]\!] : (s, s') \in R\}\,.$$

(iv) The surjective abstraction function $\alpha\colon S \to I$ is representable by a set of predicates $\{\psi_i \in \mathcal{L} \mid i \in I\}$ such that, for all $i \in I$, we have $[\![\psi_i]\!] = \{s \in S \mid \alpha(s) = i\}$. (This is, for example, the case in predicate abstraction.)

The first three conditions may be relaxed, as familiar in the judicious over- and under-approximation of precise abstract models for undecidable logics.

Pre-abstraction. A may-transition from i to i' exists iff $\psi_i \wedge \mathrm{pre}(\psi_{i'})$ is satisfiable. If satisfiability is undecidable, we ensure soundness but may lose precision by adding such a may-transition whenever the satisfiability of $\psi_i \wedge \mathrm{pre}(\psi_{i'})$ is unknown. A must-transition from i to D exists iff $\psi_i \Rightarrow (\bigvee_{i' \in D} \mathrm{pre}(\psi_{i'}))$ is valid. In case of the undecidability of validity, we add a must-transition only if the validity of $\psi_i \Rightarrow (\bigvee_{i' \in D} \mathrm{pre}(\psi_{i'}))$ is known. A predicate literal l (either some p or some $\neg p$) is in $L(i)$ iff $\psi_i \Rightarrow l$ is valid. As in the treatment of must-transitions, l is ruled to be a member of $L(i)$ only if the validity of $\psi_i \Rightarrow l$ can be established.

Post-abstraction. The transitions from BRANCH-states and the labeling function rely on an implementation of set membership, as specified in Definition 10. A transition from the OR-state $i \in I$ to the BRANCH-state $b \in \mathbb{P}(I \cup \mathrm{AP})$ exists iff

$$\psi_i \wedge \bigwedge_{i' \in b \cap I} \mathrm{pre}(\psi_{i'}) \wedge \bigwedge_{i' \in I \setminus b} \neg\mathrm{pre}(\psi_{i'}) \wedge \bigwedge_{p \in b \cap \mathrm{AP}} p \wedge \bigwedge_{p \in \mathrm{AP} \setminus b} \neg p \qquad (1)$$

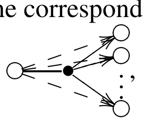
is satisfiable. If satisfiability is undecidable, we over-approximate in the same manner as described above.

7 Expressiveness of pre- and post-abstractions

In order to obtain a fair comparison between pre- and post-abstraction, we take into account the removal of hypertransitions whose targets are supersets of other hypertransition targets, as well as the sharing of identical hypertransition targets for different hypertransition sources through, what we refer to below as, *"division points"*.

Pre-abstraction can be less complex than post-abstraction:

Example 8. Consider the Kripke structure with state set $(\mathbb{P}(X) \setminus \{\{\}\}) \cup X$ where $X = \{x_1, ..x_n\}$, transition relation $\{(X', x) \mid X' \in \mathbb{P}(X) \setminus \{\{\}\} \; \& \; x \in X'\}$, and arbitrary labeling function. The partition, which determines α, is given by $\{\mathbb{P}(X) \setminus \{\{\}\}, \{x_1\}, ..., \{x_n\}\}$. Then the post-abstraction yields the μ-automaton with the same structure as the considered Kripke structure, except that an additional OR-state per partition element is being used. In particular, that μ-automaton has at least 2^n BRANCH-states (depending on the labeling function, up to $n-1$ further BRANCH-states can exist), $n+1$ OR-states (of which n are trivial, i.e., have exactly one outgoing transition), at least 2^n transitions leading to OR-states, and $n2^{n-1}$ transitions leading to BRANCH-states.

On the other hand, the corresponding pre-abstraction yields the generalized Kripke modal transition system , which has $n+1$ states, one division point, and $n+1$ transitions to and from that division point, and n may-transitions.

Post-abstractions, in turn, can be less complex than pre-abstractions:

Example 9. Consider the Kripke structure with state set $X \cup X'$ with $X = \{x_1, \ldots, x_n\}$ and $X' = \{x'_1, \ldots, x'_n\}$, transition relation $\{(x_i, X' \setminus \{x'_i\}) \mid i \in \{1, ..., n\}\}$, and arbitrary labeling function. The partition, determining α, is given by $\{X, \{x'_1\}, ..., \{x'_n\}\}$. Then the post-abstraction yields the μ-automaton with the same structure as the considered Kripke structure, except that an additional OR-state per partition element is being used. In particular, that μ-automaton has $2n$ BRANCH-states, $n+1$ OR-states (where n of them are trivial in the sense aforementioned), and n^2+n transitions (of which n result from trivial OR-states).

On the other hand, the corresponding pre-abstraction yields a generalized Kripke modal transition system with $n+1$ states, $(n^2-n)/2$ division points (all must-hypertransitions having a target set consisting of two elements exist), $(3n^2-3n)/2$ transitions to and from division points, and n may-transitions.

Examples 8 and 9 illustrate that either pre- or post-abstractions can yield smaller abstractions. Taking the size of such abstractions and cost issues of their synthesis aside, both notions can verify the same properties of the concrete models they abstract.

Theorem 4 (Equal expressiveness). *Let $K = (S, R, L)$ be a Kripke structure, $\alpha\colon S \to I$ a surjective function, and (A, q) a rooted alternating tree automaton with $(K, s) \models^{\alpha}_{+} (A, q)$. Then the pre-abstraction of K with respect to α' satisfies (A, q) where $\alpha'\colon S \to \{\mathsf{BR}^K_\alpha(s) \mid s \in S\}$ is defined by $\alpha'(s) = \mathsf{BR}^K_\alpha(s)$.*

Note that the converse of the expressiveness stated in Theorem 4 follows from Theorem 1. Theorem 4 may suggest that post-abstraction is a redundant notion. But in terms of efficiency, pre-abstractions with respect to the derived abstraction α' are more complex than post-abstractions with respect to the original abstraction α, since the calculation of must-hypertransitions leads to an additional exponential blow up (the number of configurations for the model-checking game is exponentially larger). Must-hypertransitions are essential: Theorem 4 won't hold if only must-transitions that point to singletons are allowed.

There is a curious difference between generalized Kripke modal transition systems and μ-automata with respect to their "equational theories". We write $\phi = \psi$ for formulas ϕ and ψ of the modal μ-calculus if, for a given notion of model and satisfaction game, all states in all models have the same satisfaction instances for A_ϕ and for A_ψ — the alternating tree automata encoding the respective formulas.[2] For example, we have $p\tilde{\vee}\neg p = \mathit{true}$ and $\mathsf{EX}\, p\tilde{\vee}\mathsf{AX}\,\neg p = \mathit{true}$ for all μ-automata and $p \in \mathrm{AP}$, but these equations won't hold in generalized Kripke modal transition systems: neither $p\tilde{\vee}\neg p$ nor $\mathsf{EX}\, p\tilde{\vee}\mathsf{AX}\,\neg p$ holds in state $\hat{\imath}'''$ of the generalized Kripke modal transition system from Figure 3. There are equations familiar from basic modal logic that hold neither for μ-automata nor for generalized Kripke modal transition systems, e.g. $\mathsf{EX}\, q_1\tilde{\vee}\mathsf{EX}\, q_2 = \mathsf{EX}\,(q_1\tilde{\vee}q_2)$ which is valid over Kripke structures.

8 Abstraction refinement

The precise abstractions proposed in this paper are suitable for counter-example-guided abstraction-refinement since already checked properties can be reused. For both pre- and post-satisfaction, all previously valid satisfaction instances remain to be valid if abstract states are refined by further splitting:

Theorem 5 (Incremental analysis). *Let $K = (S, R, L)$ be a Kripke structure and both $\alpha_1 \colon S \to I_1$, $f_2 \colon I_1 \to I_2$ surjective functions. Then for all s and all (A, q) we have*

$$(K, s) \models_{-}^{f_2\circ\alpha_1} (A, q) \Rightarrow (K, s) \models_{-}^{\alpha_1} (A, q)$$
$$(K, s) \models_{+}^{f_2\circ\alpha_1} (A, q) \Rightarrow (K, s) \models_{+}^{\alpha_1} (A, q)\,.$$

The above theorem also guarantees confluence of abstractions. Finite-state abstractions, if they exist, can always be found in principle, regardless of the particular history of incremental refinements of an initially chosen abstraction:

Theorem 6 (Confluence). *Let $K = (S, R, L)$ be a Kripke structure and both $\alpha_1 \colon S \to I_1$, $\alpha_2 \colon S \to I_2$ surjective functions. Then there exist surjective functions $\alpha_3 \colon S \to I_3$ and $f : I_3 \to I_2$ such that $\alpha_2 = f \circ \alpha_3$ and, for all s and (A, q), we have*

$$(K, s) \models_{-}^{\alpha_1} (A, q) \Rightarrow (K, s) \models_{-}^{\alpha_3} (A, q)$$
$$(K, s) \models_{+}^{\alpha_1} (A, q) \Rightarrow (K, s) \models_{+}^{\alpha_3} (A, q)\,.$$

[2] We did not define true or A_{true} but they hold in all states of all (kinds of) models.

9 Discussion

The calculation of pre-abstractions may well be more efficient than the calculation of post-abstractions, even if the resulting state space is larger, as it is, e.g., the case in Example 9. The reason being that more complex formulas have to be checked by a theorem prover in the synthesis of the post-abstraction, as seen in Section 6. Nevertheless, whether pre- or post-abstractions work better heavily depends on the Kripke structure and the alternating tree automata one wishes to check. We now sketch how a combination of pre- and post-abstraction may result in a more precise abstraction without increasing the cost of the abstraction synthesis too much. First an approximation of the pre-abstraction avoids the expensive calculation of must-hypertransitions. Then, *before* the abstraction function is being refined, a post-abstraction on the just computed approximation of a pre-abstraction is calculated, where the information encoded in said approximative pre-abstraction is being reused. More precisely, the post-abstraction is calculated, locally, at those abstract states where the currently existing must-transition information is insufficient to verify or falsify the property. This needs only to consider those subsets of $\mathbb{P}(I)$ that are supersets of the local must-transition targets and subsets of the local may-transition targets, speeding up the synthesis of the abstraction.

In order to obtain precise models for this abstraction process, a new kind of model has to be developed: μ-automata in which there are also may-transitions and must-transitions between OR-states. In order to reduce the post-abstraction calculation with respect to predicates in a similar way as described above, there should be a predicate labeling function, as in generalized Kripke modal transition systems, over OR-states. Note that this kind of model is really different from that of the modal automata in [9], where may-transitions are allowed from BRANCH-states to OR-states only, and the labeling function is over BRANCH-states and not over OR-states.

10 Conclusions

Using parity games and avoiding any appeal to particular kinds of models, we presented two notions of precision for partition-based abstractions. We proved that our new notion of post-abstraction is generally more precise than the already established one based on pre-abstraction, and corresponds to the use of μ-automata as abstractions. For functional abstractions, pre-abstraction is shown to be an adaptation of the abstraction games of de Alfaro et al., and to coincide with the algebraic notion of precision given recently by Shoham & Grumberg. The relative tradeoffs of these two notions have been investigated along a number of dimensions: model size, cost of synthesis, and equational theories of abstractions. A combination of both approaches has been discussed informally as subject for future work.

Acknowledgments. This work is in part financially supported by the DFG project *Refism* (FE 942/1-1) and by UK Engineering and Physical Sciences Research Council (EP/D50595X/1).

References

1. L. de Alfaro, P. Godefroid, and R. Jagadeesan. Three-valued abstractions of games: Uncertainty, but with precision. In *Proc. of LICS'04*, pages 170–179. IEEE Computer Society Press, 2004.
2. G. Bruns and P. Godefroid. Model Checking Partial State Spaces with 3-Valued Temporal Logics. In *Proc. of CAV'99*, LNCS 1633, pp. 274–287, Springer, 1999.
3. G. Bruns and P. Godefroid. Generalized model checking: Reasoning about partial state spaces. In *Proc. of CONCUR'00*, LNCS 1877, pp. 168–182. Springer, 2000.
4. E. M. Clarke and E. A. Emerson. Synthesis of synchronization skeletons for branching time temporal logic. In *Logic of Programs Workshop*, LNCS 131, pp. 244–263. Springer, 1981.
5. R. Cleaveland, S. P. Iyer, and D. Yankelevich. Optimality in abstractions of model checking. In *Proc. of SAS'95*, LNCS 983, pp. 51–63. Springer, 1995.
6. P. Cousot and R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs. In *Proc. of POPL'77*, pp. 238–252, ACM Press, 1977.
7. D. Dams. *Abstract interpretation and partition refinement for model checking*. PhD thesis, Technische Universiteit Eindhoven, The Netherlands, 1996.
8. D. Dams, R. Gerth, and O. Grumberg. Abstract interpretation of reactive systems. *ACM TOPLAS*, 19(2):253–291, 1997.
9. D. Dams and K. S. Namjoshi. Automata as abstractions. In *Proc. of VMCAI'05*, LNCS 3385, pp. 216–232. Springer, 2005.
10. P. Godefroid, M. Huth, and R. Jagadeesan. Abstraction-based Model Checking using Modal Transition Systems. In *Proc. of CONCUR'01*, LNCS 2154, pp. 426–440, Springer, 2001.
11. P. Godefroid and R. Jagadeesan. Automatic abstraction using generalized model checking. In *Proc. of CAV'04*, LNCS 2404, pp. 137–150. Springer, 2002.
12. P. Godefroid and M. Huth. Model checking vs. generalized model checking: semantic minimization for temporal logics. In *Proc. of LICS'05*, pp. 158–167, IEEE Computer Society Press.
13. S. Graf and H. Saidi. Construction of abstract state graphs with PVS. In *Proc. of CAV'97*, LNCS 1254, pp. 72–83, Springer, 1997.
14. A. Gurfinkel, O. Wei, and M. Chechik. Systematic construction of abstractions for model-checking. In *Proc. of VMCAI'06*, LNCS 3855, pp. 381–397. Springer, 2006.
15. H. Fecher and M. Huth. Ranked Predicate Abstraction for Branching Time: Complete, Incremental, and Precise. In *Proc. of ATVA'06*, LNCS, Springer, to appear.
16. D. Janin and I. Walukiewicz. Automata for the modal mu-calculus and related results. In *Proc. of MFCS'95*, LNCS 969, pp. 552–562. Springer, 1995.
17. M. Jurdziński. Deciding the winner in parity games is in UP∩co-UP. *Information Processing Letters*, 68(3):119–124, 1998.
18. D. Kozen. Results on the propositional μ-calculus. *Theoretical Computer Science* 27:333–354, 1983.
19. K. G. Larsen and L. Xinxin. Equation solving using modal transition systems. In *Proc. of LICS'90*, pp. 108–117. IEEE Computer Society Press, 1990.
20. J. P. Quielle and J. Sifakis. Specification and verification of concurrent systems in CESAR. In *Proc. of the 5th International Symposium on Programming*, 1981.
21. D. A. Schmidt. Closed and logical relations for over- and under-approximation of powersets. In *Proc. of SAS'04*, LNCS 3148, pp. 22–37, Springer, 2004.
22. S. Shoham and O. Grumberg. 3-valued abstraction: More precision at less cost. In *Proc. of LICS'06*, pp. 399–410. IEEE Computer Society Press, 2006.
23. S. Shoham and O. Grumberg. Monotonic abstraction-refinement for CTL. In *Proc. of TACAS'04*, LNCS 2988, pp. 546–560. Springer, 2004.
24. Th. Wilke. Alternating tree automata, parity games, and modal μ-calculus. *Bull. Soc. Math. Belg.*, 8(2):359–391, May 2001.

A Proof sketches

In all satisfaction games of this paper, the Verifier (resp. the Refuter) has a winning strategy iff the Verifier (resp. the Refuter) has a memoryless winning strategy. This is so since all these games are parity games.

Proof (Theorem 1). We show the two points of the theorem separately.

- $(K, s) \models^{\alpha}_{-} (A, q) \Rightarrow (K, s) \models^{\alpha}_{+} (A, q)$:
  Let γ_{-} be a winning strategy for the Verifier with respect to the satisfaction game $(K, s) \models^{\alpha}_{-} (A, q)$. Let U be the set of configurations (s', q') such that γ_{-} is a winning strategy for $(\alpha(s'), q')$. A winning strategy for the Verifier on any $(s', q') \in U$ is defined as follows: If $\delta(q') = q_1 \tilde{\wedge} q_2$ or if $\delta(q') = \mathsf{EX}\, q''$, the Verifier plays according to γ_{-} at $(\alpha(s'), q')$.
  It is easily checked that a play beginning in a configuration of U and played by the Verifier according to the memoryless strategy described above stays within U and is won by the Verifier.
- $(K, s) \models^{\alpha}_{+} (A, q) \Rightarrow (K, s) \models (A, q)$:
  Let γ_{+} be a winning strategy for the Verifier with respect to the satisfaction game $(K, s) \models^{\alpha}_{+} (A, q)$. Let V be the set of configurations (s', q') such that γ_{+} is a winning strategy for (s', q'). A winning strategy for the Verifier on any $(s', q') \in V$ is defined as follows: If $\delta(q') = q_1 \tilde{\wedge} q_2$ or if $\delta(q') = \mathsf{EX}\, q''$, the Verifier plays according to γ_{+} at (s', q').
  It is easily checked that a play beginning in a configuration of V and played by the Verifier according to the memoryless strategy described above stays within V and is won by the Verifier. ■

Proof (Theorem 2). We show the two implications of the theorem separately.

- $(K, s) \models^{\alpha}_{-} (A, q) \Rightarrow (G^{K}_{\alpha}, \alpha(s)) \models (A, q)$:
  Let γ_{-} be a winning strategy for the Verifier with respect to the satisfaction game $(K, s) \models^{\alpha}_{-} (A, q)$. Let U be the set of configurations (i', q') such that γ_{-} is a winning strategy for (i', q'). A winning strategy for the Verifier on set U is defined as follows:
  - If $\delta(q') = q_1 \tilde{\wedge} q_2$, the Verifier plays according to γ_{-} at (i', q').
  - Now suppose $\delta(q') = \mathsf{EX}\, q''$. Let $s_{s'}$ be the move of the Verifier according to strategy γ_{-} at (i', q') after the Refuter picked s' with $\alpha(s') = i'$. By the definition of pre-abstraction $(i', \{s_{s'} \mid \alpha(s') = i'\}) \in R^{-}_{K,\alpha}$. The Verifier replies with this set $\{s_{s'} \mid \alpha(s') = i'\}$.

  It is easily checked that a play beginning in a configuration of U and played by the Verifier according to the memoryless strategy described in the two items above stays within U and is won by the Verifier.
- $(K, s) \models^{\alpha}_{-} (A, q) \Leftarrow (G^{K}_{\alpha}, \alpha(s)) \models (A, q)$:
  Let γ_{g} be a winning strategy for the Verifier with respect to the satisfaction game $(G^{K}_{\alpha}, \alpha(s)) \models (A, q)$. Let V be the set of configurations (i', q') such that γ_{g} is a winning strategy for (i', q'). A winning strategy for the Verifier on any $(i', q') \in V$ is defined as follows:

- If $\delta(q') = q_1 \tilde{\wedge} q_2$, the Verifier plays according to γ_g at (i', q').
- Consider $\delta(q') = \mathsf{EX}\, q''$. Suppose the Refuter picked s' with $\alpha(s') = i'$. Let D' be the reply of the Verifier according to strategy γ_g at (i', q'). Then, by the definition of pre-abstraction, there is some s'' in $\{s'\}.R$ such that $\alpha(s'') \in D'$. The Verifier now selects one such s'' and replies with this s''.

It is easily checked that a play beginning in a configuration of V and played by the Verifier according to the memoryless strategy described in the two items above stays within V and is won by the Verifier. ■

Proof (Theorem 3). We show the two implications of the theorem separately.

- $(K, s) \models^{\alpha}_{+} (A, q) \Rightarrow (M^K_\alpha, \mathsf{BR}^K_\alpha(s)) \models (A, q)$:
  Let γ_+ be a winning strategy for the Verifier with respect to the satisfaction game $(K, s) \models^{\alpha}_{+} (A, q)$. Define f to be a partial function from $\mathbb{P}(I \cup \mathrm{AP})$ into S such that

  - $f(b)$ is undefined if there is no $s' \in S$ with $\mathsf{BR}^K_\alpha(s') = b$ and
  - if there is some $s' \in S$ with $\mathsf{BR}^K_\alpha(s') = b$, we select one such s' and define $f(b)$ to be that s'.

  Let U be the set of configurations (β', q') such that

  (a) $\beta' \in I$ and, for all $s' \in \alpha.\{\beta'\}$, strategy γ_+ is winning for (s', q') *or*
  (b) $\beta' \in \mathbb{P}(I \cup \mathrm{AP})$ and γ_+ is a winning strategy for $(f(\beta'), q')$.

  A winning strategy for the Verifier on any $(\beta', q') \in U$ is defined as follows:

  - If $\delta(q') = q_1 \tilde{\wedge} q_2$, the Verifier plays according to γ_+ at $(f(\beta'), q')$.
  - Consider $\delta(q') = \mathsf{EX}\, q''$. Let s'' be the reply of the Verifier via strategy γ_+ at $(f(\beta'), q')$. Thus $s'' \in \{f(\beta')\}.R$. By the constraint of f and by the definition of BR^K_α we get $\alpha(s'') \in \beta'$. Hence $\beta' \rightarrow^K_\alpha \alpha(s'')$ and so the Verifier can, and will, reply with $\alpha(s'')$.

  It is easily checked that a play beginning in a configuration of U and played by the Verifier according to the memoryless strategy described in the two items above stays within U and is won by the Verifier.
- $(K, s) \models^{\alpha}_{+} (A, q) \Leftarrow (M^K_\alpha, \mathsf{BR}^K_\alpha(s)) \models (A, q)$:
  Let γ_μ be a winning strategy for the Verifier with respect to the satisfaction game $(M^K_\alpha, \mathsf{BR}^K_\alpha(s)) \models (A, q)$. Let V be the set of configurations (s', q') such that γ_μ is a winning strategy for $(\mathsf{BR}^K_\alpha(s'), q')$. A winning strategy for the Verifier on any $(s', q') \in V$ is defined as follows:

  - If $\delta(q') = q_1 \tilde{\wedge} q_2$, the Verifier plays according to γ_μ at $(\mathsf{BR}^K_\alpha(s'), q')$.
  - Now suppose $\delta(q') = \mathsf{EX}\, q''$. Let i'' be the reply of the Verifier via strategy γ_μ at $(\mathsf{BR}^K_\alpha(s'), q')$. By the definition of the post-abstraction and BR^K_α, we have that $i'' \in \alpha(\{s'\}.R)$. Hence there is some $s'' \in \{s'\}.R$ with $\alpha(s'') = i''$. The Verifier selects some such s'' and replies with that s''.

It is easily checked that a play beginning in a configuration of V and played by the Verifier according to the memoryless strategy described in the two items above stays within V and is won by the Verifier. ■

Proof (Theorem 4). By Theorem 3, we have

$$(K,s) \models^{\alpha}_{+} (A,q) \quad\Leftrightarrow\quad (M^{K}_{\alpha}, \mathsf{BR}^{K}_{\alpha}(s)) \models (A,q)$$

Let γ_μ be a winning strategy for the Verifier with respect to the satisfaction game $(M^{K}_{\alpha}, \mathsf{BR}^{K}_{\alpha}(s)) \models (A,q)$. Let U be the set of configurations (b',q') such that γ_μ is a winning strategy for (b',q'). A winning strategy for the Verifier on any $(b',q') \in U$ is defined as follows:

- If $\delta(q') = q_1 \tilde{\wedge} q_2$, the Verifier plays according to γ_μ at (b',q').
- Consider the case $\delta(q') = \mathsf{EX}\, q''$. Suppose the Refuter picked s' with $\mathsf{BR}^{K}_{\alpha}(s') = b'$. Let i'' be the reply of the Verifier via strategy γ_μ at (b',q'). Then, by the definition of the post-abstraction and BR^{K}_{α}, there is some $s'' \in \{s'\}.R$ with $\alpha(s'') = i''$. The Verifier selects some such s'' and replies with s''.

It is easily checked that a play beginning in a configuration of U and played by the Verifier according to the memoryless strategy described in the two items above stays within U and is won by the Verifier. ■

Proof (Theorem 5). We show the two points of the theorem separately.

- $(K,s) \models^{f_2 \circ \alpha_1}_{-} (A,q) \Rightarrow (K,s) \models^{\alpha_1}_{-} (A,q)$:
  Let γ_- be a winning strategy for the Verifier with respect to the satisfaction game $(K,s) \models^{f_2 \circ \alpha_1}_{-}$. Let U be the set of configurations (i'_1, q') such that γ_- is a winning strategy for $(f_2(i'_1), q')$. A winning strategy for the Verifier on any $(i'_1, q') \in U$ is defined as follows:
  - If $\delta(q') = q_1 \tilde{\wedge} q_2$, the Verifier plays according to γ_- at $(f_2(i'_1), q')$.
  - Now consider $\delta(q') = \mathsf{EX}\, q''$. Suppose the Refuter picked s' with $\alpha_1(s') = i'_1$. Let s'' be the reply of the Verifier via strategy γ_- at $(f_2(i'_1), q')$ after the Refuter picked s'. The Verifier therefore replies with s''.

  It is easily checked that a play beginning in a configuration of U and played by the Verifier according to the memoryless strategy described in the two items above stays within U and is won by the Verifier.
- $(K,s) \models^{f_2 \circ \alpha_1}_{+} (A,q) \Rightarrow (K,s) \models^{\alpha_1}_{+} (A,q)$:
  Let γ_- be a winning strategy for the Verifier with respect to the satisfaction game $(K,s) \models^{f_2 \circ \alpha_1}_{+} (A,q)$. Let V be the set of configurations (s',q') such that γ_+ is a winning strategy for (s',q'). A winning strategy for the Verifier on any $(s',q') \in V$ is defined as follows: If $\delta(q') = q_1 \tilde{\wedge} q_2$ or if $\delta(q') = \mathsf{EX}\, q''$, the Verifier plays according to γ_+ at (s',q').
  It is easily checked that a play beginning in a configuration of V and played by the Verifier according to the memoryless strategy described above stays within V and is won by the Verifier. ■

Proof (Theorem 6). Define $I' = I_1 \times I_2$ and let the projection $f = \pi_2 \colon I' \to I_2$ be given by $\pi_2(i_1, i_2) = i_2$. Furthermore, let the abstraction function $\alpha_3 \colon S \to I'$ be defined by $\alpha_3(s) = (\alpha_1(s), \alpha_2(s))$. Let I be the restriction of I' onto the image of α_3. Then the first constraint $\alpha_2 = f \circ \alpha_3$ is satisfied. Furthermore, $(K,s) \models^{\alpha_1}_{-} (A,q) \Leftrightarrow (K,s) \models^{\pi_1 \circ \alpha_3}_{-} (A,q)$, where $\pi_1(i_1,i_2) = i_1$. By Theorem 5, $(K,s) \models^{\pi_1 \circ \alpha_3}_{-} (A,q) \Rightarrow (K,s) \models^{\alpha_3}_{-} (A,q)$, as required.

The second point of the theorem is shown in a similar manner. ■