Automated Reasoning in Artificial Intelligence:

# Introduction to Description Logic

**Szymon Klarman**

(*part of the content based on the tutorial by* **Stefan Schlobach**)

szymon.klarman@gmail.com

VU University Amsterdam, 2009-2012

# Plan for today

- Tableau algorithm for $\mathcal{ALC}$ with empty TBoxes
- Soundness, completeness, termination
- Reasoning w.r.t. non-empty TBoxes

# Reasoning over DL knowledge bases

There are many *different reasoning problems* but we would strongly prefer having one *universal reasoner* (generic problem solver).

General strategy:

1. Choose one type of problems — $\varphi$ — and design a reasoner for solving it.
2. For any problem $\psi$, reduce $\psi$ to $\varphi$, so that:

   **answer to $\psi$? is YES $\Leftrightarrow$ answer to $\varphi$? is YES.**

3. Solve $\varphi$ using the reasoner and translate the answer adequately.

Problem solver (DL reasoner):

*Tableau algorithm* deciding *consistency of the ABox w.r.t. the TBox*.

# Reasoning as model finding

Recall that for $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, we say that $\mathcal{A}$ is *consistent* w.r.t. $\mathcal{T}$ *iff* there *exists a model* for $\mathcal{A}$ and $\mathcal{T}$, i.e. an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ satisfying all axioms in $\mathcal{A}$ and $\mathcal{T}$.

Note: this problem is also called *deciding satisfiability* of $\mathcal{K}$.

The most natural way of solving this problem is to... try to *find a model* for $\mathcal{A}$ and $\mathcal{T}$.

Let's try:

Decide whether $\mathcal{A}$ is consistent w.r.t. $\mathcal{T}$, where:

$$\begin{array}{ll} \mathcal{T}: & Artist \equiv \exists created.Sculpture \sqcup \exists painted.Artwork \\ & Painting \sqsubseteq Artwork \sqcap \neg Sculpture \\ & Painter \sqsubseteq Artist \sqcap \forall created.Painting \\ \mathcal{A}: & rembrandt : Painter \\ & (rembrandt, nightwatch) : created \end{array}$$

# Tableau algorithm: overview

Tableau is a *refutation proof system*. It performs a search through the tree of possible models of the input. It *succeeds* (delivers a proof) *iff* the input is *inconsistent* (there is no model).

Input: ABox $\mathcal{A}$: *for now we assume* $\mathcal{T} = \emptyset$

Procedure:

- Set $\mathcal{A}$ as the root of the tree.
- Apply *tableau expansion rules* to the formulas on the branches.
  (*) Rules add new assertions on a branch and/or create new branches.
- **IF** a branch contains a clash: $\{a : A, a : \neg A\}$ or $\{a : \bot\}$
  **THEN** mark the branch as closed;
  **ELSE** continue expansion until no more rules apply.

Output:

- **IF** all branches close **RETURN**: $\mathcal{A}$ is **INCONSISTENT**.
- **IF** there exists an open branch **RETURN**: $\mathcal{A}$ is **CONSISTENT**.

# Negation Normal Form

To reduce the number of tableau rules we can assume that all concepts in the input appear in *Negation Normal Form* (*NNF*).

$$\begin{aligned}
\neg\top &\Rightarrow \bot \\
\neg\bot &\Rightarrow \top \\
\neg A &\Rightarrow \neg A \\
\neg(\neg C) &\Rightarrow C \\
\neg(C \sqcap D) &\Rightarrow \neg C \sqcup \neg D \\
\neg(C \sqcup D) &\Rightarrow \neg C \sqcap \neg D \\
\neg\exists r.C &\Rightarrow \forall r.\neg C \\
\neg\forall r.C &\Rightarrow \exists r.\neg C
\end{aligned}$$

Example:

$$\begin{aligned}
NNF(A \sqcap \neg\exists r.((D \sqcap \forall r.E) \sqcup \neg C)) &= A \sqcap \forall r.\neg((D \sqcap \forall r.E) \sqcup \neg C) \\
&= A \sqcap \forall r.(\neg(D \sqcap \forall r.E) \sqcap \neg\neg C) \\
&= A \sqcap \forall r.((\neg D \sqcup \neg\forall r.E) \sqcap C) \\
&= A \sqcap \forall r.((\neg D \sqcup \exists r.\neg E) \sqcap C)
\end{aligned}$$

# Tableau rules

A *branch* of a tableau is a set of ABox assertions. For any branch $S$, the following rules apply:

$\Rightarrow_{\sqcap}$ **IF** $(a : C \sqcap D) \in S$ **THEN** $S' := S \cup \{a : C, a : D\}$

$\Rightarrow_{\sqcup}$ **IF** $(a : C \sqcup D) \in S$ **THEN** $S' := S \cup \{a : C\}$ **or** $S' := S \cup \{a : D\}$

$\Rightarrow_{\exists}$ **IF** $(a : \exists r.C) \in S$ **THEN** $S' := S \cup \{(a, b) : r,\ b : C\}$
where $b$ is a 'fresh' individual name in $S$

$\Rightarrow_{\forall}$ **IF** $(a : \forall r.C) \in S$ **and** $(a, b) : r \in S$ **THEN** $S' := S \cup \{b : C\}$

$\Rightarrow_{\times}$ **IF** $\{a : A, a : \neg A\} \subseteq S$ **or** $(a : \bot) \in S$ **THEN** mark the branch as CLOSED

Note:

- A rule should fire only once on a given match.
- The order in which the rules are applied is not determined in principle. We only assume "fairness".

# Example

Problem: Is $\exists r.A \sqcap \exists r.B$ subsumed by $\exists r.(A \sqcap B)$?

Reduction: Is $(\exists r.A \sqcap \exists r.B) \sqcap \neg\exists r.(A \sqcap B)$ unsatisfiable?
Is $\mathcal{A} = \{a : \exists r.A \sqcap \exists r.B \sqcap \neg\exists r.(A \sqcap B)\}$ inconsistent?

Input: $NNF(\mathcal{A}) = \{a : \exists r.A \sqcap \exists r.B \sqcap \forall r.(\neg A \sqcup \neg B)\}$

Procedure: ...compute a tableau proof for $\mathcal{A}$

# Example

Tableau proof:

| | | |
|---|---|---|
| 1. | $a : \exists r.A \sqcap \exists r.B \sqcap \forall r.(\neg A \sqcup \neg B)$ | $\mathcal{A}$ |

# Example

Tableau proof:

| | | |
|---|---|---|
| 1. | $a : \exists r.A \sqcap \exists r.B \sqcap \forall r.(\neg A \sqcup \neg B)$ | $\mathcal{A}$ |
| 2. | $a : \exists r.A$ | $(\Rightarrow_{\sqcap}\!: 1)$ |
| 3. | $a : \exists r.B$ | $(\Rightarrow_{\sqcap}\!: 1)$ |
| 4. | $a : \forall r.(\neg A \sqcup \neg B)$ | $(\Rightarrow_{\sqcap}\!: 1)$ |

# Example

Tableau proof:

| | | |
|---|---|---|
| 1. | $a : \exists r.A \sqcap \exists r.B \sqcap \forall r.(\neg A \sqcup \neg B)$ | $\mathcal{A}$ |
| 2. | $a : \exists r.A$ | $(\Rightarrow_{\sqcap}\colon\ 1)$ |
| 3. | $a : \exists r.B$ | $(\Rightarrow_{\sqcap}\colon\ 1)$ |
| 4. | $a : \forall r.(\neg A \sqcup \neg B)$ | $(\Rightarrow_{\sqcap}\colon\ 1)$ |
| 5. | $(a, b) : r$ | $(\Rightarrow_{\exists}\colon\ 2)$ |
| 6. | $b : A$ | $(\Rightarrow_{\exists}\colon\ 2)$ |

# Example

Tableau proof:

| | | |
|---|---|---|
| 1. | $a : \exists r.A \sqcap \exists r.B \sqcap \forall r.(\neg A \sqcup \neg B)$ | $\mathcal{A}$ |
| 2. | $a : \exists r.A$ | $(\Rightarrow_{\sqcap}\colon\ 1)$ |
| 3. | $a : \exists r.B$ | $(\Rightarrow_{\sqcap}\colon\ 1)$ |
| 4. | $a : \forall r.(\neg A \sqcup \neg B)$ | $(\Rightarrow_{\sqcap}\colon\ 1)$ |
| 5. | $(a, b) : r$ | $(\Rightarrow_{\exists}\colon\ 2)$ |
| 6. | $b : A$ | $(\Rightarrow_{\exists}\colon\ 2)$ |
| 7. | $(a, c) : r$ | $(\Rightarrow_{\exists}\colon\ 3)$ |
| 8. | $c : B$ | $(\Rightarrow_{\exists}\colon\ 3)$ |

# Example

Tableau proof:

| | | |
|---|---|---|
| 1. | $a : \exists r.A \sqcap \exists r.B \sqcap \forall r.(\neg A \sqcup \neg B)$ | $\mathcal{A}$ |
| 2. | $a : \exists r.A$ | $(\Rightarrow_{\sqcap}\text{: } 1)$ |
| 3. | $a : \exists r.B$ | $(\Rightarrow_{\sqcap}\text{: } 1)$ |
| 4. | $a : \forall r.(\neg A \sqcup \neg B)$ | $(\Rightarrow_{\sqcap}\text{: } 1)$ |
| 5. | $(a, b) : r$ | $(\Rightarrow_{\exists}\text{: } 2)$ |
| 6. | $b : A$ | $(\Rightarrow_{\exists}\text{: } 2)$ |
| 7. | $(a, c) : r$ | $(\Rightarrow_{\exists}\text{: } 3)$ |
| 8. | $c : B$ | $(\Rightarrow_{\exists}\text{: } 3)$ |
| 9. | $b : \neg A \sqcup \neg B$ | $(\Rightarrow_{\forall}\text{: } 4{,}5)$ |

# Example

Tableau proof:

| | | |
|---|---|---|
| 1. | $a : \exists r.A \sqcap \exists r.B \sqcap \forall r.(\neg A \sqcup \neg B)$ | $\mathcal{A}$ |
| 2. | $a : \exists r.A$ | $(\Rightarrow_{\sqcap}\text{: } 1)$ |
| 3. | $a : \exists r.B$ | $(\Rightarrow_{\sqcap}\text{: } 1)$ |
| 4. | $a : \forall r.(\neg A \sqcup \neg B)$ | $(\Rightarrow_{\sqcap}\text{: } 1)$ |
| 5. | $(a, b) : r$ | $(\Rightarrow_{\exists}\text{: } 2)$ |
| 6. | $b : A$ | $(\Rightarrow_{\exists}\text{: } 2)$ |
| 7. | $(a, c) : r$ | $(\Rightarrow_{\exists}\text{: } 3)$ |
| 8. | $c : B$ | $(\Rightarrow_{\exists}\text{: } 3)$ |
| 9. | $b : \neg A \sqcup \neg B$ | $(\Rightarrow_{\forall}\text{: } 4,5)$ |
| 10. | $c : \neg A \sqcup \neg B$ | $(\Rightarrow_{\forall}\text{: } 4,7)$ |

# Example

Tableau proof:

| | | |
|---|---|---|
| 1. | $a : \exists r.A \sqcap \exists r.B \sqcap \forall r.(\neg A \sqcup \neg B)$ | $\mathcal{A}$ |
| 2. | $a : \exists r.A$ | $(\Rightarrow_{\sqcap}\text{: } 1)$ |
| 3. | $a : \exists r.B$ | $(\Rightarrow_{\sqcap}\text{: } 1)$ |
| 4. | $a : \forall r.(\neg A \sqcup \neg B)$ | $(\Rightarrow_{\sqcap}\text{: } 1)$ |
| 5. | $(a, b) : r$ | $(\Rightarrow_{\exists}\text{: } 2)$ |
| 6. | $b : A$ | $(\Rightarrow_{\exists}\text{: } 2)$ |
| 7. | $(a, c) : r$ | $(\Rightarrow_{\exists}\text{: } 3)$ |
| 8. | $c : B$ | $(\Rightarrow_{\exists}\text{: } 3)$ |
| 9. | $b : \neg A \sqcup \neg B$ | $(\Rightarrow_{\forall}\text{: } 4,5)$ |
| 10. | $c : \neg A \sqcup \neg B$ | $(\Rightarrow_{\forall}\text{: } 4,7)$ |

11. $b : \neg A$ $(\Rightarrow_{\sqcup}\text{: } 9)$

$\times$ $(6, 11)$

12. $b : \neg B$ $(\Rightarrow_{\sqcup}\text{: } 9)$

# Example

Tableau proof:

| | | |
|---|---|---|
| 1. | $a : \exists r.A \sqcap \exists r.B \sqcap \forall r.(\neg A \sqcup \neg B)$ | $\mathcal{A}$ |
| 2. | $a : \exists r.A$ | $(\Rightarrow_{\sqcap}\!: 1)$ |
| 3. | $a : \exists r.B$ | $(\Rightarrow_{\sqcap}\!: 1)$ |
| 4. | $a : \forall r.(\neg A \sqcup \neg B)$ | $(\Rightarrow_{\sqcap}\!: 1)$ |
| 5. | $(a, b) : r$ | $(\Rightarrow_{\exists}\!: 2)$ |
| 6. | $b : A$ | $(\Rightarrow_{\exists}\!: 2)$ |
| 7. | $(a, c) : r$ | $(\Rightarrow_{\exists}\!: 3)$ |
| 8. | $c : B$ | $(\Rightarrow_{\exists}\!: 3)$ |
| 9. | $b : \neg A \sqcup \neg B$ | $(\Rightarrow_{\forall}\!: 4,5)$ |
| 10. | $c : \neg A \sqcup \neg B$ | $(\Rightarrow_{\forall}\!: 4,7)$ |

11. $b : \neg A$ $(\Rightarrow_{\sqcup}\!: 9)$

$\times\ (6, 11)$

12. $b : \neg B$ $(\Rightarrow_{\sqcup}\!: 9)$

13. $c : \neg A$ $(\Rightarrow_{\sqcup}\!: 10)$

14. $c : \neg B$ $(\Rightarrow_{\sqcup}\!: 10)$

$\times\ (8, 14)$

# Example

...there exists an open branch in the tableau, hence:

Output: $\{a : \exists r.A \sqcap \exists r.B \sqcap \forall r.(\neg A \sqcup \neg B)\}$ is **CONSISTENT**
$(\exists r.A \sqcap \exists r.B) \sqcap (\neg \exists r.(A \sqcap B))$ is **SATISFIABLE**
$\exists r.A \sqcap \exists r.B$ is **NOT SUBSUMED** by $\exists r.(A \sqcap B)$

Can we be sure?

Construct a *canonical model* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ of the ABox from the open branch:

- use the individual names from the branch to define the domain,
- use the atomic assertions of the form $a : A$ and $(a, b) : r$ to define an interpretation of the vocabulary.

# Example

| Branch | Canonical model |
| --- | --- |
| $a : \exists r.A \sqcap \exists r.B \sqcap \forall r.(\neg A \sqcup \neg B)$ | |
| $a : \exists r.A$ | |
| $a : \exists r.B$ | |
| $a : \forall r.(\neg A \sqcup \neg B)$ | |
| $(a, b) : r$ | $\Delta^{\mathcal{I}} = \{a, b, c\}$ |
| $b : A$ | $A^{\mathcal{I}} = \{b\}$ |
| $(a, c) : r$ | $B^{\mathcal{I}} = \{c\}$ |
| $c : B$ | $r^{\mathcal{I}} = \{(a, b), (a, c)\}$ |
| $b : \neg A \sqcup \neg B$ | |
| $c : \neg A \sqcup \neg B$ | |
| $b : \neg B$ | |
| $c : \neg A$ | |

Clearly, $\mathcal{I}$ is a model of $\{a : \exists r.A \sqcap \exists r.B \sqcap \forall r.(\neg A \sqcup \neg B)\}$.

# Exercise: reasoning

Problem: Is $\forall created.Painting \sqcap \exists created.\top$ subsumed by $\exists created.Painting$ ?

# Exercise: reasoning

Problem: Is $\forall created.Painting \sqcap \exists created.\top$ subsumed by $\exists created.Painting$ ?

Input: $\{a : \forall created.Painting \sqcap \exists created.\top \sqcap \forall created.\neg Painting\}$

Proof:

| | | |
|---|---|---|
| 1. | $a : \forall created.Painting \sqcap \exists created.\top \sqcap \forall created.\neg Painting$ | $\mathcal{A}$ |
| 2. | $a : \forall created.Painting$ | $(\Rightarrow_{\sqcap}$: 1) |
| 3. | $a : \exists created.\top$ | $(\Rightarrow_{\sqcap}$: 1) |
| 4. | $a : \forall created.\neg Painting$ | $(\Rightarrow_{\sqcap}$: 1) |
| 5. | $(a, b) : created$ | $(\Rightarrow_{\exists}$: 3) |
| 6. | $b : \top$ | $(\Rightarrow_{\exists}$: 3) |
| 7. | $b : Painting$ | $(\Rightarrow_{\forall}$: 2,5) |
| 8. | $b : \neg Painting$ | $(\Rightarrow_{\forall}$: 4,5) |
| | $\times\ (7, 8)$ | |

Output:

# Exercise: reasoning

Problem: Is $\forall created.Painting \sqcap \exists created.\top$ subsumed by $\exists created.Painting$ ?

Input: $\{a : \forall created.Painting \sqcap \exists created.\top \sqcap \forall created.\neg Painting\}$

Proof:

| | | |
|---|---|---|
| 1. | $a : \forall created.Painting \sqcap \exists created.\top \sqcap \forall created.\neg Painting$ | $\mathcal{A}$ |
| 2. | $a : \forall created.Painting$ | $(\Rightarrow_{\sqcap}$: 1) |
| 3. | $a : \exists created.\top$ | $(\Rightarrow_{\sqcap}$: 1) |
| 4. | $a : \forall created.\neg Painting$ | $(\Rightarrow_{\sqcap}$: 1) |
| 5. | $(a, b) : created$ | $(\Rightarrow_{\exists}$: 3) |
| 6. | $b : \top$ | $(\Rightarrow_{\exists}$: 3) |
| 7. | $b : Painting$ | $(\Rightarrow_{\forall}$: 2,5) |
| 8. | $b : \neg Painting$ | $(\Rightarrow_{\forall}$: 4,5) |
| | $\times\ (7, 8)$ | |

Output: Yes, it is **SUBSUMED**.

# Exercise: constructing canonical model

Construct a canonical model for the following open branch:

$a : \forall created.Painting$
$a : \exists created.\top$
$a : \forall created.\neg Sculpture$
$(a, b) : created$
$b : \top$
$b : Painting$
$b : \neg Sculpture$

Solution:

# Exercise: constructing canonical model

Construct a canonical model for the following open branch:

$a : \forall created.Painting$
$a : \exists created.\top$
$a : \forall created.\neg Sculpture$
$(a, b) : created$
$b : \top$
$b : Painting$
$b : \neg Sculpture$

Solution:

- $\Delta^{\mathcal{I}} = \{a, b\}$
- $Painting^{\mathcal{I}} = \{b\}$
- $Sculpture^{\mathcal{I}} = \emptyset$
- $created^{\mathcal{I}} = \{(a, b)\}$

# Correctness

The key computational properties are given via three theorems:

- soundness: *the algorithm proves only conclusions that are "really" true*
  - ⇔ IF the tableau proof succeeds, i.e. it closes on some input, THEN the input is inconsistent
  - ⇔ IF the input is consistent THEN the tableau does not close

    *Proof*: show that the tableau *rules preserve consistency*, i.e. whenever the branch is consistent before an application of a rule then after it, there must still exist at least one consistent branch.
- completeness: *the algorithm can prove every true conclusion*
  - ⇔ IF the input is inconsistent THEN the algorithm proves it, i.e. it closes
  - ⇔ IF the algorithm does not close THEN the input is consistent

    *Proof*: show that it is always possible to *construct a model of the input* from an open branch.

# Termination

- termination: *The algorithm always returns an answer after a finite number of inference steps.*

  *Proof*: The size of the resulting tableau is bounded by the size of the input (which is finite):

  - applications of $\Rightarrow_{\sqcup}$ and $\Rightarrow_{\sqcap}$ result in strictly shorter formulas,
  - for any $a$, the number of its successors generated by $\Rightarrow_{\exists}$ rule is limited by the number of assertions of type $a : \exists r.C$,
  - for any $(a, b) : r$, the number of possible applications of $\Rightarrow_{\forall}$ rule is limited by the number of assertions of type $a : \forall r.C$.

- complexity: with empty TBoxes, the tableau algorithm is PSPACE-complete.

# Reasoning with non-empty TBoxes

In order to account for the TBox $\mathcal{T}$, the tableau procedure has to be extended as follows:

Input:

- Replace every $C \equiv D \in \mathcal{T}$ with $C \sqsubseteq D$ and $D \sqsubseteq C$.
- Replace every $C \sqsubseteq D \in \mathcal{T}$ with $\top \equiv NNF(\neg C \sqcup D)$.
- Add $\mathcal{T}$ to the root of the tableau.

Tableau rule:

$\Rightarrow_{\equiv}$ **IF** $(\top \equiv C) \in S$ **and** an individual $a$ occurs in $S$
**THEN** $S' := S \cup \{a : C\}$

Note: $\top \equiv C$ indeed means that EVERY individual in any model must be $C$.

# Example

Problem: Is $C$ satisfiable w.r.t. $\mathcal{T} = \{C \sqsubseteq D, C \sqsubseteq \neg D\}$ ?

Tableau proof:

| | | |
|---|---|---|
| 1. | $\top \equiv \neg C \sqcup D$ | $\mathcal{T}$ |
| 2. | $\top \equiv \neg C \sqcup \neg D$ | $\mathcal{T}$ |
| 3. | $a : C$ | $\mathcal{A}$ |

# Example

Problem: Is $C$ satisfiable w.r.t. $\mathcal{T} = \{C \sqsubseteq D, C \sqsubseteq \neg D\}$ ?

Tableau proof:

| | | |
|---|---|---|
| 1. | $\top \equiv \neg C \sqcup D$ | $\mathcal{T}$ |
| 2. | $\top \equiv \neg C \sqcup \neg D$ | $\mathcal{T}$ |
| 3. | $a : C$ | $\mathcal{A}$ |
| 4. | $a : \neg C \sqcup D$ | $(\Rightarrow_{\equiv}\text{: } 1, 3)$ |
| 5. | $a : \neg C \sqcup \neg D$ | $(\Rightarrow_{\equiv}\text{: } 2, 3)$ |

# Example

Problem: Is $C$ satisfiable w.r.t. $\mathcal{T} = \{C \sqsubseteq D, C \sqsubseteq \neg D\}$ ?

Tableau proof:

| | | |
|---|---|---|
| 1. | $\top \equiv \neg C \sqcup D$ | $\mathcal{T}$ |
| 2. | $\top \equiv \neg C \sqcup \neg D$ | $\mathcal{T}$ |
| 3. | $a : C$ | $\mathcal{A}$ |
| 4. | $a : \neg C \sqcup D$ | $(\Rightarrow_{\equiv}\text{: } 1, 3)$ |
| 5. | $a : \neg C \sqcup \neg D$ | $(\Rightarrow_{\equiv}\text{: } 2, 3)$ |

6. $a : \neg C$ $(\Rightarrow_{\sqcup}\text{: } 4)$
$\times\ (1, 6)$

7. $a : D$ $(\Rightarrow_{\sqcup}\text{: } 4)$

# Example

Problem: Is $C$ satisfiable w.r.t. $\mathcal{T} = \{C \sqsubseteq D, C \sqsubseteq \neg D\}$ ?

Tableau proof:

1. $\top \equiv \neg C \sqcup D$ $\mathcal{T}$
2. $\top \equiv \neg C \sqcup \neg D$ $\mathcal{T}$
3. $a : C$ $\mathcal{A}$
4. $a : \neg C \sqcup D$ $(\Rightarrow_{\equiv}\text{: } 1, 3)$
5. $a : \neg C \sqcup \neg D$ $(\Rightarrow_{\equiv}\text{: } 2, 3)$

6. $a : \neg C$ $(\Rightarrow_{\sqcup}\text{: } 4)$
   $\times\ (1, 6)$

7. $a : D$ $(\Rightarrow_{\sqcup}\text{: } 4)$

8. $a : \neg C$ $(\Rightarrow_{\sqcup}\text{: } 5)$
   $\times\ (1, 8)$

9. $a : \neg D$ $(\Rightarrow_{\sqcup}\text{: } 5)$
   $\times\ (7, 9)$

Output: $C$ is **UNSATISFIABLE** w.r.t. $\mathcal{T}$

# Reasoning with non-empty TBoxes

- Soundness and completeness hold with small changes in proofs.
- The complexity is actually NExpTime-complete.
- Termination requires an additional supporting mechanism.

Termination:

Consider any input containing an axiom of the form: $\top \equiv \ldots \exists r.C \ldots$ A straightforward application of the rules $\Rightarrow_{\exists}$ and $\Rightarrow_{\equiv}$ might lead to an infinite expansion of the tableau tree.

Solution: Detect cycles and prevent further application of the $\Rightarrow_{\exists}$ rule. This is achieved by a special *blocking rule*.

# Blocking example

Problem: Is $B$ satisfiable w.r.t. $\mathcal{T} = \{\top \equiv \exists r.C\}$ ?

Tableau proof:

| | | |
|---|---|---|
| 1. | $\top \equiv \exists r.C$ | $\mathcal{T}$ |
| 2. | $a : B$ | $\mathcal{A}$ |

# Blocking example

Problem: Is $B$ satisfiable w.r.t. $\mathcal{T} = \{\top \equiv \exists r.C\}$ ?

Tableau proof:

| | | |
|---|---|---|
| 1. | $\top \equiv \exists r.C$ | $\mathcal{T}$ |
| 2. | $a : B$ | $\mathcal{A}$ |
| 3. | $a : \exists r.C$ | $(\Rightarrow_{\equiv}$: 1, 2) |

# Blocking example

Problem: Is $B$ satisfiable w.r.t. $\mathcal{T} = \{\top \equiv \exists r.C\}$ ?

Tableau proof:

| | | |
|---|---|---|
| 1. | $\top \equiv \exists r.C$ | $\mathcal{T}$ |
| 2. | $a : B$ | $\mathcal{A}$ |
| 3. | $a : \exists r.C$ | $(\Rightarrow_{\equiv}\text{: } 1, 2)$ |
| 4. | $(a, b) : r$ | $(\Rightarrow_{\exists}\text{: } 3)$ |
| 5. | $b : C$ | $(\Rightarrow_{\exists}\text{: } 3)$ |

# Blocking example

Problem: Is $B$ satisfiable w.r.t. $\mathcal{T} = \{\top \equiv \exists r.C\}$ ?

Tableau proof:

| | | |
|---|---|---|
| 1. | $\top \equiv \exists r.C$ | $\mathcal{T}$ |
| 2. | $a : B$ | $\mathcal{A}$ |
| 3. | $a : \exists r.C$ | $(\Rightarrow_{\equiv}\text{: } 1, 2)$ |
| 4. | $(a, b) : r$ | $(\Rightarrow_{\exists}\text{: } 3)$ |
| 5. | $b : C$ | $(\Rightarrow_{\exists}\text{: } 3)$ |
| 6. | $b : \exists r.C$ | $(\Rightarrow_{\equiv}\text{: } 1, 4)$ |

# Blocking example

Problem: Is $B$ satisfiable w.r.t. $\mathcal{T} = \{\top \equiv \exists r.C\}$ ?

Tableau proof:

| | | |
|---|---|---|
| 1. | $\top \equiv \exists r.C$ | $\mathcal{T}$ |
| 2. | $a : B$ | $\mathcal{A}$ |
| 3. | $a : \exists r.C$ | $(\Rightarrow_{\equiv}\text{: } 1, 2)$ |
| 4. | $(a, b) : r$ | $(\Rightarrow_{\exists}\text{: } 3)$ |
| 5. | $b : C$ | $(\Rightarrow_{\exists}\text{: } 3)$ |
| 6. | $b : \exists r.C$ | $(\Rightarrow_{\equiv}\text{: } 1, 4)$ |
| 7. | $(b, c) : r$ | $(\Rightarrow_{\exists}\text{: } 6)$ |
| 8. | $c : C$ | $(\Rightarrow_{\exists}\text{: } 6)$ |

# Blocking example

Problem: Is $B$ satisfiable w.r.t. $\mathcal{T} = \{\top \equiv \exists r.C\}$ ?

Tableau proof:

| | | |
|---|---|---|
| 1. | $\top \equiv \exists r.C$ | $\mathcal{T}$ |
| 2. | $a : B$ | $\mathcal{A}$ |
| 3. | $a : \exists r.C$ | $(\Rightarrow_{\equiv}\text{: } 1, 2)$ |
| 4. | $(a, b) : r$ | $(\Rightarrow_{\exists}\text{: } 3)$ |
| 5. | $b : C$ | $(\Rightarrow_{\exists}\text{: } 3)$ |
| 6. | $b : \exists r.C$ | $(\Rightarrow_{\equiv}\text{: } 1, 4)$ |
| 7. | $(b, c) : r$ | $(\Rightarrow_{\exists}\text{: } 6)$ |
| 8. | $c : C$ | $(\Rightarrow_{\exists}\text{: } 6)$ |
| 9. | $c : \exists r.C$ | $(\Rightarrow_{\equiv}\text{: } 1, 7)$ |
| 10. | $\dots$ | |

# Blocking example

Problem: Is $B$ satisfiable w.r.t. $\mathcal{T} = \{\top \equiv \exists r.C\}$ ?

Tableau proof:

| | | |
|---|---|---|
| 1. | $\top \equiv \exists r.C$ | $\mathcal{T}$ |
| 2. | $a : B$ | $\mathcal{A}$ |
| 3. | $a : \exists r.C$ | $(\Rightarrow_{\equiv}\text{: } 1, 2)$ |
| 4. | $(a, b) : r$ | $(\Rightarrow_{\exists}\text{: } 3)$ |
| 5. | $b : C$ | $(\Rightarrow_{\exists}\text{: } 3)$ |
| 6. | $b : \exists r.C$ | $(\Rightarrow_{\equiv}\text{: } 1, 4)$ |
| 7. | $(b, c) : r$ | $(\Rightarrow_{\exists}\text{: } 6)$ |
| 8. | $c : C$ | $(\Rightarrow_{\exists}\text{: } 6)$ |
| 9. | $c : \exists r.C$ | $(\Rightarrow_{\equiv}\text{: } 1, 7)$ |
| 10. | $\ldots$ | |

$\Rightarrow_{\mathrm{B}}$ **IF** $b$ is a (possibly indirect) successor of $a$ in $S$ **and** it is the case that:

$$\{C \mid b : C \in S\} \subseteq \{D \mid a : D \in S\}$$

**THEN** mark $b$ as BLOCKED by $a$ in $S$ and do not apply $\Rightarrow_{\exists}$ to $b$

# Blocking example

Problem: Is $B$ satisfiable w.r.t. $\mathcal{T} = \{\top \equiv \exists r.C\}$ ?

Tableau proof:

| | | | |
|---|---|---|---|
| 1. | $\top \equiv \exists r.C$ | $\mathcal{T}$ | |
| 2. | $a : B$ | $\mathcal{A}$ | |
| 3. | $a : \exists r.C$ | $(\Rightarrow_{\equiv}\text{: } 1, 2)$ | $L_a = \{B, \exists r.C\}$ |
| 4. | $(a, b) : r$ | $(\Rightarrow_{\exists}\text{: } 3)$ | $L_b = \{C, \exists r.C\}$ |
| 5. | $b : C$ | $(\Rightarrow_{\exists}\text{: } 3)$ | $L_c = \{C, \exists r.C\}$ |
| 6. | $b : \exists r.C$ | $(\Rightarrow_{\equiv}\text{: } 1, 4)$ | |
| 7. | $(b, c) : r$ | $(\Rightarrow_{\exists}\text{: } 6)$ | |
| 8. | $c : C$ | $(\Rightarrow_{\exists}\text{: } 6)$ | $c$ *is BLOCKED by* $b$ |
| 9. | $c : \exists r.C$ | $(\Rightarrow_{\equiv}\text{: } 1, 7)$ | $\leftarrow$ do not expand this anymore! |

$\Rightarrow_{\text{B}}$ **IF** $b$ is a (possibly indirect) successor of $a$ in $S$ **and** it is the case that:

$$\{C \mid b : C \in S\} \subseteq \{D \mid a : D \in S\}$$

**THEN** mark $b$ as BLOCKED by $a$ in $S$ and do not apply $\Rightarrow_{\exists}$ to $b$

# Blocking example

Problem: Is $B$ satisfiable w.r.t. $\mathcal{T} = \{\top \equiv \exists r.C\}$ ?

Tableau proof:

| | | | |
|---|---|---|---|
| 1. | $\top \equiv \exists r.C$ | $\mathcal{T}$ | |
| 2. | $a : B$ | $\mathcal{A}$ | |
| 3. | $a : \exists r.C$ | $(\Rightarrow_{\equiv}$: 1, 2) | $L_a = \{B, \exists r.C\}$ |
| 4. | $(a, b) : r$ | $(\Rightarrow_{\exists}$: 3) | $L_b = \{C, \exists r.C\}$ |
| 5. | $b : C$ | $(\Rightarrow_{\exists}$: 3) | $L_c = \{C, \exists r.C\}$ |
| 6. | $b : \exists r.C$ | $(\Rightarrow_{\equiv}$: 1, 4) | |
| 7. | $(b, c) : r$ | $(\Rightarrow_{\exists}$: 6) | |
| 8. | $c : C$ | $(\Rightarrow_{\exists}$: 6) | $c$ *is BLOCKED by* $b$ |
| 9. | $c : \exists r.C$ | $(\Rightarrow_{\equiv}$: 1, 7) | $\leftarrow$ do not expand this anymore! |

Output: $C$ is **SATISFIABLE** w.r.t. $\mathcal{T}$

*Warning*: to ensure completeness, the blocking rule can be applied ONLY when no other rules (apart from $\Rightarrow_{\exists}$) apply anymore on the branch.

# Data structures for tableaux

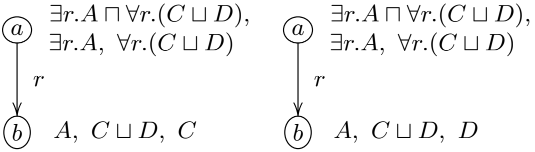
Practical implementations of the tableau algorithm for DLs often use different *data structures* — closer to DL models. An open branch is represented as a *labeled graph*, where:

| | |
|---|---|
| *nodes* ⇝ individuals | *edges* ⇝ role relationships |
| *node labels* ⇝ concepts | *edge labels* ⇝ role names |

Example:

1. $a : \exists r.A \sqcap \forall r.(C \sqcup D)$
2. $a : \exists r.A$
3. $a : \forall r.(C \sqcup D)$
4. $(a, b) : r$
5. $b : A$
6. $b : C \sqcup D$

7. $b : C$      8. $b : D$




Note: The branching $\Rightarrow_{\sqcup}$ rule involves duplicating of the branch.

# Summary

- All basic reasoning problems for $\mathcal{ALC}$ can be turned into a task of *finding a model* of the ABox and the TBox.
- Tableau algorithm is a *decision procedure*, i.e. sound, complete and terminating algorithm, employing exactly this strategy.
- Whenever the algorithm terminates and tableau is open, we can construct a *canonical model* of the input.

Resources:

F. Baader, U. Sattler. *An Overview of Tableau Algorithms for Description Logics*. In: Studia Logica 69(1), 2001.

(see Blackboard)

Description Logic resources: `http://dl.kr.org/`

Next:

- LoTREC tutorial and handing in the assignment.

▷ *Please bring laptops with LoTREC installed*

`http://www.irit.fr/Lotrec/`