More precise partition abstractions

Harald Fecher[1] **Michael Huth**[2]

[1]Christian-Albrechts-University at Kiel, Germany

[2]Imperial College London, United Kingdom

Nice, VMCAI 2007, 15 January 2007

Motivation

- Branching time: multiple system observers; biological systems; modeling faulty systems
- Branching time logics: computation tree logic, modal mu-calculus (modal logic with least and greatest fixpoints)
- Model checking not directly applicable on large or infinite systems
- Counter-example-guided abstraction refinement (CEGAR): initial abstraction; model check; spurious counterexample → refinement; loop
- Abstraction technique: predicate abstraction (synthesized automatically using theorem provers)

Precision

- Predicate abstraction yields partition of concrete state space, i.e., abstraction function from the concrete to abstract state space.
- Precise abstraction: best abstraction of concrete system with respect to abstraction function.
- Precision usually defined to be best with respect to given class of abstract models.
- Definition of precision, independent of model structure of abstractions, yields better characterization and can lead to better abstractions.

Main contributions

- Develop **two** abstract-model-independent characterizations of precision for abstractions based on partitions.
- Uses satisfaction games over concrete state space: refuter may switch between states abstracted to the same element.
- Develop precise abstractions for these two characterizations and study their expressive power and size.

Mu-calculus: alternating tree automata

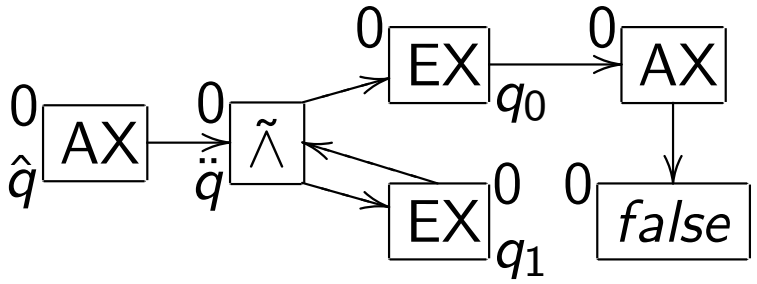
Alternating tree automaton over AP, including *false* and *true*:

- $Q(\ni q)$ a finite, nonempty set of states
- δ_A a transition relation mapping Q to

$$p \mid \neg p \mid q' \tilde{\wedge}\, q'' \mid q' \tilde{\vee}\, q'' \mid \mathrm{EX}\, q' \mid \mathrm{AX} q'$$

  with $q', q'' \in Q$ and $p \in \mathrm{AP}$
- Θ an acceptance condition mapping Q to a finite set of natural numbers.

  Infinite sequence accepted iff maximal acceptance number of those that occur infinitely often is even.



Kripke structure satisfaction

Concrete models: *Kripke structure* (S, R, L) over finite set of propositions AP

Game-rules (at config. $(s, q) \in S \times Q$, case analysis over q)

p: *Verifier wins if* $p \in L(s)$; *Refuter wins if* $p \notin L(s)$

$\neg p$: *Verifier wins if* $p \notin L(s)$; *Refuter wins if* $p \in L(s)$

$q_1 \tilde{\wedge} q_2$: *Refuter picks a* q' *from* $\{q_1, q_2\}$; *the next config. is* (s, q')

$q_1 \tilde{\vee} q_2$: *Verifier picks a* q' *from* $\{q_1, q_2\}$; *the next config. is* (s, q')

$\mathsf{EX}\, q'$: *Verifier picks* $s' \in \{s\}.R$; *the next config. is* (s', q')

$\mathsf{AX}\, q'$: *Refuter picks* $s' \in \{s\}.R$; *the next config. is* (s', q').

Infinite play won by Verifier iff maximal acceptance number that occurs infinitely often is even.

Pre-game

Refuter may switch to related state **before** literal/quantifier moves.

Formally, where $\alpha\colon S \to I$ is an abstraction function:

Game-rules (at config. $(s, q) \in S \times Q$)

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

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

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

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

Post-game

Refuter may switch to related state **after** literal/quantifier moves.

Formally, where $\alpha\colon S \to I$ is an abstraction function:

Game-rules (at config. $(s, q) \in S \times Q$)

p: *Verifier wins if* $p \in L(s)$; *Refuter wins if* $p \notin L(s)$

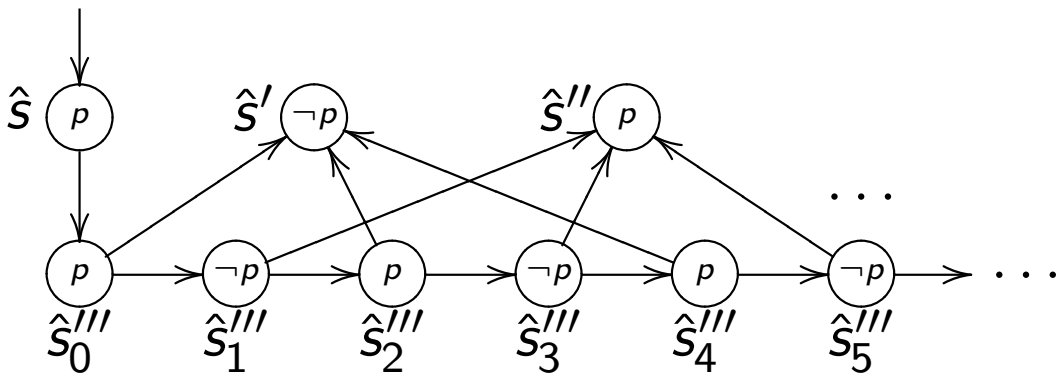
$\neg p$: *Verifier wins if* $p \notin L(s)$; *Refuter wins if* $p \in L(s)$

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

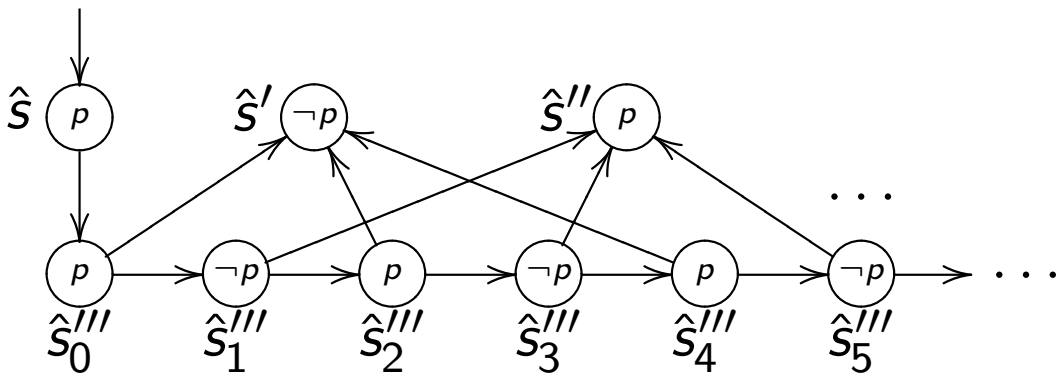
AX q': *Refuter picks* $s' \in \{s\}.R$ *and then some* $s'' \in S$ *with* $\alpha(s') = \alpha(s'')$; *the next config. is* (s', q').

Satisfaction relations $\models_{\mathrm{foo}}$ defined as usual in terms of winning strategies of these games.

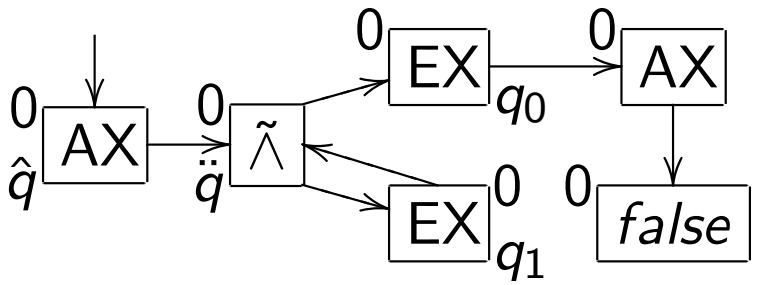
Satisfaction example



Satisfaction example

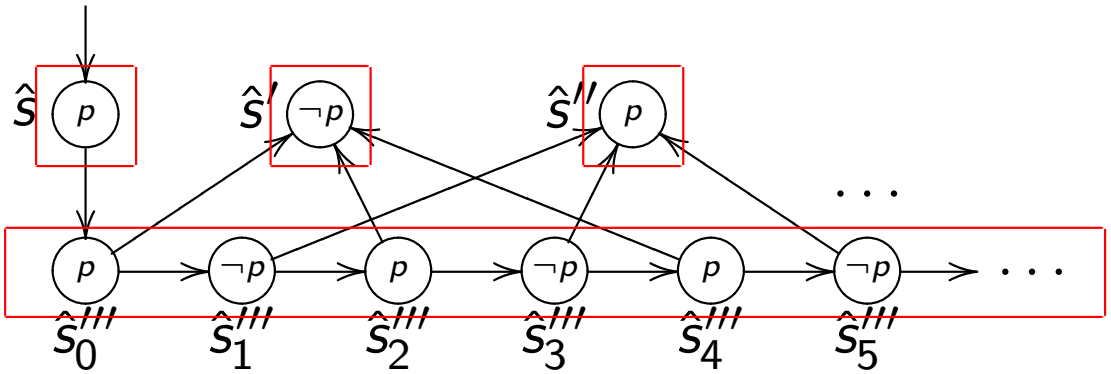


$\top_{\text{Kripke}}$

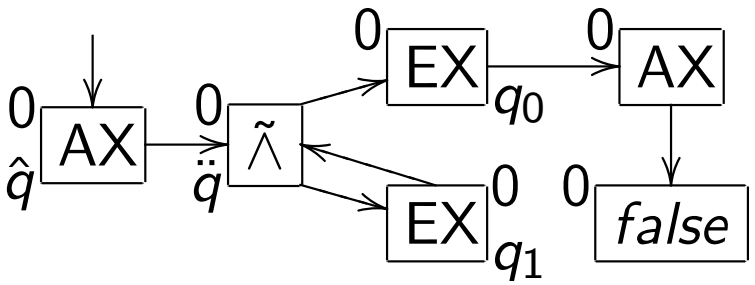


$$\widehat{=}\ \mathsf{AX}\ \ \nu Y.((\mathsf{EX}\ \ Y) \wedge (\mathsf{EX}\ \ \mathsf{AX}\ \ \mathit{false}))$$

Satisfaction example



$\top_{\text{Kripke}}$ $\quad$ $\overset{\alpha}{\top}_{\text{pre}}$ $\quad$ $\overset{\alpha}{\top}_{\text{post}}$



$$\widehat{=}\ \mathsf{AX}\ \ \nu Y.((\mathsf{EX}\ \ Y) \wedge (\mathsf{EX}\ \ \mathsf{AX}\ \ \mathit{false}))$$

Soundness of abstract satisfaction games

Theorem

Pre-satisfaction implies post-satisfaction, which implies satisfaction:

$$\begin{aligned} (K,s) \models^{\alpha}_{\text{pre}} (A,q) \quad &\Rightarrow \quad (K,s) \models^{\alpha}_{\text{post}} (A,q) \\ (K,s) \models^{\alpha}_{\text{post}} (A,q) \quad &\Rightarrow \quad (K,s) \models_{\text{Kripke}} (A,q)\,. \end{aligned}$$

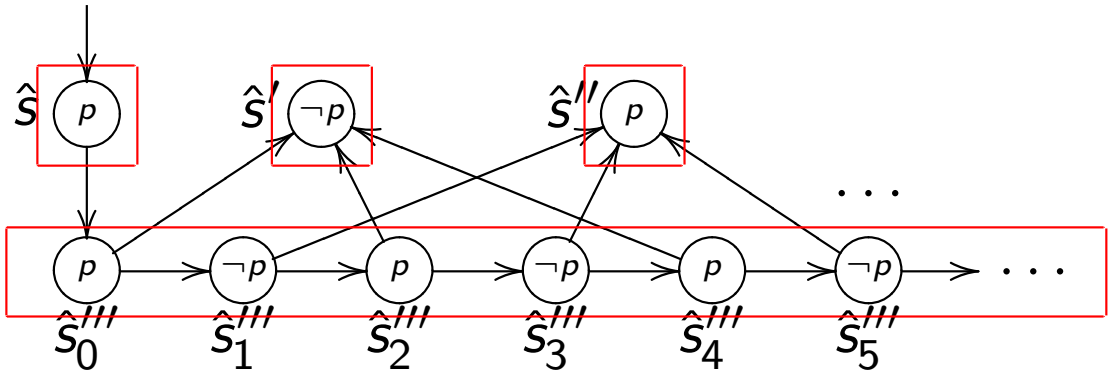
Soundness of abstract satisfaction games

Theorem

Pre-satisfaction implies post-satisfaction, which implies satisfaction:

$$(K, s) \models^{\alpha}_{\text{pre}} (A, q) \quad \Rightarrow \quad (K, s) \models^{\alpha}_{\text{post}} (A, q)$$
$$(K, s) \models^{\alpha}_{\text{post}} (A, q) \quad \Rightarrow \quad (K, s) \models_{\text{Kripke}} (A, q)\,.$$

The implications are strict.



$\models_{\text{Kripke}} \mathsf{EX}\, p$
$\not\models^{\alpha}_{\text{post}}$

$\models^{\alpha}_{\text{post}} \mathsf{EX}\,(p \vee \mathsf{EX}\, p)$
$\not\models^{\alpha}_{\text{pre}}$

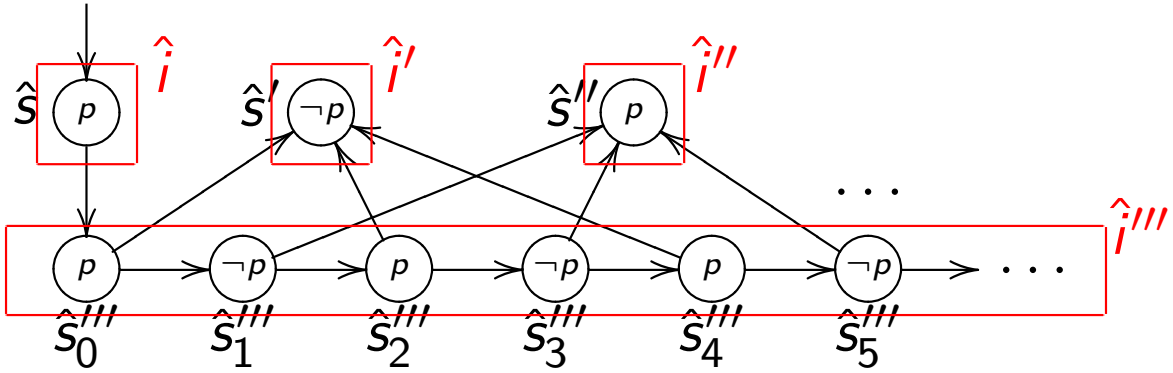
Precise abstractions for pre-satisfaction

Generalized Kripke modal transition systems have must-predicate labels, may-transitions (dashed arrows), and must-hypertransitions (solid arrows).

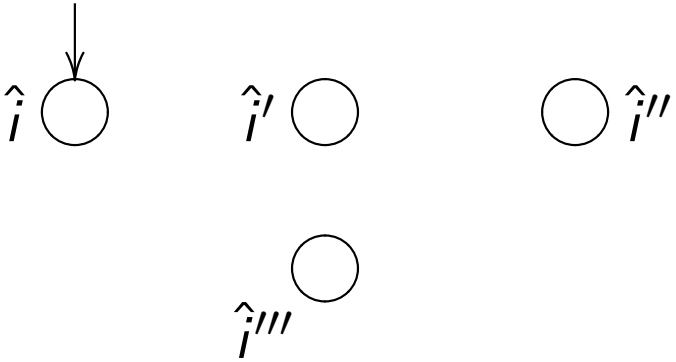
Precise abstractions for pre-satisfaction

Generalized Kripke modal transition systems have must-predicate labels, may-transitions (dashed arrows), and must-hypertransitions (solid arrows).

Kripke structure



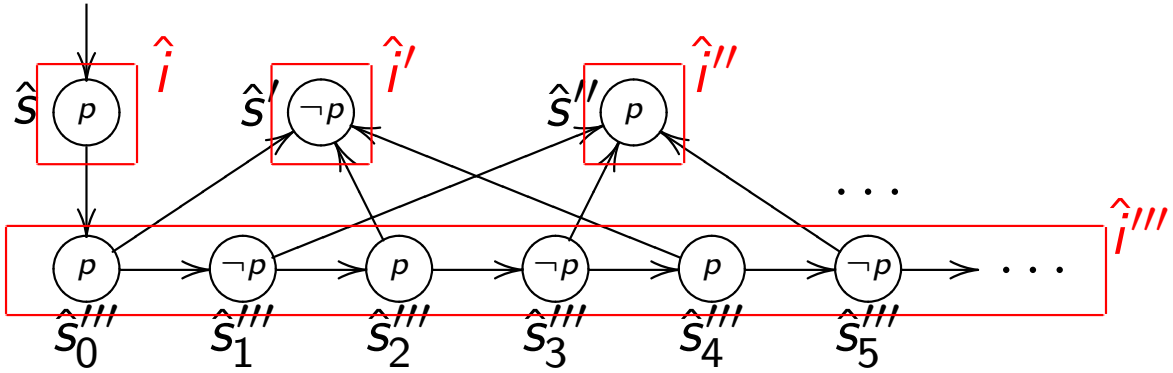
abstracted to Generalized Kripke modal transition system



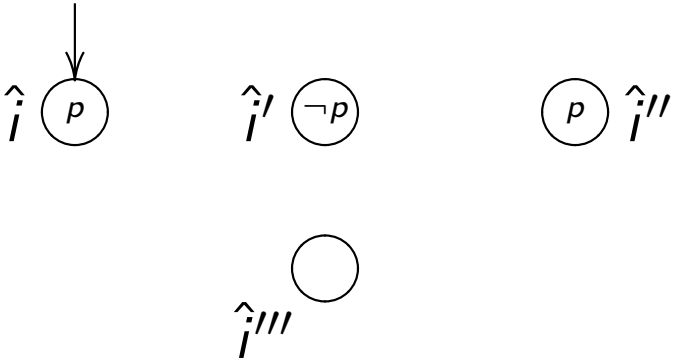
Precise abstractions for pre-satisfaction

Generalized Kripke modal transition systems have must-predicate labels, may-transitions (dashed arrows), and must-hypertransitions (solid arrows).

Kripke structure



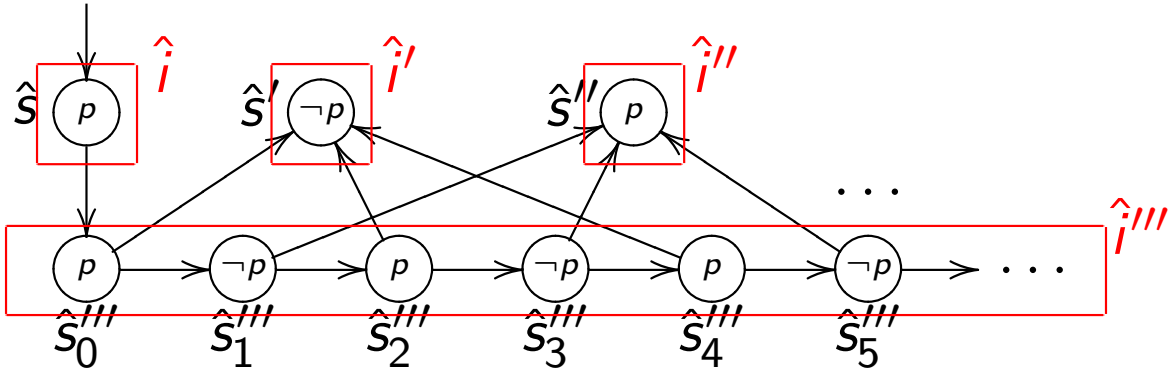
abstracted to Generalized Kripke modal transition system



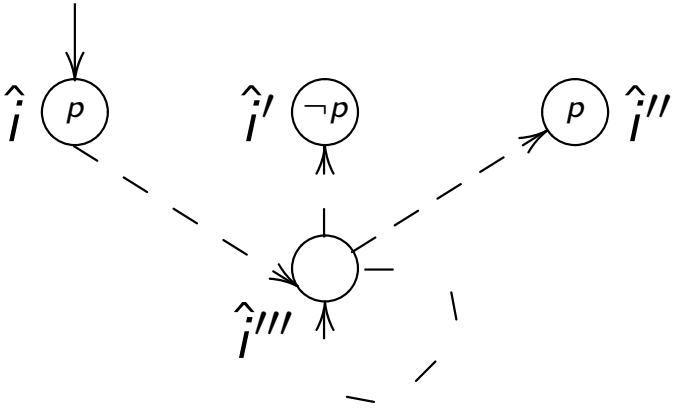
Precise abstractions for pre-satisfaction

Generalized Kripke modal transition systems have must-predicate labels, may-transitions (dashed arrows), and must-hypertransitions (solid arrows).

Kripke structure



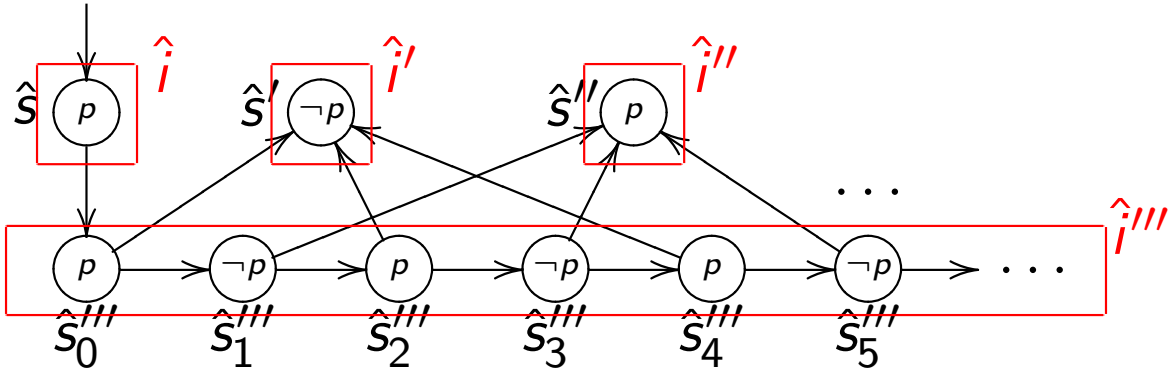
abstracted to Generalized Kripke modal transition system



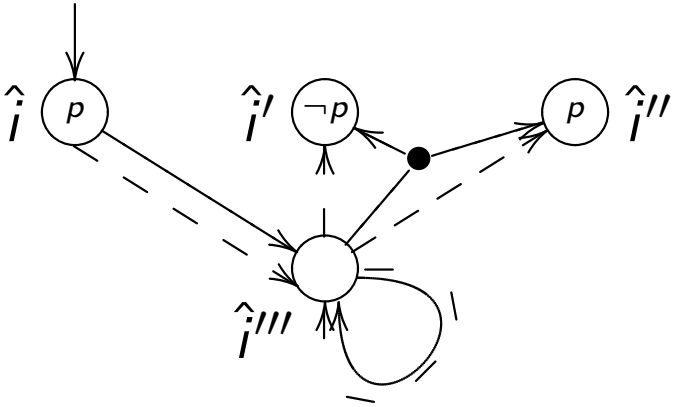
Precise abstractions for pre-satisfaction

Generalized Kripke modal transition systems have must-predicate labels, may-transitions (dashed arrows), and must-hypertransitions (solid arrows).

Kripke structure



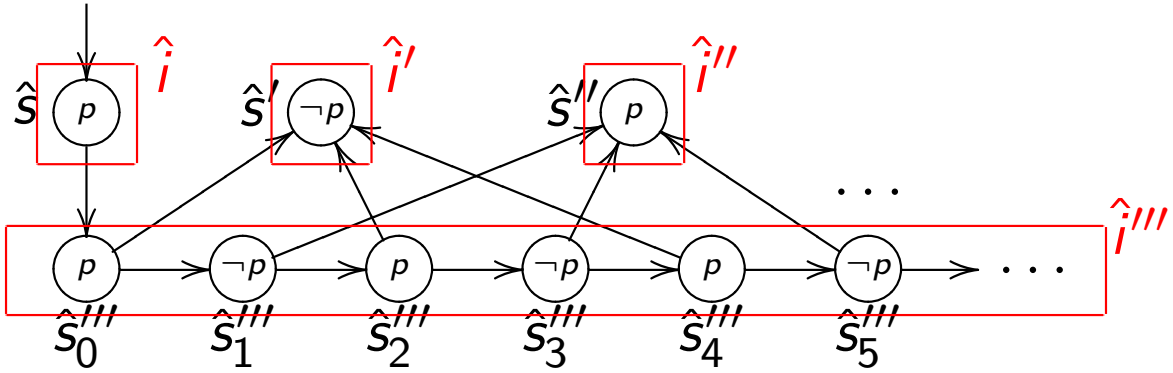
abstracted to Generalized Kripke modal transition system



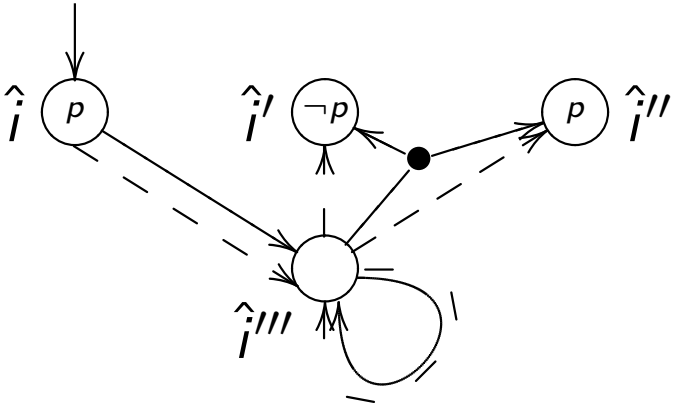
Precise abstractions for pre-satisfaction

Generalized Kripke modal transition systems have must-predicate labels, may-transitions (dashed arrows), and must-hypertransitions (solid arrows). Their satisfaction relation $\models_{\mathrm{GKMTS}}$ given in paper.

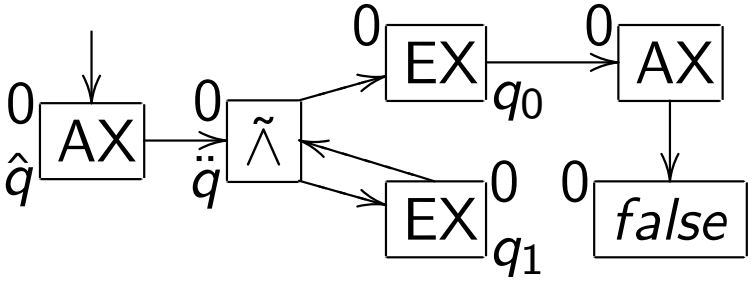
Kripke structure



abstracted to Generalized Kripke modal transition system



$\models_{\mathrm{GKMTS}}$

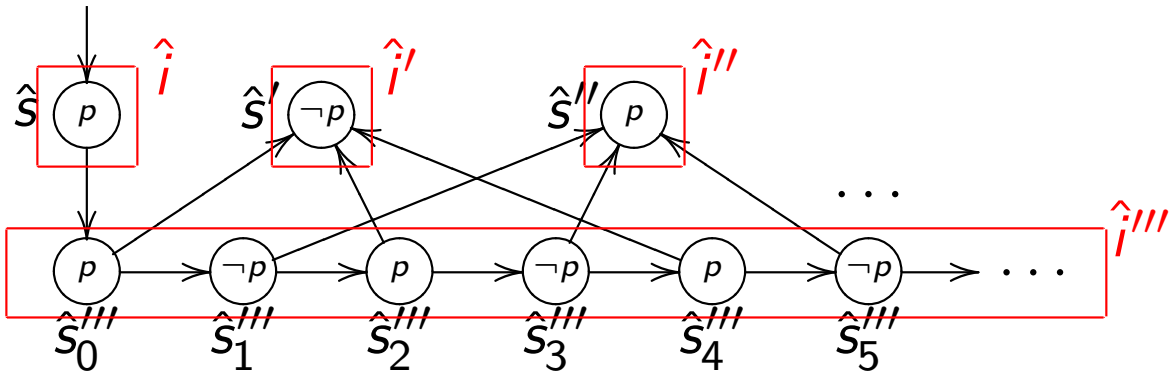


Precise abstractions for post-satisfaction

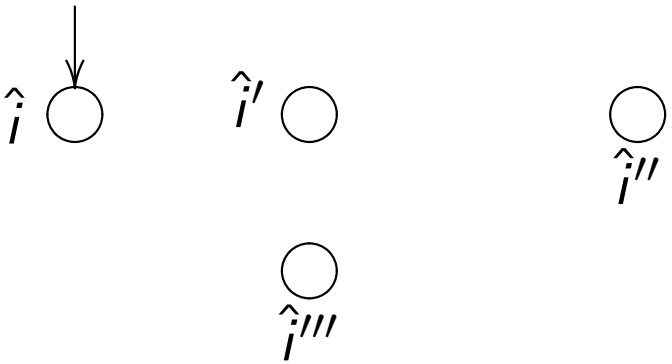
Use μ-automata with OR (unfilled circles) and BRANCH (filled circles)-states, predicate labeling over BRANCH, OR-relation from OR to BRANCH, and BRANCH-relation from BRANCH to OR.

Precise abstractions for post-satisfaction

Use μ-automata with OR (unfilled circles) and BRANCH (filled circles)-states, predicate labeling over BRANCH, OR-relation from OR to BRANCH, and BRANCH-relation from BRANCH to OR.

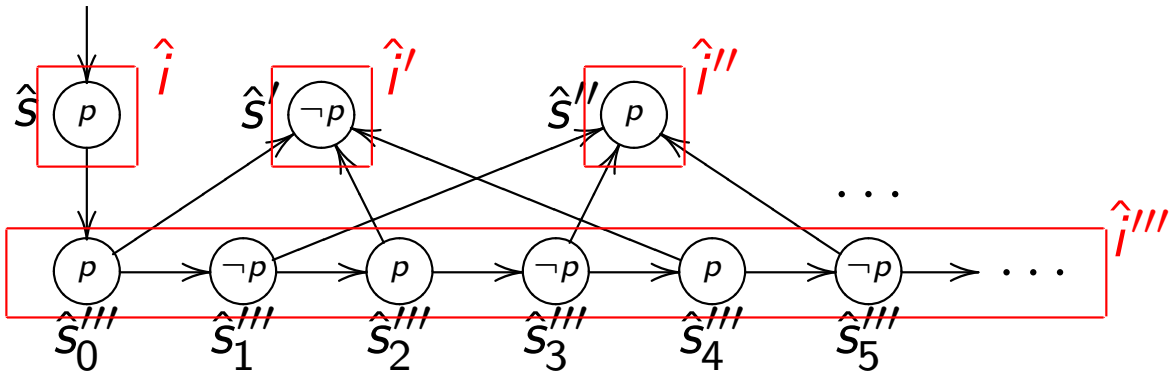


abstracted to μ-automaton

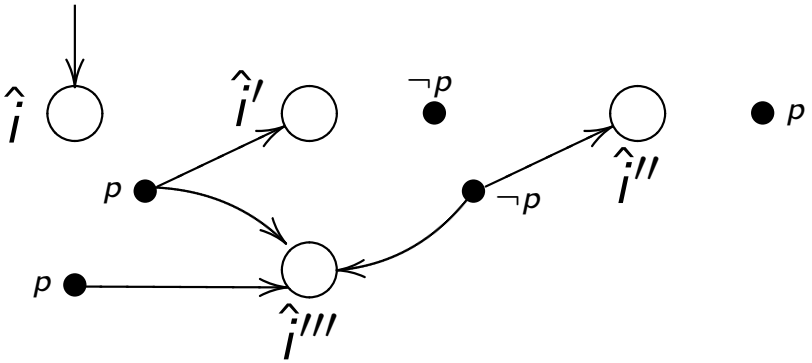


Precise abstractions for post-satisfaction

Use μ-automata with OR (unfilled circles) and BRANCH (filled circles)-states, predicate labeling over BRANCH, OR-relation from OR to BRANCH, and BRANCH-relation from BRANCH to OR.

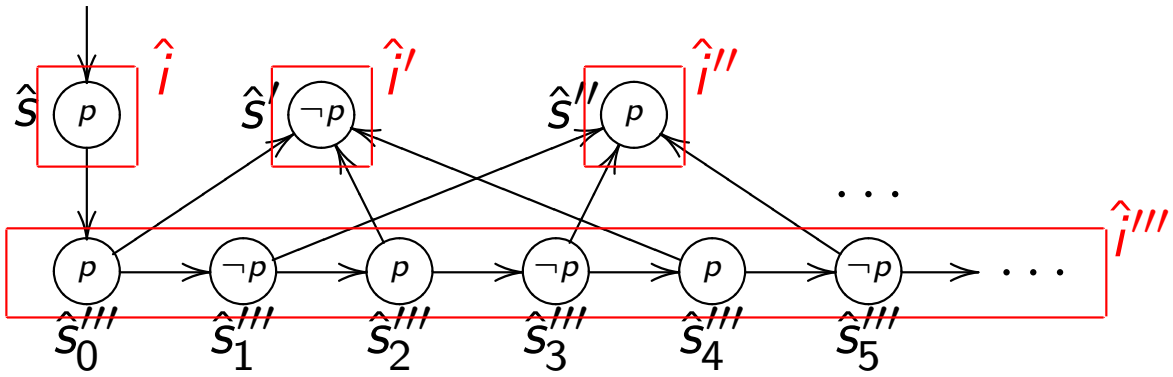


abstracted to μ-automaton

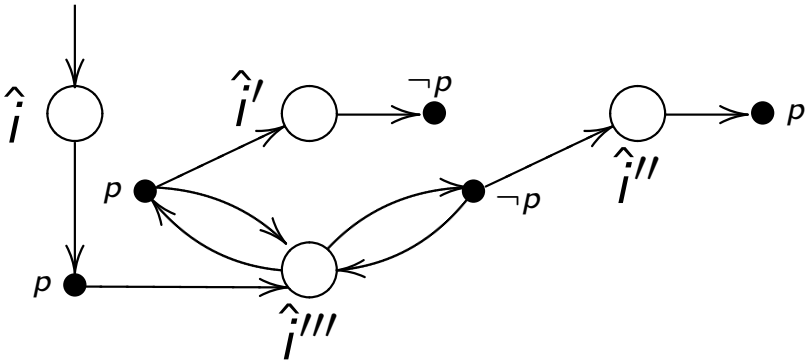


Precise abstractions for post-satisfaction

Use μ-automata with OR (unfilled circles) and BRANCH (filled circles)-states, predicate labeling over BRANCH, OR-relation from OR to BRANCH, and BRANCH-relation from BRANCH to OR.

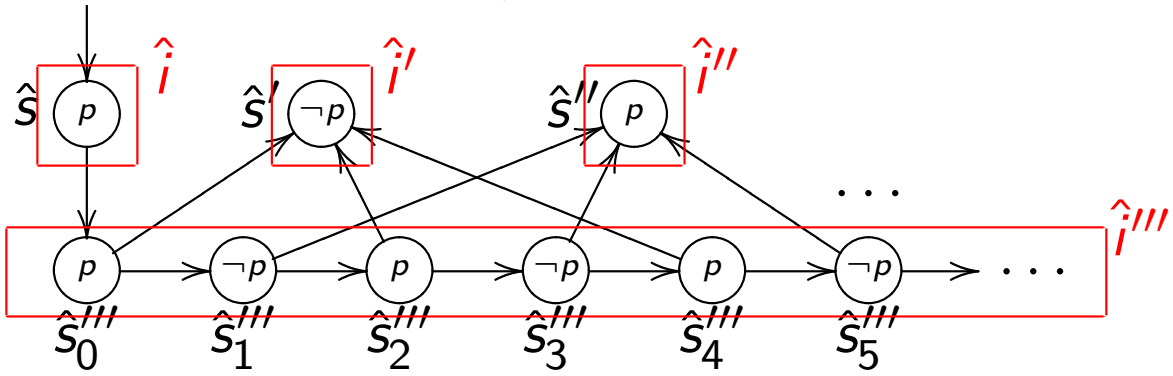


abstracted to μ-automaton

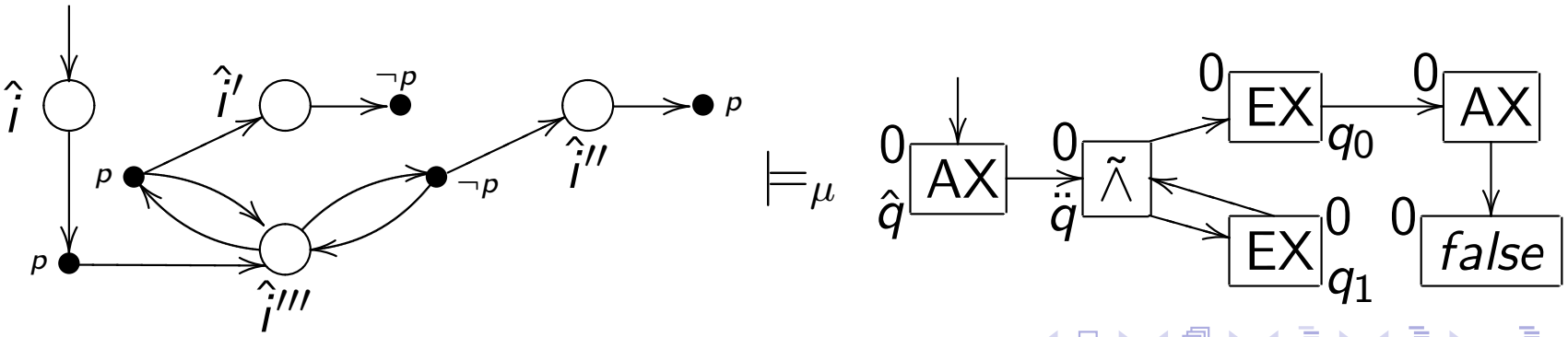


Precise abstractions for post-satisfaction

Use μ-automata with OR (unfilled circles) and BRANCH (filled circles)-states, predicate labeling over BRANCH, OR-relation from OR to BRANCH, and BRANCH-relation from BRANCH to OR. Their satisfaction relation $\models_\mu$ defined in paper.



abstracted to μ-automaton



Correspondence of games and abstractions

Theorem

Pre-game corresponds to pre-abstraction satisfaction:

$$(K, s) \models^{\alpha}_{\text{pre}} (A, q) \iff \textit{pre-abstr}_{\alpha}(K, s) \models_{\text{GKMTS}} (A, q)$$

Theorem

Post-game corresponds to post-abstraction satisfaction:

$$(K, s) \models^{\alpha}_{\text{post}} (A, q) \iff \textit{post-abstr}_{\alpha}(K, s) \models_{\mu} (A, q)$$

In other words, these abstractions are precise for their respective notions of games.

Synthesis

Pre- and post-abstraction can be synthesised with standard application of theorem prover.

Size of abstract models can be exponentially better for either kind.

Synthesis

Pre- and post-abstraction can be synthesised with standard application of theorem prover.

Size of abstract models can be exponentially better for either kind.

Theorem (Equal expressiveness)

Every properties shown by post-abstraction can be shown by pre-abstraction.

Proof.

Construct more complex abstraction function that corresponds to state space of μ-automata abstraction. □

Synthesis

Pre- and post-abstraction can be synthesised with standard application of theorem prover.

Size of abstract models can be exponentially better for either kind.

Theorem (Equal expressiveness)

Every properties shown by post-abstraction can be shown by pre-abstraction.

Proof.

Construct more complex abstraction function that corresponds to state space of μ-automata abstraction. □

Post-abstraction **not** redundant: exponential blow-up may occur through hypertransitions.

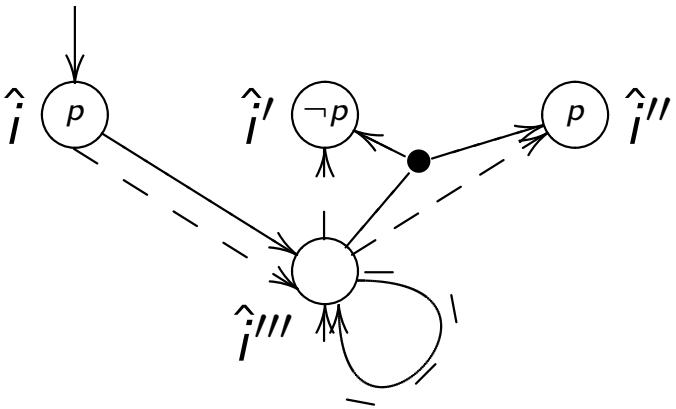
Equational theory

μ-automata: for any $p \in \mathrm{AP}$ we have $p \vee \neg p = \mathit{true}$ and $\mathsf{EX}\, p \vee \mathsf{AX}\, \neg p = \mathit{true}$ for satisfaction relation $\models_\mu$

Equational theory

μ-automata: for any $p \in \mathrm{AP}$ we have $p \vee \neg p = \mathit{true}$ and $\mathsf{EX}\, p \vee \mathsf{AX}\, \neg p = \mathit{true}$ for satisfaction relation $\models_\mu$

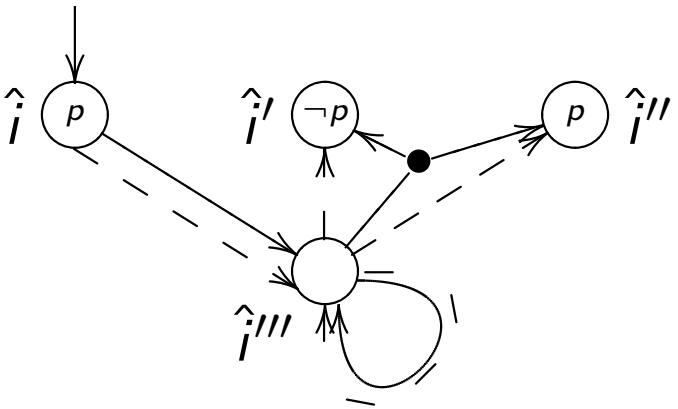
generalized Kripke modal transition systems: neither $p \vee \neg p = \mathit{true}$ nor $\mathsf{EX}\, p \vee \mathsf{AX}\, \neg p = \mathit{true}$ for $\models_{\mathrm{GMTS}}$ in state $\hat{i}'''$ of



Equational theory

μ-automata: for any $p \in \mathrm{AP}$ we have $p \vee \neg p = \mathit{true}$ and $\mathsf{EX}\, p \vee \mathsf{AX}\, \neg p = \mathit{true}$ for satisfaction relation $\models_\mu$

generalized Kripke modal transition systems: neither $p \vee \neg p = \mathit{true}$ nor $\mathsf{EX}\, p \vee \mathsf{AX}\, \neg p = \mathit{true}$ for $\models_{\mathrm{GMTS}}$ in state $\hat{i}'''$ of



$\mathsf{EX}\, q_1 \vee \mathsf{EX}\, q_2 = \mathsf{EX}\,(q_1 \vee q_2)$ **valid** in Kripke structures, but **neither** in μ-automata nor in generalized Kripke modal transition systems.

Abstraction refinement

Theorem (Incremental analysis)

Let $\alpha_1 \colon S \to I_1$, $f_2 \colon I_1 \to I_2$ *both surjective. Then*

$$(K, s) \models^{f_2 \circ \alpha_1}_{\text{pre}} (A, q) \quad \Rightarrow \quad (K, s) \models^{\alpha_1}_{\text{pre}} (A, q)$$
$$(K, s) \models^{f_2 \circ \alpha_1}_{\text{post}} (A, q) \quad \Rightarrow \quad (K, s) \models^{\alpha_1}_{\text{post}} (A, q)\,.$$

Theorem (Confluence)

Let $\alpha_1 \colon S \to I_1$, $\alpha_2 \colon S \to I_2$ *both surjective. Then there exist surjective* $\alpha_3 \colon S \to I_3$ *and* $f : I_3 \to I_2$ *such that* $\alpha_2 = f \circ \alpha_3$ *and*

$$(K, s) \models^{\alpha_1}_{\text{pre}} (A, q) \quad \Rightarrow \quad (K, s) \models^{\alpha_3}_{\text{pre}} (A, q)$$
$$(K, s) \models^{\alpha_1}_{\text{post}} (A, q) \quad \Rightarrow \quad (K, s) \models^{\alpha_3}_{\text{post}} (A, q)\,.$$

So CEGAR appropriately supported: refinement increases satisfaction, possibility of sucess does not depend on previous refinement decisions.

Conclusion

- Developed: abstract-model-independent characterization of precise partition-based abstraction, pre- and post-satisfaction games, where post-games are more precise
- Showed: existing abstraction into generalized Kripke modal transition system is precise for pre-abstraction
- Developed: new abstraction into μ-automata that is precise for post-abstraction
- Showed: both abstractions can be synthesised and are incomparable in size.
- Showed: abstractions suitable for counter-example-guided abstraction-refinement.

Related work

- [G. Bruns and P. Godefroid. Generalized model checking: Reasoning about partial state spaces. In *Proc. of CONCUR'00*] consider generalized model checking, yielding precision of model checking algorithms wrt. thorough semantics (ψ holds iff it holds for all concrete ref.).
- [P. Godefroid and M. Huth. Model checking vs. generalized model checking: semantic minimization for tempor al logics. In *Proc. of LICS'05*] determines semantically self-minimizing formulas, where compositional check coincide with the thorough semantics.
- [S. Shoham and O. Grumberg. 3-valued abstraction: More precision at less cost. In *Proc. of LICS'06*] define precision independent of the abstract models in terms of properties of algebraic operators (coincides with pre-abstraction for partition based abstractions).
- [H. Fecher and M. Huth. Ranked Predicate Abstraction for Branching Time: Complete, Incremental, and Precise. In *Proc. of ATVA'06*] develop a model for which precise, finite-state abstractions can be computed by a generalization of predicate abstraction.

Future work

- Tool development for verification based on pre- and post-abstraction.
- Combinations of pre- and post-abstraction. Promising sketch already in this paper.
- Examination of precision for relational abstraction (not partition-based). Here: more complex abstract models are required.
- Precision of satisfaction relation. Better understanding of which property equations hold for what kind of abstract models.