

Temporal Abstract Interpretation¹

Interprétation abstraite temporelle

Patrick COUSOT

DI, École normale supérieure
45 rue d'Ulm, 75230 Paris cedex 05, France

<mailto:Patrick.Cousot@ens.fr>

<http://www.di.ens.fr/~cousot>

IRISA, Rennes, Salle Michel Métivier — Mardi 11 janvier 2000

¹ Join work with R. Cousot, LIX.

1. Introductory motivations ...

Common believes

- The **verification** of a temporal specification for a transition system by **model-checking** is sound and **complete**²:

A temporal property holds if (soundness) and only if (completeness) it can be model-checked.

² at least for finite state systems.

Common believes

- The **verification** of a temporal specification for a transition system by **model-checking** is sound and **complete**²:

A temporal property holds if (soundness) and only if (completeness) it can be model-checked.

- The **verification** of a temporal specification for a program given by its small-step operational semantics by **static analysis** is sound and **incomplete**³;

² at least for finite state systems.

³ even for finite state systems.

Common believes

- The **verification** of a temporal specification for a transition system by **model-checking** is sound and **complete**²:
A temporal property holds if (soundness) and only if (completeness) it can be model-checked.
- The **verification** of a temporal specification for a program given by its small-step operational semantics by **static analysis** is sound and **incomplete**³;
- so **model-checking** is to be preferred to **program static analysis**.

² at least for finite state systems.

³ even for finite state systems.

Our point of view

Both for model-checking and program static analysis:

Our point of view

Both for model-checking and program static analysis:

- Approximations are involved;

Our point of view

Both for model-checking and program static analysis:

- Approximations are involved;
- So (in)completeness is relative (to an implicit reference temporal semantics);

Our point of view

Both for model-checking and program static analysis:

- Approximations are involved;
- So (in)completeness is relative (to an implicit reference temporal semantics);
- Abstract interpretation can help in understanding and comparing the approximations involved in each case.

Our point of view

Both for **model-checking** and **program static analysis**:

- **Approximations** are involved;
- So **(in)completeness** is relative (to an implicit reference temporal semantics);
- **Abstract interpretation** can help in understanding and comparing the approximations involved in each case.

Indeed both **model-checking** and **program static analysis** are abstract interpretations based on similar approximations.

**Why bother? we already have
“abstract model checking”!**

**Why bother? we already have
“abstract model checking”!**

Yes, but

Why bother? we already have “abstract model checking”!

Yes, but abstract model checking [1]:

- is based on a state-to-state abstraction which is not general enough!

Reference

[1] E. Clarke O. Grumberg & D. Long. Model checking and abstraction. *TOPLAS* 16 1994.

Why bother? we already have “abstract model checking”!

Yes, but abstract model checking [1]:

- is based on a state-to-state abstraction which is not general enough!
- e.g. it cannot take into account polyhedral model checking of hybrid systems à la Halbwachs et al. [2].

Reference

[1] E. Clarke O. Grumberg & D. Long. Model checking and abstraction. *TOPLAS* 16 1994.

[2] N. Halbwachs, J.-É. Proy, & P. Raymond. Verification of linear hybrid systems by means of convex approximations. *SAS '94*, LNCS 864, 1994.

A more general point of view is needed...

A more general point of view is needed...

- We would like to have to have a continuum of techniques ranging from model-checking to static program analysis;

A more general point of view is needed...

- We would like to have to have a continuum of techniques ranging from model-checking to static program analysis;
- **Abstract interpretation** can help with this general point of view.

A more general point of view is needed...

- We would like to have to have a continuum of techniques ranging from model-checking to static program analysis;
 - **Abstract interpretation** can help with this general point of view.
- ⇒ We consider a very general **temporal specification language**;

A more general point of view is needed...

- We would like to have to have a continuum of techniques ranging from model-checking to static program analysis;
 - **Abstract interpretation** can help with this general point of view.
- ⇒ We consider a very general **temporal specification language**;
- ⇒ We study its **abstractions**.

2. Temporal specification language

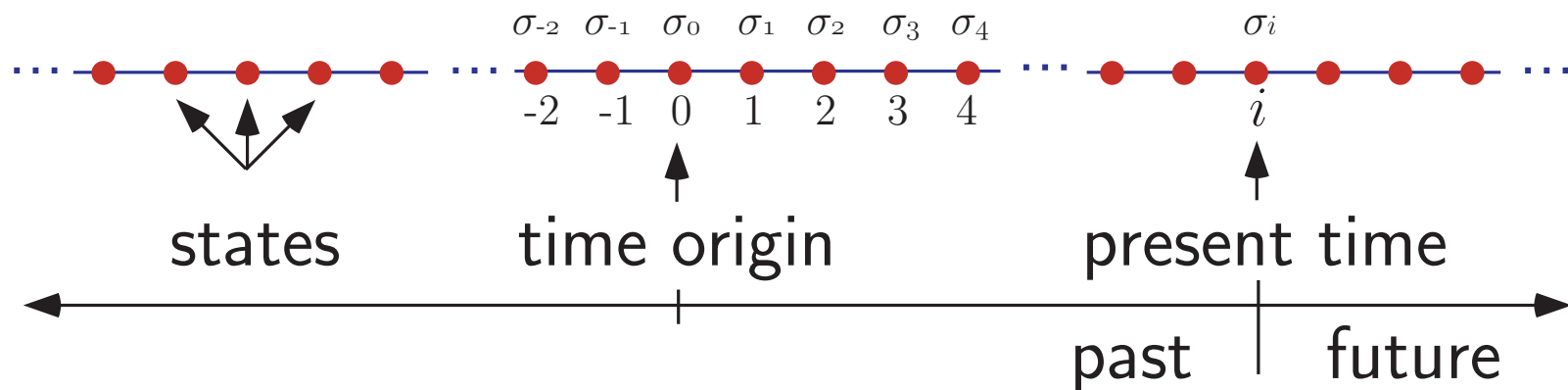
Temporal logics and calculi

- Temporal logics:
 - CTL^* ,
 - CTL ;
- Temporal calculi:
 - propositional μ -calculus;

are all generalized by the reversible $\hat{\mu}^*$ -calculus.

Semantic domain for the reversible $\hat{\mu}^*$ -calculus

- The semantics of a formula of the reversible $\hat{\mu}^*$ -calculus is a **model** that is a set of infinite time-symmetric traces;
- An infinite time-symmetric **trace** $\langle i, \sigma \rangle$:

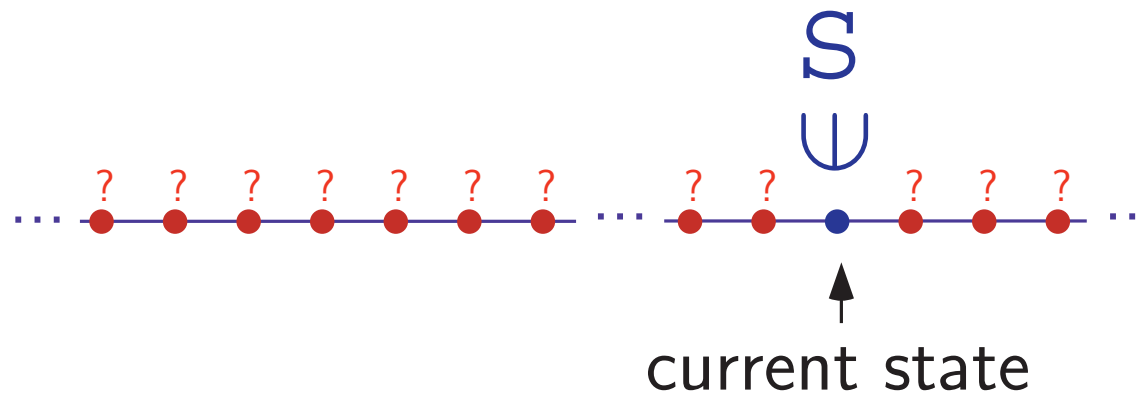


The reversible μ^* -calculus

$\varphi ::=$	σ_S	$S \in \wp(\mathbb{S})$	state predicate
	π_t	$t \in \wp(\mathbb{S} \times \mathbb{S})$	transition predicate
	$\oplus \varphi_1$		next
	$\varphi_1^\curvearrowright$		reversal
	$\varphi_1 \vee \varphi_2$		disjunction
	$\neg \varphi_1$		negation
	X	$X \in \mathbb{X}$	variable
	$\mu X \cdot \varphi_1$		least fixpoint
	$\nu X \cdot \varphi_1$		greatest fixpoint
	$\forall \varphi_1 : \varphi_2$		universal state closure

State predicates σ_S

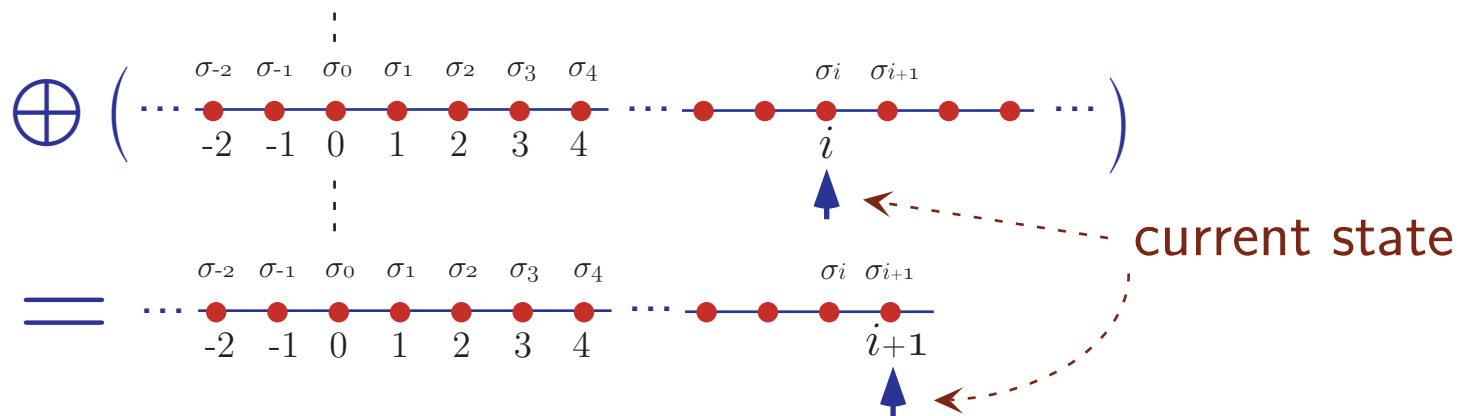
- The state predicate σ_S denotes all traces with current state in set S ⁴:



⁴ In this talk we identify a reversible $\hat{\mu}^*$ -calculus formula φ with its semantics/interpretation $\llbracket \varphi \rrbracket$.

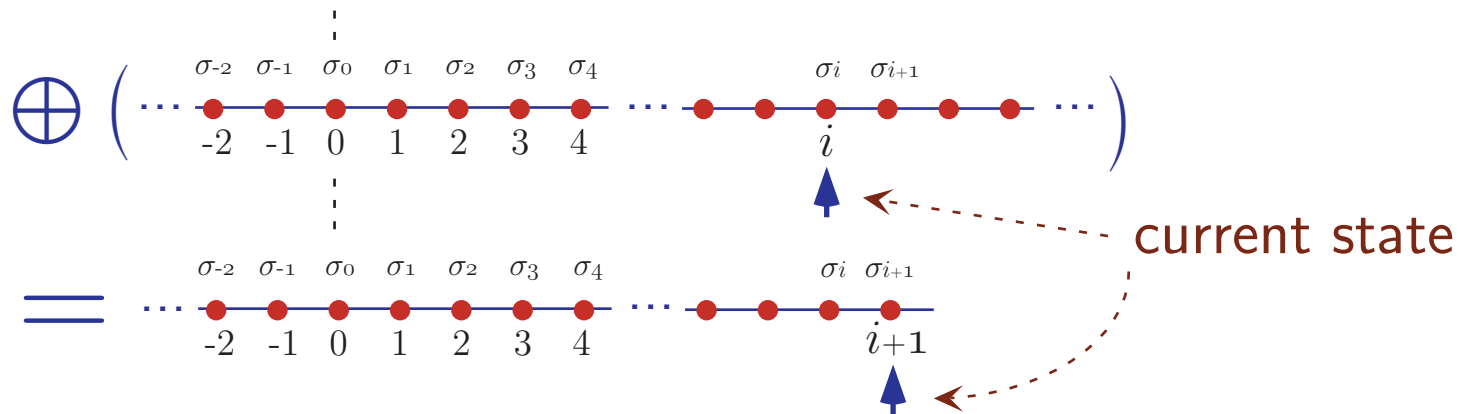
Next \oplus

- Trace next time:



Next \oplus

- Trace next time:



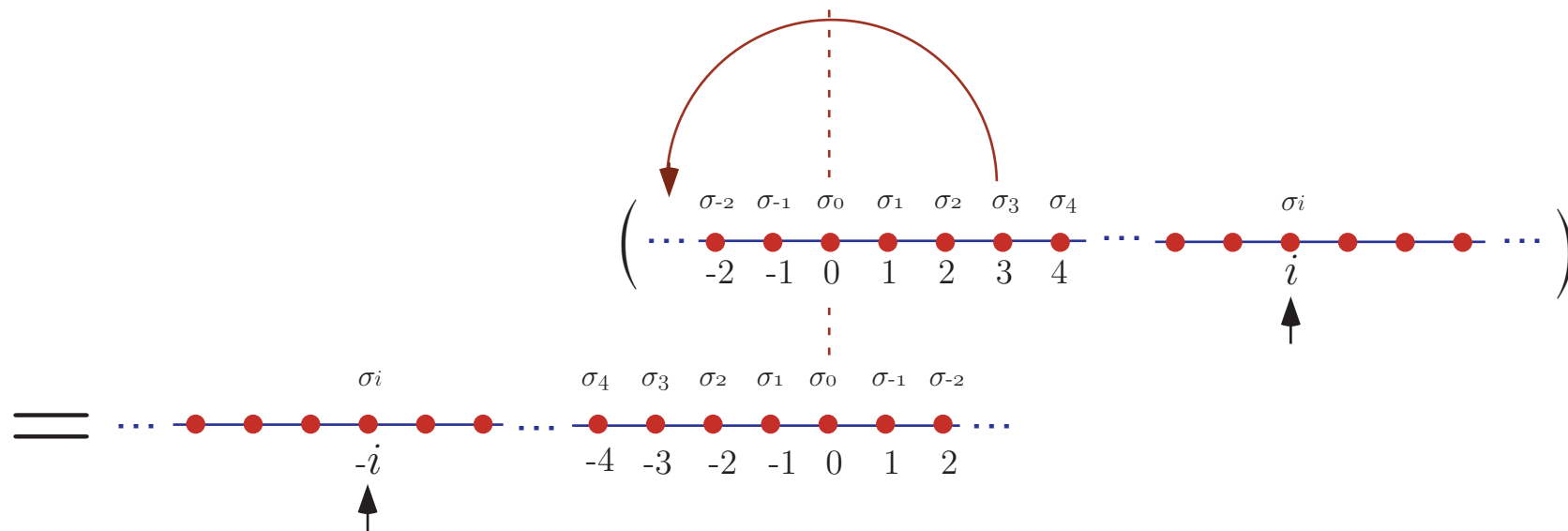
- Model next time:

$$\oplus M \triangleq \{ \langle i, \sigma \rangle \mid \oplus \langle i, \sigma \rangle \in M \}$$

A trace of $\oplus M$ will, at next time, be a trace of M .

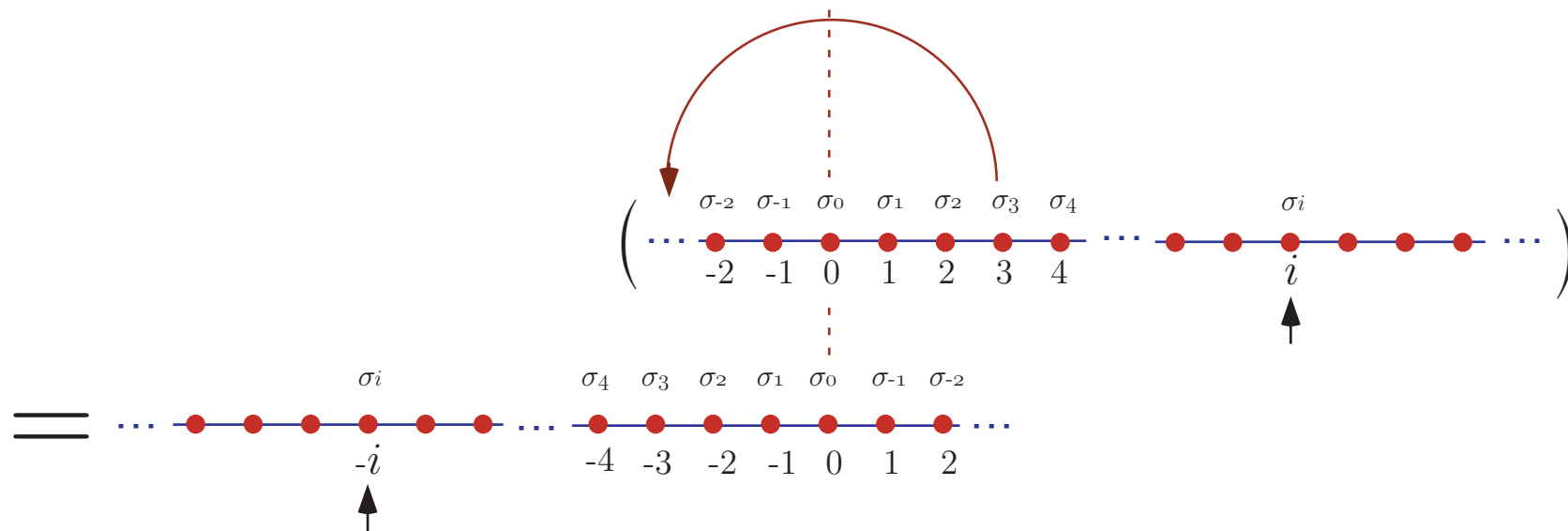
Reversal ↻

- Trace reversal:



Reversal ↻

- Trace reversal:



- Model reversal:

$$M^{\curvearrowright} \triangleq \{ \langle i, \sigma \rangle \mid \langle i, \sigma \rangle^{\curvearrowright} \in M \}$$

Universal \forall and existential \exists state closures

- The **universal state closure** $\forall \varphi_1 : \varphi_2$ is the set of traces $\langle i, \sigma \rangle$ of φ_1 such that all traces in φ_1 with the same current state σ_i belong to φ_2 ;

Universal \forall and existential \exists state closures

- The **universal state closure** $\forall \varphi_1 : \varphi_2$ is the set of traces $\langle i, \sigma \rangle$ of φ_1 such that all traces in φ_1 with the same current state σ_i belong to φ_2 ;
- The **existential state closure** $\exists \varphi_1 : \varphi_2 = \neg(\forall \varphi_1 : \neg \varphi_2)$ is the set of traces $\langle i, \sigma \rangle$ of φ_1 such that some trace in φ_1 with the same current state σ_i belongs to φ_2 .

Abbreviations (examples)

$$\varphi_1 \mathbf{U} \varphi_2 \stackrel{\Delta}{=} \mu X \cdot (\varphi_2 \vee (\varphi_1 \wedge \oplus X)) \quad \text{until}$$

Abbreviations (examples)

$$\varphi_1 \mathbf{U} \varphi_2 \stackrel{\Delta}{=} \mu X \cdot (\varphi_2 \vee (\varphi_1 \wedge \oplus X)) \quad \text{until}$$

$$\varphi_1 \mathbf{S} \varphi_2 \stackrel{\Delta}{=} (\varphi_1^{\curvearrowright} \mathbf{U} \varphi_2^{\curvearrowright})^{\curvearrowright} \quad \text{since}$$

Subcalculi

(example: Kozen's propositional μ -calculus)

$$\varphi ::= \sigma_S \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \wedge \varphi_2 \mid \neg \varphi_1 \mid \Box \varphi_1 \mid \Diamond \varphi_1 \mid \\ X \mid \mu X \cdot \varphi_1 \mid \nu X \cdot \varphi_1$$

where:

τ : transition relation (program SOS semantics);

$\Box \varphi_1 \triangleq \forall \pi_\tau : \oplus \varphi_1$ always (after next step);

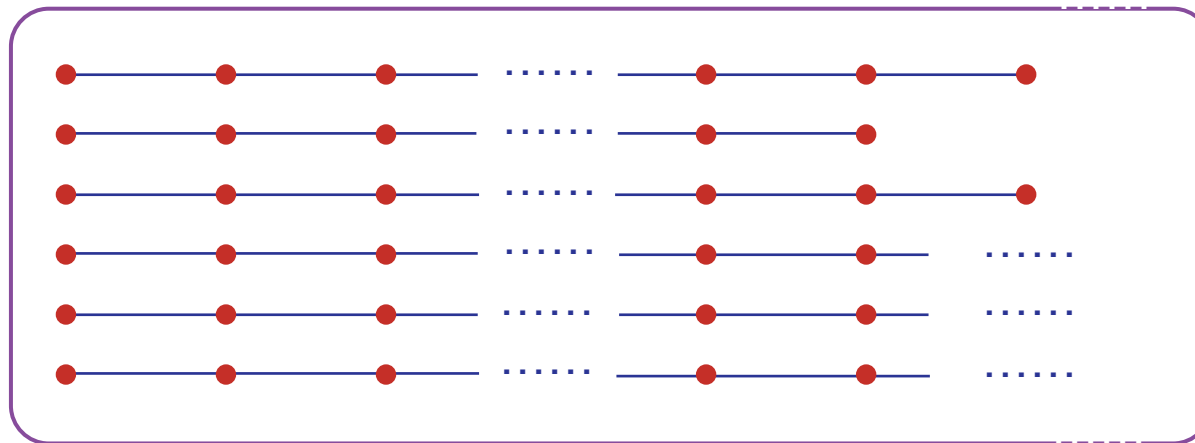
$\Diamond \varphi_1 \triangleq \exists \pi_\tau : \oplus \varphi_1$ sometime (after next step).

On the reversible μ^\star -calculus

- Generalization of previous temporal logics and calculi;
- Contrary to previous propositions:
 - Every logical statement is explicit (e.g. no implicit underlying Kripke structure),
 - A single temporal operator \curvearrowright to handle past and future,
 - Completely time-symmetric.

3. An intuitive example of abstraction/ approximation

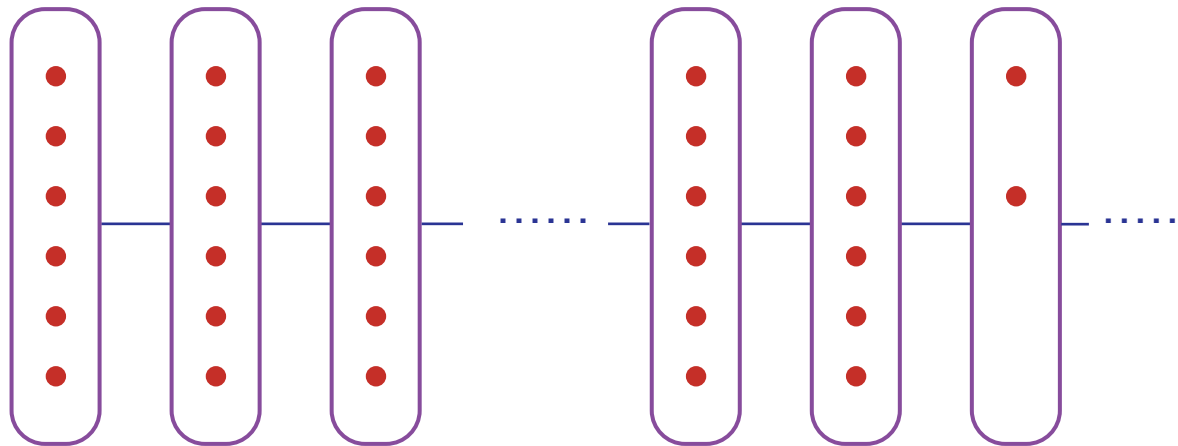
An example of approximation: a set of sequences of states



can be approximated/abstracted by .../...

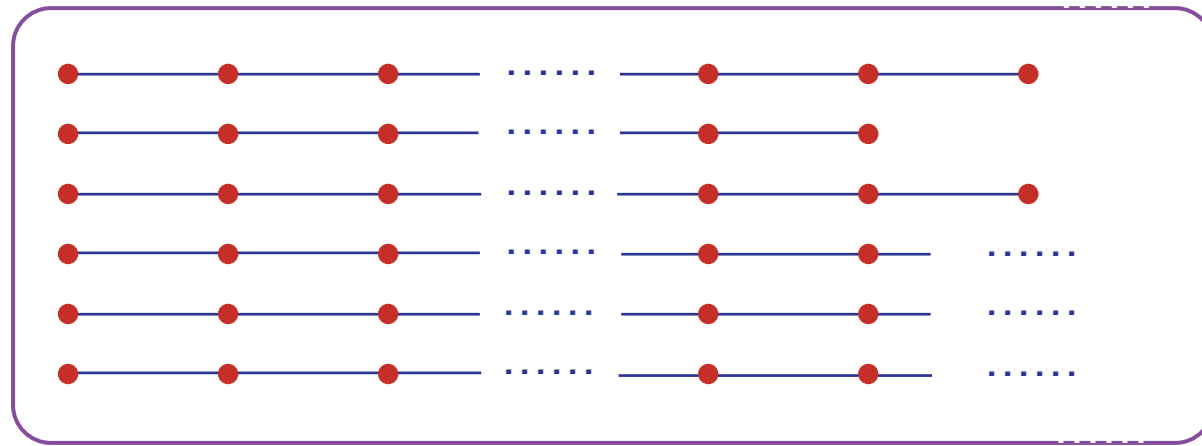
An example of approximation (cont'd)

a sequence of sets of states



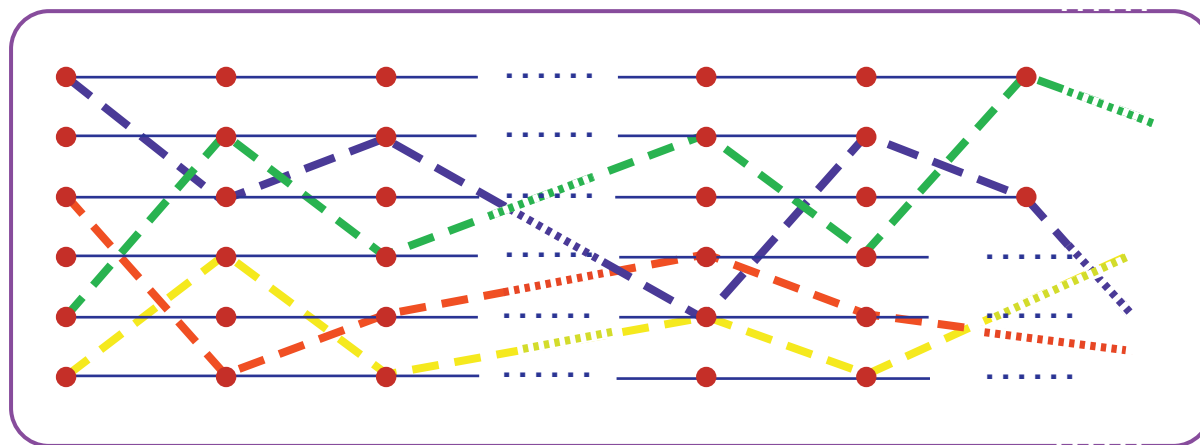
Back to a set of sequences of states

- The concretization contains all original traces:



Back to a set of sequences of states (cont'd)

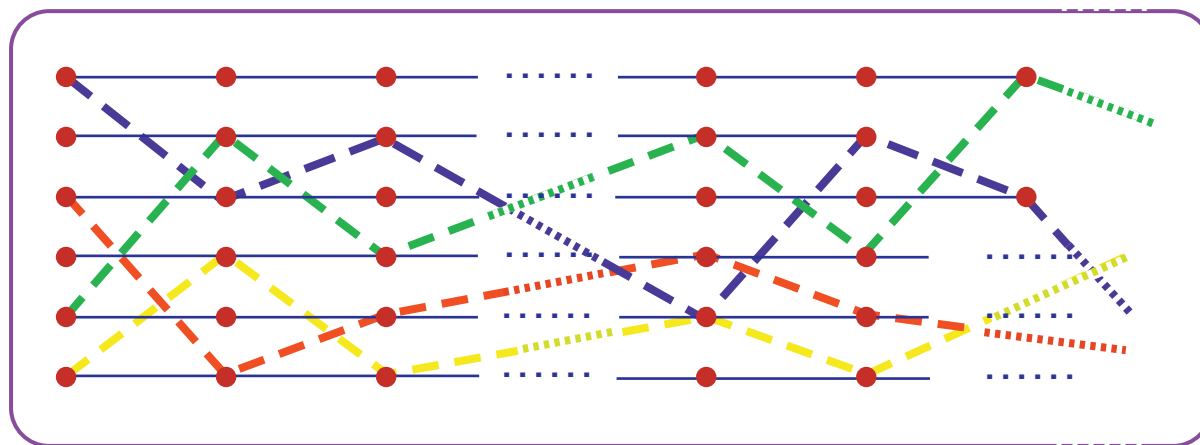
- The concretization contains all original traces:



plus (unrealistic) additional ones (— — — — — , — — — — — , — — — — — , — — — — — , ...);

Back to a set of sequences of states (cont'd)

- The concretization contains all original traces:

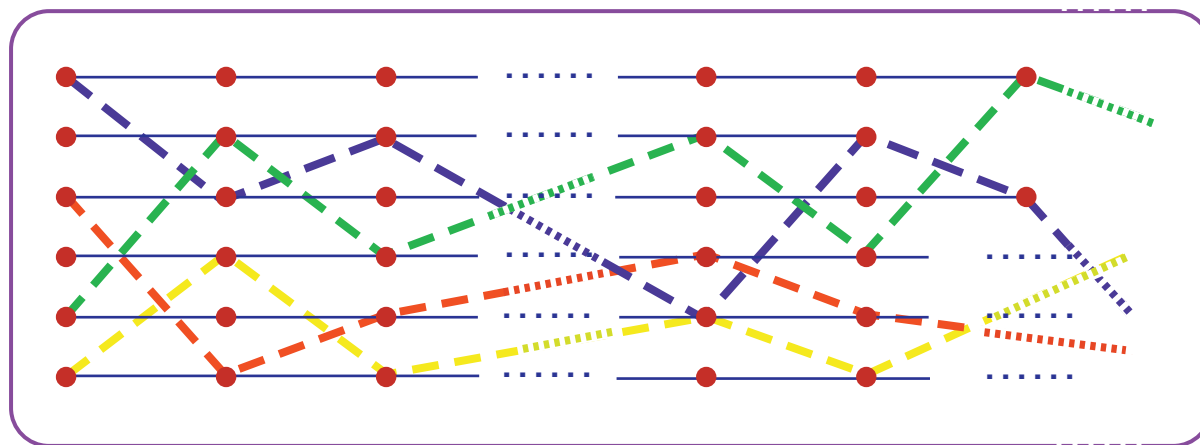


plus (unrealistic) additional ones (— — —, — — —, — — —, — — —, ...);

- This particular approximation is therefore from above;

Back to a set of sequences of states (cont'd)

- The concretization contains all original traces:

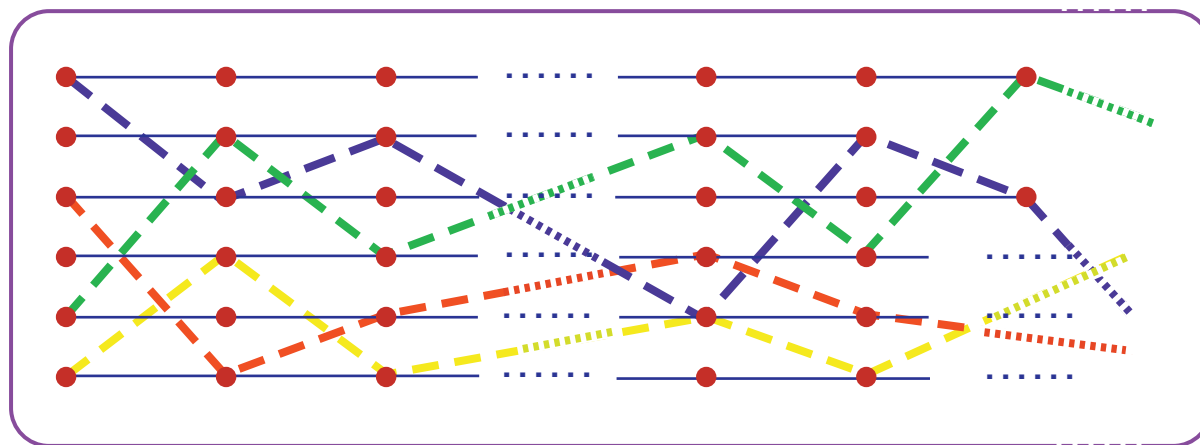


plus (unrealistic) additional ones (— — —, — — —, — — —, — — —, ...);

- This particular approximation is therefore from above;
- It contains more traces than possible.

Back to a set of sequences of states (cont'd)

- The **concretization** contains all original traces:

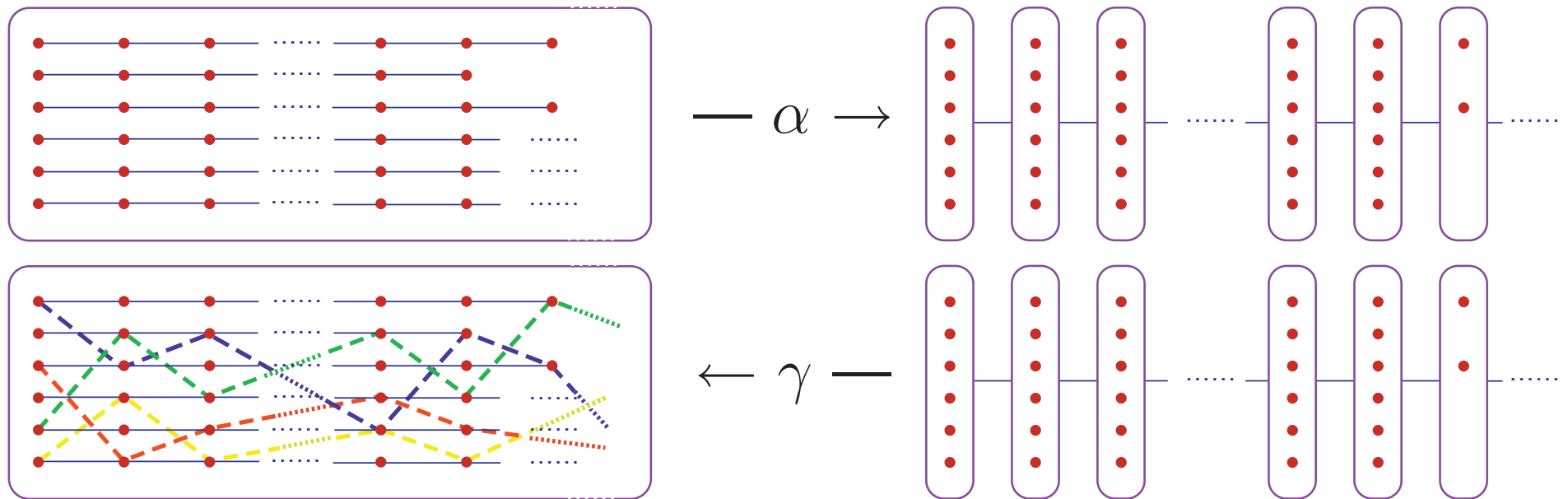


plus (unrealistic) additional ones (---, ---, ---, ---, ...);

- This particular **approximation** is therefore **from above**;
- It contains **more traces than possible**. These additional traces would yield the **same** abstraction anyway!

Set-based abstraction

Let us call this abstraction the **set-based abstraction**:



4. Introduction to abstraction soundness/completeness

Intuition for soundness

For a given class of properties, **soundness** means that:

- Any property of the abstract world (in the given class) must hold in the concrete world;

Intuition for soundness

For a given class of properties, **soundness** means that:

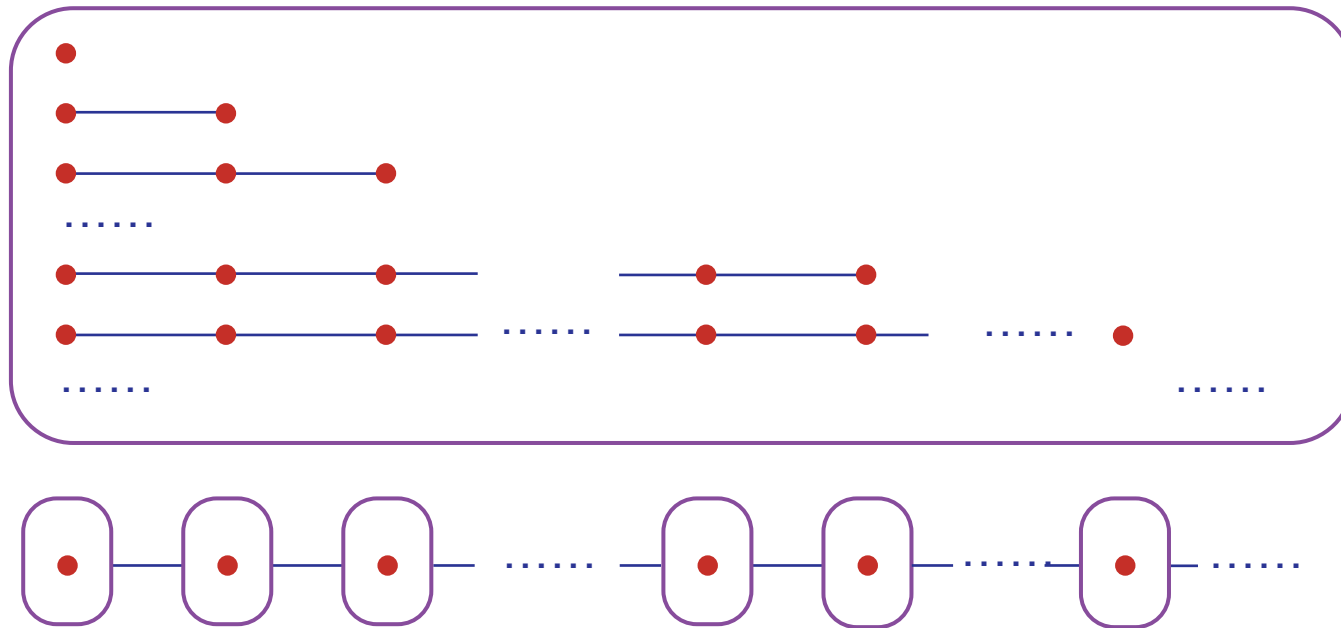
- Any property (in the given class) of the abstract world must hold in the concrete world;
- For the set-based abstraction:
 - Example: “on any trace, state a can never be immediately followed by state b ”;

Intuition for soundness

For a given class of properties, **soundness** means that:

- Any property (in the given class) of the abstract world must hold in the concrete world;
- For the set-based abstraction:
 - Example: “on any trace, state a can never be immediately followed by state b ”;
 - Counter-example: “all traces are infinite”;

Example for unsoundness



All abstract traces are infinite but not the concrete ones!

Intuition for completeness

For a given class of properties, **completeness** means that:

- Any property (in the given class) of the concrete world must hold in the abstract world;

Intuition for completeness

For a given class of properties, **completeness** means that:

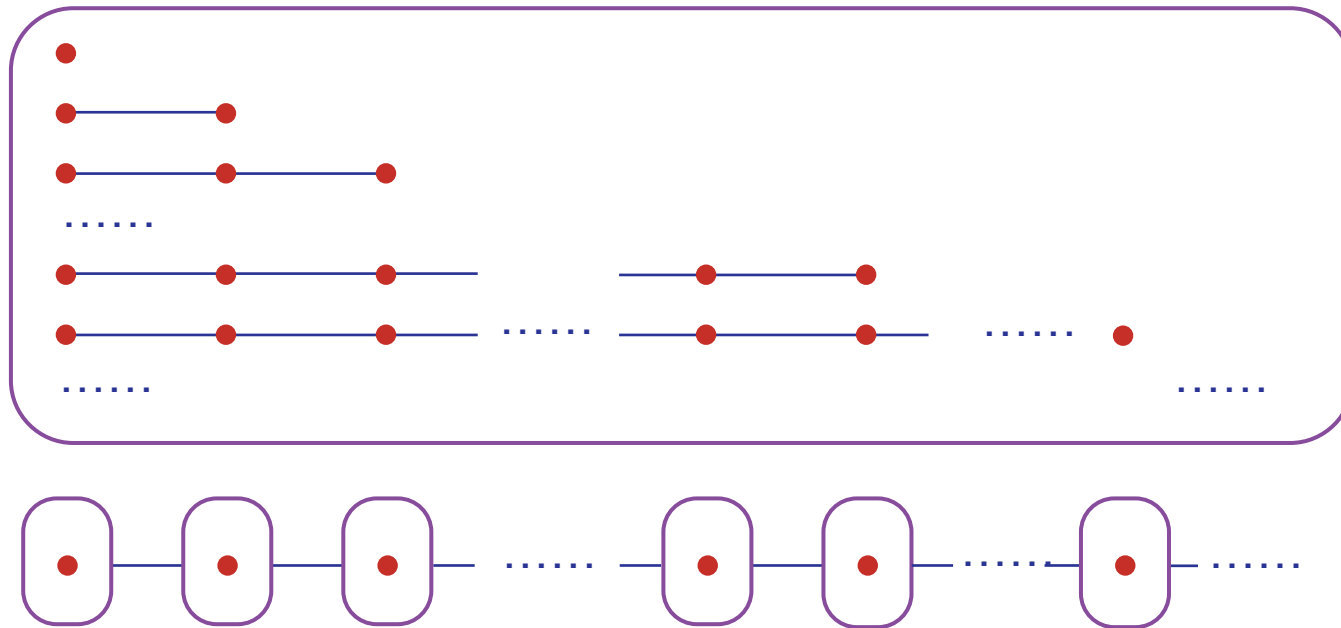
- Any property (in the given class) of the concrete world must hold in the abstract world;
- For the set-based abstraction:
 - Example: “execution from state a must eventually be followed by states b or c ”

Intuition for completeness

For a given class of properties, **completeness** means that:

- Any property (in the given class) of the concrete world must hold in the abstract world;
- For the set-based abstraction:
 - Example: “execution from state a must eventually be followed by states b or c ”
 - Counter-example: “all traces are finite”;

Example for uncompleteness



All concrete traces are finite but not the abstract ones!

4.1 Classical temporal-logics/calculi involve implicit abstractions

Temporal-logics/calculi involve implicit abstractions

- In general, temporal-logic/calculi cannot express all properties of models, but only specific ones (e.g. [3]);

Reference

[3] Emerson, E. & Halpern, J. “Sometimes” and “Not Never” revisited: On branching time versus linear time. *TOPLAS* 33 (1986), 151–178.

Temporal-logics/calculi involve implicit abstractions

- In general, temporal-logic/calculi cannot express all properties of models, but only specific ones (e.g. [3]);
- Some concrete properties of the model can only be approximated in the abstract temporal-logic/calculus;

Reference

[3] Emerson, E. & Halpern, J. “Sometimes” and “Not Never” revisited: On branching time versus linear time. *TOPLAS* 33 (1986), 151–178.

Temporal-logics/calculi involve implicit abstractions

- In general, temporal-logic/calculi cannot express all properties of models, but only specific ones (e.g. [3]);
- Some concrete properties of the model can only be approximated in the abstract temporal-logic/calculus;
- The semantics of the temporal-logic/calculus can be understood as an abstraction of the concrete semantics (of the models).

Reference

[3] Emerson, E. & Halpern, J. “Sometimes” and “Not Never” revisited: On branching time versus linear time. *TOPLAS* 33 (1986), 151–178.

Abstraction closedness

- A temporal logic/calculus \mathcal{T} is closed for an abstraction $\alpha_{\mathcal{T}}$ iff this abstraction leaves all temporal specifications φ of \mathcal{T} invariant:

$$\alpha_{\mathcal{T}}(\varphi) = \varphi$$

Abstraction closedness

- A temporal logic/calculus \mathcal{T} is closed for an abstraction $\alpha_{\mathcal{T}}$ iff this abstraction leaves all temporal specifications φ of \mathcal{T} invariant:

$$\alpha_{\mathcal{T}}(\varphi) = \varphi$$

- For example Kozen's propositional μ -calculus is closed for the set-based abstraction.

Unexpressivity

- The concrete properties P such that $\alpha_{\mathcal{T}}(P) \neq P$ cannot be expressed by the temporal logic/calculus \mathcal{T} ;

Unexpressivity

- The concrete properties P such that $\alpha_{\mathcal{T}}(P) \neq P$ cannot be expressed by the temporal logic/calculus \mathcal{T} ;
- So e.g. the propositional μ -calculus is not expressive enough to capture all concrete models (as expressible e.g. by the reversible $\hat{\mu}^*$ -calculus);

5. Is model checking a complete temporal abstract interpretation?

Current state abstraction

- The model-checking algorithms for the propositional μ -calculus can be derived by classical abstract interpretation techniques⁵ with the *current state abstraction*⁶:

$$\alpha^*(M) \triangleq \{\sigma_i \mid \langle i, \sigma \rangle \in M\}$$

$$\gamma^*(S) \triangleq \{\langle i, \sigma \rangle \mid \sigma_i \in S\} = \sigma_S$$

⁵ Galois connections, fixpoint transfert(/approximations), chaotic iterations, etc.

⁶ or the corresponding boolean characteristic functions represented e.g. as BDDs.

Current state abstraction

- The model-checking algorithms for the propositional μ -calculus can be derived by classical abstract interpretation techniques⁵ with the *current state abstraction*⁶:

$$\alpha^*(M) \triangleq \{\sigma_i \mid \langle i, \sigma \rangle \in M\}$$

$$\gamma^*(S) \triangleq \{\langle i, \sigma \rangle \mid \sigma_i \in S\} = \sigma_S$$

\implies Model-checking is a sound and complete abstract interpretation.

⁵ Galois connections, fixpoint transfert(/approximations), chaotic iterations, etc.

⁶ or the corresponding boolean characteristic functions represented e.g. as BDDs.

Not convincing!

- The completeness result is **relative** to the set-based abstraction closed semantics of the propositional μ -calculus!

Not convincing!

- The completeness result is **relative** to the set-based abstraction closed semantics of the propositional μ -calculus!
- The completeness is **relative** to the abstract world **not** to the concrete world!

Not convincing!

- The completeness result is **relative** to the set-based abstraction closed semantics of the propositional μ -calculus!
- The completeness is **relative** to the abstract world **not** to the concrete world!
- This set-based abstraction is itself **incomplete** (e.g. for the reversible $\hat{\mu}^*$ -calculus);

Not convincing!

- The completeness result is **relative** to the set-based abstraction closed semantics of the propositional μ -calculus!
- The completeness is **relative** to the abstract world **not** to the concrete world!
- This **set-based abstraction** is itself **incomplete** (e.g. for the reversible $\hat{\mu}^*$ -calculus);
- **Intuition:** with general temporal specifications, model-checking algorithms could not only deal with **sets of states** only and would have to handle **sets of traces** (which would be too costly).

6. Model checking is an incomplete temporal abstract interpretation!

Universal checking abstraction

- State projection:

$$M_{\downarrow s} \triangleq \{ \langle i, \sigma \rangle \in M \mid \sigma_i = s \}$$

- Universal checking abstraction:

$$\alpha_M^\forall(\phi) \triangleq \{ s \mid M_{\downarrow s} \subseteq \phi \}$$

- Universal checking concretization:

$$\gamma_M^\forall(S) \triangleq \{ \langle i, \sigma \rangle \mid \langle i, \sigma \rangle \in M \wedge \sigma_i \in S \}$$

- Galois connection:

$$\langle \mathbb{M}, \supseteq \rangle \xrightleftharpoons[\alpha_M^\forall]{\gamma_M^\forall} \langle \wp(\mathbb{S}), \supseteq \rangle$$

Existential checking abstraction

- Dual existential checking abstraction:

$$\begin{aligned}\alpha_M^\exists(\phi) &\triangleq \neg \alpha_M^\forall(\neg \phi) \\ &= \{s \mid (M_{\downarrow s} \cap \phi) \neq \emptyset\}\end{aligned}$$

- Existential checking concretization:

$$\begin{aligned}\gamma_M^\exists(\phi) &\triangleq \neg \gamma_M^\forall(\neg \phi) \\ &= \{\langle i, \sigma \rangle \mid (\langle i, \sigma \rangle \in M) \implies (\sigma_i \in S)\}\end{aligned}$$

- Galois connection:

$$\langle \mathbb{M}, \subseteq \rangle \begin{array}{c} \xleftarrow{\gamma_M^\exists} \\ \xrightarrow{\alpha_M^\exists} \end{array} \langle \wp(S), \subseteq \rangle$$

Model checking (algorithms)

- Fix the model M to be generated by a transition relation (Kripke structure)⁷:

$$M = \boxplus \pi_t$$

⁷ may be with fairness conditions also expressible with the reversible $\hat{\mu}^*$ -calculus.

Model checking (algorithms)

- Fix the **model** M to be generated by a transition relation (Kripke structure)⁷:

$$M = \boxplus \pi_t$$

- Define the abstraction by structural inductively on formulae:
 - **Basic temporal operators** are defined by universal/existential abstraction of concrete ones ;

⁷ may be with fairness conditions also expressible with the reversible $\hat{\mu}^*$ -calculus.

Model checking (algorithms)

- Fix the **model** M to be generated by a transition relation (Kripke structure)⁷:

$$M = \boxplus \pi_t$$

- Define the abstraction by structural inductively on formulae:
 - **Basic temporal operators** are defined by universal/existential abstraction of concrete ones ;
 - **Inductive combination** with abstraction closed operations (e.g. join, meet, complement, fixpoints, etc.).

⁷ may be with fairness conditions also expressible with the reversible $\hat{\mu}$ -calculus.

Example: propositional μ -calculus

- $\alpha_{\pi_t}^{\forall}(\sigma_S) = \alpha_{\pi_t}^{\exists}(\sigma_S) = S$;
- If φ_1 and φ_2 are $\alpha_{\pi_t}^{\forall}$ or $\alpha_{\pi_t}^{\exists}$ abstraction closed, i.e.:
$$\alpha_{\pi_t}(\varphi_i) = \varphi_i, i = 1, \dots, 2$$

then so are:

$$\varphi_1 \vee \varphi_2, \varphi_1 \wedge \varphi_2, \neg \varphi_1, \Box \varphi_1, \Diamond \varphi_1, \mu X \cdot \varphi_1, \\ \nu X \cdot \varphi_1.$$

7. Abstraction completeness for sub-logics and calculi

Model checking algorithms incompleteness

- The classical model-checking algorithms are set of states based (but not set of traces based);

Model checking algorithms incompleteness

- The classical model-checking algorithms are set of states based (but not set of traces based);
- In general, this abstraction is incomplete (e.g. for the full complete reversible $\hat{\mu}^*$ -calculus);

Model checking algorithms incompleteness

- The classical model-checking algorithms are set of states based (but not set of traces based);
- In general, this abstraction is incomplete (e.g. for the full complete reversible $\hat{\mu}^*$ -calculus);
- We can identify sub-calculi (whence logics) for which the model-checking abstractions are complete;

Example: μ_+^\forall -calculus

$\psi ::= \sigma_S \mid \psi_1 \vee \psi_2 \mid \psi_1 \wedge \psi_2 \mid \neg \psi_1 \mid \forall \varphi$

state formulae

$\varphi ::= \psi \mid \pi_t \mid \oplus \varphi_1 \mid \varphi_1 \wedge \varphi_2 \mid$

path formulae

$\psi_1 \vee \varphi_2 \mid \varphi_1 \vee \psi_2 \mid X \mid \mu X \cdot \varphi_1 \mid \nu X \cdot \varphi_1$

(with $\forall \varphi \triangleq \forall \boxplus \pi_t : \varphi$) is such that:

$$[\varphi]^\forall = \vec{\alpha}_{\mathcal{M}_\tau}^\forall([\varphi])$$

- This covers $\forall\text{CTL}$ (but not $\forall\text{CTL}^\star$);
- Same for $\exists\text{CTL}$ (but not $\exists\text{CTL}^\star$) and inductive combination with joins, ... to get completeness for CTL (but not CTL^\star).

8. Conclusion

More in the forthcoming POPL'00 paper ...

- Compositional **abstract interpretation** of generic μ -calculi (independently of a particular semantics, including for non-monotone operators);
- Study of the **model-checking abstractions**;
- Study of (sufficient) abstraction **completeness conditions**;
- **Applications** to:
 - Abstract model checking;
 - Dataflow analysis (and the soundness of live variables).

Perspectives

- Anyway, model-checking is an **incomplete** abstract interpretation;
- So for **infinite state systems**:

Perspectives

- Anyway, model-checking is an **incomplete** abstract interpretation;
- So for **infinite state systems**:
 - **other abstractions** can be used (e.g. as in abstract testing);

Perspectives

- Anyway, model-checking is an **incomplete** abstract interpretation;
- So for **infinite state systems**:
 - **other abstractions** can be used (e.g. as in abstract testing);
 - because of incompleteness, **other algorithms** should be used [4] (the common model-checking algorithms are not the most precise ones).

Reference

[4] P. Cousot and R. Cousot. Abstract interpretation and application to logic programs. *Journal of Logic Programming*, 13(2–3):103–179, 1992.