Description Logics – Decidability and Complexity

Decidability & Undecidability

Polynomial Cases

Complexity of $\mathcal{ALC}$ Subsumption

Expressive Power vs. Complexity

The Complexity of Subsumption in TBoxes

Outlook

Decidability

$L_2$ is the fragment of first-order predicate logic using only two different variable names (note: variable names can be reused!).

$L_2^=$ the same including equality.

Theorem

$L_2^=$ is decidable.

Corollary

Subsumption and satisfiability of concept descriptions is decidable in description logics using only the following concept and role forming operators: $C \sqcap D$, $C \sqcup D$, $\neg C$, $\forall r.C$, $\exists r.C$, $r \sqsubseteq s$, $r \sqcap s$, $r \sqcup s$, $\neg r$, $r^{-1}$.

Potential problems: Role composition and cardinality restrictions for role fillers. Cardinality restrictions, however, are not a real problem.

Undecidability

- $r \circ s$, $r \sqcap s$, $\neg r$, $1$ [Schild 88]
- not relevant; Tarski had shown that already! – for relation algebras
- $r \circ s$, $r = s$, $C \sqcap D$, $\forall r.C$ [Schmidt-Schauß 89]
- This is in fact a fragment of the early description logic $\mathcal{KL}$-ONE, where people had hoped to come up with a complete subsumption algorithm
Decidable, Polynomial-Time Cases

- $FL^-$ has obviously a polynomial subsumption problem (in the empty TBox) – the SUB algorithm needs only quadratic time.
- Donini et al [IJCAI 91] have shown that in the following languages subsumption can be decided using only polynomial time (and they are maximal wrt. this property):

\[
C \rightarrow A | \neg A | C \sqcap C' | \forall r.C | (\geq n r) | (\leq n r), \ r \rightarrow t | r^{-1}
\]

and

\[
C \rightarrow A | C \sqcap C' | \forall r.C | \exists r, \ r \rightarrow t | r^{-1} | r \sqcap r' | r \circ r'
\]

Open:

\[
C \rightarrow A | C \sqcap C' | \forall r.C | (\geq n r) | (\leq n r), \ r \rightarrow t | r \circ r'.
\]
Computational Complexity of \( ALC \) Subsumption

**Lemma (Upper Bound for \( ALC \))**

\( ALC \) subsumption, unsatisfiability and satisfiability are all in PSPACE.

**Proof.**

This follows from the tableau algorithm for \( ALC \). Although there may be exponentially many closed constraint systems, we can visit them step by step generating only one at a time. When closing a system, we have to consider only one role at a time – resulting in an only polynomial space requirement, i.e., satisfiability can be decided in PSPACE.

**Theorem (Complexity of \( ALC \))**

\( ALC \) subsumption, unsatisfiability and satisfiability are all PSPACE-complete.

Further Consequences of the Reducibility of \( K \) to \( ALC \)

- In the reduction we used only one role symbol. Are there modal logics that would require more than one such role symbol?
- The multi-modal logic \( K(n) \) has \( n \) different Box operators \( \Box_i \) (for \( n \) different agents)
- \( ALC \) is a notational variant of \( K(n) \) [Schild, IJCAI-91]
- Are there perhaps other modal logics that correspond to other descriptions logics?
- propositional dynamic logic (PDL), e.g., transitive closure, composition, role inverse, ...
- DL can be thought as fragments of first-order predicate logic. However, they are much more similar to modal logics
- Algorithms and complexity results can be borrowed. Works also the other way around

Expressive Power vs. Complexity

- Of course, one wants to have a description logic with high expressive power. However, high expressive power implies usually that the computational complexity of the reasoning problems might also be high, e.g., \( FL^- \) vs. \( ALC \)
- Does it make sense to use a language such as \( ALC \) or even extensions (corresponding to PDL) with higher complexity?
- There are three approaches to this problem:
  1. Use only small description logics with complete inference algorithms
  2. Use expressive description logics, but employ incomplete inference algorithms
  3. Use expressive description logics with complete inference algorithms
- For a long time, only options 1 and 2 were studied. Meanwhile, most researcher concentrate on option 3!

Is Subsumption in the Empty TBox Enough?

- We have shown that we can reduce concept subsumption in a given TBox to concept subsumption in the empty TBox.
- However, it is not obvious that this can be done in polynomial time
- In particular, in the following example unfolding leads to an exponential blowup:

\[
C_1 \equiv \forall r. C_0 \sqcap \forall s. C_0 \\
C_2 \equiv \forall r. C_1 \sqcap \forall s. C_1 \\
\vdots \\
C_n \equiv \forall r. C_{n-1} \sqcap \forall s. C_{n-1}
\]

- Unfolding \( C_n \) leads to a concept description with a size \( \Omega(2^n) \)
- Is it possible to avoid this blowup?
- Can we avoid exponential preprocessing?
TBox Subsumption for Small Languages

▶ **Question**: Can we decide in polynomial time TBox subsumption for a description logic such as $\mathcal{FL}^-$, for which concept subsumption in the empty TBox can be decided in polynomial time?

▶ Let us consider $\mathcal{FL}_0$: $C \sqcap D, \forall r.C$ with terminological axioms.

▶ Subsumption without a TBox can be done easily, using a structural subsumption algorithm.

▶ Unfolding + structural subsumption gives us an exponential algorithm.

The Complexity of TBox Subsumption

**Theorem (Complexity of TBox subsumption)**

TBox subsumption for $\mathcal{FL}_0$ is NP-hard.

Proof sketch.

We use the NDFA-equivalence problem, which is NP-complete for cycle-free automata and PSPACE-complete for general NDFAs. We transform a cycle-free NDFA to a $\mathcal{FL}_0$-terminology with the mapping $\pi$ as follows:

- automaton $A \mapsto$ terminology $T_A$
- state $q \mapsto$ concept name $q$
- terminal state $q_f \mapsto$ concept name $q_f$
- input symbol $r \mapsto$ role name $r$
- $r$-transition from $q$ to $q'$ $\mapsto q \sqcap \forall r. q'$

In general, we have: $L(q) \subseteq L(q')$ iff $q' \sqsubseteq_T q$, from which the correctness of the reduction and the complexity result follows.

What Does This Complexity Result Mean?

▶ Note that for expressive languages such as $\mathcal{ALC}$, we do not notice any difference!

▶ The TBox subsumption complexity result for less expressive languages does not play a large role in practice

▶ Pathological situations do not happen very often

▶ In fact, if the definition depth is logarithmic in the size of the TBox, the whole problem vanishes.

▶ However, in order to protect oneself against such problems, one often uses lazy unfolding

▶ Similarly, also for the $\mathcal{ALC}$ concept descriptions, one notices that they are usually very well behaved.
Outlook

- Description logics have a long history (Tarski’s relation algebras and Brachman’s KL-ONE)
- Early on, either small languages with provably easy reasoning problems (e.g., the system CLASSIC) or large languages with incomplete inference algorithms (e.g., the system Loom) were used.
- Meanwhile, one uses complete algorithms on very large description logics (e.g., SHIQ), e.g. in the systems FaCT and RACER
- RACER can handle KBs with up to 160,000 concepts (example from unified medical language system) in reasonable time (less than one day computing time)
- Description logics are used as the semantic backbone for OWL (a Web-language extending RDF)

Literature