Resolution-Based Uniform Interpolation and Forgetting for Expressive Description Logics

Patrick Koopmann

December 12, 2017
Predicate forgetting

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

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

Given $\mathcal{L}$ sentence $\phi$, predicate $P$, compute $\phi^{-P}$ s.t.
- $P$ does not occur in $\phi^{-P}$
- for every $\mathcal{L}$ sentence $\psi$ without $P$:
  - $\phi^{-P} \models \psi$ iff $\phi \models \psi$

Theorem for first order logic:
- Iff $\phi^{-P}$ 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 $G$
Introduction

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 $\Sigma$ of symbols

compute uniform interpolant (UI) $F^\Sigma$ s.t.
- $F^\Sigma$ only uses symbols from $\Sigma$
- for every $\psi$ in $\Sigma$, $F \models \psi$ iff $F^\Sigma \models \psi$
Introduction

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 $\Sigma$ of symbols

compute uniform interpolant (UI) $F^\Sigma$ s.t.
- $F^\Sigma$ only uses symbols from $\Sigma$
- for every $\psi$ in $\Sigma$, $F \models \psi$ iff $F^\Sigma \models \psi$

Dual to Forgetting:
- UI for $\Sigma \Leftrightarrow$ forget everything not in $\Sigma$
Resolution-Based Uniform Interpolation and Forgetting for Expressive Description Logics

Introduction

Uniform Interpolation

Input Ontology

\[
\begin{align*}
\text{Male} \sqcap \text{Female} & \sqsubseteq \bot \\
\top & \sqsubseteq \forall \text{hasParent.} \text{Parent} \\
\text{Parent} & \sqsubseteq \text{Male} \sqcup \text{Female} \\
\text{Father} & \equiv \text{Parent} \sqcap \text{Male} \\
\text{Mother} & \equiv \text{Parent} \sqcap \text{Female} \\
\text{Orphan} & \equiv \forall \text{hasParent.} \neg \text{Alive} \\
\text{hasParent}(\text{peter, thomas}) \\
\text{Male}(\text{thomas}) & \quad \text{Alive}(\text{thomas}) \\
\text{hasParent}(\text{thomas, ingrid})
\end{align*}
\]

Uniform Interpolant

\[
\begin{align*}
\text{Father} \sqcap \text{Mother} & \sqsubseteq \bot \\
\neg \text{Orphan(} \text{peter} \text{)} \\
\text{Father(} \text{thomas} \text{)} \\
(\text{Father} \sqcap \text{Mother})(\text{ingrid})
\end{align*}
\]
Resolution-Based Uniform Interpolation and Forgetting for Expressive Description Logics

Introduction

Motivation

Ontology Reuse

Big General Ontology

New Ontology
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 $\Sigma$:
  - $\text{logDiff}(\mathcal{T}_1, \mathcal{T}_2, \Sigma) = \{\alpha \in \mathcal{T}_2^\Sigma \mid \mathcal{T}_1 \not\models \alpha\}$

- $\Sigma$: 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:

    \[ \forall p : \Box \Box p \rightarrow \Box p \iff \forall xyz. (r(x, y) \land r(y, z) \rightarrow r(x, z)) \]

- Requires elimination to preserve all models
- UI only preserves entailments in language under consideration
### Expressive Description Logics

#### Concepts $\mathcal{ALC}$

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\bot$</td>
<td>False</td>
</tr>
<tr>
<td>$\top$</td>
<td>True</td>
</tr>
<tr>
<td>$A$</td>
<td>Concept</td>
</tr>
<tr>
<td>$\neg C$</td>
<td>Negation</td>
</tr>
<tr>
<td>$C \sqcup D$</td>
<td>Union</td>
</tr>
<tr>
<td>$C \sqcap D$</td>
<td>Intersection</td>
</tr>
<tr>
<td>$\exists r.C$</td>
<td>Existential Role</td>
</tr>
<tr>
<td>$\forall r.C$</td>
<td>Universal Role</td>
</tr>
</tbody>
</table>

#### TBox Axioms $\mathcal{ALC}$

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C \sqsubseteq D$</td>
<td>Subsumption</td>
</tr>
<tr>
<td>$C \equiv D$</td>
<td>Equivalence</td>
</tr>
</tbody>
</table>

#### ABox Axioms $\mathcal{ALC}$

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C(a)$</td>
<td>ConceptAssertion</td>
</tr>
<tr>
<td>$r(a, b)$</td>
<td>RoleAssertion</td>
</tr>
</tbody>
</table>

#### Extensions

- **$\mathcal{ALCH}$**: Role Hierarchies
  - $r \sqsubseteq s$

- **$\mathcal{ALCF}$**: Local Functionality
  - $\leq 1r.\top, \geq 2r.\top$

- **$\mathcal{SH}$**: Transitive Roles
  - $\text{trans}(r)$

- **$\mathcal{SHQ}$**: 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

∃hasTopping.⊤ ⊑ Pizza
∃hasSpiciness.(Pizza ⊔ PizzaTopping) ⊑ ⊥

NamedPizza ⊑ Pizza
MozzarellaTopping ⊑ PizzaTopping ⊓ ∃hasSpiciness.Mild
OliveTopping ⊑ VegetableTopping ⊓ ∃hasSpiciness.Mild
TomatoTopping ⊑ VegetableTopping ⊓ ∃hasSpiciness.Mild

Pizza ⊓ Mild ⊑ ⊥
MozzarellaTopping ⊓ VegetableTopping ⊑ ⊥
OliveTopping ⊓ TomatoTopping ⊑ ⊥
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 $\Diamond_r \sim_-. C'$
  - number restrictions correspond to graded modalities
  - transitivity as in $S4$ 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 $\phi^{-P}$ 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:

  $$\exists A. \forall x \exists y z. (r(x, y) \land A(y) \land r(x, z) \land \neg A(z))$$
  $$\equiv \forall x \exists y z. (r(x, y) \land r(x, z) \land y \neq z)$$

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 $\mu$-calculus [D’Agostino, Hollenberg, 1996]

- In most DLs, TBoxes break this property
  - Consider:
    \[ A \sqsubseteq B \quad B \sqsubseteq \exists r. B \quad \Sigma = \{ A, r \} \]

  - UI for $\Sigma$:
    \[ A \sqsubseteq \exists r. \exists r. \exists r. \exists r. \exists r. \exists r. \exists r. \exists r. \exists r. \exists r. \exists r. \exists r. \exists r. \exists r. \exists r. . . . \]
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 $2\text{EXPTIME}$-complete

- A second challenge is size
  - If exists, $\mathcal{TS}$ can have size $O\left(2^{2^{|T|}}\right)$
  - Already for lightweight DL $\mathcal{EL}$

$\mathcal{ALC}$: [Lutz, Wolter, 2010], $\mathcal{EL}$: [Nikitina, Rudolph, 2014]
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 $2^{2^{|T|}}$
  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 $\exists$- and $\forall$-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
Disadvantages of approach:

- Result big disjunction
  - Unusual representation for ontologies
- Expansions not \textit{goal-oriented}
- Expensive termination condition
Resolution addresses short-comings

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

<table>
<thead>
<tr>
<th>Prop. Resolution</th>
<th>First Order Resolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_1 \lor p \quad C_2 \lor \neg p$</td>
<td>$C_1 \lor P(s_1, \ldots, s_n) \quad C_2 \lor \neg P(t_1, \ldots, t_n)$</td>
</tr>
<tr>
<td>$C_1 \lor C_2$</td>
<td>$C_1 \lor C_2 \lor s_1 \neq t_1 \lor \ldots \lor s_n \neq t_n$</td>
</tr>
</tbody>
</table>
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
Let’s try it!

We want to forget $B$ from following ontology:

\[ A \sqsubseteq \forall r. B \quad C \sqsubseteq \exists r. \neg B \]
Using SCAN to Compute Uniform Interpolants

Let's try it!

We want to forget $B$ from following ontology:

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

Representation as First-Order clauses:

1. $\neg A(x) \lor \neg r(x, y) \lor B(y)$
2. $\neg C(x) \lor r(x, f(x))$
3. $\neg C(x) \lor \neg B(f(x))$
Using SCAN to Compute Uniform Interpolants

Representation as First-Order clauses:

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

Inferences on \( B \):

4. \( \neg A(x) \lor \neg r(x, y) \lor \neg C(x) \lor y \neq f(x) \) (Resolution 1,3)
5. \( \neg A(x) \lor \neg r(x, f(x)) \lor \neg C(x) \) (Constr. Elim.)
Using SCAN to Compute Uniform Interpolants

Representation as First-Order clauses:

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

Inferences on \( B \):

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

\( \Rightarrow \) SCAN terminates, but we have insufficient information for UI!
Using SCAN to Compute Uniform Interpolants

Representation as First-Order clauses:

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

Inferences on \( B \):

4. \( \neg A(x) \lor \neg r(x, y) \lor \neg C(x) \lor y \neq f(x) \) \hspace{1cm} (Resolution 1,3)  
5. \( \neg A(x) \lor \neg r(x, f(x)) \lor \neg C(x) \) \hspace{1cm} (Constr. Elim. 4)

Additional steps complete the picture:

6. \( \neg A(x) \lor \neg C(x) \lor f(x) \neq f(x) \) \hspace{1cm} (Resolution 2,5)  
7. \( \neg A(x) \lor \neg C(x) \) \hspace{1cm} (Constr. Elim. 6)
Using SCAN to Compute Uniform Interpolants

Complete Clause Set:

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

Uniform Interpolant:

\[
\begin{align*}
C \sqsubseteq \exists r. \top & \quad A \sqcap C \sqsubseteq \bot
\end{align*}
\]
Using Resolution to Compute Uniform Interpolants

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

- Needed: More than just SOQE
  ⇒ 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 $\mathbf{K}$

- Based on resolution calculus for modal logics by [Enjalbert and Fariñas, 1985]

- Allow to resolve on arbitrary levels of formula:

\[
C_1 \lor \lozenge \Box (C_2 \lor p) \\
C_3 \lor \lozenge \lozenge (C_4 \lor \neg p) \implies C_1 \lor C_3 \lor \lozenge \lozenge (C_2 \lor C_4)
\]

- 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: $\Box (T_1 \lor \ldots \lor T_n)$

- Base rules:
  \[
  C_1 \lor \Box \bot, \quad C_2 \lor \Diamond E \quad \implies_\alpha C_1 \lor C_2
  \]
  \[
  C_1 \lor p, \quad C_2 \lor \neg p \quad \implies_\alpha C_1 \lor C_2
  \]

- Extended rules provided $C_1, C_2 \implies_\alpha C_3$ / $C_1 \implies_\alpha C_2$
  \[
  C_1' \lor \Box C_1, \quad C_2' \lor \Diamond (C_2, E) \quad \implies_\alpha C_1' \lor C_2' \lor \Diamond (C_2, E, C_3)
  \]
  \[
  C_1' \lor \Box C_1, \quad C_2' \lor \Box C_2 \quad \implies_\alpha C_1' \lor C_2' \lor \Box C_3
  \]
  \[
  C \lor \Diamond (C_1, C_2, E) \quad \implies_\alpha C \lor \Diamond (C_1, C_2, E, C_3)
  \]
  \[
  C \lor \Diamond (C_1, E) \quad \implies_\alpha C \lor \Diamond (C_1, E, C_2)
  \]
  \[
  C \lor \Box C_1 \quad \implies_\alpha C \lor \Box C_2
  \]
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 $ALC$ in [Ludwig and Konev, 2014]:
  - First practical method for UI in $ALC$
  - 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

\[
A \sqsubseteq B \quad B \sqsubseteq C \uplus \exists r. B \quad A \sqsubseteq \forall r. \forall r. \bot
\]

\[\Sigma = \{A, C, r\}\]

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

\[
A \sqsubseteq C \uplus \exists r. C \quad A \sqsubseteq \forall r. \forall r. \bot
\]

- Probably in general no easy solution
Computing UIs using Resolution

- What to do about termination problem?
What to do about termination problem?
⇒ Move to language that has uniform interpolation!
New concept constructor $\nu X.C[X]$
- $C[X]$: concept that contains $X$ only positively
- Allows to represent *loops*:

\[
A \sqcap \nu X.(C \sqcap \exists r.X) \iff A \sqcap [C \sqcap \exists r.\square]
\]

\[
\iff A \sqcap (C \sqcap \exists r.(C \sqcap \exists r.(C \sqcap \exists r.(\ldots))))
\]
Example

- Consider the following ontology:

\[ A \sqsubseteq B \sqcup C \quad B \sqsubseteq \exists r.B \quad C \sqsubseteq \forall r.\neg B \]

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

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

\[ C \sqsubseteq \forall r. (\neg A \sqcup C) \quad A \sqsubseteq C \sqcup \nu X. (\neg C \sqcap \exists r.X) \]
Greatest fixpoint operators give us uniform interpolation in \( \mathcal{ALC} \sim \mathcal{ALCHI} \)

They can be easily approximated:

\[
\nu X. C[X] \approx C[C[C[C[T]]]]
\]

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 $\textit{ALCH}_\nu$ ($\textit{ALCH} + $greatest fixpoints)
  - Fixpoints can then be approximated or simulated in $\textit{ALCH}$
Normal form, $\mathcal{ALCH}$

**$\mathcal{ALCH}$ Clause**

$r \sqsubseteq s$

$\top \sqsubseteq L_1 \sqcup \ldots \sqcup L_n$

$L_i$: $\mathcal{ALC}$ 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:

\[ C_1 \sqcup Qr.C_2 \iff C_1 \sqcup Qr.D_1, \quad \neg D_1 \sqcup C_2 \]

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

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

Input Language

\[ O \]

Input

\[ O_{\Sigma} \]

Result

Finitely Bounded Representation

\[ N \]

Derive Implicit Knowledge

\[ N^+ \]

Filter Concepts & Roles

\[ N^\Sigma \]

Result

\[ O_{\Sigma} \]

Input Language Finitely Bounded Representation

\[ O \]

Input

\[ O_{\Sigma} \]

Result

translate

Derive Implicit Knowledge

Filter Concepts & Roles

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

<table>
<thead>
<tr>
<th>Resolution</th>
<th>Role Propagation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_1 \sqcup A \quad C_2 \sqcup \neg A$</td>
<td>$C_1 \sqcup \forall r.D_1 \quad C_2 \sqcup Qr.D_2$</td>
</tr>
<tr>
<td>$C_1 \sqcup C_2$</td>
<td>$C_1 \sqcup C_2 \sqcup Qr.D_{12}$</td>
</tr>
</tbody>
</table>

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

Assume the following ontology:

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

Normalisation brings four clauses:

\[ \neg C_1 \sqcup \exists r.D_1 \]
\[ \neg C_2 \sqcup \forall r.D_2 \]
\[ \neg D_1 \sqcup A \]
\[ \neg D_2 \sqcup B \sqcup \neg A \]
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

¬D₁ ⊃ A
¬C₁ ⊃ ∃r·D₁

¬D₂ ⊃ B ⊃ ¬A
¬C₂ ⊃ ∀r·D₂
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

Resolution-Based Uniform Interpolation and Forgetting for Expressive Description Logics

Flattened Approach

Basic Calculus

Cannot resolve due invariant

┐ ¬D₁ ⊔ A
 │ ¬C₁ ⊔ ∃r.D₁
 └ ¬D₂ ⊔ B ⊔ ¬A
     ¬C₂ ⊔ ∀r.D₂

.combine

¬C₁ ⊔ ¬C₂ ⊔ ∃r.D₁₂
¬D₁₂ ⊔ A
¬D₁₂ ⊔ B ⊔ ¬A

Resolves to ¬D₁₂ ⊔ B
Example

Final clause set:

\[ \neg C_1 \sqcup \exists r. D_1 \]
\[ \neg C_2 \sqcup \forall r. D_2 \]
\[ \neg C_1 \sqcup \neg C_2 \sqcup \exists r. D_{12} \]
\[ \neg D_{12} \sqcup D_1 \]
\[ \neg D_{12} \sqcup B \]
\[ \neg D_1 \sqcup A \]
\[ \neg D_2 \sqcup B \sqcup \neg A \]

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

\[ C_1 \sqsubseteq \exists r. T \]
\[ C_2 \sqsubseteq \forall r. T \]
\[ C_1 \cap C_2 \sqsubseteq \exists r. B \]
Forgetting Concept and Role Names in $ALCH$

<table>
<thead>
<tr>
<th>$\exists$-elimination</th>
<th>Role hierarchy</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ C \sqcup \exists r.D \quad \neg D ] \hspace{1cm} [ C ] \hspace{1cm} [ r \sqsubseteq s \quad s \sqsubseteq t ] \hspace{1cm} [ r \sqsubseteq t ]</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Universal roles</th>
<th>Existential roles</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ C_1 \sqcup \forall s.D_1 \quad r \sqsubseteq s ] \hspace{1cm} [ C_1 \sqcup \forall r.D_1 ] \hspace{1cm} [ C_1 \sqcup \exists s.D_1 \quad s \sqsubseteq r ] \hspace{1cm} [ C_1 \sqcup \exists r.D_1 ]</td>
<td></td>
</tr>
</tbody>
</table>

$\Rightarrow$ 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

\[
\begin{align*}
C_0 \sqcup \forall r.D_0 & \ldots C_n \sqcup \forall r.D_n & C \sqcup \exists r.D \\
\hline
C_0 \sqcup \ldots \sqcup C_n \sqcup C
\end{align*}
\]

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

⇒ 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 \textit{tautological fixpoints}: $\nu X. C[X]$ where $C[\top] = \top$
Flattened Approach

- General structure of calculus:
  1. Resolution-like rule (Resolution, $\exists$-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

<table>
<thead>
<tr>
<th>Universalisation</th>
<th>( C_1 \sqcup \exists r.D_1 )</th>
<th>( C_2 \sqcup \leq_{1r}.\top )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( C_1 \sqcup C_2 \sqcup \forall r.D_1 )</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( \exists\exists )-Role Propagation</th>
<th>( C_1 \sqcup \exists r.D_1 )</th>
<th>( C_2 \sqcup \exists r.D_2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( C_1 \sqcup C_2 \sqcup \exists r.D_{12} \sqcup \geq_{2r}.\top )</td>
<td></td>
</tr>
</tbody>
</table>
Example Functional Role Restrictions

Example:

\[ A \sqsubseteq \exists r. B \quad A \sqsubseteq \exists r. \neg B \]

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

Example:

\[ A \sqsubseteq \exists r. B \quad A \sqsubseteq \exists r. \neg B \]

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

<table>
<thead>
<tr>
<th>Combination</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\leq \leq)-Combination:</td>
</tr>
</tbody>
</table>
| \[
\begin{align*}
C_1 \sqcup \leq n_1 r_1.\neg D_1 & \quad C_2 \sqcup \leq n_2 r_2.\neg D_2 & \quad r \sqsubseteq r_1 & \quad r \sqsubseteq r_2 \\
C_1 \sqcup C_2 \sqsubseteq (n_1 + n_2) r.\neg D_{12}
\end{align*}
\] |
| \(\geq \leq\)-Combination: |
| \[
\begin{align*}
C_1 \sqcup \geq n_1 r_1.\neg D_1 & \quad (D_1 \sqcup \ldots \sqcup D_m) & \quad C_2 \sqcup \leq n_2 r_2.\neg D_a & \quad r_1 \sqsubseteq r_2 \\
C_1 \sqcup C_2 \sqsubseteq (n_1 + n_2) r_1.\neg (D_{1a} \sqcup \ldots \sqcup D_{ma})
\end{align*}
\] |
| \(\leq \geq\)-Combination: |
| \[
\begin{align*}
C_1 \sqcup \leq n_1 r_1.\neg D_1 & \quad C_2 \sqcup \geq n_2 r_2. D_2 & \quad r_2 \sqsubseteq r_1 & \quad n_1 \geq n_2 \\
C_1 \sqcup C_2 \sqsubseteq (n_1 - n_2) r_1.\neg (D_1 \sqcup D_2) \sqcup \geq r_1. D_{12} \\
\vdots \\
C_1 \sqcup C_2 \sqsubseteq (n_1 - 1) r_1.\neg (D_1 \sqcup D_2) \sqcup \geq n_2 r_1. D_{12}
\end{align*}
\] |
| \(\geq \geq\)-Combination: |
| \[
\begin{align*}
C_1 \sqcup \geq n_1 r_1. D_1 & \quad C_2 \sqcup \geq n_2 r_2. D_2 & \quad r_1 \sqsubseteq r & \quad r_2 \sqsubseteq r \\
C_1 \sqcup C_2 \sqsubseteq (n_1 + n_2) r.\neg (D_1 \sqcup D_2) \sqcup \geq r_1. D_{12} \\
\vdots \\
C_1 \sqcup C_2 \sqsubseteq (n_1 + 1) r.\neg (D_1 \sqcup D_2) \sqcup \geq n_2 r. D_{12}
\end{align*}
\] |
Limits of Approach

- Approach has been extended to DLs supporting:
  - local functionality
  - number restrictions (graded modalities)
  - transitive roles (as in modal logic S4)
  - inverse roles (converse modalities)
  - ABoxes

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

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

### ALC

<table>
<thead>
<tr>
<th>Forgetting</th>
<th>Success Rate:</th>
<th>Without Fixpoints:</th>
<th>Duration Mean:</th>
<th>Duration Median:</th>
<th>Duration 90th percentile:</th>
</tr>
</thead>
<tbody>
<tr>
<td>50 symbols</td>
<td>91.10%</td>
<td>95.29%</td>
<td>7.68 sec.</td>
<td>2.74 sec.</td>
<td>12.45 sec.</td>
</tr>
<tr>
<td>100 symbols</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50 symbols</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100 symbols</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### ALC w. ABoxes

<table>
<thead>
<tr>
<th>Forgetting</th>
<th>Success Rate:</th>
<th>Without Fixpoints:</th>
<th>Duration Mean:</th>
<th>Duration Median:</th>
<th>Duration 90th percentile:</th>
</tr>
</thead>
<tbody>
<tr>
<td>50 symbols</td>
<td>94.79%</td>
<td>92.91%</td>
<td>23.94 sec.</td>
<td>3.01 sec.</td>
<td>29.00 sec.</td>
</tr>
<tr>
<td>100 symbols</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50 symbols</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100 symbols</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### SHQ

<table>
<thead>
<tr>
<th>Forgetting</th>
<th>Success Rate:</th>
<th>Without Fixpoints:</th>
<th>Duration Mean:</th>
<th>Duration Median:</th>
<th>Duration 90th percentile:</th>
</tr>
</thead>
<tbody>
<tr>
<td>50 concept symbols</td>
<td>95.83%</td>
<td>93.40%</td>
<td>7.62 sec.</td>
<td>1.04 sec.</td>
<td>4.89 sec.</td>
</tr>
<tr>
<td>100 concept symbols</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50 concept symbols</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100 concept symbols</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### SHQ w. ABox

<table>
<thead>
<tr>
<th>Forgetting</th>
<th>Timeouts:</th>
<th>Fixpoints:</th>
<th>Duration Mean:</th>
<th>Duration Median:</th>
<th>Duration 90th percentile:</th>
</tr>
</thead>
<tbody>
<tr>
<td>50 concept symbols</td>
<td>90.77%</td>
<td>91.99%</td>
<td>13.51 sec.</td>
<td>1.60 sec.</td>
<td>11.65 sec.</td>
</tr>
<tr>
<td>100 concept symbols</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>50 concept symbols</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100 concept symbols</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

**Timeout**  30 minutes
Evaluation of Uniform Interpolation

<table>
<thead>
<tr>
<th>ALC Knowledge Bases, $#S = 50$</th>
<th>ALC Knowledge Bases, $#S = 100$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Success Rate:</strong></td>
<td>84.78%</td>
</tr>
<tr>
<td><strong>Without Fixpoints:</strong></td>
<td>96.06%</td>
</tr>
<tr>
<td><strong>Duration Mean:</strong></td>
<td>113.90 sec.</td>
</tr>
<tr>
<td><strong>Duration Median:</strong></td>
<td>29.58 sec.</td>
</tr>
<tr>
<td><strong>Duration 90th percent.:</strong></td>
<td>330.56 sec.</td>
</tr>
<tr>
<td><strong>Axioms Mean:</strong></td>
<td>198.52</td>
</tr>
<tr>
<td><strong>Axioms Median:</strong></td>
<td>31.00</td>
</tr>
<tr>
<td><strong>Axioms 90th percent.:</strong></td>
<td>426.00</td>
</tr>
<tr>
<td><strong>Ax. Size Mean:</strong></td>
<td>6.15</td>
</tr>
<tr>
<td><strong>Ax. Size Median:</strong></td>
<td>3.00</td>
</tr>
<tr>
<td><strong>Ax. Size 90th percent.:</strong></td>
<td>5.59</td>
</tr>
</tbody>
</table>

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

**Timeout**  30 minutes
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**


**Foundations of Uniform Interpolation in DL:**

References

**Uniform Interpolation with Tableaux**


**Resolution-Based Second-Order Quantifier Elimination**


**Modal Resolution**

Resolution-Based Uniform Interpolation


Conclusion


Methods for other DLs Mentioned