Combination of Quantifier-free Uniform Interpolants using Beth Definability (Abridged Version)

Diego Calvanese[1], Silvio Ghilardi[2], **Alessandro Gianola**[1], Marco Montali[1], Andrey Rivkin[1]

[1] KRDB Research Centre for Knowledge and Data
Free University of Bozen-Bolzano, Italy

[2] Dipartimento di Matematica
Università degli Studi di Milano, Italy

TACL 2022

June 22, 2022

Outline

Outline

Aim of the Talk

- Our general problem: studying **combination of (quantifier-free) Uniform Interpolants (UIs)**.

Aim of the Talk

- Our general problem: studying **combination of (quantifier-free) Uniform Interpolants (UIs)**.
- Let T be a logic or a theory and L a suitable fragment (propositional, first-order quantifier-free, etc.) of its language. Given an L-formula $\phi(\underline{x}, \underline{y})$, a *uniform interpolant* of ϕ (w.r.t. $\underline{y}$) is an L-formula $\phi'(\underline{x})$ where only the $\underline{x}$ occur, and that satisfies the following two properties:

Aim of the Talk

- Our general problem: studying **combination of (quantifier-free) Uniform Interpolants (UIs)**.
- Let T be a logic or a theory and L a suitable fragment (propositional, first-order quantifier-free, etc.) of its language. Given an L-formula $\phi(\underline{x}, \underline{y})$, a *uniform interpolant* of ϕ (w.r.t. $\underline{y}$) is an L-formula $\phi'(\underline{x})$ where only the $\underline{x}$ occur, and that satisfies the following two properties:
  - $\phi(\underline{x}, \underline{y}) \vdash_T \phi'(\underline{x})$;

Aim of the Talk

- Our general problem: studying **combination of (quantifier-free) Uniform Interpolants (UIs)**.
- Let T be a logic or a theory and L a suitable fragment (propositional, first-order quantifier-free, etc.) of its language. Given an L-formula $\phi(\underline{x}, \underline{y})$, a *uniform interpolant* of ϕ (w.r.t. $\underline{y}$) is an L-formula $\phi'(\underline{x})$ where only the $\underline{x}$ occur, and that satisfies the following two properties:
  - $\phi(\underline{x}, \underline{y}) \vdash_T \phi'(\underline{x})$;
  - for any further L-formula $\psi(\underline{x}, \underline{z})$ such that $\phi(\underline{x}, \underline{y}) \vdash_T \psi(\underline{x}, \underline{z})$, we have $\phi'(\underline{x}) \vdash_T \psi(\underline{x}, \underline{z})$.

Motivation (I)

- Infinite-state model checking $\implies$ sets of (*reachable*) states and transitions represented **symbolically**.

Motivation (I)

- Infinite-state model checking $\implies$ sets of (*reachable*) states and transitions represented **symbolically**.
- *Precise* computations of the set of reachable states through quantifier elimination (QE).

Motivation (I)

- Infinite-state model checking $\Longrightarrow$ sets of (*reachable*) states and transitions represented **symbolically**.
- *Precise* computations of the set of reachable states through quantifier elimination (QE).
- Usually, QE is **computationally intractable**.

Motivation (I)

- Infinite-state model checking $\Longrightarrow$ sets of (*reachable*) states and transitions represented **symbolically**.
- *Precise* computations of the set of reachable states through quantifier elimination (QE).
- Usually, QE is **computationally intractable**.
- In contrast, methods for *symbol elimination* (e.g., predicate abstraction or ordinary interpolation), used to **approximate** states, are quite *efficient*. But the computation is *not* exact.

Motivation (I)

- Infinite-state model checking $\Longrightarrow$ sets of (*reachable*) states and transitions represented **symbolically**.
- *Precise* computations of the set of reachable states through quantifier elimination (QE).
- Usually, QE is **computationally intractable**.
- In contrast, methods for *symbol elimination* (e.g., predicate abstraction or ordinary interpolation), used to **approximate** states, are quite *efficient*. But the computation is *not* exact.
- However, QE has strict relations with **uniform** interpolation (or, **covers** [GM08]), largely studied in non-classical logics since the nineties, and becomes **tractable** in significant cases [CGG^{+}19].

Motivation (II)

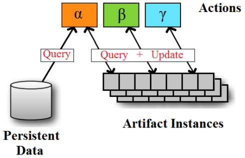
- Modeling and verifying **data-aware processes** $\Longrightarrow$ **combination of different theories**, e.g., (i) for the *read-only data storage*; (ii) for elements from value domains (like *arithmetical values*)

Motivation (II)

- Modeling and verifying **data-aware processes** $\Longrightarrow$ **combination of different theories**, e.g., (i) for the *read-only data storage*; (ii) for elements from value domains (like *arithmetical values*)
- Important question: is it **possible** (and, if so, **under which conditions**) to transfer UIs from two theories T_1, T_2 to the combined theory $T_1 \cup T_2$?

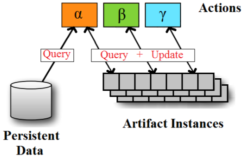
Motivation (II)

- Modeling and verifying **data-aware processes** $\Longrightarrow$ **combination of different theories**, e.g., (i) for the *read-only data storage*; (ii) for elements from value domains (like *arithmetical values*)
- Important question: is it **possible** (and, if so, **under which conditions**) to transfer UIs from two theories T_1, T_2 to the combined theory $T_1 \cup T_2$?
- Example: *Simple Artifact Systems (SAS)* [CGG$^+$19]



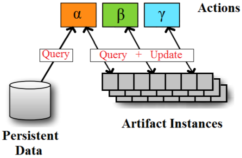
Motivation (II)

- Modeling and verifying **data-aware processes** $\implies$ **combination of different theories**, e.g., (i) for the *read-only data storage*; (ii) for elements from value domains (like *arithmetical values*)
- Important question: is it **possible** (and, if so, **under which conditions**) to transfer UIs from two theories T_1, T_2 to the combined theory $T_1 \cup T_2$?
- Example: *Simple Artifact Systems (SAS)* [CGG+19]
  - States: $\phi(\underline{x})$ (**quantifier-free**)



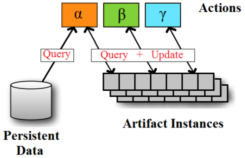
Motivation (II)

- Modeling and verifying **data-aware processes** $\Longrightarrow$ **combination of different theories**, e.g., (i) for the *read-only data storage*; (ii) for elements from value domains (like *arithmetical values*)
- Important question: is it **possible** (and, if so, **under which conditions**) to transfer UIs from two theories T_1, T_2 to the combined theory $T_1 \cup T_2$?
- Example: *Simple Artifact Systems (SAS)* [CGG+19]
  - States: $\phi(\underline{x})$ **(quantifier-free)**
  - Transitions: $\tau(\underline{x}, \underline{x}') \equiv \exists \underline{d}, \underline{i}(G(\underline{x}, \underline{d}, \underline{i}) \wedge U(\underline{x}, \underline{x}', \underline{d}, \underline{i}))$ **(existential)**



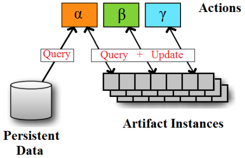
Motivation (II)

- Modeling and verifying **data-aware processes** $\Longrightarrow$ **combination of different theories**, e.g., (i) for the *read-only data storage*; (ii) for elements from value domains (like *arithmetical values*)
- Important question: is it **possible** (and, if so, **under which conditions**) to transfer UIs from two theories T_1, T_2 to the combined theory $T_1 \cup T_2$?
- Example: *Simple Artifact Systems (SAS)* [CGG+19]
  - States: $\phi(\underline{x})$ **(quantifier-free)**
  - Transitions: $\tau(\underline{x}, \underline{x}') \equiv \exists \underline{d}, \underline{i}(G(\underline{x}, \underline{d}, \underline{i}) \wedge U(\underline{x}, \underline{x}', \underline{d}, \underline{i}))$ **(existential)**
  - $\underline{d}$: *Persistent Data* from DB;



Motivation (II)

- Modeling and verifying **data-aware processes** $\Longrightarrow$ **combination of different theories**, e.g., (i) for the *read-only data storage*; (ii) for elements from value domains (like *arithmetical values*)
- Important question: is it **possible** (and, if so, **under which conditions**) to transfer UIs from two theories T_1, T_2 to the combined theory $T_1 \cup T_2$?
- Example: *Simple Artifact Systems (SAS)* [CGG+19]
  - States: $\phi(\underline{x})$ (**quantifier-free**)
  - Transitions: $\tau(\underline{x}, \underline{x}') \equiv \exists \underline{d}, \underline{i}(G(\underline{x}, \underline{d}, \underline{i}) \wedge U(\underline{x}, \underline{x}', \underline{d}, \underline{i}))$ (**existential**)
  - $\underline{d}$: *Persistent Data* from DB;
  - $\underline{i}$: elements from *arithmetical domains*.



Motivation (III): Verification of SASs

Given a state formula ϕ for states $S^{(i)}$, we symbolically define $T^{-1}(S^{(i)})$:

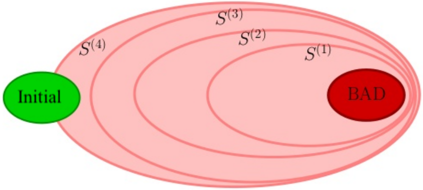
$$Pre(\tau, \phi) \equiv \exists \underline{x}'(\tau(\underline{x}, \underline{x}') \wedge \phi(\underline{x}'))$$

Motivation (III): Verification of SASs

Given a state formula ϕ for states $S^{(i)}$, we symbolically define $T^{-1}(S^{(i)})$:

$$Pre(\tau, \phi) \equiv \exists \underline{x}'(\tau(\underline{x}, \underline{x}') \wedge \phi(\underline{x}'))$$

Backward-Reachability ($S^{(0)} \equiv$ "bad states")	
Safety Check	If $S^{(i)}$ contains an initial, return **unsafe**
Next States	Compute $S^{(i+1)} := S^{(i)} \cup T^{-1}(S^{(i)})$
Fix-Point Check	If $S^{(i+1)} \equiv S^{(i)}$, return **safe**



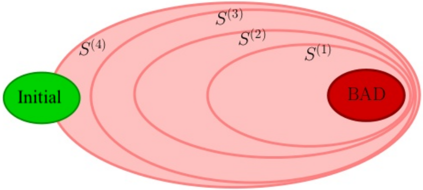
Motivation (III): Verification of SASs

Given a state formula ϕ for states $S^{(i)}$, we symbolically define $T^{-1}(S^{(i)})$:

$$Pre(\tau, \phi) \equiv \exists \underline{x}'(\tau(\underline{x}, \underline{x}') \wedge \phi(\underline{x}'))$$

Backward-Reachability ($S^{(0)} \equiv$ "bad states")

Safety Check	If $S^{(i)}$ contains an initial, return **unsafe**
Next States	Compute $S^{(i+1)} := S^{(i)} \cup T^{-1}(S^{(i)})$
Fix-Point Check	If $S^{(i+1)} \equiv S^{(i)}$, return **safe**



$$S^{(0)} : \phi \implies S^{(1)} : Pre(\tau, \phi) \equiv \exists \underline{d}, \underline{i}, \underline{x}'(G(\underline{x}, \underline{d}, \underline{i}) \wedge U(\underline{x}, \underline{x}', \underline{d}, \underline{i}) \wedge \phi(\underline{x}'))$$

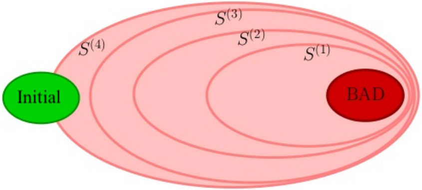
Motivation (III): Verification of SASs

Given a state formula ϕ for states $S^{(i)}$, we symbolically define $T^{-1}(S^{(i)})$:

$$Pre(\tau, \phi) \equiv \exists \underline{x}'(\tau(\underline{x}, \underline{x}') \wedge \phi(\underline{x}'))$$

Backward-Reachability ($S^{(0)} \equiv$ "bad states")

Safety Check	If $S^{(i)}$ contains an initial, return **unsafe**
Next States	Compute $S^{(i+1)} := S^{(i)} \cup T^{-1}(S^{(i)})$
Fix-Point Check	If $S^{(i+1)} \equiv S^{(i)}$, return **safe**



$$S^{(0)} : \phi \implies S^{(1)} : Pre(\tau, \phi) \equiv \exists \underline{d}, \underline{i}, \underline{x}'(G(\underline{x}, \underline{d}, \underline{i}) \wedge U(\underline{x}, \underline{x}', \underline{d}, \underline{i}) \wedge \phi(\underline{x}'))$$

$S^{(1)}$ is **NOT** a state formula! The existential quantifiers can be 'eliminated' [CGG+19] by computing ***combined* UIs**!

Our contributions

- General algorithm for computing **combined UIs** in case of **convex** component theories.

Our contributions

- General algorithm for computing **combined UIs** in case of **convex** component theories.
- The **hypothesis** under which this algorithm is correct is the same needed to transfer quantifier-free interpolation: the **equality interpolating condition**.

Our contributions

- General algorithm for computing **combined UIs** in case of **convex** component theories.
- The **hypothesis** under which this algorithm is correct is the same needed to transfer quantifier-free interpolation: the **equality interpolating condition**.
- We prove that the equality interpolating condition is also **necessary** for transferring UIs.

Our contributions

- General algorithm for computing **combined UIs** in case of **convex** component theories.
- The **hypothesis** under which this algorithm is correct is the same needed to transfer quantifier-free interpolation: the **equality interpolating condition**.
- We prove that the equality interpolating condition is also **necessary** for transferring UIs.
- The algorithm relies on the extensive use of the **Beth definability property** for primitive fragments.

Our contributions

- General algorithm for computing **combined UIs** in case of **convex** component theories.
- The **hypothesis** under which this algorithm is correct is the same needed to transfer quantifier-free interpolation: the **equality interpolating condition**.
- We prove that the equality interpolating condition is also **necessary** for transferring UIs.
- The algorithm relies on the extensive use of the **Beth definability property** for primitive fragments.
- **Counterexample** showing **non-transfer** of UIs for non-convex theories in general, even in case combined quantifier-free interpolants do exist.

Outline

Preliminaries

Definition

Given a FO theory T and two quantifier-free FO formulae $\alpha(\underline{x}, \underline{y})$, $\beta(\underline{y}, \underline{z})$ such that $\vdash_T \alpha \to \beta$, a **quantifier-free** FO formula $\gamma(\underline{y})$ is a **T-quantifier-free interpolant** if $\vdash_T \alpha \to \gamma$ and $\vdash_T \gamma \to \beta$ hold.

Preliminaries

Definition

Given a FO theory T and two quantifier-free FO formulae $\alpha(\underline{x}, \underline{y})$, $\beta(\underline{y}, \underline{z})$ such that $\vdash_T \alpha \to \beta$, a **quantifier-free** FO formula $\gamma(\underline{y})$ is a **T-quantifier-free interpolant** if $\vdash_T \alpha \to \gamma$ and $\vdash_T \gamma \to \beta$ hold.

If every pair $\alpha(\underline{x}, \underline{y}), \beta(\underline{y}, \underline{z})$ has a quantifier-free interpolant, then T enjoys the quantifier-free interpolation property.

Preliminaries

Definition

Given a FO theory T and two quantifier-free FO formulae $\alpha(\underline{x}, \underline{y})$, $\beta(\underline{y}, \underline{z})$ such that $\vdash_T \alpha \to \beta$, a **quantifier-free** FO formula $\gamma(\underline{y})$ is a **T-quantifier-free interpolant** if $\vdash_T \alpha \to \gamma$ and $\vdash_T \gamma \to \beta$ hold.

If every pair $\alpha(\underline{x}, \underline{y}), \beta(\underline{y}, \underline{z})$ has a quantifier-free interpolant, then T enjoys the quantifier-free interpolation property.

Definition

A theory T is *stably infinite* iff every T-satisfiable **constraint** is satisfiable in an **infinite** model of T.

Preliminaries

Definition

Given a FO theory T and two quantifier-free FO formulae $\alpha(\underline{x}, \underline{y})$, $\beta(\underline{y}, \underline{z})$ such that $\vdash_T \alpha \to \beta$, a **quantifier-free** FO formula $\gamma(\underline{y})$ is a **T-quantifier-free interpolant** if $\vdash_T \alpha \to \gamma$ and $\vdash_T \gamma \to \beta$ hold.

If every pair $\alpha(\underline{x}, \underline{y}), \beta(\underline{y}, \underline{z})$ has a quantifier-free interpolant, then T enjoys the quantifier-free interpolation property.

Definition

A theory T is *stably infinite* iff every T-satisfiable **constraint** is satisfiable in an **infinite** model of T.

Definition

A theory T is *convex* iff for every **constraint** δ, if $T \vdash \delta \to \bigvee_{i=1}^{n} x_i = y_i$ then $T \vdash \delta \to x_i = y_i$ holds for some $i \in \{1, ..., n\}$.

A convex theory is 'almost' stably infinite (for constraints satisfiable in models with at least two elements)

Uniform Quantifier-Free Interpolation (Covers)

Fix a theory T and an existential formula $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$.

- A quantifier-free (qf) formula $\psi(\underline{y})$ is a T-***uniform (qf) interpolant*** (or, T-***cover***) of $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$ iff

Uniform Quantifier-Free Interpolation (Covers)

Fix a theory T and an existential formula $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$.

- A quantifier-free (qf) formula $\psi(\underline{y})$ is a T-***uniform (qf) interpolant*** (or, T-***cover***) of $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$ iff

  (i) $\psi(\underline{y}) \in Res(\exists \underline{e}\, \phi) := \{\theta(\underline{y}, \underline{z}) \mid T \models \phi(\underline{e}, \underline{y}) \to \theta(\underline{y}, \underline{z})\}$,

Uniform Quantifier-Free Interpolation (Covers)

Fix a theory T and an existential formula $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$.

- A quantifier-free (qf) formula $\psi(\underline{y})$ is a T-***uniform (qf) interpolant*** (or, T-***cover***) of $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$ iff

  (i) $\psi(\underline{y}) \in Res(\exists \underline{e}\, \phi) := \{\theta(\underline{y}, \underline{z}) \mid T \models \phi(\underline{e}, \underline{y}) \to \theta(\underline{y}, \underline{z})\}$,

  (ii) $\psi(\underline{y})$ implies (modulo T) all the formulae in $Res(\exists \underline{e}\, \phi)$.

Uniform Quantifier-Free Interpolation (Covers)

Fix a theory T and an existential formula $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$.

- A quantifier-free (qf) formula $\psi(\underline{y})$ is a T-***uniform (qf) interpolant*** (or, T-***cover***) of $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$ iff

  (i) $\psi(\underline{y}) \in Res(\exists \underline{e}\, \phi) := \{\theta(\underline{y}, \underline{z}) \mid T \models \phi(\underline{e}, \underline{y}) \rightarrow \theta(\underline{y}, \underline{z})\}$,

  (ii) $\psi(\underline{y})$ implies (modulo T) all the formulae in $Res(\exists \underline{e}\, \phi)$.

  We say that a theory T has *uniform (qf) interpolation* iff every existential formula $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$ has a T-uniform (qf) interpolant.

Uniform Quantifier-Free Interpolation (Covers)

Fix a theory T and an existential formula $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$.

- A quantifier-free (qf) formula $\psi(\underline{y})$ is a T-***uniform (qf) interpolant*** (or, T-***cover***) of $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$ iff

  (i) $\psi(\underline{y}) \in Res(\exists \underline{e}\, \phi) := \{\theta(\underline{y}, \underline{z}) \mid T \models \phi(\underline{e}, \underline{y}) \to \theta(\underline{y}, \underline{z})\}$,

  (ii) $\psi(\underline{y})$ implies (modulo T) all the formulae in $Res(\exists \underline{e}\, \phi)$.

  We say that a theory T has *uniform (qf) interpolation* iff every existential formula $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$ has a T-uniform (qf) interpolant.

- A T-cover is a T-quantifier-free interpolant and is, intuitively, the strongest formula implied by $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$.

Uniform Quantifier-Free Interpolation (Covers)

Fix a theory T and an existential formula $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$.

- A quantifier-free (qf) formula $\psi(\underline{y})$ is a T-***uniform (qf) interpolant*** (or, T-***cover***) of $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$ iff

  (i) $\psi(\underline{y}) \in Res(\exists \underline{e}\, \phi) := \{\theta(\underline{y}, \underline{z}) \mid T \models \phi(\underline{e}, \underline{y}) \rightarrow \theta(\underline{y}, \underline{z})\}$,

  (ii) $\psi(\underline{y})$ implies (modulo T) all the formulae in $Res(\exists \underline{e}\, \phi)$.

  We say that a theory T has *uniform (qf) interpolation* iff every existential formula $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$ has a T-uniform (qf) interpolant.

- A T-cover is a T-quantifier-free interpolant and is, intuitively, the strongest formula implied by $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$.

- In the cover $\psi(\underline{y})$, the variables $\underline{e}$ have been 'eliminated', in some sense.

Uniform Quantifier-Free Interpolation (Covers)

Fix a theory T and an existential formula $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$.

- A quantifier-free (qf) formula $\psi(\underline{y})$ is a T-***uniform (qf) interpolant*** (or, T-***cover***) of $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$ iff

  (i) $\psi(\underline{y}) \in Res(\exists \underline{e}\, \phi) := \{\theta(\underline{y}, \underline{z}) \mid T \models \phi(\underline{e}, \underline{y}) \rightarrow \theta(\underline{y}, \underline{z})\}$,

  (ii) $\psi(\underline{y})$ implies (modulo T) all the formulae in $Res(\exists \underline{e}\, \phi)$.

  We say that a theory T has *uniform (qf) interpolation* iff every existential formula $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$ has a T-uniform (qf) interpolant.
- A T-cover is a T-quantifier-free interpolant and is, intuitively, the strongest formula implied by $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$.
- In the cover $\psi(\underline{y})$, the variables $\underline{e}$ have been 'eliminated', in some sense.
- But, in general, $\psi(\underline{y})$ does *not* imply $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$. Hence, usually $\psi(\underline{y})$ and $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$ are not T-equivalent.

UIs and Model Completions

A *universal* Σ-theory T has a ***model completion*** iff there is a stronger theory $T^* \supseteq T$ (in the same signature Σ) such that (i) every Σ-constraint that is satisfiable in a model of T is satisfiable in a model of T^*; (ii) T^* eliminates quantifiers.

UIs and Model Completions

A *universal* Σ-theory T has a ***model completion*** iff there is a stronger theory $T^* \supseteq T$ (in the same signature Σ) such that (i) every Σ-constraint that is satisfiable in a model of T is satisfiable in a model of T^*; (ii) T^* eliminates quantifiers.

Theorem (UIs and QE [CGG+19])

*Suppose that T is a universal theory. Then, T has a model completion T^* **iff** T has uniform quantifier-free interpolation. If this happens, T^* is axiomatized by the infinitely many sentences $\forall \underline{y}\,(\psi(\underline{y}) \to \exists \underline{e}\,\phi(\underline{e}, \underline{y}))$, where $\exists \underline{e}\,\phi(\underline{e}, \underline{y})$ is a primitive formula and ψ is a **UI** of it.*

UIs and Model Completions

A *universal* Σ-theory T has a ***model completion*** iff there is a stronger theory $T^* \supseteq T$ (in the same signature Σ) such that (i) every Σ-constraint that is satisfiable in a model of T is satisfiable in a model of T^*; (ii) T^* eliminates quantifiers.

Theorem (UIs and QE [CGG+19])

Suppose that T is a universal theory. Then, T has a model completion T^* ***iff*** *T has uniform quantifier-free interpolation. If this happens, T^* is axiomatized by the infinitely many sentences $\forall \underline{y}\, (\psi(\underline{y}) \to \exists \underline{e}\, \phi(\underline{e}, \underline{y}))$, where $\exists \underline{e}\, \phi(\underline{e}, \underline{y})$ is a primitive formula and ψ is a* ***UI*** *of it.*

Hence, **computing UIs** in a theory T
is **equivalent** to
eliminating quantifiers in its model completion T^*.

Outline

Equality Interpolating Condition

Definition ([YM05])

A convex universal theory T is *equality interpolating* iff for every pair y_1, y_2 of variables and for every pair of *constraints* $\delta_1(\underline{x}, \underline{z}_1, y_1)$, $\delta_2(\underline{x}, \underline{z}_2, y_2)$ such that $T \vdash \delta_1(\underline{x}, \underline{z}_1, y_1) \wedge \delta_2(\underline{x}, \underline{z}_2, y_2) \to y_1 = y_2$, **there exists** a term $t(\underline{x})$ such that $T \vdash \delta_1(\underline{x}, \underline{z}_1, y_1) \wedge \delta_2(\underline{x}, \underline{z}_2, y_2) \to y_1 = t(\underline{x}) \wedge y_2 = t(\underline{x})$.

Theorem ([BGR14])

A universal theory T has the strong amalgamation property iff it is equality interpolating.

Equality Interpolating Condition

Definition ([YM05])

A convex universal theory T is *equality interpolating* iff for every pair y_1, y_2 of variables and for every pair of *constraints* $\delta_1(\underline{x}, \underline{z}_1, y_1)$, $\delta_2(\underline{x}, \underline{z}_2, y_2)$ such that $T \vdash \delta_1(\underline{x}, \underline{z}_1, y_1) \wedge \delta_2(\underline{x}, \underline{z}_2, y_2) \to y_1 = y_2$, **there exists** a term $t(\underline{x})$ such that $T \vdash \delta_1(\underline{x}, \underline{z}_1, y_1) \wedge \delta_2(\underline{x}, \underline{z}_2, y_2) \to y_1 = t(\underline{x}) \wedge y_2 = t(\underline{x})$.

Theorem ([BGR14])

A universal theory T has the strong amalgamation property iff it is equality interpolating.

Examples of universal **quantifier-free interpolating** and **equality interpolating** theories:

- $\mathcal{EUF}(\Sigma)$, given a signature Σ;
- recursive data theories;
- linear arithmetics.

Transfer of Quantifier-free Interpolants

Theorem (Sufficient Condition [YM05, BGR14])

Let T_1 and T_2 be two universal, convex, stably infinite theories over disjoint signatures Σ_1 and Σ_2. If both T_1 and T_2 are equality interpolating and have quantifier-free interpolation property, then so does $T_1 \cup T_2$.

Transfer of Quantifier-free Interpolants

Theorem (Sufficient Condition [YM05, BGR14])

Let T_1 and T_2 be two universal, convex, stably infinite theories over disjoint signatures Σ_1 and Σ_2. If both T_1 and T_2 are equality interpolating and have quantifier-free interpolation property, then so does $T_1 \cup T_2$.

There is a **converse** [BGR14] of the previous result, in the sense that the **equality interpolating property** is already required for transferring *quantifier-free interpolation* in the **minimal combinations** with signatures adding uninterpreted symbols ($\mathcal{EUF}(\Sigma)$).

Beth Definability and Equality Interpolating Condition

Equality interpolating can be characterized using Beth definability.

Beth Definability and Equality Interpolating Condition

Equality interpolating can be characterized using Beth definability.

Given a primitive formula $\exists \underline{z} \phi(\underline{x}, \underline{z}, y)$, we say that:

Beth Definability and Equality Interpolating Condition

Equality interpolating can be characterized using Beth definability.

Given a primitive formula $\exists \underline{z} \phi(\underline{x}, \underline{z}, y)$, we say that:

- $\exists \underline{z}\, \phi(\underline{x}, \underline{z}, y)$ *implicitly defines* y in T iff the following formula is T-valid: $\forall y \, \forall y' \, (\exists \underline{z} \phi(\underline{x}, \underline{z}, y) \wedge \exists \underline{z} \phi(\underline{x}, \underline{z}, y') \rightarrow y = y')$;

Beth Definability and Equality Interpolating Condition

Equality interpolating can be characterized using Beth definability.

Given a primitive formula $\exists \underline{z} \phi(\underline{x}, \underline{z}, y)$, we say that:

- $\exists \underline{z}\, \phi(\underline{x}, \underline{z}, y)$ *implicitly defines* y in T iff the following formula is T-valid: $\forall y\, \forall y'\; (\exists \underline{z} \phi(\underline{x}, \underline{z}, y) \wedge \exists \underline{z} \phi(\underline{x}, \underline{z}, y') \rightarrow y = y')$;
- $\exists \underline{z} \phi(\underline{x}, \underline{z}, y)$ *explicitly defines* y in T iff there is a term $t(\underline{x})$ such that the formula is T-valid: $\forall y\; (\exists \underline{z} \phi(\underline{x}, \underline{z}, y) \rightarrow y = t(\underline{x}))$;

Beth Definability and Equality Interpolating Condition

Equality interpolating can be characterized using Beth definability.

Given a primitive formula $\exists \underline{z}\phi(\underline{x}, \underline{z}, y)$, we say that:

- $\exists \underline{z}\, \phi(\underline{x}, \underline{z}, y)$ *implicitly defines* y in T iff the following formula is T-valid: $\forall y\, \forall y'\, (\exists \underline{z}\phi(\underline{x}, \underline{z}, y) \wedge \exists \underline{z}\phi(\underline{x}, \underline{z}, y') \rightarrow y = y')$;
- $\exists \underline{z}\phi(\underline{x}, \underline{z}, y)$ *explicitly defines* y in T iff there is a term $t(\underline{x})$ such that the formula is T-valid: $\forall y\, (\exists \underline{z}\phi(\underline{x}, \underline{z}, y) \rightarrow y = t(\underline{x}))$;
- a theory T has the ***Beth definability property*** *for primitive formulae* iff whenever a primitive formula $\exists \underline{z}\, \phi(\underline{x}, \underline{z}, y)$ *implicitly* defines the variable y then it also *explicitly* defines it.

Beth Definability and Equality Interpolating Condition

Equality interpolating can be characterized using Beth definability.

Given a primitive formula $\exists \underline{z} \phi(\underline{x}, \underline{z}, y)$, we say that:

- $\exists \underline{z}\, \phi(\underline{x}, \underline{z}, y)$ *implicitly defines* y in T iff the following formula is T-valid: $\forall y\, \forall y' \,(\exists \underline{z} \phi(\underline{x}, \underline{z}, y) \wedge \exists \underline{z} \phi(\underline{x}, \underline{z}, y') \rightarrow y = y')$;
- $\exists \underline{z} \phi(\underline{x}, \underline{z}, y)$ *explicitly defines* y in T iff there is a term $t(\underline{x})$ such that the formula is T-valid: $\forall y \,(\exists \underline{z} \phi(\underline{x}, \underline{z}, y) \rightarrow y = t(\underline{x}))$;
- a theory T has the ***Beth definability property*** *for primitive formulae* iff whenever a primitive formula $\exists \underline{z}\, \phi(\underline{x}, \underline{z}, y)$ *implicitly* defines the variable y then it also *explicitly* defines it.

Theorem (Key Theorem [BGR14])

*A convex theory T having quantifier-free interpolation is **equality interpolating** **iff** it has the **Beth definability property** for primitive formulae.*

Outline

Convex Theories

- Every Σ_i-theory T_i from now on is ***convex, stably infinite, equality interpolating, universal and admitting a model completion*** T_i^*.

Convex Theories

- Every Σ_i-theory T_i from now on is ***convex, stably infinite, equality interpolating, universal and admitting a model completion*** T_i^*.
- For $i = 1, \ldots, n$, we let the formula $\texttt{ImplDef}^T_{\phi,y_i}(\underline{x})$ be the quantifier-free formula equivalent in T^* to the formula

$$\forall \underline{y}\, \forall \underline{y}' (\phi(\underline{x}, \underline{y}) \wedge \phi(\underline{x}, \underline{y}') \to y_i = y_i')$$

where the $\underline{y}'$ are renamed copies of the $\underline{y}$.

Convex Theories

- Every Σ_i-theory T_i from now on is ***convex, stably infinite, equality interpolating, universal and admitting a model completion*** T_i^*.
- For $i = 1, \ldots, n$, we let the formula $\texttt{ImplDef}^T_{\phi,y_i}(\underline{x})$ be the quantifier-free formula equivalent in T^* to the formula

$$\forall \underline{y}\, \forall \underline{y}' (\phi(\underline{x}, \underline{y}) \wedge \phi(\underline{x}, \underline{y}') \rightarrow y_i = y_i')$$

  where the $\underline{y}'$ are renamed copies of the $\underline{y}$.

The following Lemma supplies terms used as ingredients in the combined covers algorithm:

Lemma (Useful Terms)

Let $L_{i1}(\underline{x}) \vee \cdots \vee L_{ik_i}(\underline{x})$ *be the disjunctive normal form (DNF) of* $\texttt{ImplDef}^T_{\phi,y_i}(\underline{x})$. *Then, for every* $j = 1, \ldots, k_i$, *there is a* $\Sigma(\underline{x})$*-term* $t_{ij}(\underline{x})$ *such that* $T \vdash L_{ij}(\underline{x}) \wedge \phi(\underline{x}, \underline{y}) \rightarrow y_i = t_{ij}$

The terms t_{ij} are obtained thanks to the Beth definability property, that holds because of the Key Theorem.

Computing Combined UIs

- Given a Σ_1-theory T_1 and a Σ_2-theory T_2, we want to compute a $T_1 \cup T_2$-**cover** for $\exists \underline{e}\, \phi(\underline{x}, \underline{e})$ (Initial Formula).

Computing Combined UIs

- Given a Σ_1-theory T_1 and a Σ_2-theory T_2, we want to compute a $T_1 \cup T_2$-**cover** for $\exists \underline{e}\, \phi(\underline{x}, \underline{e})$ (Initial Formula).
- By applying rewriting purification steps, we can assume that ϕ is of the kind $\phi_1 \wedge \phi_2$, where ϕ_i is a Σ_i-formula $(i = 1, 2)$.

Computing Combined UIs

- Given a Σ_1-theory T_1 and a Σ_2-theory T_2, we want to compute a $T_1 \cup T_2$-**cover** for $\exists \underline{e}\, \phi(\underline{x}, \underline{e})$ (Initial Formula).
- By applying rewriting purification steps, we can assume that ϕ is of the kind $\phi_1 \wedge \phi_2$, where ϕ_i is a Σ_i-formula $(i = 1, 2)$.
- Assume that ϕ_1 and ϕ_2 contain $e_i \neq e_j$ (for $i \neq j$): guess a partition of the $\underline{e}$ and replace each e_i with the representative element of its equivalence class.

Computing Combined UIs

- Given a Σ_1-theory T_1 and a Σ_2-theory T_2, we want to compute a $T_1 \cup T_2$-**cover** for $\exists \underline{e}\, \phi(\underline{x}, \underline{e})$ (Initial Formula).
- By applying rewriting purification steps, we can assume that ϕ is of the kind $\phi_1 \wedge \phi_2$, where ϕ_i is a Σ_i-formula $(i = 1, 2)$.
- Assume that ϕ_1 and ϕ_2 contain $e_i \neq e_j$ (for $i \neq j$): guess a partition of the $\underline{e}$ and replace each e_i with the representative element of its equivalence class.
- The algorithm employs **acyclic explicit definitions** $\texttt{ExplDef}(\underline{z}, \underline{x})$ $\bigwedge_{i=1}^{m} z_i = t_i(z_1, \ldots, z_{i-1}, \underline{x})$ where the term t_i is pure.

Computing Combined UIs

- Given a Σ_1-theory T_1 and a Σ_2-theory T_2, we want to compute a $T_1 \cup T_2$**-cover** for $\exists \underline{e}\, \phi(\underline{x}, \underline{e})$ (Initial Formula).
- By applying rewriting purification steps, we can assume that ϕ is of the kind $\phi_1 \wedge \phi_2$, where ϕ_i is a Σ_i-formula ($i = 1, 2$).
- Assume that ϕ_1 and ϕ_2 contain $e_i \neq e_j$ (for $i \neq j$): guess a partition of the $\underline{e}$ and replace each e_i with the representative element of its equivalence class.
- The algorithm employs **acyclic explicit definitions** $\texttt{ExplDef}(\underline{z}, \underline{x})$ $\bigwedge_{i=1}^{m} z_i = t_i(z_1, \ldots, z_{i-1}, \underline{x})$ where the term t_i is pure.
- A *working formula* is $\exists \underline{z}\, (\texttt{ExplDef}(\underline{z}, \underline{x}) \wedge \exists \underline{e}\, (\psi_1(\underline{x}, \underline{z}, \underline{e}) \wedge \psi_2(\underline{x}, \underline{z}, \underline{e})))$, where ψ_i is a Σ_i-formula ($i = 1, 2$) and $\underline{x}$ are called *parameters*, $\underline{z}$ *defined variables* and $\underline{e}$ *(truly) existential variables*. ψ_1, ψ_2 always contain the literals $e_i \neq e_j$ (for distinct e_i, e_j from $\underline{e}$) as a conjunct.

Computing Combined UIs

- Given a Σ_1-theory T_1 and a Σ_2-theory T_2, we want to compute a $T_1 \cup T_2$**-cover** for $\exists \underline{e}\, \phi(\underline{x}, \underline{e})$ (Initial Formula).
- By applying rewriting purification steps, we can assume that ϕ is of the kind $\phi_1 \wedge \phi_2$, where ϕ_i is a Σ_i-formula ($i = 1, 2$).
- Assume that ϕ_1 and ϕ_2 contain $e_i \neq e_j$ (for $i \neq j$): guess a partition of the $\underline{e}$ and replace each e_i with the representative element of its equivalence class.
- The algorithm employs **acyclic explicit definitions** $\texttt{ExplDef}(\underline{z}, \underline{x})$ $\bigwedge_{i=1}^{m} z_i = t_i(z_1, \ldots, z_{i-1}, \underline{x})$ where the term t_i is pure.
- A *working formula* is $\exists \underline{z}\, (\texttt{ExplDef}(\underline{z}, \underline{x}) \wedge \exists \underline{e}\, (\psi_1(\underline{x}, \underline{z}, \underline{e}) \wedge \psi_2(\underline{x}, \underline{z}, \underline{e})))$, where ψ_i is a Σ_i-formula ($i = 1, 2$) and $\underline{x}$ are called *parameters*, $\underline{z}$ *defined variables* and $\underline{e}$ *(truly) existential variables*. ψ_1, ψ_2 always contain the literals $e_i \neq e_j$ (for distinct e_i, e_j from $\underline{e}$) as a conjunct.
- A working formula is *terminal* iff for every $e_i \in \underline{e}$

$$T_1 \vdash \psi_1 \to \neg \texttt{ImplDef}^{T_1}_{\psi_1, e_i}(\underline{x}, \underline{z}) \text{ and } T_2 \vdash \psi_2 \to \neg \texttt{ImplDef}^{T_2}_{\psi_2, e_i}(\underline{x}, \underline{z})$$

Combined UIs Algorithm

Lemma (Main Lemma)

*Every **working formula** is equivalent (modulo $T_1 \cup T_2$) to a disjunction of **terminal** **working formulae**.*

Combined UIs Algorithm

Lemma (Main Lemma)

*Every **working formula** is equivalent (modulo $T_1 \cup T_2$) to a disjunction of **terminal** **working formulae**.*

Start from an Initial Formula. The non-deterministic procedure to compute the terminal working formulae applies one of the following **alternatives**:

(1) Add to ψ_1 a disjunct from the DNF of $\bigwedge_{e_i \in \underline{e}} \neg \mathtt{ImplDef}^{T_1}_{\psi_1, e_i}(\underline{x}, \underline{z})$ and to ψ_2 a disjunct from the DNF of $\bigwedge_{e_i \in \underline{e}} \neg \mathtt{ImplDef}^{T_2}_{\psi_2, e_i}(\underline{x}, \underline{z})$;

(2.i) Select $e_i \in \underline{e}$ and $h \in \{1, 2\}$; then add to ψ_h a disjunct L_{ij} from the DNF of $\mathtt{ImplDef}^{T_h}_{\psi_h, e_i}(\underline{x}, \underline{z})$; add $e_i = t_{ij}$ (where t_{ij} is the term mentioned in **Useful Terms Lemma**) to $\mathtt{ExplDef}(\underline{z}, \underline{x})$; the variable e_i becomes *defined*.

Combined UIs Algorithm

Lemma (Main Lemma)

*Every **working formula** is equivalent (modulo $T_1 \cup T_2$) to a disjunction of **terminal working formulae**.*

Start from an Initial Formula. The non-deterministic procedure to compute the terminal working formulae applies one of the following **alternatives**:

(1) Add to ψ_1 a disjunct from the DNF of $\bigwedge_{e_i \in \underline{e}} \neg \mathtt{ImplDef}^{T_1}_{\psi_1, e_i}(\underline{x}, \underline{z})$ and to ψ_2 a disjunct from the DNF of $\bigwedge_{e_i \in \underline{e}} \neg \mathtt{ImplDef}^{T_2}_{\psi_2, e_i}(\underline{x}, \underline{z})$;

(2.i) Select $e_i \in \underline{e}$ and $h \in \{1, 2\}$; then add to ψ_h a disjunct L_{ij} from the DNF of $\mathtt{ImplDef}^{T_h}_{\psi_h, e_i}(\underline{x}, \underline{z})$; add $e_i = t_{ij}$ (where t_{ij} is the term mentioned in **Useful Terms Lemma**) to $\mathtt{ExplDef}(\underline{z}, \underline{x})$; the variable e_i becomes *defined*.

The output is the disjunction of all possible outcomes.

Transfer of UIs

Proposition

*A **UI** of a terminal working formula can be obtained by unravelling the explicit definitions of the variables $\underline{z}$ from*
$\exists \underline{z}\ (\texttt{ExplDef}(\underline{z}, \underline{x}) \wedge \theta_1(\underline{x}, \underline{z}) \wedge \theta_2(\underline{x}, \underline{z}))$*, where* $\theta_1(\underline{x}, \underline{z})$ *is the* T_1*-**cover** of* $\exists \underline{e} \psi_1(\underline{x}, \underline{z}, \underline{e})$ *and* $\theta_2(\underline{x}, \underline{z})$ *is the* T_2*-**cover** of* $\exists \underline{e} \psi_2(\underline{x}, \underline{z}, \underline{e})$*.*

Transfer of UIs

Proposition

*A **UI** of a terminal working formula can be obtained by unravelling the explicit definitions of the variables* $\underline{z}$ *from* $\exists \underline{z}\ (\mathtt{ExplDef}(\underline{z}, \underline{x}) \wedge \theta_1(\underline{x}, \underline{z}) \wedge \theta_2(\underline{x}, \underline{z}))$*, where* $\theta_1(\underline{x}, \underline{z})$ *is the* T_1*-**cover** of* $\exists \underline{e} \psi_1(\underline{x}, \underline{z}, \underline{e})$ *and* $\theta_2(\underline{x}, \underline{z})$ *is the* T_2*-**cover** of* $\exists \underline{e} \psi_2(\underline{x}, \underline{z}, \underline{e})$*.*

From the Main Lemma, the Proposition and the 'UIs and QE' Theorem:

Transfer of UIs

Proposition

*A **UI** of a terminal working formula can be obtained by unravelling the explicit definitions of the variables* $\underline{z}$ *from*
$\exists \underline{z}\, (\mathtt{ExplDef}(\underline{z}, \underline{x}) \wedge \theta_1(\underline{x}, \underline{z}) \wedge \theta_2(\underline{x}, \underline{z}))$*, where* $\theta_1(\underline{x}, \underline{z})$ *is the* T_1*-**cover** of* $\exists \underline{e} \psi_1(\underline{x}, \underline{z}, \underline{e})$ *and* $\theta_2(\underline{x}, \underline{z})$ *is the* T_2*-**cover** of* $\exists \underline{e} \psi_2(\underline{x}, \underline{z}, \underline{e})$*.*

From the Main Lemma, the Proposition and the 'UIs and QE' Theorem:

Theorem

Let T_1, T_2 *be **convex, stably infinite, equality interpolating, universal theories over disjoint signatures admitting a model completion**. Then* $T_1 \cup T_2$ *admits a model completion too. UIs in* $T_1 \cup T_2$ *can be effectively **computed** as shown above.*

Transfer of UIs

Proposition

*A **UI** of a terminal working formula can be obtained by unravelling the explicit definitions of the variables* $\underline{z}$ *from* $\exists \underline{z}\,(\texttt{ExplDef}(\underline{z},\underline{x}) \wedge \theta_1(\underline{x},\underline{z}) \wedge \theta_2(\underline{x},\underline{z}))$*, where* $\theta_1(\underline{x},\underline{z})$ *is the* T_1*-**cover** of* $\exists \underline{e}\psi_1(\underline{x},\underline{z},\underline{e})$ *and* $\theta_2(\underline{x},\underline{z})$ *is the* T_2*-**cover** of* $\exists \underline{e}\psi_2(\underline{x},\underline{z},\underline{e})$*.*

From the Main Lemma, the Proposition and the 'UIs and QE' Theorem:

Theorem

Let T_1, T_2 *be **convex, stably infinite, equality interpolating, universal theories over disjoint signatures admitting a model completion**. Then* $T_1 \cup T_2$ *admits a model completion too. UIs in* $T_1 \cup T_2$ *can be effectively **computed** as shown above.*

In [CGG+22], it is also shown that equality interpolating is a **necessary condition** for obtaining UI transfer: already required for **minimal combinations** with signatures adding uninterpreted symbols.

Outline

Non-transfer of UIs in the Non-convex case

Convexity hypothesis cannot be eliminated.

Non-transfer of UIs in the Non-convex case

Convexity hypothesis cannot be eliminated.
Consider the **UI transfer** for $T_1 \cup T_2$, where:

- $T_1 :=$ integer difference logic $\mathcal{IDL}$ (integer numbers with successor and predecessor, 0 and the strict order $<$): it is *not* convex, but it satisfies the equality interpolating condition for non-convex theories.
- $T_2 := \mathcal{EUF}(\Sigma_f)$, where Σ_f has only one unary free function symbol f (not belonging to the signature of T_1).

Non-transfer of UIs in the Non-convex case

Convexity hypothesis cannot be eliminated.
Consider the **UI transfer** for $T_1 \cup T_2$, where:

- $T_1 :=$ integer difference logic $\mathcal{IDL}$ (integer numbers with successor and predecessor, 0 and the strict order $<$): it is *not* convex, but it satisfies the equality interpolating condition for non-convex theories.
- $T_2 := \mathcal{EUF}(\Sigma_f)$, where Σ_f has only one unary free function symbol f (not belonging to the signature of T_1).

Proposition

*Let T_1, T_2 be as above; the formula $\exists e\ (0 < e \wedge e < x \wedge f(e) = 0)$ does **not** have a UI in $T_1 \cup T_2$.*

Non-transfer of UIs in the Non-convex case

Convexity hypothesis cannot be eliminated.
Consider the **UI transfer** for $T_1 \cup T_2$, where:

- $T_1 :=$ integer difference logic $\mathcal{IDL}$ (integer numbers with successor and predecessor, 0 and the strict order $<$): it is *not* convex, but it satisfies the equality interpolating condition for non-convex theories.
- $T_2 := \mathcal{EUF}(\Sigma_f)$, where Σ_f has only one unary free function symbol f (not belonging to the signature of T_1).

Proposition

*Let T_1, T_2 be as above; the formula $\exists e\ (0 < e \wedge e < x \wedge f(e) = 0)$ does **not** have a UI in $T_1 \cup T_2$.*

The counterexample still applies when replacing integer difference logic with *linear integer arithmetics*.

Outline

Conclusions

- Problem of **combined UIs.**

Conclusions

- Problem of **combined UIs.**
- Sufficient and necessary conditions for transferring UIs to **combinations** in the *convex* case.

Conclusions

- Problem of **combined UIs.**
- Sufficient and necessary conditions for transferring UIs to **combinations** in the *convex* case.
- General method and algorithm for computing **combined UIs** for *convex* theories, based on the use of Beth definability.

Conclusions

- Problem of **combined UIs.**
- Sufficient and necessary conditions for transferring UIs to **combinations** in the *convex* case.
- General method and algorithm for computing **combined UIs** for *convex* theories, based on the use of Beth definability.
- Non-transfer of UIs in the *non-convex* case, in general.

Further Directions

- Investigate **UI transfer** for 'tame' theory combinations (*codomain* sorts are shared) [CGG+22];
- **UI transfer** properties for non-disjoint signatures combinations;

References

R. Bruttomesso, S. Ghilardi, and S. Ranise.
Quantifier-free interpolation in combinations of equality interpolating theories.
ACM Trans. Comput. Log., 15(1):5:1–5:34, 2014.

D. Calvanese, S. Ghilardi, A. Gianola, M. Montali, and A. Rivkin.
Model completeness, covers and superposition.
In *Proc. of CADE*, volume 11716 of *LNCS*. Springer, 2019.

D. Calvanese, S. Ghilardi, A. Gianola, M. Montali, and A. Rivkin.
Combination of uniform interpolants via Beth definability.
J. Autom. Reason., 2022.

S. Gulwani and M. Musuvathi.
Cover algorithms and their combination.
In *Proc. of ESOP, Held as Part of ETAPS*, pages 193–207, 2008.

G. Yorsh and M. Musuvathi.
A combination method for generating interpolants.
In *Proc. of CADE-20*, LNCS, pages 353–368. 2005.

THANKS FOR YOUR ATTENTION!

Combined Algorithm: an Example

Let T_1 be $\mathcal{EUF}(\Sigma)$ and T_2 be linear real arithmetic.

Combined Algorithm: an Example

Let T_1 be $\mathcal{EUF}(\Sigma)$ and T_2 be linear real arithmetic.

Covers are computed in real arithmetic by **quantifier elimination**, whereas for $\mathcal{EUF}(\Sigma)$ one can apply the **superposition-based algorithm** from [CGG⁺19].

Combined Algorithm: an Example

Let T_1 be $\mathcal{EUF}(\Sigma)$ and T_2 be linear real arithmetic.

Covers are computed in real arithmetic by **quantifier elimination**, whereas for $\mathcal{EUF}(\Sigma)$ one can apply the **superposition-based algorithm** from [CGG+19].

Consider the formula:

$$\exists e_1 \cdots \exists e_4 \quad \begin{pmatrix} e_1 = f(x_1) \ \wedge \ e_2 = f(x_2) \wedge \\ \wedge \ f(e_3) = e_3 \ \wedge \ \ f(e_4) = x_1 \wedge \\ \wedge \ x_1 + e_1 \leq e_3 \ \wedge \ e_3 \leq x_2 + e_2 \ \wedge \ e_4 = x_2 + e_3 \end{pmatrix}$$

Combined Algorithm: an Example

Let T_1 be $\mathcal{EUF}(\Sigma)$ and T_2 be linear real arithmetic.
Covers are computed in real arithmetic by **quantifier elimination**, whereas for $\mathcal{EUF}(\Sigma)$ one can apply the **superposition-based algorithm** from [CGG+19].
Consider the formula:

$$\exists e_1 \cdots \exists e_4 \quad \begin{pmatrix} e_1 = f(x_1) \ \wedge \ e_2 = f(x_2) \wedge \\ \wedge \ f(e_3) = e_3 \ \wedge \ \ f(e_4) = x_1 \wedge \\ \wedge \ x_1 + e_1 \leq e_3 \ \wedge \ e_3 \leq x_2 + e_2 \ \wedge \ e_4 = x_2 + e_3 \end{pmatrix}$$

Applying exhaustively Step (1) and Step (2.i), we get:

$$\begin{aligned} & [x_2 = 0 \ \wedge \ f(x_1) = x_1 \ \wedge \ x_1 \leq 0 \ \wedge \ x_1 \leq f(0)] \ \vee \\ & \vee \ [x_1 + f(x_1) < x_2 + f(x_2) \ \wedge \ x_2 \neq 0] \ \vee \\ & \vee \begin{bmatrix} x_2 \neq 0 \wedge x_1 + f(x_1) = x_2 + f(x_2) \ \wedge \ f(2x_2 + f(x_2)) = x_1 \wedge \\ \wedge f(x_1 + f(x_1)) = x_1 + f(x_1) \end{bmatrix} \end{aligned}$$

Artifact-Centric Systems

Artifact-Centric Systems: process-centric paradigm + data (**artifact** = *lifecycle* + *information model*).

Artifact-Centric Systems

Artifact-Centric Systems: process-centric paradigm + data
(**artifact** = *lifecycle* + *information model*).
They **can** be formalized using three components:

Artifact-Centric Systems

Artifact-Centric Systems: process-centric paradigm + data (**artifact** = *lifecycle* + *information model*).

They **can** be formalized using three components:

- *a read-only database (DB)*;

Artifact-Centric Systems

Artifact-Centric Systems: process-centric paradigm + data (**artifact** = *lifecycle* + *information model*).
They **can** be formalized using three components:

- *a read-only database (DB)*;
- *an artifact working memory (e.g., artifact variables + artifact relations)*;

Artifact-Centric Systems

Artifact-Centric Systems: process-centric paradigm + data (**artifact** = *lifecycle* + *information model*).
They **can** be formalized using three components:

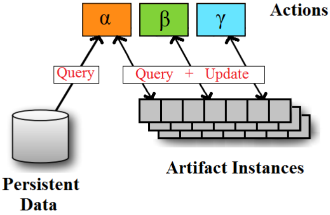
- *a read-only database (DB)*;
- *an artifact working memory (e.g., artifact variables + artifact relations)*;
- *actions* (also called *services*).

Artifact-Centric Systems

Artifact-Centric Systems: process-centric paradigm + data (**artifact** = *lifecycle* + *information model*).

They **can** be formalized using three components:

- *a read-only database (DB)*;
- *an artifact working memory (e.g., artifact variables + artifact relations)*;
- *actions* (also called *services*).

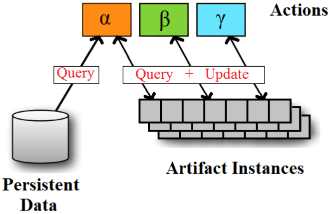


Artifact-Centric Systems

Artifact-Centric Systems: process-centric paradigm + data (**artifact** = *lifecycle* + *information model*).

They **can** be formalized using three components:

- *a read-only database (DB)*;
- *an artifact working memory (e.g., artifact variables + artifact relations)*;
- *actions* (also called *services*).



Artifact-Centric Systems $\Longrightarrow$ Array-based Systems $\Longrightarrow$ SMT-based tool **Model Checker Modulo Theories (MCMT)**

DB schemas

DB schemas: *read-only DB* of Artifact-Centric Systems, incorporating primary keys and foreign keys dependencies

DB schemas

DB schemas: *read-only DB* of Artifact-Centric Systems, incorporating primary keys and foreign keys dependencies

Definition

A DB schema is a pair (Σ, T), where:

- Σ is a *DB signature*, that is, a finite multi-sorted signature with equality, unary functions, n-ary relations and constants;
- T is a *DB theory*, that is, a set of universal Σ-sentences.

DB schemas

DB schemas: *read-only DB* of Artifact-Centric Systems, incorporating primary keys and foreign keys dependencies

Definition

A DB schema is a pair (Σ, T), where:

- Σ is a *DB signature*, that is, a finite multi-sorted signature with equality, unary functions, n-ary relations and constants;
- T is a *DB theory*, that is, a set of universal Σ-sentences.

In a *basic DB schema*, T is empty. $G(\Sigma)$: characteristic graph capturing the dependencies induced by functions over sorts.

DB schemas

DB schemas: *read-only DB* of Artifact-Centric Systems, incorporating primary keys and foreign keys dependencies

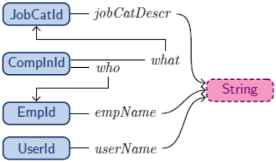
Definition

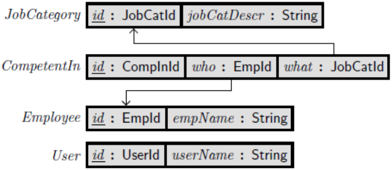
A DB schema is a pair (Σ, T), where:

- Σ is a *DB signature*, that is, a finite multi-sorted signature with equality, unary functions, n-ary relations and constants;
- T is a *DB theory*, that is, a set of universal Σ-sentences.

In a *basic DB schema*, T is empty. $G(\Sigma)$: characteristic graph capturing the dependencies induced by functions over sorts.

Example:





Array-based Artifact-Centric Systems: a simplified version

A **SAS** (**Simple Artifact Systems**) is a tuple
$\mathcal{S} = \langle \Sigma, T, \underline{x}, \iota(\underline{x}), \tau(\underline{x}, \underline{x}') \rangle$, where:

- (Σ, T) is a DB schema;
- $\underline{x}$ are individual FO variables representing the current state;
- ι is a Σ-formula representing the initialization;
- $\tau(\underline{x}, \underline{x}')$ is a Σ-formula representing the transitions from the **current state** $\underline{x}$ to the **new state** $\underline{x}'$.

Array-based Artifact-Centric Systems: a simplified version

A **SAS** (**Simple Artifact Systems**) is a tuple
$\mathcal{S} = \langle \Sigma, T, \underline{x}, \iota(\underline{x}), \tau(\underline{x}, \underline{x}') \rangle$, where:

- (Σ, T) is a DB schema;
- $\underline{x}$ are individual FO variables representing the current state;
- ι is a Σ-formula representing the initialization;
- $\tau(\underline{x}, \underline{x}')$ is a Σ-formula representing the transitions from the **current state** $\underline{x}$ to the **new state** $\underline{x}'$.

Individual variables $\underline{x}$

Artifact Variables (Working Memory)

Array-based Artifact-Centric Systems: a simplified version

A **SAS** (**Simple Artifact Systems**) is a tuple
$\mathcal{S} = \langle \Sigma, T, \underline{x}, \iota(\underline{x}), \tau(\underline{x}, \underline{x}') \rangle$, where:

- (Σ, T) is a DB schema;
- $\underline{x}$ are individual FO variables representing the current state;
- ι is a Σ-formula representing the initialization;
- $\tau(\underline{x}, \underline{x}')$ is a Σ-formula representing the transitions from the **current state** $\underline{x}$ to the **new state** $\underline{x}'$.

Individual variables $\underline{x}$ ⟷ Artifact Variables (Working Memory)

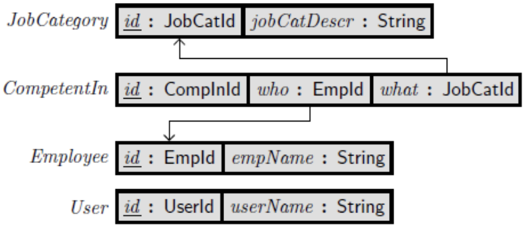
Individual variables change their value over the time, according to the *transitions* formula!

A simple example

Job Hiring Process:

$$\iota := (\mathsf{Applicant} = undef \wedge \mathsf{JobPos} = undef)$$

$$\tau := \exists \mathsf{UserID}, \mathsf{JobID} \begin{pmatrix} \mathsf{UserID} \neq undef \wedge \mathsf{JobID} \neq undef \wedge \mathsf{Applicant} = undef \wedge \\ \mathsf{JobPos} = undef \wedge \mathsf{Applicant}' := \mathsf{UserID} \wedge \mathsf{JobPos}' := \mathsf{JobID} \end{pmatrix}$$

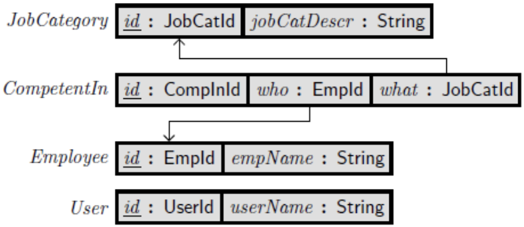


A simple example

Job Hiring Process:

$$\iota := (\mathsf{Applicant} = undef \wedge \mathsf{JobPos} = undef)$$

$$\tau := \exists \mathsf{UserID}, \mathsf{JobID} \begin{pmatrix} \mathsf{UserID} \neq undef \wedge \mathsf{JobID} \neq undef \wedge \mathsf{Applicant} = undef \wedge \\ \mathsf{JobPos} = undef \wedge \mathsf{Applicant}' := \mathsf{UserID} \wedge \mathsf{JobPos}' := \mathsf{JobID} \end{pmatrix}$$

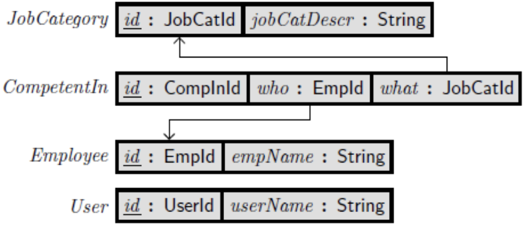


A simple example

Job Hiring Process:

$$\iota := (\mathsf{Applicant} = undef \wedge \mathsf{JobPos} = undef)$$

$$\tau := \exists \mathsf{UserID}, \mathsf{JobID} \begin{pmatrix} \mathsf{UserID} \neq undef \wedge \mathsf{JobID} \neq undef \wedge \mathsf{Applicant} = undef \wedge \\ \mathsf{JobPos} = undef \wedge \mathsf{Applicant}' := \mathsf{UserID} \wedge \mathsf{JobPos}' := \mathsf{JobID} \end{pmatrix}$$



Verification of safety in a SAS $\mathcal{S}$

A *safety* formula for $\mathcal{S}$: *generic* quantifier-free formula $\upsilon(\underline{x}) \implies$ *undesired states* of $\mathcal{S}$.

Verification of safety in a SAS $\mathcal{S}$

A *safety* formula for $\mathcal{S}$: *generic* quantifier-free formula $\upsilon(\underline{x}) \implies$ *undesired states* of $\mathcal{S}$.

$\mathcal{S}$ is safe wrt υ **iff** in no model $\mathcal{M}$ of (Σ, T), for no $k \geq 0$ and for no assignment in $\mathcal{M}$ to $\underline{x}^0, \ldots, \underline{x}^k$ (1) is true ($\underline{x}^i$ are renamed copies of $\underline{x}$):

$$\iota(\underline{x}^0) \wedge \tau(\underline{x}^0, \underline{x}^1) \wedge \cdots \wedge \tau(\underline{x}^{k-1}, \underline{x}^k) \wedge \upsilon(\underline{x}^k) \tag{1}$$

Verification of safety in a SAS $\mathcal{S}$

A *safety* formula for $\mathcal{S}$: *generic* quantifier-free formula $\upsilon(\underline{x}) \implies$ *undesired states* of $\mathcal{S}$.

$\mathcal{S}$ is safe wrt υ **iff** in no model $\mathcal{M}$ of (Σ, T), for no $k \geq 0$ and for no assignment in $\mathcal{M}$ to $\underline{x}^0, \dots, \underline{x}^k$ (1) is true ($\underline{x}^i$ are renamed copies of $\underline{x}$):

$$\iota(\underline{x}^0) \wedge \tau(\underline{x}^0, \underline{x}^1) \wedge \cdots \wedge \tau(\underline{x}^{k-1}, \underline{x}^k) \wedge \upsilon(\underline{x}^k) \tag{1}$$

Safety problem for $\mathcal{S}$: *given υ, decide if $\mathcal{S}$ is safe wrt υ.*

Verification of safety in a SAS $\mathcal{S}$

A *safety* formula for $\mathcal{S}$: *generic* quantifier-free formula $\upsilon(\underline{x}) \implies$ *undesired states* of $\mathcal{S}$.

$\mathcal{S}$ is safe wrt υ **iff** in no model $\mathcal{M}$ of (Σ, T), for no $k \geq 0$ and for no assignment in $\mathcal{M}$ to $\underline{x}^0, \ldots, \underline{x}^k$ (1) is true ($\underline{x}^i$ are renamed copies of $\underline{x}$):

$$\iota(\underline{x}^0) \wedge \tau(\underline{x}^0, \underline{x}^1) \wedge \cdots \wedge \tau(\underline{x}^{k-1}, \underline{x}^k) \wedge \upsilon(\underline{x}^k) \tag{1}$$

Safety problem for $\mathcal{S}$: *given* υ, *decide if* $\mathcal{S}$ *is safe wrt* υ.

Theorem (Soundness and Completeness)

Backward search is effective, correct and complete (the last one w.r.t. detecting unsafety) for the safety problems for SASs. If $G(\Sigma)$ is acyclic, backward search always terminates and it is a full decision procedure.