Resolution-Based Uniform Interpolation and Forgetting for Expressive Description Logics

Patrick Koopmann

December 12, 2017

Forgetting

Predicate forgetting

Given $\mathcal{L}$ sentence ϕ, predicate P,
compute ϕ^{-P} s.t.

- P does not occur in ϕ^{-P}
- for every $\mathcal{L}$ sentence ψ without P:
  - $\phi^{-P} \models \psi$ iff $\phi \models \psi$

Forgetting

Predicate forgetting

Given $\mathcal{L}$ sentence ϕ, predicate P,
compute ϕ^{-P} s.t.

- P does not occur in ϕ^{-P}
- for every $\mathcal{L}$ sentence ψ without P:
  - $\phi^{-P} \models \psi$ iff $\phi \models \psi$

Theorem for first order logic:

- Iff ϕ^{-P} exists, then $\phi^{-P} \equiv \exists P.\phi$

Uniform Interpolation

Craig Interpolation:

- Given $F \models G$, compute *interpolant* I s.t.
  - $F \models I$,
  - $I \models G$
  - I contains only symbols common to F and G

Uniform Interpolation

Craig Interpolation:

- Given $F \models G$, compute *interpolant* I s.t.
  - $F \models I$,
  - $I \models G$
  - I contains only symbols common to F and G

Uniform Interpolation

Given

- formula F
- signature Σ of symbols

compute *uniform interpolant* (UI) F^Σ s.t.

- F^Σ only uses symbols from Σ
- for every ψ in Σ, $F \models \psi$ iff $F^\Sigma \models \psi$

Uniform Interpolation

Craig Interpolation:

- Given $F \models G$, compute *interpolant* I s.t.
  - $F \models I$,
  - $I \models G$
  - I contains only symbols common to F and G

Uniform Interpolation

Given

- formula F
- signature Σ of symbols

compute *uniform interpolant* (UI) F^{Σ} s.t.

- F^{Σ} only uses symbols from Σ
- for every ψ in Σ, $F \models \psi$ iff $F^{\Sigma} \models \psi$

Dual to Forgetting:

- UI for $\Sigma \Leftrightarrow$ forget everything not in Σ

Uniform Interpolation

Input Ontology

$$Male \sqcap Female \sqsubseteq \bot$$
$$\top \sqsubseteq \forall hasParent.Parent$$
$$Parent \sqsubseteq Male \sqcup Female$$
$$\mathbf{Father} \equiv Parent \sqcap Male$$
$$\mathbf{Mother} \equiv Parent \sqcap Female$$
$$\mathbf{Orphan} \equiv \forall hasParent.\neg Alive$$

$$hasParent(\mathbf{peter}, \mathbf{thomas})$$
$$Male(\mathbf{thomas}) \qquad Alive(\mathbf{thomas})$$
$$hasParent(\mathbf{thomas}, \mathbf{ingrid})$$

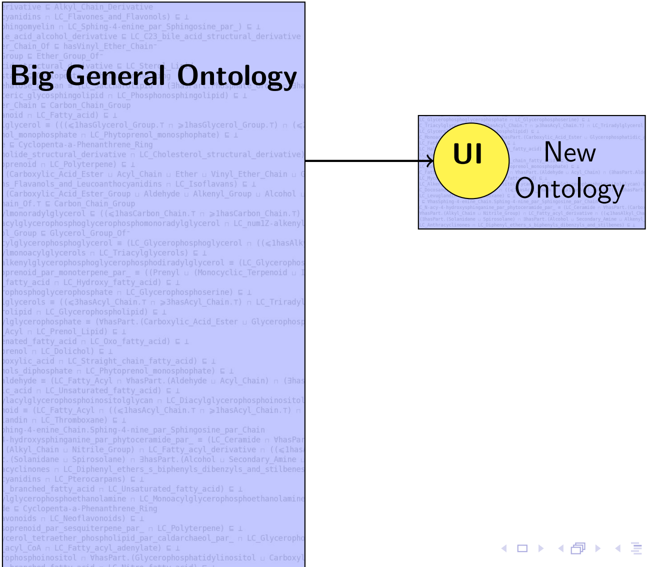
⇒

Uniform Interpolant

$$\mathbf{Father} \sqcap \mathbf{Mother} \sqsubseteq \bot$$

$$\neg\mathbf{Orphan}(\mathbf{peter})$$
$$\mathbf{Father}(\mathbf{thomas})$$
$$(\mathbf{Father} \sqcup \mathbf{Mother})(\mathbf{ingrid})$$

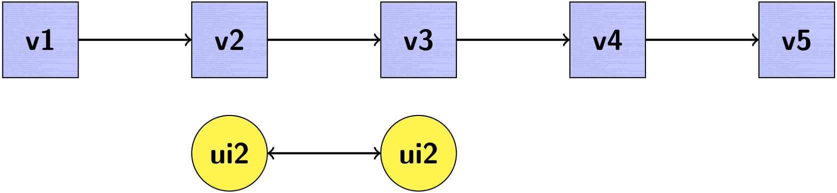
Ontology Reuse



Explore Hidden Relations

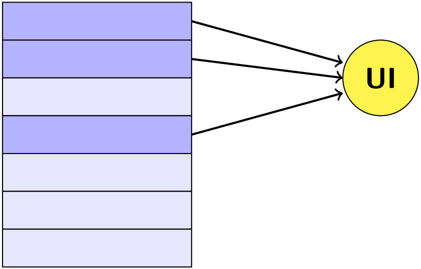
- Select concept and role names of interest
- Make relations between them explicit

Logical Difference



- Compare ontology versions
- Capture all new entailments in signature Σ:
  - $\mathsf{logDiff}(\mathcal{T}_1, \mathcal{T}_2, \Sigma) = \{\alpha \in \mathcal{T}_2^{\Sigma} \mid \mathcal{T}_1 \not\models \alpha\}$
- Σ: common signature, or set of "core" symbols

Module Extraction



Subsumption modules:

- Subset of the ontology preserving entailments in signature
- UI + axiom pinpointing/justification

Applications of UI

- Further applications:
  - Multi-agent systems
  - Conflict resolution
  - Abduction (see later talk)
- Similar applications in modal logics
  - Most techniques presented here also apply to modal logics

Applications of UI

- Further applications:
  - Multi-agent systems
  - Conflict resolution
  - Abduction (see later talk)
- Similar applications in modal logics
  - Most techniques presented here also apply to modal logics
- Not an application: modal correspondence
  - Apply SOQE to obtain *frame properties*:

$$\forall p : \Box\Box p \rightarrow \Box p \qquad \Longleftrightarrow \qquad \forall xyz.(r(x, y) \wedge r(y, z) \rightarrow r(x, z))$$

  - Requires elimination to preserve *all models*
  - UI only preserves entailments *in language under consideration*

Expressive Description Logics

Concepts $\mathcal{ALC}$

$$\bot \mid \top \mid A \mid \neg C \mid C \sqcup D \mid C \sqcap D \mid \exists r.C \mid \forall r.C$$

TBox Axioms $\mathcal{ALC}$

$$C \sqsubseteq D \mid C \equiv D$$

ABox Axioms $\mathcal{ALC}$

$$C(a) \mid r(a, b)$$

$\mathcal{ALCH}$:	Role Hierarchies	$r \sqsubseteq s$
$\mathcal{ALCF}$:	Local Functionality	$\leq 1r.\top, \geq 2r.\top$
$\mathcal{SH}$:	Transitive Roles	$trans(r)$
$\mathcal{SHQ}$:	Number Restrictions	$\geq nr.C, \leq nr.C$
$\mathcal{SHI}$:	Inverse Roles	r^{-1}

Example

UI of Pizza ontology, for 10 most frequent concept and role names

$\exists \mathit{hasTopping}.\top \sqsubseteq \mathit{Pizza}$ $\quad \top \sqsubseteq \forall \mathit{hasTopping}.\mathit{PizzaTopping}$

$\exists \mathit{hasSpiciness}.(\mathit{Pizza} \sqcup \mathit{PizzaTopping}) \sqsubseteq \bot$

$\mathit{NamedPizza} \sqsubseteq \mathit{Pizza}$ $\quad \mathit{VegetableTopping} \sqsubseteq \mathit{PizzaTopping}$

$\mathit{MozzarellaTopping} \sqsubseteq \mathit{PizzaTopping} \sqcap \exists \mathit{hasSpiciness}.\mathit{Mild}$

$\mathit{OliveTopping} \sqsubseteq \mathit{VegetableTopping} \sqcap \exists \mathit{hasSpiciness}.\mathit{Mild}$

$\mathit{TomatoTopping} \sqsubseteq \mathit{VegetableTopping} \sqcap \exists \mathit{hasSpiciness}.\mathit{Mild}$

$\mathit{Pizza} \sqcap \mathit{Mild} \sqsubseteq \bot$ $\quad \mathit{Pizza} \sqcap \mathit{PizzaTopping} \sqsubseteq \bot$ $\quad \mathit{PizzaTopping} \sqcap \mathit{Mild} \sqsubseteq \bot$

$\mathit{MozzarellaTopping} \sqcap \mathit{VegetableTopping} \sqsubseteq \bot$

$\mathit{OliveTopping} \sqcap \mathit{TomatoTopping} \sqsubseteq \bot$

Relation to Modal Logic

- There is a direct relation to multi-modal logics:
  - $\exists r.C$ corresponds to $\Diamond_r.C'$
  - $\forall r.C$ corresponds to $\Box_r.C'$
  - $\exists r^{-}.C$ corresponds to $\Diamond_{\breve{r}}.C'$
  - number restrictions correspond to graded modalities
  - transitivity as in **S**4 for selected roles
- Concepts correspond to modal logic formulae
- But: TBox axioms hold globally

Relation to Second-Order Quantifier Elimination

In first order logic, forgetting corresponds to SOQE:

- Iff ϕ^{-P} exists, then $\phi^{-P} \equiv \exists P.\phi$

This does not apply in the logics considered

- Consider $\top \sqsubseteq \exists r.A \sqcap \exists r.\neg A$
- Forgetting A from the FO-representation yields:

$$\begin{aligned} &\exists A.\forall x \exists yz.\big(r(x,y) \wedge A(y) \wedge r(x,z) \wedge \neg A(z)\big) \\ \equiv\ &\forall x \exists yz.\big(r(x,y) \wedge r(x,z) \wedge y \neq z\big) \end{aligned}$$

In $\mathcal{ALC}$, the UI is just:

$$\top \sqsubseteq \exists r.\top$$

Challenges Uniform Interpolation

- A lot of modal logics have uniform interpolation:
  - **K**, **IPC**, **GL**, **S4Grz** [Visser, 1996]
  - modal μ-calculus [D'Agostino, Hollenberg, 1996]
- In most DLs, TBoxes break this property
  - Consider:

$$A \sqsubseteq B \qquad B \sqsubseteq \exists r.B \qquad\qquad \Sigma = \{A, r\}$$

  - UI for Σ:

$$A \sqsubseteq \exists r.\exists r.\exists r.\exists r.\exists r.\exists r.\exists r.\exists r.\exists r.\exists r.\exists r.\exists r.\exists r.\exists r.\exists r.\exists r.\exists r.\ldots$$

Challenges Uniform Interpolation in DLs

- In general, we may have to approximate or to use more expressive DLs
- Deciding existence of UIs in $\mathcal{ALC}$ is $2\textsc{ExpTime}$-complete
- A second challenge is *size*
  - If exists, $\mathcal{T}^{\Sigma}$ can have size $O\left(2^{2^{2^{|\mathcal{T}|}}}\right)$
    - Already for lightweight DL $\mathcal{EL}$

$\mathcal{ALC}$: [Lutz, Wolter, 2010], $\mathcal{EL}$: [Nikitina, Rudolph, 2014]

Computing Uniform Interpolants Practically

Can we compute uniform interpolants practically?

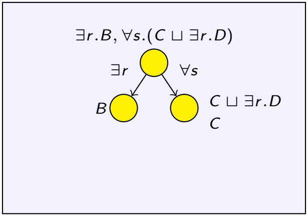
- Upper bound on size directly gives us a method for computing UIs:
  1. Iterate over all axioms in signature of size $2^{2^{2^{|\mathcal{T}|}}}$
  2. Collect all those that are entailed

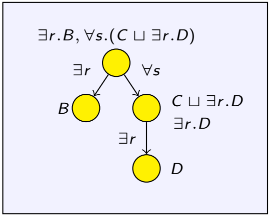
⇒ However, this is not practical at all!

Using Tableaux to Compute Uniform Interpolants

First more "practical" idea: use tableaux

- Directly generate entailed axioms
- Each tree corresponds to disjunct in result
- Different edges for ∃- and ∀-restrictions





Modal Logic: [Kracht, 2007], $\mathcal{ALC}$: [Wang, Wang et al, 2010]

Using Tableaux to Compute Uniform Interpolants

Obtain $\top \sqsubseteq C_1 \sqcup \ldots \sqcup C_n$ from tableau

- Each C_i constructed from one tree
- Only keep what is in signature

With TBox, tableau might not be finite

- Allows to compute arbitrary *approximations*
- Equivalence test to check for termination
  - last approximation equivalent to current

Using Tableaux to Compute Uniform Interpolants

Disadvantages of approach:

- Result big disjunction
  - Unusual representation for ontologies
- Expansions not *goal-oriented*
- Expensive termination condition

Using Resolution to Compute Uniform Interpolation

Resolution addresses short-comings

- Usually works on *conjunctive* normal forms
  - Conjunction of disjunctions
  - Closer to typical shape of ontologies
- Infers information on specific symbol

Prop. Resolution

$$\frac{C_1 \vee p \quad C_2 \vee \neg p}{C_1 \vee C_2}$$

First Order Resolution

$$\frac{C_1 \vee P(s_1, \ldots, s_n) \quad C_2 \vee \neg P(t_1, \ldots, t_n)}{C_1 \vee C_2 \vee s_1 \neq t_1 \vee \ldots \vee s_n \neq t_n}$$

Using SCAN to Compute Uniform Interpolants

Main idea used by SOQE method SCAN [Gabbay, Ohlbach, 1992]

1. Clausify input formula
2. Infer all inferences on predicate to eliminate
3. Filter out occurrences of that predicate
4. Deskolemise resulting set of clauses

Using SCAN to Compute Uniform Interpolants

Let's try it!

We want to forget B from following ontology:

$$A \sqsubseteq \forall r.B \qquad\qquad C \sqsubseteq \exists r.\neg B$$

Using SCAN to Compute Uniform Interpolants

Let's try it!

We want to forget B from following ontology:

$$A \sqsubseteq \forall r.B \qquad C \sqsubseteq \exists r.\neg B$$

Representation as First-Order clauses:

$$1. \neg A(x) \vee \neg r(x,y) \vee B(y) \qquad 2. \neg C(x) \vee r(x, f(x))$$
$$3. \neg C(x) \vee \neg B(f(x))$$

Using SCAN to Compute Uniform Interpolants

Representation as First-Order clauses:

1. $\neg A(x) \vee \neg r(x, y) \vee B(y)$
2. $\neg C(x) \vee r(x, f(x))$
3. $\neg C(x) \vee \neg B(f(x))$

Inferences on B:

4. $\neg A(x) \vee \neg r(x, y) \vee \neg C(x) \vee y \neq f(x)$ (Resolution 1,3)
5. $\neg A(x) \vee \neg r(x, f(x)) \vee \neg C(x)$ (Constr. Elim.)

Using SCAN to Compute Uniform Interpolants

Representation as First-Order clauses:

1. $\neg A(x) \vee \neg r(x,y) \vee B(y)$
2. $\neg C(x) \vee r(x, f(x))$
3. $\neg C(x) \vee \neg B(f(x))$

Inferences on B:

4. $\neg A(x) \vee \neg r(x,y) \vee \neg C(x) \vee y \neq f(x)$ (Resolution 1,3)
5. $\neg A(x) \vee \neg r(x, f(x)) \vee \neg C(x)$ (Constr. Elim.)

$\Rightarrow$ SCAN terminates, but we have insufficient information for UI!

Using SCAN to Compute Uniform Interpolants

Representation as First-Order clauses:

1. $\neg A(x) \vee \neg r(x,y) \vee B(y)$
2. $\neg C(x) \vee r(x, f(x))$
3. $\neg C(x) \vee \neg B(f(x))$

Inferences on B:

4. $\neg A(x) \vee \neg r(x,y) \vee \neg C(x) \vee y \neq f(x)$ (Resolution 1,3)
5. $\neg A(x) \vee \neg r(x, f(x)) \vee \neg C(x)$ (Constr. Elim. 4)

Additional steps complete the picture:

6. $\neg A(x) \vee \neg C(x) \vee f(x) \neq f(x)$ (Resolution 2,5)
7. $\neg A(x) \vee \neg C(x)$ (Constr. Elim. 6)

Using SCAN to Compute Uniform Interpolants

Complete Clause Set:

1. $\neg A(x) \vee \neg r(x, y) \vee B(y)$
2. $\neg C(x) \vee r(x, f(x))$
3. $\neg C(x) \vee \neg B(f(x))$
4. $\neg A(x) \vee \neg r(x, y) \vee \neg C(x) \vee y \neq f(x)$
5. $\neg A(x) \vee \neg r(x, f(x)) \vee \neg C(x)$
6. $\neg A(x) \vee \neg C(x) \vee f(x) \neq f(x)$
7. $\neg A(x) \vee \neg C(x)$

Uniform Interpolant:

$$C \sqsubseteq \exists r.\top \qquad A \sqcap C \sqsubseteq \bot$$

Using Resolution to Compute Uniform Interpolants

- Downsides of SCAN:
  - Infers too much
  - Infers too little

- Needed: More than just SOQE

$\Rightarrow$ Infer consequences that are
  - in target signature
  - translate to logic under consideration

- More direct approach:
  - Stay in logic under consideration

Uniform Interpolation Using Modal Resolution

- Idea first followed by [Herzig and Mengin, 2008] for modal logic **K**
- Based on resolution calculus for modal logics by [Enjalbert and Fariñas, 1985]
- Allow to resolve on arbitrary levels of formula:

$$\begin{array}{l} C_1 \vee \Diamond\Box(C_2 \vee p) \\ C_3 \vee \Diamond\Diamond(C_4 \vee \neg p) \end{array} \implies C_1 \vee C_3 \vee \Diamond\Diamond(C_2 \vee C_4)$$

- Idea: use system of “meta”-rules to generate rules with arbitrary nesting depth

Modal Resolution after Enjalbert and Fariñas

- Normal form assumes DNF/CNF on each level of formula
  - CNF under diamond: $\Diamond(C_1, \ldots, C_n)$
  - DNF under box: $\Box(T_1 \vee \ldots \vee T_n)$
- Base rules:

$$C_1 \vee \Box\bot, \quad C_2 \vee \Diamond E \Longrightarrow_\alpha C_1 \vee C_2$$
$$C_1 \vee p, \quad C_2 \vee \neg p \Longrightarrow_\alpha C_1 \vee C_2$$

- Extended rules provided $C_1, C_2 \Longrightarrow_\alpha C_3$ / $C_1 \Longrightarrow_\alpha C_2$

$$C_1' \vee \Box C_1, \quad C_2' \vee \Diamond(C_2, E) \Longrightarrow_\alpha C_1' \vee C_2' \vee \Diamond(C_2, E, C_3)$$
$$C_1' \vee \Box C_1, \quad C_2' \vee \Box C_2 \Longrightarrow_\alpha C_1' \vee C_2' \vee \Box C_3$$
$$C \vee \Diamond(C_1, C_2, E) \Longrightarrow_\alpha C \vee \Diamond(C_1, C_2, E, C_3)$$
$$C \vee \Diamond(C_1, E) \Longrightarrow_\alpha C \vee \Diamond(C_1, E, C_2)$$
$$C \vee \Box C_1 \Longrightarrow_\alpha C \vee \Box C_2$$

Computing UIs using Modal Resolution

- UI computed similar to SCAN:
  - Compute all resolvents on symbols to forget
- Forms complete method for modal logic **K**
- Termination assured by maximum nesting depth in **K**

- Extended to $\mathcal{ALC}$ in [Ludwig and Konev, 2014]:
  - First practical method for UI in $\mathcal{ALC}$
  - Additional rules to handle TBox axioms
  - Termination cannot be guaranteed, but arbitrary approximations computed

Computing UIs by Modal Resolution

- Goal-oriented approach allows for practicality
- The cost is completeness

$$A \sqsubseteq B \qquad B \sqsubseteq C \sqcup \exists r.B \qquad A \sqsubseteq \forall r.\forall r.\bot$$
$$\Sigma = \{A, C, r\}$$

- Computing inferences only on B will not terminate
- However, there is a uniform interpolant for $\{A, C, r\}$:

$$A \sqsubseteq C \sqcup \exists r.C \qquad A \sqsubseteq \forall r.\forall r.\bot$$

- Probably in general no easy solution

Computing UIs using Resolution

- What to do about termination problem?

Computing UIs using Resolution

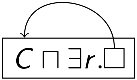
- What to do about termination problem?

$\Rightarrow$ Move to language that *has* uniform interpolation!

DLs With Greatest Fixpoint Operators

- New concept constructor $\nu X.C[X]$
  - $C[X]$: concept that contains X only positively
- Allows to represent *loops*:

$$A \sqcap \nu X.(C \sqcap \exists r.X) \iff A \sqcap$$



$$\iff A \sqcap (C \sqcap \exists r.(C \sqcap \exists r.(C \sqcap \exists r.(\ldots))))$$

Example

- Consider the following ontology:

$$A \sqsubseteq B \sqcup C \qquad B \sqsubseteq \exists r.B \qquad C \sqsubseteq \forall r.\neg B$$

- No UI for $\Sigma = \{A, C, r\}$ in $\mathcal{ALC}$

- However, in $\mathcal{ALC}\nu$, we have the following UI:

$$C \sqsubseteq \forall r.(\neg A \sqcup C) \qquad A \sqsubseteq C \sqcup \nu X.(\neg C \sqcap \exists r.X)$$

DLs with Greatest Fixpoint Operators

- Greatest fixpoint operators give us uniform interpolation in $\mathcal{ALC}$–$\mathcal{ALCHI}$
- They can be easily approximated:

$$\nu X.C[X] \approx C[C[C[C[C[\top]]]]]$$

- They can be "simulated" using auxiliary concept names:

$$A \sqsubseteq \nu X.C[X] \quad \text{becomes} \quad A \sqsubseteq D, \quad D \sqsubseteq C[D]$$

Flattened Approach

Final approach:

- Uses resolution
- Uses *flattened* normal form to ensure termination
- Always terminates for $\mathcal{ALCH}\nu$ ($\mathcal{ALCH}$+greatest fixpoints)
    - Fixpoints can then be approximated or simulated in $\mathcal{ALCH}$

Normal form, $\mathcal{ALCH}$

$\mathcal{ALCH}$ Clause

$$r \sqsubseteq s$$

$$\top \sqsubseteq L_1 \sqcup \ldots \sqcup L_n \qquad L_i\text{: } \mathcal{ALC} \text{ literal}$$

$\mathcal{ALC}$ Literal

$$A \mid \neg A \mid \exists r.D \mid \forall r.D$$

A: any concept name, D: definer symbol

- Definer symbols: special concept names, not part of signature
- Invariant: max 1 negative definer symbol per clause

  $\Rightarrow$ $\neg D_1 \sqcup \exists r.D_2 \sqcup \neg B$, ~~$\neg D_1 \sqcup \neg D_2 \sqcup A$~~

Definer symbols

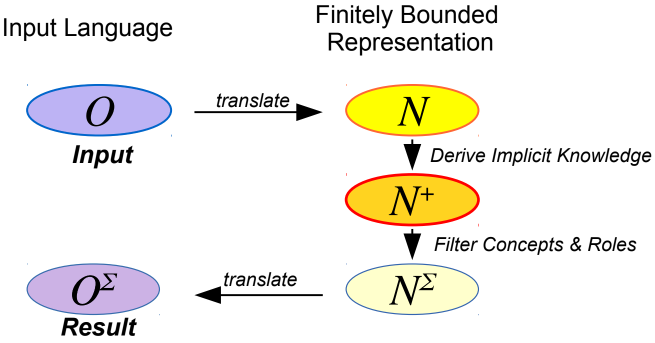
Invariant: max 1 negative definer symbol per clause

- Allows easy translation to clausal form and back:

$$C_1 \sqcup Qr.C_2 \quad \Longleftrightarrow \quad C_1 \sqcup Qr.D_1, \quad \neg D_1 \sqcup C_2$$
$$C_1 \sqcup \nu X.C_2[X] \quad \Longleftrightarrow \quad C_1 \sqcup Qr.D_1, \quad \neg D_1 \sqcup C_2[D]$$

- New definer symbols introduced by calculus
  - At most exponentially many

Basic Method



Rules of the Calculus

Concept forgetting in $\mathcal{ALC}$ uses two rules

Resolution

$$\frac{C_1 \sqcup A \qquad C_2 \sqcup \neg A}{C_1 \sqcup C_2}$$

Role Propagation

$$\frac{C_1 \sqcup \forall r.D_1 \qquad C_2 \sqcup Qr.D_2}{C_1 \sqcup C_2 \sqcup Qr.D_{12}}$$

- where $Q \in \{\forall, \exists\}$
- D_{12} is a possibly new definer representing $D_1 \sqcap D_2$
- side condition: $C_1 \sqcup C_2$ does not contain more than one negative definer literal

Example

Assume the following ontology:

$$C_1 \sqsubseteq \exists r.A$$
$$C_2 \sqsubseteq \forall r.(B \sqcup \neg A)$$

Normalisation brings four clauses:

$$\neg C_1 \sqcup \exists r.D_1 \qquad \neg D_1 \sqcup A$$
$$\neg C_2 \sqcup \forall r.D_2 \qquad \neg D_2 \sqcup B \sqcup \neg A$$

Example

$\neg D_1 \sqcup A$
$\neg C_1 \sqcup \exists r.D_1$

$\neg D_2 \sqcup B \sqcup \neg A$
$\neg C_2 \sqcup \forall r.D_2$

Example

Cannot resolve due invariant

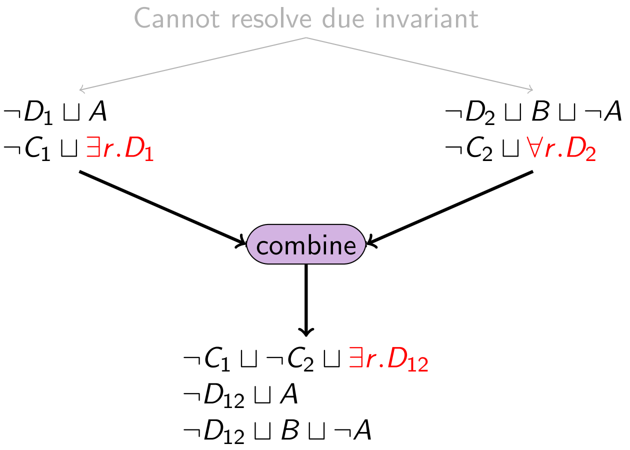
$\neg D_1 \sqcup A$

$\neg C_1 \sqcup \exists r.D_1$

$\neg D_2 \sqcup B \sqcup \neg A$

$\neg C_2 \sqcup \forall r.D_2$

Example


Cannot resolve due invariant
$\neg D_1 \sqcup A$
$\neg C_1 \sqcup \exists r.D_1$
$\neg D_2 \sqcup B \sqcup \neg A$
$\neg C_2 \sqcup \forall r.D_2$
combine
$\neg C_1 \sqcup \neg C_2 \sqcup \exists r.D_{12}$
$\neg D_{12} \sqcup A$
$\neg D_{12} \sqcup B \sqcup \neg A$

Example

Cannot resolve due invariant

$\neg D_1 \sqcup A$

$\neg C_1 \sqcup \exists r.D_1$

$\neg D_2 \sqcup B \sqcup \neg A$

$\neg C_2 \sqcup \forall r.D_2$

combine

$\neg C_1 \sqcup \neg C_2 \sqcup \exists r.D_{12}$

$\neg D_{12} \sqcup A$

$\neg D_{12} \sqcup B \sqcup \neg A$

Resolves to $\neg D_{12} \sqcup B$

Example

Final clause set:

$\neg C_1 \sqcup \exists r.D_1$

$\neg C_2 \sqcup \forall r.D_2$

$\neg C_1 \sqcup \neg C_2 \sqcup \exists r.D_{12}$

$\neg D_{12} \sqcup D_1$

$\neg D_{12} \sqcup B$

$\neg D_1 \sqcup A$

$\neg D_2 \sqcup B \sqcup \neg A$

$\neg D_{12} \sqcup D_2$

We obtain as uniform interpolant for $\{r, B, C_1, C_2\}$:

$$C_1 \sqsubseteq \exists r.\top \qquad C_2 \sqsubseteq \forall r.\top \qquad C_1 \sqcap C_2 \sqsubseteq \exists r.B$$

Forgetting Concept and Role Names in $\mathcal{ALCH}$

∃-elimination

$$\frac{C \sqcup \exists r.D \qquad \neg D}{C}$$

Role hierarchy

$$\frac{r \sqsubseteq s \qquad s \sqsubseteq t}{r \sqsubseteq t}$$

Universal roles

$$\frac{C_1 \sqcup \forall s.D_1 \qquad r \sqsubseteq s}{C_1 \sqcup \forall r.D_1}$$

Existential roles

$$\frac{C_1 \sqcup \exists s.D_1 \qquad s \sqsubseteq r}{C_1 \sqcup \exists r.D_1}$$

$\Rightarrow$ Rules form *refutational* and *interpolation* complete calculus

Forgetting Role Names

Alternative rule allows for more convenient implementation

Provided $\mathcal{T} \models D_0 \sqcap \ldots \sqcap D_n \sqcap D \sqsubseteq \bot$, apply:

Role Restriction Resolution

$$\frac{C_0 \sqcup \forall r.D_0 \qquad \ldots \qquad C_n \sqcup \forall r.D_n \qquad C \sqcup \exists r.D}{C_0 \sqcup \ldots \sqcup C_n \sqcup C}$$

- Side condition: $C_0 \sqcup \ldots \sqcup C_n \sqcup C$ does not contain more than one negative definer literal

$\Rightarrow$ Use external reasoner

Forgetting Algorithm

To eliminate (concept/role) name X:

1. Determine literals that allow for inference on name
2. If result would break invariant:
   - Check whether role propagation makes inference possible
   - Evt. recursively call Step 2

Forgetting Algorithm

To eliminate (concept/role) name X:

1. Determine literals that allow for inference on name
2. If result would break invariant:
   - Check whether role propagation makes inference possible
   - Evt. recursively call Step 2

General Algorithm:

1. Process names by number of occurrences
2. Use simplification heuristics at each step to keep result small
   - Determine *tautological fixpoints*: $\nu X.C[X]$ where $C[\top] = \top$

Flattened Approach

- General structure of calculus:
  1. Resolution-like rule (Resolution, ∃-elimination, etc.)
  2. Combination rule (role propagation rule)
- Purpose of combination rule is to introduce definers
- More combination rules possible in more expressive DLs

Functional Role Restrictions

$\mathcal{ALCF}$ has constructors $\leq 1r.\top$ and $\geq 2r.\top$

$\Rightarrow$ local functionality and its complement

Universalisation

$$\frac{C_1 \sqcup \exists r.D_1 \qquad C_2 \sqcup \leq 1r.\top}{C_1 \sqcup C_2 \sqcup \forall r.D_1}$$

∃∃-Role Propagation

$$\frac{C_1 \sqcup \exists r.D_1 \qquad C_2 \sqcup \exists r.D_2}{C_1 \sqcup C_2 \sqcup \exists r.D_{12} \sqcup \geq 2r.\top}$$

Example Functional Role Restrictions

Example:

$$A \sqsubseteq \exists r.B \qquad A \sqsubseteq \exists r.\neg B$$

Clauses:

1. $\neg A \sqcup \exists r.D_1$
2. $\neg D_1 \sqcup A$
3. $\neg A \sqcup \exists r.D_2$
4. $\neg D_2 \sqcup \neg A$

Example Functional Role Restrictions

Clauses:

1. $\neg A \sqcup \exists r.D_1$
2. $\neg D_1 \sqcup A$
3. $\neg A \sqcup \exists r.D_2$
4. $\neg D_2 \sqcup \neg A$

Inferences:

5. $\neg A \sqcup \exists r.D_{12} \sqcup \geq 2 r.\top$ (∃∃-Role Prop. 1,3)

Example Functional Role Restrictions

Clauses:

1. $\neg A \sqcup \exists r.D_1$
2. $\neg D_1 \sqcup A$
3. $\neg A \sqcup \exists r.D_2$
4. $\neg D_2 \sqcup \neg A$

Inferences:

5. $\neg A \sqcup \exists r.D_{12} \sqcup \geq 2 r.\top$ ($\exists\exists$-Role Prop. 1,3)
6. $\neg D_{12} \sqcup A$ ($D_{12} \sqsubseteq D_1$)
7. $\neg D_{12} \sqcup \neg A$ ($D_{12} \sqsubseteq D_2$)

Example Functional Role Restrictions

Clauses:

1. $\neg A \sqcup \exists r.D_1$
2. $\neg D_1 \sqcup A$
3. $\neg A \sqcup \exists r.D_2$
4. $\neg D_2 \sqcup \neg A$

Inferences:

5. $\neg A \sqcup \exists r.D_{12} \sqcup \geq 2r.\top$ (∃∃-Role Prop. 1,3)
6. $\neg D_{12} \sqcup A$ ($D_{12} \sqsubseteq D_1$)
7. $\neg D_{12} \sqcup \neg A$ ($D_{12} \sqsubseteq D_2$)
8. $\neg D_{12}$ (Resolution 6,7)

Example Functional Role Restrictions

Clauses:

1. $\neg A \sqcup \exists r.D_1$
2. $\neg D_1 \sqcup A$
3. $\neg A \sqcup \exists r.D_2$
4. $\neg D_2 \sqcup \neg A$

Inferences:

5. $\neg A \sqcup \exists r.D_{12} \sqcup \geq 2r.\top$ (∃∃-Role Prop. 1,3)
6. $\neg D_{12} \sqcup A$ ($D_{12} \sqsubseteq D_1$)
7. $\neg D_{12} \sqcup \neg A$ ($D_{12} \sqsubseteq D_2$)
8. $\neg D_{12}$ (Resolution 6,7)
9. $\neg A \sqcup \geq 2r.\top$ (∃-elimination 5,8)

Functional Role Restrictions

Example:

$$A \sqsubseteq \exists r.B \qquad A \sqsubseteq \exists r.\neg B$$

Clauses:

1. $\neg A \sqcup \exists r.D_1$
2. $\neg D_1 \sqcup A$
3. $\neg A \sqcup \exists r.D_2$
4. $\neg D_2 \sqcup \neg A$
5. $\neg A \sqcup \exists r.D_{12} \sqcup {\geq}2r.\top$
6. $\neg D_{12} \sqcup A$
7. $\neg D_{12} \sqcup \neg A$
8. $\neg D_{12}$
9. $\neg A \sqcup {\geq}2r.\top$

Uniform interpolant for $\Sigma = \{A, r\}$:

$$A \sqsubseteq {\geq}2r.\top$$

General Number Restrictions

Rules can be generalised to support qualified number restrictions

$\leq\leq$-Combination:

$$\frac{C_1 \sqcup {\leq} n_1 r_1.\neg D_1 \qquad C_2 \sqcup {\leq} n_2 r_2.\neg D_2 \qquad r \sqsubseteq r_1 \qquad r \sqsubseteq r_2}{C_1 \sqcup C_2 \sqcup {\leq}(n_1 + n_2) r.\neg D_{12}}$$

$\geq\leq$-Combination:

$$\frac{C_1 \sqcup {\geq} n_1 r_1.(D_1 \sqcup \ldots \sqcup D_m) \qquad C_2 \sqcup {\leq} n_2 r_2.\neg D_a \qquad r_1 \sqsubseteq_{\mathcal{R}} r_2}{C_1 \sqcup C_2 \sqcup {\geq}(n_1 - n_2) r_1.(D_{1a} \sqcup \ldots \sqcup D_{ma})}$$

$\leq\geq$-Combination:

$$\frac{C_1 \sqcup {\leq} n_1 r_1.\neg D_1 \qquad C_2 \sqcup {\geq} n_2 r_2.D_2 \qquad r_2 \sqsubseteq_{\mathcal{R}} r_1 \qquad n_1 \geq n_2}{\begin{array}{c} C_1 \sqcup C_2 \sqcup {\leq}(n_1 - n_2) r_1.\neg(D_1 \sqcup D_2) \sqcup {\geq} 1 r_1.D_{12} \\ \vdots \\ C_1 \sqcup C_2 \sqcup {\leq}(n_1 - 1) r_1.\neg(D_1 \sqcup D_2) \sqcup {\geq} n_2 r_1.D_{12} \end{array}}$$

$\geq\geq$-Combination:

$$\frac{C_1 \sqcup {\geq} n_1 r_1.\mathcal{D}_1 \qquad C_2 \sqcup {\geq} n_2 r_2.\mathcal{D}_2 \qquad r_1 \sqsubseteq_{\mathcal{R}} r \qquad r_2 \sqsubseteq_{\mathcal{R}} r}{\begin{array}{c} C_1 \sqcup C_2 \sqcup {\geq}(n_1 + n_2) r.(\mathcal{D}_1 \sqcup \mathcal{D}_2) \sqcup {\geq} 1 r.\mathcal{D}_{12} \\ \vdots \\ C_1 \sqcup C_2 \sqcup {\geq}(n_1 + 1) r.(\mathcal{D}_1 \sqcup \mathcal{D}_2) \sqcup {\geq} n_2 r.\mathcal{D}_{12} \end{array}}$$

Limits of Approach

- Approach has been extended to DLs supporting:
  - local functionality
  - number restrictions (graded modalities)
  - transitive roles (as in modal logic **S**4)
  - inverse roles (converse modalities)
  - ABoxes

⇒ Complete methods for $\mathcal{SHI}\nu$, $\mathcal{SIF}\nu$ and $\mathcal{SHQ}\nu$
  - Transitive roles cannot be eliminated
  - $\mathcal{SHQ}$: only forgetting concept names

- Combining rules further breaks completeness
  - Possibly limit of resolution approach
  - Might require support for *role conjunctions*

Evaluation of Forgetting

$\mathcal{ALCH}$, forget 50 symbols	
Success Rate:	91.10%
Without Fixpoints:	95.29%
Duration Mean:	7.68 sec.
Duration Median:	2.74 sec.
Duration 90th percentile:	12.45 sec.

$\mathcal{ALCH}$, forget 100 symbols	
Success Rate:	88.10%
Without Fixpoints:	93.27%
Duration Mean:	18.03 sec.
Duration Median:	3.81 sec.
Duration 90th percentile:	21.17 sec.

$\mathcal{ALC}$ w. ABoxes, forget 50 symbols	
Success Rate:	94.79%
Without Fixpoints:	92.91%
Duration Mean:	23.94 sec.
Duration Median:	3.01 sec.
Duration 90th percentile:	29.00 sec.

$\mathcal{ALC}$ w. ABoxes, forget 100 symbols	
Success Rate:	91.37%
Fixpoints:	92.48%
Duration Mean:	57.87 sec.
Duration Median:	6.43 sec.
Duration 90th percentile:	99.26 sec.

$\mathcal{SHQ}$, forget 50 concept symbols	
Success Rate:	95.83%
Without Fixpoints:	93.40%
Duration Mean:	7.62 sec.
Duration Median:	1.04 sec.
Duration 90th percentile:	4.89 sec.

$\mathcal{SHQ}$, forget 100 concept symbols	
Timeouts:	90.77%
Fixpoints:	91.99%
Duration Mean:	13.51 sec.
Duration Median:	1.60 sec.
Duration 90th percentile:	11.65 sec.

Corpus Respective fragments of 306 ontologies from BioPortal having at most 100,000 axioms.

Timeout 30 minutes

Evaluation of Uniform Interpolation

$\mathcal{ALC}$ Knowledge Bases, $\#\mathcal{S} = 50$	
Success Rate:	84.78%
Without Fixpoints:	96.06%
Duration Mean:	113.90 sec.
Duration Median:	29.58 sec.
Duration 90th percent.:	330.56 sec.
Axioms Mean:	198.52
Axioms Median:	31.00
Axioms 90th percent.:	426.00
Ax. Size Mean:	6.15
Ax. Size Median:	3.00
Ax. Size 90th percent.:	5.59

$\mathcal{ALC}$ Knowledge Bases, $\#\mathcal{S} = 100$	
Success Rate:	80.54%
Without Fixpoints:	95.04%
Duration Mean:	313.28 sec.
Duration Median:	214.56 sec.
Duration 90th percent.:	780.30 sec.
Axioms Mean:	302.78
Axioms Median:	84.00
Axioms 90th percent.:	709.00
Ax. Size Mean:	4.66
Ax. Size Median:	3.04
Ax. Size 90th percent.:	5.82

Corpus Respective fragments of 306 ontologies from BioPortal having at most 100,000 axioms.

Timeout 30 minutes

Conclusion

- UI has many applications in DLs, but also in modal logics
- Resolution often allows to compute UIs practically
- Method implemented in tool/library LETHE, available online
- Calculi might have applications outside UI
- Not covered in this tutorial:
  - Forgetting with ABoxes
  - Forgetting with background knowledge

Thank you!

References

Uniform Interpolation in Modal Logics

[Visser, 1996]A Visser. *Uniform interpolation and layered bisimulation.* In Godel'96, 1996.

[D'Agostino, Hollenberg, 1996]. G D'Agostino, M. Hollenberg. *Uniform interpolation, automata and the modal μ-calculus.* Logic Group Preprint Series, 165, 1996.

Foundations of Uniform Interpolation in DL:

[Lutz,Wolter,2010] C. Lutz, F. Wolter. *Foundations for Uniform Interpolation and Forgetting in Expressive Description Logics*, In Proceedings of ICJAI 2011.

[Nikitina,Rudolph,2014] N. Nikitina, S. Rudolph. *(Non-)Succinctness of Uniform Interpolants of General Terminologies in the Description Logic $\mathcal{EL}$.* In Artificial Intelligence, 2014.

References

Uniform Interpolation with Tableaux

[Kracht, 2007] M. Kracht. *Modal Consequence Relations*. In Handbook of Modal Logic, chapter 8, 2007.

[Wang, Wang, et al, 2010] Z. Wang, K. Wang, R. Topor, X. Zhang. *Tableau-Based Forgetting in $\mathcal{ALC}$ Ontologies*. In Proceedings of ECAI, 2010.

Resolution-Based Second-Order Quantifier Elimination

[Gabbay, Ohlbach, 1992] D. Gabbay, Hans Jürgen Ohlbach. *Quantifier Elimination in Second-Order Predicate Logic*. In Proceedings of KR, 1992.

Modal Resolution

[Enjalbert and Fariñas, 1985] P. Enjalbert, L. Fariñas del Cerro. *Modal Resolution in Clausal Form*. Theoretical Computer Science, 65(1):1–33, 1989.

References

Resolution-Based Uniform Interpolation

[Herzig and Mengin, 2008] A. Herzig, J. Mengin. *Uniform Interpolation by Resolution in Modal Logic.* In Proceedings of JELIA, 2008.

[Ludwig and Konev, 2014] M. Ludwig, B. Konev. *Practical Uniform Interpolation and Forgetting for $\mathcal{ALC}$* TBoxes with Applications to Logical Difference. In Proceedings of KR, 2014.

[Koopmann, Schmidt, 2013] P. Koopmann, R. A. Schmidt. *Forgetting Concept and Role Symbols in $\mathcal{ALCH}$-Ontologies*. In Proceedings of LPAR, 2013.

[Koopmann, Schmidt, 2014] P. Koopmann, R. A. Schmidt. *Count and Forget: Uniform Interpolation of $\mathcal{SHQ}$-Ontologies*. In Proceedings of IJCAR, 2014.

[Koopmann, Schmidt, 2015] P. Koopmann, R. A. Schmidt. Uniform Interpolation and Forgetting for $\mathcal{ALC}$ Ontologies with ABoxes. In Proceedings of AAAI, 2015.

Methods for other DLs Mentioned

[Koopmann, 2015] P. Koopmann. *Practical Uniform Interpolation for Expressive Description Logics*. PhD-Thesis, University of Manchester, 2015.