The Impact of Craig's Interpolation Theorem

in Computer Science

Cesare Tinelli

tinelli@cs.uiowa.edu

The University of Iowa

The Role of Logic in Computer Science

Mathematical logic is central to Computer Science

The Role of Logic in Computer Science

Mathematical logic is central to Computer Science

It provides formal foundations for

- Programming languages
- Relational databases
- Computational complexity
- Hardware design and validation
- Formal methods in software engineering
- Artificial Intelligence
- . . .

Craig's Interpolation Theorem in CS

- has had a strong and lasting impact in several CS areas, both at the theoretical and the practical level

Craig's Interpolation Theorem in CS

- has had a strong and lasting impact in several CS areas, both at the theoretical and the practical level
- has been generalized to many other logics used in CS (sorted, equational, modal, intuitionistic, . . .)

Craig's Interpolation Theorem in CS

- has had a strong and lasting impact in several CS areas, both at the theoretical and the practical level
- has been generalized to many other logics used in CS (sorted, equational, modal, intuitionistic, . . .)
- together with compactness, is considered a crucial property of any new logic for CS

Craig's Interpolation Theorem in CS

- has had a strong and lasting impact in several CS areas, both at the theoretical and the practical level
- has been generalized to many other logics used in CS (sorted, equational, modal, intuitionistic, . . .)
- together with compactness, is considered a crucial property of any new logic for CS
- comes up in any formal method based on modular decomposition of complex systems

Craig's Interpolation

Some applications:

- **Hardware/software specification** (Diaconescu et al.,'93, Rosu & Goguen, 2000, Bicarregui et al., 2000)

Craig’s Interpolation

Some applications:

- **Hardware/software specification** (Diaconescu et al.,’93, Rosu & Goguen, 2000, Bicarregui et al., 2000)
- **Reasoning with large knowledge bases** (Amir & McIlraith, 2005)

Craig's Interpolation

Some applications:

- **Hardware/software specification** (Diaconescu et al.,'93, Rosu & Goguen, 2000, Bicarregui et al., 2000)
- **Reasoning with large knowledge bases** (Amir & McIlraith, 2005)
- **Type inference** (Jhala et al., 2007)

Craig's Interpolation

Some applications:

- **Hardware/software specification** (Diaconescu et al.,'93, Rosu & Goguen, 2000, Bicarregui et al., 2000)
- **Reasoning with large knowledge bases** (Amir & McIlraith, 2005)
- **Type inference** (Jhala et al., 2007)
- **Combination of theorem provers for different theories** (Nelson& Oppen, 1979; Tinelli, 2003; Ghilardi, 2005)

Craig's Interpolation

Some applications:

- **Hardware/software specification** (Diaconescu et al.,'93, Rosu & Goguen, 2000, Bicarregui et al., 2000)
- **Reasoning with large knowledge bases** (Amir & McIlraith, 2005)
- **Type inference** (Jhala et al., 2007)
- **Combination of theorem provers for different theories** (Nelson& Oppen, 1979; Tinelli, 2003; Ghilardi, 2005)
- **Model checking of finite- and infinite-state systems** (McMillan, 2003, Henzinger et al., 2004)

Craig's Interpolation

Some applications:

- **Hardware/software specification** (Diaconescu et al.,'93, Rosu & Goguen, 2000, Bicarregui et al., 2000)
- **Reasoning with large knowledge bases** (Amir & McIlraith, 2005)
- **Type inference** (Jhala et al., 2007)
- **Combination of theorem provers for different theories** (Nelson& Oppen, 1979; Tinelli, 2003; Ghilardi, 2005)
- **Model checking of finite- and infinite-state systems** (McMillan, 2003, Henzinger et al., 2004)

The Essence of Craig's Interpolation for CS

Craig's Interpolation: If φ_1 and φ_2 are inconsistent, there is a φ in their shared language such that

$$\varphi_1 \models \psi \quad \text{and} \quad \psi \wedge \varphi_2 \text{ is inconsistent.}$$

The Essence of Craig's Interpolation for CS

Craig's Interpolation: If φ_1 and φ_2 are inconsistent, there is a φ in their shared language such that

$$\varphi_1 \models \psi \quad \text{and} \quad \psi \wedge \varphi_2 \text{ is inconsistent.}$$

Intuitively,

- ψ is an abstraction of φ_1 from the viewpoint of φ_2;
- ψ summarizes and translates in the shared language why φ_1 is inconsistent with φ_2.

Part I: Craig Interpolation for Prover Combinations

Satisfiability Modulo Theories

- In some areas of Computer Science, one is interested in the satisfiability in a particular theory of certain classes of formulas.

Satisfiability Modulo Theories

- In some areas of Computer Science, one is interested in the satisfiability in a particular theory of certain classes of formulas.
  - Microprocessors design: theory of equality, atoms like $f(g(a, b), c) = g(c, a)$.

Satisfiability Modulo Theories

- In some areas of Computer Science, one is interested in the satisfiability in a particular theory of certain classes of formulas.
  - Microprocessors design: theory of equality, atoms like $f(g(a,b),c) = g(c,a)$.
  - Timed automata, planning: theory of integers/reals, atoms like $x - y < 2$.

Satisfiability Modulo Theories

- In some areas of Computer Science, one is interested in the satisfiability in a particular theory of certain classes of formulas.
  - Microprocessors design: theory of equality, atoms like $f(g(a, b), c) = g(c, a)$.
  - Timed automata, planning: theory of integers/reals, atoms like $x - y < 2$.
  - Software verification/model checking: combination of theories, atoms like $5 + \mathit{first}((x + 2) :: l) = a[j] + 1$.

Satisfiability Modulo Theories

- In some areas of Computer Science, one is interested in the satisfiability in a particular theory of certain classes of formulas.
  - Microprocessors design: theory of equality, atoms like $f(g(a, b), c) = g(c, a)$.
  - Timed automata, planning: theory of integers/reals, atoms like $x - y < 2$.
  - Software verification/model checking: combination of theories, atoms like $5 + \mathit{first}((x + 2) :: l) = a[j] + 1$.
- We refer to this general problem as Satisfiability Modulo Theories, or SMT.

Satisfiability Modulo Theories

- In some areas of Computer Science, one is interested in the satisfiability in a particular theory of certain classes of formulas.
  - Microprocessors design: theory of equality, atoms like $f(g(a,b),c) = g(c,a)$.
  - Timed automata, planning: theory of integers/reals, atoms like $x - y < 2$.
  - Software verification/model checking: combination of theories, atoms like $5 + \mathit{first}((x+2) :: l) = a[j] + 1$.
- We refer to this general problem as Satisfiability Modulo Theories, or SMT.

Satisfiability Modulo Theories

Let T be a first-order theory of signature Σ and $\mathcal{L}^\Sigma$ a class of Σ-formulas.

Satisfiability Modulo Theories

Let T be a first-order theory of signature Σ and $\mathcal{L}^{\Sigma}$ a class of Σ-formulas.

The T-satisfiability problem for $\mathcal{L}^{\Sigma}$ consists in deciding for any formula $\varphi[\mathbf{x}] \in \mathcal{L}^{\Sigma}$ whether $T \cup \{\exists \mathbf{x}.\, \varphi\}$ is satisfiable.

Satisfiability Modulo Theories

Let T be a first-order theory of signature Σ and $\mathcal{L}^\Sigma$ a class of Σ-formulas.

The T-satisfiability problem for $\mathcal{L}^\Sigma$ consists in deciding for any formula $\varphi[\mathbf{x}] \in \mathcal{L}^\Sigma$ whether $T \cup \{\exists \mathbf{x}.\, \varphi\}$ is satisfiable.

Some relevant theories in SMT

- Equality with "Uninterpreted Function Symbols"
- Linear Arithmetic (Real and Integer)
- Arrays (i.e., updatable maps)
- Bit vectors
- Finite trees

Solving Combined SMT Problems

For many theories T and some formula classes $\mathcal{L}$ there exist (efficient) decision procedures for the T-satisfiability problem for $\mathcal{L}^{\Sigma}$.

Solving Combined SMT Problems

For many theories T and some formula classes $\mathcal{L}$ there exist (efficient) decision procedures for the T-satisfiability problem for $\mathcal{L}^{\Sigma}$.

Problem: In practice, we often need to deal with *mixed* formulas in $\mathcal{L}^{\Sigma_1 \cup \cdots \cup \Sigma_n}$ modulo a *combined theory* $T_1 \cup \cdots \cup T_n$.

Solving Combined SMT Problems

For many theories T and some formula classes $\mathcal{L}$ there exist (efficient) decision procedures for the T-satisfiability problem for $\mathcal{L}^{\Sigma}$.

Problem: In practice, we often need to deal with *mixed* formulas in $\mathcal{L}^{\Sigma_1 \cup \dots \cup \Sigma_n}$ modulo a *combined theory* $T_1 \cup \dots \cup T_n$.

In that case, it helps if we can

> combine modularly decision procedures for the individual $T_1, \dots, T_n$ into a decision procedure for $T_1 \cup \dots \cup T_n$.

The General Combined Satisfiability Problem

For $i = 1, 2$,

- let T_i a first-order theory of signature Σ_i and
- let $\mathcal{L}^{\Sigma_i}$ be a class of Σ_i-formulas

such that the T_i-satisfiability problem for $\mathcal{L}^{\Sigma_i}$ is decidable.

The General Combined Satisfiability Problem

For $i = 1, 2$,

- let T_i a first-order theory of signature Σ_i and
- let $\mathcal{L}^{\Sigma_i}$ be a class of Σ_i-formulas

such that the T_i-satisfiability problem for $\mathcal{L}^{\Sigma_i}$ is decidable.

Combination methods apply to languages $\mathcal{L}^{\Sigma_1 \cup \Sigma_2}$ that are effectively purifiable for T_1 and T_2

The General Combined Satisfiability Problem

For $i = 1, 2$,

- let T_i a first-order theory of signature Σ_i and
- let $\mathcal{L}^{\Sigma_i}$ be a class of Σ_i-formulas

such that the T_i-satisfiability problem for $\mathcal{L}^{\Sigma_i}$ is decidable.

Combination methods apply to languages $\mathcal{L}^{\Sigma_1 \cup \Sigma_2}$ that are effectively purifiable for T_1 and T_2 , i.e., such that

the $(T_1 \cup T_2)$-satisfiability of a formula $\varphi \in \mathcal{L}^{\Sigma_1 \cup \Sigma_2}$
is effectively reducible to
the $(T_1 \cup T_2)$-satisfiability of formulas of the form $\varphi_1 \wedge \varphi_2$ with $\varphi_i \in \mathcal{L}^{\Sigma_i}$ for $i = 1, 2$.

The General Combined Satisfiability Problem

For $i = 1, 2$,

- let T_i a first-order theory of signature Σ_i and
- let $\mathcal{L}^{\Sigma_i}$ be a class of Σ_i-formulas

such that the T_i-satisfiability problem for $\mathcal{L}^{\Sigma_i}$ is decidable.

Combination methods apply to languages $\mathcal{L}^{\Sigma_1 \cup \Sigma_2}$ that are effectively purifiable for T_1 and T_2

Observation: For purifiable languages, $(T_1 \cup T_2)$-satisfiability is at heart an interpolation problem.

Combined Satisfiability as Interpolation

For $i = 1, 2$, let T_i-be a Σ_i-theory and $\varphi_i[\mathbf{x}_i]$ a Σ_i-formula.

$\varphi_1 \wedge \varphi_2$ is $(T_1 \cup T_2)$-unsatisfiable

Combined Satisfiability as Interpolation

For $i = 1, 2$, let T_i-be a Σ_i-theory and $\varphi_i[\mathbf{x}_i]$ a Σ_i-formula.

$\varphi_1 \wedge \varphi_2$ is $(T_1 \cup T_2)$-unsatisfiable

iff

$$T_1, \varphi_1, T_2, \varphi_2 \models \bot$$

Combined Satisfiability as Interpolation

For $i = 1, 2$, let T_i-be a Σ_i-theory and $\varphi_i[\mathbf{x}_i]$ a Σ_i-formula.

$\varphi_1 \wedge \varphi_2$ is $(T_1 \cup T_2)$-unsatisfiable

iff

$$T_1, \varphi_1, T_2, \varphi_2 \models \bot$$

iff, by an application of Craig's interpolation theorem,

there is a $(\Sigma_1 \cap \Sigma_2)$-formula $\varphi(\mathbf{x})$ with $\mathbf{x} = \mathbf{x}_1 \cap \mathbf{x}_2$ s.t.

$$T_1, \varphi_1 \models \varphi \quad \text{and} \quad T_2, \varphi_2, \varphi \models \bot$$

Combined Satisfiability as Interpolation

For $i = 1, 2$, let T_i-be a Σ_i-theory and $\varphi_i[\mathbf{x}_i]$ a Σ_i-formula.

$$\varphi_1 \wedge \varphi_2 \text{ is } (T_1 \cup T_2)\text{-unsatisfiable}$$

iff

$$T_1, \varphi_1, T_2, \varphi_2 \models \bot$$

iff, by an application of Craig's interpolation theorem,

there is a $(\Sigma_1 \cap \Sigma_2)$-formula $\varphi(\mathbf{x})$ with $\mathbf{x} = \mathbf{x}_1 \cap \mathbf{x}_2$ s.t.

$$T_1, \varphi_1 \models \varphi \quad \text{and} \quad T_2, \varphi_2, \varphi \models \bot$$

The problem then is "just" computing the interpolant φ.

Combined Satisfiability as Interpolation

For $i = 1, 2$, let T_i-be a Σ_i-theory and $\varphi_i[\mathbf{x}_i]$ a Σ_i-formula.

$\varphi_1 \wedge \varphi_2$ is $(T_1 \cup T_2)$-unsatisfiable

iff

$$T_1, \varphi_1, T_2, \varphi_2 \models \bot$$

iff, by an application of Craig's interpolation theorem,

there is a $(\Sigma_1 \cap \Sigma_2)$-formula $\varphi(\mathbf{x})$ with $\mathbf{x} = \mathbf{x}_1 \cap \mathbf{x}_2$ s.t.

$$T_1, \varphi_1 \models \varphi \quad \text{and} \quad T_2, \varphi_2, \varphi \models \bot$$

All existing combination methods are in essence ways to compute φ, possibly incrementally, in finite time, without building a direct proof that $T_1, \varphi_1, T_2, \varphi_2 \models \bot$

Combined Satisfiability as Interpolation

For $i = 1, 2$, let T_i-be a Σ_i-theory and $\varphi_i[\mathbf{x}_i]$ a Σ_i-formula.

$\varphi_1 \wedge \varphi_2$ is $(T_1 \cup T_2)$-unsatisfiable

iff

$$T_1, \varphi_1, T_2, \varphi_2 \models \bot$$

iff, by an application of Craig's interpolation theorem,

there is a $(\Sigma_1 \cap \Sigma_2)$-formula $\varphi(\mathbf{x})$ with $\mathbf{x} = \mathbf{x}_1 \cap \mathbf{x}_2$ s.t.

$$T_1, \varphi_1 \models \varphi \quad \text{and} \quad T_2, \varphi_2, \varphi \models \bot$$

Historical note: The original correctness proof of the foremost combination method for SMT (Nelson & Oppen, 1979) relies directly on Craig's interpolation theorem.

An Effectively Purifiable Language

The class of quantifier-free formulas is effectively purifiable for any Σ_1 and Σ_2

An Effectively Purifiable Language

The class of quantifier-free formulas is effectively purifiable for any Σ_1 and Σ_2 :

Given a quantifier-free $(\Sigma_1 \cup \Sigma_2)$-formula φ

we can compute Σ_1-qffs $\varphi_1^1 \ldots \varphi_1^n$ and Σ_2-qffs $\varphi_2^1 \ldots \varphi_2^n$ s.t.

for every $(\Sigma_1 \cup \Sigma_2)$-structure $\mathcal{A}$,

φ is satisfiable in $\mathcal{A}$ iff $\varphi_1^j \wedge \varphi_2^j$ is satisfiable in $\mathcal{A}$ for some j.

An Effectively Purifiable Language

The class of quantifier-free formulas is effectively purifiable for any Σ_1 and Σ_2 .

Moreover, the T-satisfiability problem for qffs is decidable for a very large number of theories of interest in CS.

An Effectively Purifiable Language

The class of quantifier-free formulas is effectively purifiable for any Σ_1 and Σ_2 .

Moreover, the T-satisfiability problem for qffs is decidable for a very large number of theories of interest in CS.

Let's focus then on quantifier-free formulas.

An Effectively Purifiable Language

The class of quantifier-free formulas is effectively purifiable for any Σ_1 and Σ_2 .

Moreover, the T-satisfiability problem for qffs is decidable for a very large number of theories of interest in CS.

Let's focus then on quantifier-free formulas.

For simplicity, but wlog, let's consider only combined satisfiability problems of the form

$$\Gamma_1 \cup \Gamma_2$$

where each Γ_i is a finite set of Σ_i-*literals*
(i.e., atomic formulas and negated atomic formulas)

The Combined Satisfiability Problem for QFFs

For $i = 1, 2$, let T_i-be a Σ_i-theory and $\Gamma_i[\mathbf{x}_i]$ a set of Σ_i-literals.

The Combined Satisfiability Problem for QFFs

For $i = 1, 2$, let T_i-be a Σ_i-theory and $\Gamma_i[\mathbf{x}_i]$ a set of Σ_i-literals.

Let $\psi_1, \ldots, \psi_n$ be $(\Sigma_1 \cap \Sigma_2)$-formulas over $\mathbf{x} = \mathbf{x}_1 \cap \mathbf{x}_2$.

The Combined Satisfiability Problem for QFFs

For $i = 1, 2$, let T_i-be a Σ_i-theory and $\Gamma_i[\mathbf{x}_i]$ a set of Σ_i-literals.

Let $\psi_1, \ldots, \psi_n$ be $(\Sigma_1 \cap \Sigma_2)$-formulas over $\mathbf{x} = \mathbf{x}_1 \cap \mathbf{x}_2$.

$\psi_1, \ldots, \psi_n$ is an *interpolation chain* if for each $k = 1, \ldots, m$ there is an $i \in \{1, 2\}$ s.t.

$$T_i, \Gamma_i, \psi_1, \ldots, \psi_{k-1} \models \psi_k$$

The Combined Satisfiability Problem for QFFs

For $i = 1, 2$, let T_i-be a Σ_i-theory and $\Gamma_i[\mathbf{x}_i]$ a set of Σ_i-literals.

Let $\psi_1, \ldots, \psi_n$ be $(\Sigma_1 \cap \Sigma_2)$-formulas over $\mathbf{x} = \mathbf{x}_1 \cap \mathbf{x}_2$.

$\psi_1, \ldots, \psi_n$ is an *interpolation chain* if for each $k = 1, \ldots, m$ there is an $i \in \{1, 2\}$ s.t.

$$T_i, \Gamma_i, \psi_1, \ldots, \psi_{k-1} \models \psi_k$$

Under the right conditions:

1. $\Gamma_1 \cup \Gamma_2$ is $(T_1 \cup T_2)$-unsatisfiable iff there is an interpolation chain $\psi_1, \ldots, \psi_m$ with $\psi_n = \bot$, and
2. each ψ_i is a disjunction of atoms and is computable using one of the decision procedures for T_1 and T_2.

The Combined Satisfiability Problem for QFFs

Sufficient conditions on T_1 and T_2 (Ghilardi, 2005)

Where $\Sigma_0 = \Sigma_1 \cap \Sigma_2$, there is a universal Σ_0-theory T_0 that is:

The Combined Satisfiability Problem for QFFs

Sufficient conditions on T_1 and T_2 (Ghilardi, 2005)

Where $\Sigma_0 = \Sigma_1 \cap \Sigma_2$, there is a universal Σ_0-theory T_0 that is:

1. T_i-*compatible* for $i = 1, 2$:
   (a) is enclosed in T_i
   (b) admits a model completion T_0^*
   (c) every model of T_i embeds into a model of $T_i \cup T_0^*$

The Combined Satisfiability Problem for QFFs

Sufficient conditions on T_1 and T_2 (Ghilardi, 2005)

Where $\Sigma_0 = \Sigma_1 \cap \Sigma_2$, there is a universal Σ_0-theory T_0 that is:

1. T_i-*compatible* for $i = 1, 2$:
   (a) is enclosed in T_i
   (b) admits a model completion T_0^*
   (c) every model of T_i embeds into a model of $T_i \cup T_0^*$
2. *effectively locally finite*:
   For any $\mathbf{x}$ we can compute a set $\{t_1, \ldots t_n\}$ of Σ_0-terms over $\mathbf{x}$ s.t. every Σ_0-term $t[\mathbf{x}]$ is T_0-equivalent to some t_i

The Combined Satisfiability Problem for QFFs

Sufficient conditions on T_1 and T_2 (Ghilardi, 2005)

Where $\Sigma_0 = \Sigma_1 \cap \Sigma_2$, there is a universal Σ_0-theory T_0 that is:

1. T_i-*compatible* for $i = 1, 2$:
   (a) is enclosed in T_i
   (b) admits a model completion T_0^*
   (c) every model of T_i embeds into a model of $T_i \cup T_0^*$
2. *effectively locally finite*:
   For any $\mathbf{x}$ we can compute a set $\{t_1, \ldots t_n\}$ of Σ_0-terms over $\mathbf{x}$ s.t. every Σ_0-term $t[\mathbf{x}]$ is T_0-equivalent to some t_i

Nelson-Oppen Method: $\Sigma_0 = \emptyset$ and each T_i is *stably infinite*.

Stably Infinite Theories

A Σ-theory T is stably infinite iff every quantifier-free T-satisfiable formula is satisfiable in an infinite model of T.

Stably Infinite Theories

A Σ-theory T is stably infinite iff every quantifier-free T-satisfiable formula is satisfiable in an infinite model of T.

Many *interesting* theories are stably infinite:

- Theories of an infinite structure.
- Complete theories with an infinite model.
- Convex theories with no trivial models.

Stably Infinite Theories

A Σ-theory T is stably infinite iff every quantifier-free T-satisfiable formula is satisfiable in an infinite model of T.

Many *interesting* theories are stably infinite:

- Theories of an infinite structure.
- Complete theories with an infinite model.
- Convex theories with no trivial models.

But others are not:

- Theories of a finite structure.
- Theories with models of bounded cardinality.
- Some equational/Horn theories.

Beyond Stable Infiniteness

- Recent extensions of the Nelson-Oppen method partially lift the stable infiniteness requirement
(Tinelli & Zarba, 2004; Ranise et al., 2005)

Beyond Stable Infiniteness

- Recent extensions of the Nelson-Oppen method partially lift the stable infiniteness requirement
  (Tinelli & Zarba, 2004; Ranise et al., 2005)
- The trick is to require the decision procedures to also exchange finite-cardinality constraints.

Beyond Stable Infiniteness

- Recent extensions of the Nelson-Oppen method partially lift the stable infiniteness requirement
(Tinelli & Zarba, 2004; Ranise et al., 2005)
- The trick is to require the decision procedures to also exchange finite-cardinality constraints.
- These extensions are still instances of Craig interpolation.

Beyond Stable Infiniteness

- Recent extensions of the Nelson-Oppen method partially lift the stable infiniteness requirement (Tinelli & Zarba, 2004; Ranise et al., 2005)
- The trick is to require the decision procedures to also exchange finite-cardinality constraints.
- These extensions are still instances of Craig interpolation.
- However, they now consider interpolation chains that also include quantified formulas like

$$\forall x, y, z.\ x = y \vee x = z$$

The Combined Satisfiability Problem for QFFs

- SMT provers based on some variant of the Nelson-Oppen method are widely used in academia and industry.

The Combined Satisfiability Problem for QFFs

- SMT provers based on some variant of the Nelson-Oppen method are widely used in academia and industry.

- The generalized results by Ghilardi have several additional applications.

  For instance, they can be used in the combination of modals logics.

Part II: Craig Interpolation in Model Checking

Modeling Computer Systems

Software or hardware systems can be often modeled as *state transition systems* $\mathcal{M} = (S, I, R, L)$ where

- S is a set of *states*
- $I \subseteq S$ is a set of *initial states*
- $R \subseteq S \times S$ is a total *transition relation*
- $L : S \to 2^{At}$ is a *labelling function* into sets of atomic formulas in some base logic

Modeling Computer Systems

Software or hardware systems can be often modeled as *state transition systems* $\mathcal{M} = (S, I, R, L)$ where

- S is a set of *states*
- $I \subseteq S$ is a set of *initial states*
- $R \subseteq S \times S$ is a total *transition relation*
- $L : S \rightarrow 2^{At}$ is a *labelling function* into sets of atomic formulas in some base logic

Note: $\mathcal{M}$ is a Kripke model (in the sense modal logic).

Model Checking

Software or hardware systems can be often modeled as *state transition systems*, or *model*, $\mathcal{M} = (S, I, R, L)$

Model Checking

Software or hardware systems can be often modeled as *state transition systems*, or *model*, $\mathcal{M} = (S, I, R, L)$

Most system correctness properties can be expressed as a *safety* property for a suitable model $\mathcal{M}$:

> $\mathcal{M}$ is *safe* wrt a property ψ if no state R-reachable from an initial state satisfies ψ.

Model Checking

Software or hardware systems can be often modeled as *state transition systems*, or *model*, $\mathcal{M} = (S, I, R, L)$

Most system correctness properties can be expressed as a *safety* property for a suitable model $\mathcal{M}$:

> $\mathcal{M}$ is *safe* wrt a property ψ if no state R-reachable from an initial state satisfies ψ.

Model checking is one of the most successful areas of formal verification.

Model checking technologies are now routinely used in industry.

Symbolic Model Checking

A model $\mathcal{M} = (S,\ I,\ R,\ L{:}S \rightarrow 2^{At})$ can be expressed symbolically by fixing a set X of variables and a first-order Σ-structure $\mathcal{A}$ with universe A.

Symbolic Model Checking

A model $\mathcal{M} = (S,\ I,\ R,\ L{:}S \to 2^{At})$ can be expressed symbolically by fixing a set X of variables and a first-order Σ-structure $\mathcal{A}$ with universe A.

Then:

- Every state $\sigma \in S$ is a mapping in $[X \to A]$
- At is a set of atomic Σ-formulas over X
- I is characterized by a qff $\varphi_I[\mathbf{x}]$ s.t. $\sigma \in I$ iff $\mathcal{A} \models \varphi_I[\sigma]$
- R is characterized by a qff $\varphi_R[\mathbf{x}, \mathbf{x}']$ such that $(\sigma, \sigma') \in R$ iff $\mathcal{A} \models \varphi_R[\sigma, \sigma']$

Notation: if $\mathbf{x} = x_1, \ldots, x_n$ then $\psi[\sigma] = \psi[\sigma(x_1), \ldots, \sigma(x_n)]$

Some Terminology

- A state σ is *reachable (in k steps)* iff there is a sequence of states $\sigma_0, \ldots, \sigma_k = \sigma$ such that

$$\mathcal{A} \models \varphi_I[\sigma_0] \wedge \varphi_R[\sigma_0, \sigma_1] \wedge \cdots \wedge \varphi_R[\sigma_{k-1}, \sigma_k]$$

- A formula $\psi[\mathbf{x}]$ is *reachable (in k steps)* from a formula $\varphi[\mathbf{x}]$ iff there is a sequence of states $\sigma_0, \ldots, \sigma_k = \sigma$ s.t.

$$\mathcal{A} \models \varphi[\sigma_0] \wedge \varphi_R[\sigma_0, \sigma_1] \wedge \cdots \wedge \varphi_R[\sigma_{k-1}, \sigma_k] \wedge \psi[\sigma_k]$$

Some Terminology

- A state σ is *reachable (in k steps)* iff there is a sequence of states $\sigma_0, \ldots, \sigma_k = \sigma$ such that

$$\mathcal{A} \models \varphi_I[\sigma_0] \wedge \varphi_R[\sigma_0, \sigma_1] \wedge \cdots \wedge \varphi_R[\sigma_{k-1}, \sigma_k]$$

- A formula $\psi[\mathbf{x}]$ is *reachable (in k steps)* from a formula $\varphi[\mathbf{x}]$ iff there is a sequence of states $\sigma_0, \ldots, \sigma_k = \sigma$ s.t.

$$\mathcal{A} \models \varphi[\sigma_0] \wedge \varphi_R[\sigma_0, \sigma_1] \wedge \cdots \wedge \varphi_R[\sigma_{k-1}, \sigma_k] \wedge \psi[\sigma_k]$$

Observation: $\mathcal{M}$ is safe wrt ψ iff ψ is not reachable from φ_I iff

$$\varphi_I[\mathbf{x}_0] \wedge \varphi_R[\mathbf{x}_0, \mathbf{x}_1] \wedge \cdots \wedge \varphi_R[\mathbf{x}_{k-1}, \mathbf{x}_k] \wedge \psi[\mathbf{x}_k]$$

is unsatisfiable in $\mathcal{A}$ for all $k \geq 0$.

Strongest Inductive Invariant

- For a large class of systems $\mathcal{M}$, we can compute from φ_I and φ_R the *strongest inductive invariant* φ_{IR} for $\mathcal{M}$:

Strongest Inductive Invariant

- For a large class of systems $\mathcal{M}$, we can compute from φ_I and φ_R the *strongest inductive invariant* φ_{IR} for $\mathcal{M}$:

  for all $\sigma \in S$, $\mathcal{A} \models \varphi_{IR}[\sigma]$ exactly when σ is reachable.

Strongest Inductive Invariant

- For a large class of systems $\mathcal{M}$, we can compute from φ_I and φ_R the *strongest inductive invariant* φ_{IR} for $\mathcal{M}$:

  for all $\sigma \in S$, $\mathcal{A} \models \varphi_{IR}[\sigma]$ exactly when σ is reachable.

- Then, to check that $\mathcal{M}$ is safe wrt to a property ψ it suffices to check that $\varphi_{IR}[\mathbf{x}] \wedge \psi[\mathbf{x}]$ is unsatisfiable in $\mathcal{A}$.

Strongest Inductive Invariant

- For a large class of systems $\mathcal{M}$, we can compute from φ_I and φ_R the *strongest inductive invariant* φ_{IR} for $\mathcal{M}$:

  for all $\sigma \in S$, $\mathcal{A} \models \varphi_{IR}[\sigma]$ exactly when σ is reachable.

- Then, to check that $\mathcal{M}$ is safe wrt to a property ψ it suffices to check that $\varphi_{IR}[\mathbf{x}] \wedge \psi[\mathbf{x}]$ is unsatisfiable in $\mathcal{A}$.
- This can be completely automated if the satisfiability in $\mathcal{A}$ of qffs is decidable.

Strongest Inductive Invariant

- For a large class of systems $\mathcal{M}$, we can compute from φ_I and φ_R the *strongest inductive invariant* φ_{IR} for $\mathcal{M}$:

  for all $\sigma \in S$, $\mathcal{A} \models \varphi_{IR}[\sigma]$ exactly when σ is reachable.

- Then, to check that $\mathcal{M}$ is safe wrt to a property ψ it suffices to check that $\varphi_{IR}[\mathbf{x}] \wedge \psi[\mathbf{x}]$ is unsatisfiable in $\mathcal{A}$.
- This can be completely automated if the satisfiability in $\mathcal{A}$ of qffs is decidable.
- **Problem:** Computing φ_{IR} can be very expensive.

Strongest Inductive Invariant

- For a large class of systems $\mathcal{M}$, we can compute from φ_I and φ_R the *strongest inductive invariant* φ_{IR} for $\mathcal{M}$:

  for all $\sigma \in S$, $\mathcal{A} \models \varphi_{IR}[\sigma]$ exactly when σ is reachable.

- Then, to check that $\mathcal{M}$ is safe wrt to a property ψ it suffices to check that $\varphi_{IR}[\mathbf{x}] \wedge \psi[\mathbf{x}]$ is unsatisfiable in $\mathcal{A}$.
- This can be completely automated if the satisfiability in $\mathcal{A}$ of qffs is decidable.
- **Problem:** Computing φ_{IR} can be very expensive.
- **Good news:** Craig interpolation can be used to reduce this cost.

Computing Strongest Inductive Invariants

When φ_{IR} is computable it is because it is the least fix point of an *image* operator $Img : QFF \to QFF$ where

- $Img(\varphi[\mathbf{x}])$ is the strongest (wrt $\models_{\mathcal{A}}$, entailment in $\mathcal{A}$) qff $\varphi_{\mathrm{p}}[\mathbf{x}]$ such that

$$\varphi[\mathbf{x}] \wedge \varphi_R[\mathbf{x}, \mathbf{x}'] \models_{\mathcal{A}} \varphi_{\mathrm{p}}[\mathbf{x}']$$

- $\varphi_{IR} = \bigwedge_{i \geq 0} \varphi^i$ with $\varphi^0 = \varphi_I$ and $\varphi^{i+1} = \varphi^i \vee Img(\varphi^i)$

Computing Strongest Inductive Invariants

When φ_{IR} is computable it is because it is the least fix point of an *image* operator $Img : QFF \rightarrow QFF$ where

- $Img(\varphi[\mathbf{x}])$ is the strongest (wrt $\models_{\mathcal{A}}$, entailment in $\mathcal{A}$) qff $\varphi_{\mathrm{p}}[\mathbf{x}]$ such that

$$\varphi[\mathbf{x}] \wedge \varphi_R[\mathbf{x}, \mathbf{x}'] \models_{\mathcal{A}} \varphi_{\mathrm{p}}[\mathbf{x}']$$

- $\varphi_{IR} = \bigwedge_{i \geq 0} \varphi^i$ with $\varphi^0 = \varphi_I$ and $\varphi^{i+1} = \varphi^i \vee Img(\varphi^i)$

Computing Img, and so φ_{IR}, is expensive because it involves quantifier elimination.

Computing Strongest Inductive Invariants

When φ_{IR} is computable it is because it is the least fix point of an *image* operator $Img : QFF \rightarrow QFF$ where

- $Img(\varphi[\mathbf{x}])$ is the strongest (wrt $\models_{\mathcal{A}}$, entailment in $\mathcal{A}$) qff $\varphi_{\mathrm{p}}[\mathbf{x}]$ such that

$$\varphi[\mathbf{x}] \wedge \varphi_R[\mathbf{x}, \mathbf{x}'] \models_{\mathcal{A}} \varphi_{\mathrm{p}}[\mathbf{x}']$$

- $\varphi_{IR} = \bigwedge_{i \geq 0} \varphi^i$ with $\varphi^0 = \varphi_I$ and $\varphi^{i+1} = \varphi^i \vee Img(\varphi^i)$

Computing Img, and so φ_{IR}, is expensive because it involves quantifier elimination.

However, Img might be much stronger than needed for proving that a property ψ is unreachable.

Computing Strongest Inductive Invariants

When φ_{IR} is computable it is because it is the least fix point of an *image* operator $Img : QFF \rightarrow QFF$ where

- $Img(\varphi[\mathbf{x}])$ is the strongest (wrt $\models_{\mathcal{A}}$, entailment in $\mathcal{A}$) qff $\varphi_{\mathrm{p}}[\mathbf{x}]$ such that

$$\varphi[\mathbf{x}] \wedge \varphi_R[\mathbf{x}, \mathbf{x}'] \models_{\mathcal{A}} \varphi_{\mathrm{p}}[\mathbf{x}']$$

- $\varphi_{IR} = \bigwedge_{i \geq 0} \varphi^i$ with $\varphi^0 = \varphi_I$ and $\varphi^{i+1} = \varphi^i \vee Img(\varphi^i)$

Computing Img, and so φ_{IR}, is expensive because it involves quantifier elimination.

Idea (McMillan, 2003):

use interpolation to compute for each $i \geq 0$
an *adequate over-approximation* $\hat{\varphi}^i$ of φ^i wrt ψ

How to compute $\hat{\varphi}_{IR}$ for ψ incrementally

Let $k > 0$, $\hat{\varphi}^0 = \varphi_I[\mathbf{x}]$

Base Case) Let:

$$\begin{aligned}\Gamma_1 &= \hat{\varphi}^0[\mathbf{x}_0] \wedge \varphi_R[\mathbf{x}_0, \mathbf{x}_1] \\ \Gamma_2 &= \varphi_R[\mathbf{x}_1, \mathbf{x}_2] \wedge \cdots \wedge \varphi_R[\mathbf{x}_{k-1}, \mathbf{x}_k] \wedge (\psi[\mathbf{x}_1] \vee \cdots \vee \psi[\mathbf{x}_k])\end{aligned}$$

How to compute $\hat{\varphi}_{IR}$ for ψ incrementally

Let $k > 0$, $\hat{\varphi}^0 = \varphi_I[\mathbf{x}]$

Base Case) Let:

$$\begin{aligned} \Gamma_1 &= \hat{\varphi}^0[\mathbf{x}_0] \wedge \varphi_R[\mathbf{x}_0, \mathbf{x}_1] \\ \Gamma_2 &= \varphi_R[\mathbf{x}_1, \mathbf{x}_2] \wedge \cdots \wedge \varphi_R[\mathbf{x}_{k-1}, \mathbf{x}_k] \wedge (\psi[\mathbf{x}_1] \vee \cdots \vee \psi[\mathbf{x}_k]) \end{aligned}$$

- If $\Gamma_1 \wedge \Gamma_2$ is satisfiable in $\mathcal{A}$, we are done:

  ψ is reachable from φ_I in 1 to k steps.

How to compute $\hat{\varphi}_{IR}$ for ψ incrementally

Let $k > 0$, $\hat{\varphi}^0 = \varphi_I[\mathbf{x}]$

Base Case) Let:

$$\begin{aligned} \Gamma_1 &= \hat{\varphi}^0[\mathbf{x}_0] \wedge \varphi_R[\mathbf{x}_0, \mathbf{x}_1] \\ \Gamma_2 &= \varphi_R[\mathbf{x}_1, \mathbf{x}_2] \wedge \cdots \wedge \varphi_R[\mathbf{x}_{k-1}, \mathbf{x}_k] \wedge (\psi[\mathbf{x}_1] \vee \cdots \vee \psi[\mathbf{x}_k]) \end{aligned}$$

- If $\Gamma_1 \wedge \Gamma_2$ is unsatisfiable in $\mathcal{A}$, compute an interpolant $\Gamma[\mathbf{x}_1]$ (wrt to $\models_{\mathcal{A}}$).

How to compute $\hat{\varphi}_{IR}$ for ψ incrementally

Let $k > 0$, $\hat{\varphi}^0 = \varphi_I[\mathbf{x}]$

Base Case) Let:

$$\begin{aligned} \Gamma_1 &= \hat{\varphi}^0[\mathbf{x}_0] \wedge \varphi_R[\mathbf{x}_0, \mathbf{x}_1] \\ \Gamma_2 &= \varphi_R[\mathbf{x}_1, \mathbf{x}_2] \wedge \cdots \wedge \varphi_R[\mathbf{x}_{k-1}, \mathbf{x}_k] \wedge (\psi[\mathbf{x}_1] \vee \cdots \vee \psi[\mathbf{x}_k]) \end{aligned}$$

- If $\Gamma_1 \wedge \Gamma_2$ is unsatisfiable in $\mathcal{A}$,
  compute an interpolant $\Gamma[\mathbf{x}_1]$ (wrt to $\models_{\mathcal{A}}$).
- $\Gamma[\mathbf{x}]$ is an adequate over-approximation of $Img(\varphi^0)$:
  $\Gamma_1 \models_{\mathcal{A}} \Gamma[\mathbf{x}_1] \implies$ every state reachable from φ_I is in Γ
  $\Gamma \wedge \Gamma_2 \models_{\mathcal{A}} \bot \implies$ no state in Γ leads to ψ within k steps

How to compute $\hat{\varphi}_{IR}$ for ψ incrementally

Let $k > 0$, $\hat{\varphi}^0 = \varphi_I[\mathbf{x}]$

Base Case) Let:

$$\begin{aligned}\Gamma_1 &= \hat{\varphi}^0[\mathbf{x}_0] \wedge \varphi_R[\mathbf{x}_0, \mathbf{x}_1] \\ \Gamma_2 &= \varphi_R[\mathbf{x}_1, \mathbf{x}_2] \wedge \cdots \wedge \varphi_R[\mathbf{x}_{k-1}, \mathbf{x}_k] \wedge (\psi[\mathbf{x}_1] \vee \cdots \vee \psi[\mathbf{x}_k])\end{aligned}$$

- If $\Gamma_1 \wedge \Gamma_2$ is unsatisfiable in $\mathcal{A}$,
  compute an interpolant $\Gamma[\mathbf{x}_1]$ (wrt to $\models_{\mathcal{A}}$).
- $\Gamma[\mathbf{x}]$ is an adequate over-approximation of $Img(\varphi^0)$:

  $\Gamma_1 \models_{\mathcal{A}} \Gamma[\mathbf{x}_1] \implies$ every state reachable from φ_I is in Γ

  $\Gamma \wedge \Gamma_2 \models_{\mathcal{A}} \bot \implies$ no state in Γ leads to ψ within k steps
- Set $\hat{\varphi}^1 = \hat{\varphi}^0[\mathbf{x}] \vee \Gamma[\mathbf{x}]$

How to compute $\hat{\varphi}_{IR}$ for ψ incrementally

Assume we have computed $\hat{\varphi}^i$ for $i > 0$.

Step case) Let

$$\begin{aligned} \Gamma_1 &= \hat{\varphi}^i[\mathbf{x}_0] \wedge \varphi_R[\mathbf{x}_0, \mathbf{x}_1] \\ \Gamma_2 &= \varphi_R[\mathbf{x}_1, \mathbf{x}_2] \wedge \cdots \wedge \varphi_R[\mathbf{x}_{k-1}, \mathbf{x}_k] \wedge (\psi[\mathbf{x}_1] \vee \cdots \vee \psi[\mathbf{x}_k]) \end{aligned}$$

How to compute $\hat{\varphi}_{IR}$ for ψ incrementally

Assume we have computed $\hat{\varphi}^i$ for $i > 0$.

Step case) Let

$$\begin{aligned}\Gamma_1 &= \hat{\varphi}^i[\mathbf{x}_0] \wedge \varphi_R[\mathbf{x}_0, \mathbf{x}_1] \\ \Gamma_2 &= \varphi_R[\mathbf{x}_1, \mathbf{x}_2] \wedge \cdots \wedge \varphi_R[\mathbf{x}_{k-1}, \mathbf{x}_k] \wedge (\psi[\mathbf{x}_1] \vee \cdots \vee \psi[\mathbf{x}_k])\end{aligned}$$

- If $\Gamma_1 \wedge \Gamma_2$ is unsatisfiable in $\mathcal{A}$, compute an interpolant Γ as before
- Let $\hat{\varphi}^{i+1} = \hat{\varphi}^i[\mathbf{x}] \vee \Gamma[\mathbf{x}]$

How to compute $\hat{\varphi}_{IR}$ *for* ψ *incrementally*

Assume we have computed $\hat{\varphi}^i$ for $i > 0$.

Step case) Let

$$\begin{aligned} \Gamma_1 &= \hat{\varphi}^i[\mathbf{x}_0] \wedge \varphi_R[\mathbf{x}_0, \mathbf{x}_1] \\ \Gamma_2 &= \varphi_R[\mathbf{x}_1, \mathbf{x}_2] \wedge \cdots \wedge \varphi_R[\mathbf{x}_{k-1}, \mathbf{x}_k] \wedge (\psi[\mathbf{x}_1] \vee \cdots \vee \psi[\mathbf{x}_k]) \end{aligned}$$

- If $\Gamma_1 \wedge \Gamma_2$ is satisfiable in $\mathcal{A}$, ψ is reachable from φ_I in $i+1$ to $i+k$ steps in the overapproximated closure of φ_R

  So, the satisfying paths of states might not be paths in the original system $\mathcal{M}$.

How to compute $\hat{\varphi}_{IR}$ for ψ incrementally

Assume we have computed $\hat{\varphi}^i$ for $i > 0$.

Step case) Let

$$\begin{aligned} \Gamma_1 &= \hat{\varphi}^i[\mathbf{x}_0] \wedge \varphi_R[\mathbf{x}_0, \mathbf{x}_1] \\ \Gamma_2 &= \varphi_R[\mathbf{x}_1, \mathbf{x}_2] \wedge \cdots \wedge \varphi_R[\mathbf{x}_{k-1}, \mathbf{x}_k] \wedge (\psi[\mathbf{x}_1] \vee \cdots \vee \psi[\mathbf{x}_k]) \end{aligned}$$

- If $\Gamma_1 \wedge \Gamma_2$ is satisfiable in $\mathcal{A}$, ψ is reachable from φ_I in $i+1$ to $i+k$ steps in the overapproximated closure of φ_R

  So, the satisfying paths of states might not be paths in the original system $\mathcal{M}$.

- Then, increase k by 1 and restart the whole process.

Thank you