Description Logics

Franz Baader
Theoretical Computer Science
TU Dresden
Germany

1. Motivation and introduction to Description Logics
2. Tableau-based reasoning procedures
3. Automata-based reasoning procedures
4. Complexity of reasoning in Description Logics
5. Reasoning in inexpressive Description Logics
Reasoning procedures

- The procedure should be a decision procedure for the problem:
  - soundness: positive answers are correct
  - completeness: negative answers are correct
  - termination: always gives an answer in finite time

- The procedure should be as efficient as possible:
  preferably optimal w.r.t. the (worst-case) complexity of the problem

- The procedure should be practical:
  easy to implement and optimize, and behave well in applications

Note
- Satisfiability in first-order logic does not have a decision procedure.
- Satisfiability in propositional logic has a decision procedure, but the problem is NP-complete.
Tableau algorithm for $ALC$

It is sufficient to design a decision procedure for consistency of an ABox without a TBox:

- TBoxes can be eliminated by expanding concept descriptions
- satisfiability, subsumption, and the instance problem can be reduced to consistency

The tableau-based consistency algorithm tries to generate a finite model for the input ABox $A_0$:

- applies tableau rules to extend the ABox \( \text{one rule per constructor} \)
- checks for obvious contradictions
- an ABox that is complete (no rule applies) and open (no obvious contradictions) describes a model
Tableau algorithm

\[ \mathcal{T} \quad \text{GoodStudent} \equiv \text{Smart} \sqcap \text{Studious} \]

Subsumption question:
\[ \exists \text{attends}.\text{Smart} \sqcap \exists \text{attends}.\text{Studious} \sqsubseteq^? \exists \text{attends}.\text{GoodStudent} \]

Reduction to satisfiability: is the following concept unsatisfiable w.r.t. \( \mathcal{T} \)?
\[ \exists \text{attends}.\text{Smart} \sqcap \exists \text{attends}.\text{Studious} \sqcap \neg \exists \text{attends}.\text{GoodStudent} \]

Reduction to consistency: is the following ABox inconsistent w.r.t. \( \mathcal{T} \)?
\[ \{ (\exists \text{attends}.\text{Smart} \sqcap \exists \text{attends}.\text{Studious} \sqcap \neg \exists \text{attends}.\text{GoodStudent})(a) \} \]

Expansion: is the following ABox inconsistent?
\[ \{ (\exists \text{attends}.\text{Smart} \sqcap \exists \text{attends}.\text{Studious} \sqcap \neg \exists \text{attends}.(\neg \text{Smart} \sqcap \neg \text{Studious}))(a) \} \]

Negation normal form: is the following ABox inconsistent?
\[ \{ (\exists \text{attends}.\text{Smart} \sqcap \exists \text{attends}.\text{Studious} \sqcap \forall \text{attends}.(\neg \text{Smart} \sqcap \neg \text{Studious}))(a) \} \]
Tableau algorithm

example continued

Is the following ABox inconsistent?

\{ (\exists \text{attends.} \text{Smart} \sqcap \exists \text{attends.} \text{Studious} \sqcap \forall \text{attends.} (\neg \text{Smart} \sqcup \neg \text{Studious}))(a) \} 

\exists r. A \sqcap \exists r. B \sqcap \forall r. (\neg A \sqcup \neg B)  
\exists r. A, \exists r. B, \forall r. (\neg A \sqcup \neg B) 

\begin{align*} 
& a \\
& \quad r \\
& \quad \quad b \\
& \quad r \\
& \quad \quad c \\
& A \\
& \neg A \sqcup \neg B \\
& \neg A \\
& B \\
& \neg A \sqcup \neg B \\
& \neg A \\
\end{align*}

complete and open ABox

yields a model for the input ABox

and thus a counterexample to the subsumption relationship
Tableau algorithm

Input: An $\mathcal{ALC}$-ABox $\mathcal{A}_0$

Output: “yes” if $\mathcal{A}_0$ is consistent
“no” otherwise

Preprocessing:

transform all concept descriptions in $\mathcal{A}_0$ into negation normal form (NNF)
by applying the following rules:

$\neg (C \sqcap D) \Rightarrow \neg C \sqcup \neg D$

$\neg (C \sqcup D) \Rightarrow \neg C \sqcap \neg D$

$\neg \neg C \Rightarrow C$

$\neg (\exists r. C') \Rightarrow \forall r. \neg C'$

$\neg (\forall r. C') \Rightarrow \exists r. \neg C'$

The NNF can be computed in polynomial time, and it does not change the semantics of the concept.
Tableau algorithm

Data structure:
finite set of ABoxes rather than a single ABox: start with \( \{A_0\} \)

Application of tableau rules:
the rules take one ABox from the set and replace it by finitely many
new ABoxes

Termination:
if no more rules apply to any ABox in the set

Answer:
“yes” if the set contains an open ABox, i.e., an ABox not containing an
obvious contradiction of the form
\[ A(a) \text{ and } \neg A(a) \]
for some individual name \( a \)
“no” if all ABoxes in the set are closed (i.e., not open)
Tableau rules

one for every constructor (except for negation)

The $\cap$-rule

- **Condition:** $\mathcal{A}$ contains $(C \cap D)(a)$, but not both $C(a)$ and $D(a)$
- **Action:** $\mathcal{A}' := \mathcal{A} \cup \{C(a), D(a)\}$

The $\cup$-rule

- **Condition:** $\mathcal{A}$ contains $(C \cup D)(a)$, but neither $C(a)$ nor $D(a)$
- **Action:** $\mathcal{A}' := \mathcal{A} \cup \{C(a)\}$ and $\mathcal{A}'' := \mathcal{A} \cup \{D(a)\}$

The $\exists$-rule

- **Condition:** $\mathcal{A}$ contains $(\exists r.C)(a)$, but there is no $c$ with $\{r(a,c), C(c)\} \subseteq \mathcal{A}$
- **Action:** $\mathcal{A}' := \mathcal{A} \cup \{r(a,b), C(b)\}$ where $b$ is a new individual name

The $\forall$-rule

- **Condition:** $\mathcal{A}$ contains $(\forall r.C)(a)$ and $r(a,b)$, but not $C(b)$
- **Action:** $\mathcal{A}' := \mathcal{A} \cup \{C(b)\}$
Tableau algorithm is a decision procedure for consistency

1. **Local correctness:** rules preserve consistency
   - **Deterministic rule**
   - **Nondeterministic rule**

2. **Termination:** no infinite paths
   - **Complete ABoxes**

3. **Soundness:** any complete and open ABox has a model
   - **Trivial**
   - **Completeness:** closed ABoxes do not have a model
Local correctness

Rules preserve consistency

The $\exists$-rule

Condition: $A$ contains $(\exists r.C)(a)$, but there is no $c$ with $\{r(a, c), C(c)\} \subseteq A$

Action: $A' := A \cup \{r(a, b), C(b)\}$ where $b$ is a new individual name

To show: $A$ has a model iff $A'$ has a model

$\Rightarrow$ Let $I$ be a model of $A$.

Since $(\exists r.C)(a) \in A$, there is a $d \in A_I$ such that $(a^I, d) \in r^I$ and $d \in C^I$.

Let $I'$ be the interpretation that coincides with $I$, with the exception that $b^I = d$.

Since $b$ does not occur in $A$, $I'$ is a model of $A$.

By definition of $b^I$, it is also a model of $\{r(a, b), C(b)\}$.

$\Leftarrow$ trivial since $A \subseteq A'$.
Local correctness

rules preserve consistency

The \( \sqcup \)-rule

*Condition:* \( A \) contains \((C \sqcup D)(a)\), but neither \( C(a) \) nor \( D(a) \)

*Action:* \( A' := A \cup \{C(a)\} \) and \( A'' := A \cup \{D(a)\} \)

To show: \( A \) has a model iff \( A' \) has a model or \( A'' \) has a model

\[ \Rightarrow \]
Let \( I \) be a model of \( A \).

Since \((C \sqcup D)(a) \in A\), we have \( a^I \in (C \sqcup D)^I = C^I \cup D^I \).

If \( a^I \in C^I \), then \( I \) is a model of \( A' \).

If \( a^I \in D^I \), then \( I \) is a model of \( A'' \).

\[ \Leftarrow \]
trivial since \( A \subseteq A' \) and \( A \subseteq A'' \).
Termination is an easy consequence of the following facts:

The label $\mathcal{L}(a)$ of an individual name consists of the concepts in concept assertions for $a$.

1. rule application is monotonic: every application of a rule extends the label of an individual, and does not remove anything;

2. concepts in labels are subdescriptions of concepts occurring in the input ABox $\mathcal{A}_0$;

$\implies$ finite number of rule applications per individual

3. the number of new individuals that are $r$-successors of an individual is bounded by the number of existential restrictions in $\mathcal{A}_0$;

4. the length of successor chains of new individuals is bounded by the maximal size of the concepts in $\mathcal{A}_0$:
   - if $x$ is a new individual, then it has a unique predecessor $y$
   - the maximal size of concepts in $\mathcal{L}(x)$ is strictly smaller than in $\mathcal{L}(y)$

$\implies$ finitely many new individuals
Soundness

any complete and open ABox has a model

Let $\mathcal{A}$ be a complete and open ABox.

The canonical interpretation $\mathcal{I}_\mathcal{A}$ induced by $\mathcal{A}$ is defined as follows:

- $\Delta^{\mathcal{I}_\mathcal{A}} := \{ x \mid x \text{ is an individual name occurring in } \mathcal{A} \}$
- $x^{\mathcal{I}_\mathcal{A}} := x$ for all individual names occurring in $\mathcal{A}$
- $A^{\mathcal{I}_\mathcal{A}} := \{ x \mid A(x) \in \mathcal{A} \}$ for all $A \in N_C$
- $r^{\mathcal{I}_\mathcal{A}} := \{ (x, y) \mid r(x, y) \in \mathcal{A} \}$ for all $r \in N_R$

Claim

$\mathcal{I}_\mathcal{A}$ is a model of $\mathcal{A}$. 
Soundness

$\mathcal{I}_A$ is a model of $\mathcal{A}$.

- if $r(x, y) \in \mathcal{A}$, then $(x^{\mathcal{I}_A}, y^{\mathcal{I}_A}) = (x, y) \in r^{\mathcal{I}_A}$ by definition of $r^{\mathcal{I}_A}$

- for $C(x) \in \mathcal{A}$, we show $x^{\mathcal{I}_A} = x \in C^{\mathcal{I}_A}$ by induction on the size of $C$:
  - $C = A$ for $A \in N_C$: trivial by definition of $A^{\mathcal{I}_A}$
  - $C = \neg A$ for $A \in N_C$:
    since $\mathcal{A}$ is open, $A(x) \notin \mathcal{A}$, and thus $x \notin A^{\mathcal{I}_A}$ by definition of $A^{\mathcal{I}_A}$
  - $C = C_1 \cap C_2$:
    since $\mathcal{A}$ is complete, $(C_1 \cap C_2)(x) \in \mathcal{A}$ implies that $C_1(x) \in \mathcal{A}$ and $C_2(x) \in \mathcal{A}$;
    by induction, this yields $x \in C_1^{\mathcal{I}_A}$ and $x \in C_2^{\mathcal{I}_A}$, and thus $x \in (C_1 \cap C_2)^{\mathcal{I}_A}$
  - the other constructors can be treated similarly

© Franz Baader
Tableau algorithm is a decision procedure for consistency

1. Started with a finite ABox $A_0$ in NNF, the algorithm always terminates with a finite set of complete ABoxes $A_1, \ldots, A_n$

2. Local correctness: $A_0$ consistent iff one of $A_1, \ldots, A_n$ consistent

3. Answer “no”:
   none of $A_1, \ldots, A_n$ open
   $A_1, \ldots, A_n$ inconsistent
   $A_0$ inconsistent

4. Answer “yes”:
   one of $A_1, \ldots, A_n$ open
   one of $A_1, \ldots, A_n$ consistent
   $A_0$ consistent
Adding number restrictions

Number restrictions: \((\geq n \ r.C), (\leq n \ r.C)\) with semantics

\[
(\geq n \ r.C)^I := \{d \in \Delta^I \mid \text{card}\{e \mid (d, e) \in r^I \land e \in C^I\} \geq n\}
\]

\[
(\leq n \ r.C)^I := \{d \in \Delta^I \mid \text{card}\{e \mid (d, e) \in r^I \land e \in C^I\} \leq n\}
\]

Negation normal form:

\[
\neg(\geq n + 1 \ r.C) \iff (\leq n \ r.C)
\]

\[
\neg(\geq 0 \ r.C) \iff \bot
\]

\[
\neg(\leq n \ r.C) \iff (\geq n + 1 \ r.C)
\]

Extension of algorithm:

- new rules: \(\geq\)-rule and \(\leq\)-rule
- new assertions: inequality assertions of the form \(x \neq y\) with obvious semantics \(x^I \neq y^I\)
- new obvious contradictions
Adding number restrictions

The $\geq$-rule

**Condition:** $\mathcal{A}$ contains $(\geq n \, r \cdot C)(a)$, but there are no $c_1, \ldots, c_n$ with
$$\{r(a, c_1), C(c_1), \ldots, r(a, c_n), C(c_n)\} \cup \{c_i \neq c_j \mid 1 \leq i, j \leq n\} \subseteq \mathcal{A}$$

**Action:** $\mathcal{A}' := \mathcal{A} \cup \{r(a, b_1), C(c_1), \ldots, r(a, b_n), C(b_n)\} \cup \{b_i \neq b_j \mid 1 \leq i, j \leq n\}$

where $b_1, \ldots, b_n$ are new individual names

The $\leq$-rule

**Condition:** $\mathcal{A}$ contains $(\leq n \, r \cdot C)(a)$, and there are $b_1, \ldots, b_{n+1}$ with
$$\{r(a, b_1), C(b_1), \ldots, r(a, b_{n+1}), C(b_{n+1})\} \subseteq \mathcal{A},$$

but $\{b_i \neq b_j \mid 1 \leq i, j \leq n + 1\} \not\subseteq \mathcal{A}$

**Action:** for all $i < j$ with $b_i \neq b_j \not\subseteq \mathcal{A}$
$$\mathcal{A}_{i,j} := \mathcal{A}[b_i \leftarrow b_j]$$

$b_i$ replaced by $b_j$
Adding number restrictions

- \( \mathcal{A} \) contains \((\leq n \cdot r \cdot C)(a)\), and there are \(b_1, \ldots, b_{n+1}\) with
  \[ \{r(a, b_1), C(b_1), \ldots, r(a, b_{n+1}), C(b_{n+1})\} \subseteq \mathcal{A} \]
  and
  \[ \{b_i \neq b_j \mid 1 \leq i, j \leq n + 1\} \subseteq \mathcal{A} \]

- \( \mathcal{A} \) contains \(a \neq a\) for some individual name \(a\)
Adding number restrictions

does this yield a decision procedure?

To show that the algorithm obtained this way is a decision procedure for ABox consistency, we must show

1. local correctness: rules preserve consistency  
   easy to show

2. completeness: a closed ABox does not have a model  
   trivial

3. soundness: a complete and open ABox has a model  
   wrong!

4. termination: there is no infinite chain of rule applications  
   wrong!
Adding number restrictions

\{ (\geq 3 \text{ child. } T)(a), (\leq 1 \text{ child. Female})(a), (\leq 1 \text{ child. } \neg \text{Female})(a) \}

\leq\text{-rule not applicable}

no obvious contradiction

open, complete, but inconsistent

The choose-rule

\sim C: \text{ NNF of } \neg C

\text{Condition: } \mathcal{A} \text{ contains } (\leq n \text{ r. } C)(a) \text{ and } r(a, b), \text{ but neither } C(b) \text{ nor } \sim C(b)

\text{Action: } \mathcal{A}' := \mathcal{A} \cup \{C(b)\} \text{ and } \mathcal{A}'' := \mathcal{A} \cup \{\sim C(b)\}

In the presence of the choose-rule, soundness can easily be shown.
Adding number restrictions

the problem with termination

Solution:

use a strategy that applies generating rules ($\geq$-rule, $\exists$-rule) with lower priority.
Adding GCIs

\[ C \subseteq D \text{ with semantics } C^I \subseteq D^I \]

A finite set of GCIs can be encoded into one GCI of the form \( \top \subseteq C \):

\[ \{ C_1 \subseteq D_1, \ldots, C_n \subseteq D_n \} \quad \rightarrow \quad \{ \top \subseteq (\neg C_1 \cup D_1) \cap \ldots \cap (\neg C_n \cup D_n) \} \]

Consider a GCI \( \top \subseteq C \) where \( C \) is in NNF.

The GCI-rule for \( \top \subseteq C \)

**Condition:** \( \mathcal{A} \) contains the individual name \( a \), but not \( C(a) \)

**Action:** \( \mathcal{A}' := \mathcal{A} \cup \{ C(a) \} \)
Adding GCIs

does this yield a decision procedure?

- local correctness, completeness, and soundness are easy to show

- termination does not hold:

  Test consistency of \( \{P(a)\} \) w.r.t. the GCI \( \top \subseteq \exists r. P \)

  \[
  \begin{array}{c}
  a \\
  P
  \end{array} \quad \xrightarrow{r} \quad \begin{array}{c}
  P \\
  \exists r. P
  \end{array} \quad \xrightarrow{r} \quad \begin{array}{c}
  P \\
  \exists r. P
  \end{array} \quad \xrightarrow{r} \quad \begin{array}{c}
  P \\
  \exists r. P
  \end{array}
  \]

Solution: blocking

- \( y \) is blocked by \( x \) iff \( \mathcal{L}(y) \subseteq \mathcal{L}(x) \)

- to avoid cyclic blocking we fix an enumeration of the individual names, and add to the blocking condition that \( y \) comes after \( x \) in the enumeration

- generating rules are not applied to blocked individuals
Adding GCIs

does this yield a decision procedure?

- local correctness, completeness, and termination are now easy to show

- soundness must be reconsidered:
  - because of blocking, an ABox can be complete although a generating rule applies
  - requires modification in the definition of the canonical interpretation:
    the $r$-successors of a blocked individual are the $r$-successors of the least individual (in the enumeration) blocking it

consistency of $\{ (\forall r.Q)(a), P(a) \}$
w.r.t. the GCI $\top \subseteq \exists r.P$