CS 7810 - Knowledge Representation and Reasoning (for the Semantic Web)
08 - Tableau Algorithms for DLs

Adila Krisnadhi

Data Semantics Lab
Wright State University, Dayton, OH

November 10, 2016

Outline

1. Basic Idea: Example from Propositional Logic
2. Satisfiability of $\mathcal{ALC}$ Concepts
3. Satisfiability of $\mathcal{ALC}$ Knowledge Bases

Acknowledgements

Materials in this presentation are adapted from:

- Sebastian Rudolph, “Tableau Procedures I”, slides for Foundations of Semantic Web Technologies course, TU Dresden, May 23, 2014.
- Sebastian Rudolph, “Tableau Procedures II”, slides for Foundations of Semantic Web Technologies course, TU Dresden, May 30, 2014.

Outline

1. Basic Idea: Example from Propositional Logic
2. Satisfiability of $\mathcal{ALC}$ Concepts
3. Satisfiability of $\mathcal{ALC}$ Knowledge Bases

Computing Satisfiability

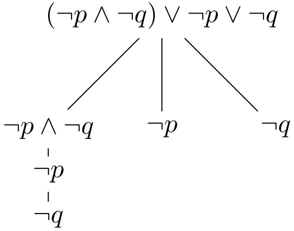
- A concept is satisfiable if it has a model, i.e., there is an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}} \neq \emptyset$
- Given a concept C, how do you decide if it is satisfiable?
  - So far: try to come up with an arbitrary model of C.
  - Can we automate it?
- Tableau algorithm: constructive decision procedure that tries to build models, if possible.
- Analogy from propositional logic:
  - Truth tables: enumerate exponentially many interpretations until finding a model
  - Tableau algorithm for propositional logic (can avoid checking exponentially many combinations)

Example from Propositional Logic

Is the following formula satisfiable: $(p \vee q) \rightarrow (\neg p \vee \neg q)$?
Negation in front of complex expressions difficult to handle, so reformulate:

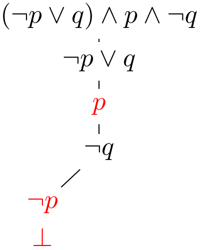
$$(p \vee q) \rightarrow (\neg p \vee \neg q)$$
$$\neg(p \vee q) \vee (\neg p \vee \neg q)$$
$$(\neg p \wedge \neg q) \vee \neg p \vee \neg q$$

Propositional Logic Tableau



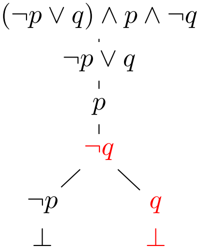
- tableau: finite set of trableau branches (paths from root to leaf)
- conjunction extends a branch with the conjuncts
- disjunction splits a branch into two, each corresponds to a disjunct
- **complete branch**: all complex expressions (conjunctions and disjunctions) in a branch have been used to extend/split the branch
- try compare it with the truth table for the formula!

Propositional Logic Tableau



- **complete branch**: (i) if $p \wedge q$ in the branch, then so are p and q; (ii) if $p \vee q$ in the branch, then p or q or both are in the branch
- **closed branch**: contains an **atomic contradiction** (**clash**)
- **closed tableau**: all of its branches are closed
- termination condition: if every branch is either closed or complete
- tableau has an open and complete branch $\rightsquigarrow$ formula is satisfiable
- from an open and complete branch, we can construct a model (see whiteboard)
- tableau is closed $\rightsquigarrow$ formula is unsatisfiable

Propositional Logic Tableau



- **complete branch**: (i) if $p \wedge q$ in the branch, then so are p and q; (ii) if $p \vee q$ in the branch, then p or q or both are in the branch
- **closed branch**: contains an **atomic contradiction** (**clash**)
- **closed tableau**: all of its branches are closed
- termination condition: if every branch is either closed or complete
- tableau has an open and complete branch $\rightsquigarrow$ formula is satisfiable
- from an open and complete branch, we can construct a model (see whiteboard)
- tableau is closed $\rightsquigarrow$ formula is unsatisfiable

Storing only One Branch in Memory

- mark disjunction with choice points, each corresponds to a branch
- all extensions of the branch due to such a choice are also marked
- when clash occurs, remove marked formulas and try next choice

$$(\neg p \vee q) \wedge p \wedge q$$

$$\neg p^{1a} \vee q^{1b}$$

$$p$$

$$q$$

$$\neg p^{1a}$$

$$\bot^{1a}$$

Storing only One Branch in Memory

- mark disjunction with choice points, each corresponds to a branch
- all extensions of the branch due to such a choice are also marked
- when clash occurs, remove marked formulas and try next choice

$$(\neg p \vee q) \wedge p \wedge q$$

$$\neg p^{1a} \vee q^{1b}$$

$$p$$

$$q$$

~~$\neg p^{1a}$~~

~~$\bot^{1a}$~~

$$q^{1b}$$

$\rightsquigarrow$ Found an open and complete branch.

Storing only One Branch in Memory

- mark disjunction with choice points, each corresponds to a branch
- all extensions of the branch due to such a choice are also marked
- when clash occurs, remove marked formulas and try next choice

$$(\neg p \vee q) \wedge p \wedge q$$
$$\neg p^{1a} \vee q^{1b}$$
$$p$$
$$q$$
$$\cancel{\neg p^{1a}}$$
$$\cancel{\bot^{1a}}$$
$$q^{1b}$$

$$(\neg p \vee q) \wedge p \wedge \neg q$$
$$\neg p^{1a} \vee q^{1b}$$
$$p$$
$$\neg q$$
$$\neg p^{1a}$$
$$\bot^{1a}$$

$\rightsquigarrow$ Found an open and complete branch.

Storing only One Branch in Memory

- mark disjunction with choice points, each corresponds to a branch
- all extensions of the branch due to such a choice are also marked
- when clash occurs, remove marked formulas and try next choice

$(\neg p \vee q) \wedge p \wedge q$	$(\neg p \vee q) \wedge p \wedge \neg q$
$\neg p^{1a} \vee q^{1b}$	$\neg p^{1a} \vee q^{1b}$
p	p
q	$\neg q$
~~$\neg p^{1a}$~~	~~$\neg p^{1a}$~~
~~$\bot^{1a}$~~	~~$\bot^{1a}$~~
q^{1b}	q^{1b}
$\rightsquigarrow$ Found an open and complete branch.	$\bot^{1b}$

Storing only One Branch in Memory

- mark disjunction with choice points, each corresponds to a branch
- all extensions of the branch due to such a choice are also marked
- when clash occurs, remove marked formulas and try next choice

$(\neg p \vee q) \wedge p \wedge q$

$\neg p^{1a} \vee q^{1b}$

p

q

~~$\neg p^{1a}$~~

~~$\bot^{1a}$~~

q^{1b}

⇝ Found an open and complete branch.

$(\neg p \vee q) \wedge p \wedge \neg q$

$\neg p^{1a} \vee q^{1b}$

p

$\neg q$

~~$\neg p^{1a}$~~

~~$\bot^{1a}$~~

~~q^{1b}~~

~~$\bot^{1b}$~~

⇝ All branches are closed.

Outline

1 Basic Idea: Example from Propositional Logic

2 Satisfiability of $\mathcal{ALC}$ Concepts

3 Satisfiability of $\mathcal{ALC}$ Knowledge Bases

Tableau for DLs

- Reasoning problem: “given a concept C, is C satisfiable?”
- We start with a simpler setting: knowledge base is empty
  $\rightsquigarrow$ C is unsatisfiable if it is contradictory “by itself”
- tableau branch: finite set of atomic propositions of the form $C(a)$, $R(a, b)$ (can be visualized as a graph involving elements of the universe)
- tableau: set of branches $\rightsquigarrow$ set of “possible graphs”
- for each existential quantifier: introduce a new domain element
- for each universal quantifier: propagate filler concept expressions to neighboring elements.
- as in propositional tableau, negations must only appear in front of atomic concepts
- **clash** occurs if (i) both propositions of the form $C(a)$ and $\neg C(a)$ is in a branch; or (ii) proposition of the form $\bot(a)$ is in a branch

Negation Normal Form

$$\begin{aligned}
\neg(C \sqcup D) &\rightsquigarrow \neg C \sqcap \neg D \\
\neg(C \sqcap D) &\rightsquigarrow \neg C \sqcup \neg D \\
\neg\neg C &\rightsquigarrow C \\
\neg(\forall R.C) &\rightsquigarrow \exists R.\neg C \\
\neg(\exists R.C) &\rightsquigarrow \forall R.\neg C \\
\neg(\leqslant n R.C) &\rightsquigarrow \geqslant (n+1) R.C \\
\neg(\geqslant n R.C) &\rightsquigarrow \leqslant (n-1) R.C, \quad n \geq 1 \\
\neg(\geqslant 0 R.C) &\rightsquigarrow \bot \\
(\geqslant 0 R.C) &\rightsquigarrow \top
\end{aligned}$$

- apply the above rules exhaustively (until none can be applied)
- result: equivalent concept in **negation normal form** (**NNF**)
- example: $\neg(\exists R.\neg C \sqcap \forall S.(\neg D \sqcup E)) \equiv \forall R.C \sqcup \exists S.(D \sqcap \neg E)$

Data structure: labeled graph where $\mathbf{V}$ is the set of nodes, $\mathbf{E}$ is the set of edges (pairs of nodes), $\mathbf{L}(v)$ is the set of labels of a node v, and $\mathbf{L}(v, v')$ is the set of labels of the edge from node v to node v'.

Input: $\mathcal{ALC}$ concept C in NNF.

Initialization: $\mathbf{V} := \{v_0\}$, $\mathbf{E} := \emptyset$, and $\mathbf{L}(v_0) := \{C\}$

Extend the graph by applying any applicable tableau rules until no more rules can be applied.

- $\sqcap$-rule: if there is a node v with $D \sqcap E \in \mathbf{L}(v)$ and $\{D, E\} \not\subseteq \mathbf{L}(v)$, then set $\mathbf{L}(v) := \mathbf{L}(v) \cup \{D, E\}$
- $\sqcup$-rule: if there is a node v with $D \sqcup E \in \mathbf{L}(v)$ and $\{D, E\} \cap \mathbf{L}(v) = \emptyset$, then choose one of $X \in \{D, E\}$ nondeterministically and set $\mathbf{L}(v) := \mathbf{L}(v) \cup \{X\}$
- $\exists$-rule: if there is a node v with $\exists R.D \in \mathbf{L}(v)$ and there is no node v' such that $\langle v, v' \rangle \in E$ and $D \in \mathbf{L}(v')$, then create a new node v', set $\mathbf{V} := \mathbf{V} \cup \{v'\}$, $\mathbf{E} := \mathbf{E} \cup \{\langle v, v' \rangle\}$, $\mathbf{L}(v') := \{D\}$, and $\mathbf{L}(v, v') := \{R\}$
- $\forall$-rule: if there are nodes v, v' with $\langle v, v' \rangle \in \mathbf{E}$, $R \in \mathbf{L}(v, v')$, $\forall R.D \in \mathbf{L}(v)$, and $D \notin \mathbf{L}(v')$, then set $\mathbf{L}(v') := \mathbf{L}(v') \cup \{D\}$

Output: "satisfiable" if we can construct a clash-free tableau where no more rules can be applied. Otherwise, "unsatisfiable"

Note: rule applications exhibit "don't care" nondeterminism; choice of disjunction exhibits "don't know" nondeterminism

Example

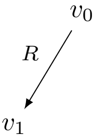
Input: $\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))$

v_0

$$\mathbf{L}(v_0) = \{\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A)) \\ \exists R.(A \sqcup \exists R.B), \exists R.\neg A, \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))\}$$

Example

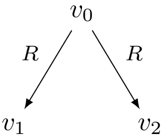
Input: $\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))$



$$\mathbf{L}(v_0) = \{\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))$$
$$\exists R.(A \sqcup \exists R.B), \exists R.\neg A, \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))\}$$
$$\mathbf{L}(v_1) = \{A \sqcup \exists R.B\}$$

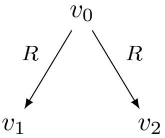
Example

Input: $\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))$



$$\begin{aligned}
\mathbf{L}(v_0) &= \{\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A)) \\
&\quad \exists R.(A \sqcup \exists R.B), \textcolor{red}{\exists R.\neg A}, \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))\} \\
\mathbf{L}(v_1) &= \{A \sqcup \exists R.B\} \\
\mathbf{L}(v_2) &= \{\textcolor{red}{\neg A}\}
\end{aligned}$$

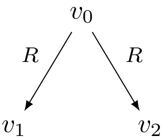
Input: $\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))$



$$\begin{aligned}
\mathbf{L}(v_0) &= \{\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A)) \\
&\qquad \exists R.(A \sqcup \exists R.B), \exists R.\neg A, \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))\} \\
\mathbf{L}(v_1) &= \{A \sqcup \exists R.B, \neg A \sqcap \forall R.(\neg B \sqcup A)\} \\
\mathbf{L}(v_2) &= \{\neg A, \neg A \sqcap \forall R.(\neg B \sqcup A)\}
\end{aligned}$$

Example

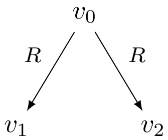
Input: $\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))$



$$
\begin{aligned}
\mathbf{L}(v_0) &= \{\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A)) \\
&\qquad \exists R.(A \sqcup \exists R.B), \exists R.\neg A, \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))\} \\
\mathbf{L}(v_1) &= \{A \sqcup \exists R.B, \neg A \sqcap \forall R.(\neg B \sqcup A), \neg A, \forall R.(\neg B \sqcup A)\} \\
\mathbf{L}(v_2) &= \{\neg A, \neg A \sqcap \forall R.(\neg B \sqcup A)\}
\end{aligned}
$$

Example

Input: $\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))$



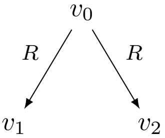
$$\mathbf{L}(v_0) = \{\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A)) \\ \exists R.(A \sqcup \exists R.B), \exists R.\neg A, \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))\}$$

$$\mathbf{L}(v_1) = \{A \sqcup \exists R.B, \neg A \sqcap \forall R.(\neg B \sqcup A), \neg A, \forall R.(\neg B \sqcup A), A\}$$

$$\mathbf{L}(v_2) = \{\neg A, \neg A \sqcap \forall R.(\neg B \sqcup A)\}$$

Example

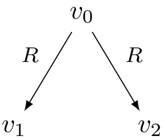
Input: $\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))$



$$
\begin{aligned}
\mathbf{L}(v_0) &= \{\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A)) \\
&\qquad \exists R.(A \sqcup \exists R.B), \exists R.\neg A, \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))\} \\
\mathbf{L}(v_1) &= \{A \sqcup \exists R.B, \neg A \sqcap \forall R.(\neg B \sqcup A), \color{red}{\neg A}, \forall R.(\neg B \sqcup A), \cancel{A}\} \\
\mathbf{L}(v_2) &= \{\neg A, \neg A \sqcap \forall R.(\neg B \sqcup A)\}
\end{aligned}
$$

Example

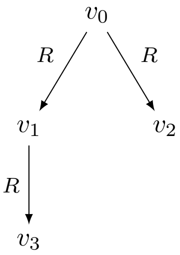
Input: $\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))$



$$\begin{aligned}
\mathbf{L}(v_0) &= \{\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A)) \\
&\quad \exists R.(A \sqcup \exists R.B), \exists R.\neg A, \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))\} \\
\mathbf{L}(v_1) &= \{{\color{red}A \sqcup \exists R.B}, \neg A \sqcap \forall R.(\neg B \sqcup A), \neg A, \forall R.(\neg B \sqcup A), \cancel{A}, \\
&\quad {\color{red}\exists R.B}\} \\
\mathbf{L}(v_2) &= \{\neg A, \neg A \sqcap \forall R.(\neg B \sqcup A)\}
\end{aligned}$$

Example

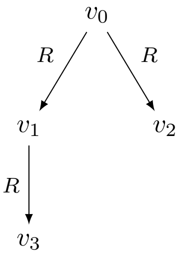
Input: $\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))$



$$
\begin{aligned}
\mathbf{L}(v_0) &= \{\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A)) \\
&\quad \exists R.(A \sqcup \exists R.B), \exists R.\neg A, \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))\} \\
\mathbf{L}(v_1) &= \{A \sqcup \exists R.B, \neg A \sqcap \forall R.(\neg B \sqcup A), \neg A, \forall R.(\neg B \sqcup A), \cancel{A}, \\
&\quad \exists R.B\} \\
\mathbf{L}(v_2) &= \{\neg A, \neg A \sqcap \forall R.(\neg B \sqcup A)\} \\
\mathbf{L}(v_3) &= \{B\}
\end{aligned}
$$

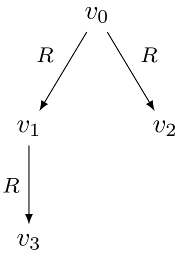
Example

Input: $\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))$



$$
\begin{aligned}
\mathbf{L}(v_0) &= \{\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A)) \\
&\qquad \exists R.(A \sqcup \exists R.B), \exists R.\neg A, \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))\} \\
\mathbf{L}(v_1) &= \{A \sqcup \exists R.B, \neg A \sqcap \forall R.(\neg B \sqcup A), \neg A, \forall R.(\neg B \sqcup A), \cancel{A}, \\
&\qquad \exists R.B\} \\
\mathbf{L}(v_2) &= \{\neg A, \neg A \sqcap \forall R.(\neg B \sqcup A)\} \\
\mathbf{L}(v_3) &= \{B, \neg B \sqcup A\}
\end{aligned}
$$

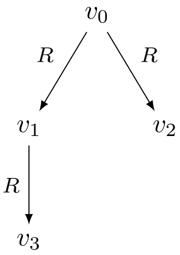
Input: $\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))$



$$\begin{aligned}
\mathbf{L}(v_0) &= \{\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A)) \\
&\qquad \exists R.(A \sqcup \exists R.B), \exists R.\neg A, \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))\} \\
\mathbf{L}(v_1) &= \{A \sqcup \exists R.B, \neg A \sqcap \forall R.(\neg B \sqcup A), \neg A, \forall R.(\neg B \sqcup A), \cancel{A}, \\
&\qquad \exists R.B\} \\
\mathbf{L}(v_2) &= \{\neg A, \neg A \sqcap \forall R.(\neg B \sqcup A)\} \\
\mathbf{L}(v_3) &= \{B, \neg B \sqcup A, \neg B\}
\end{aligned}$$

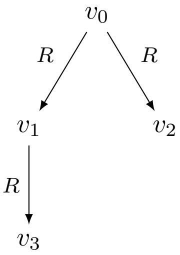
Example

Input: $\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))$



$$\begin{aligned}
\mathbf{L}(v_0) &= \{\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A)) \\
&\quad \exists R.(A \sqcup \exists R.B), \exists R.\neg A, \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))\} \\
\mathbf{L}(v_1) &= \{A \sqcup \exists R.B, \neg A \sqcap \forall R.(\neg B \sqcup A), \neg A, \forall R.(\neg B \sqcup A), \cancel{A}, \\
&\quad \exists R.B\} \\
\mathbf{L}(v_2) &= \{\neg A, \neg A \sqcap \forall R.(\neg B \sqcup A)\} \\
\mathbf{L}(v_3) &= \{B, \neg B \sqcup A, \cancel{\neg B}\}
\end{aligned}$$

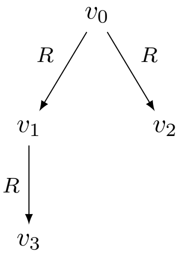
Input: $\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))$



$$\begin{aligned}
\mathbf{L}(v_0) &= \{\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A)) \\
&\qquad \exists R.(A \sqcup \exists R.B), \exists R.\neg A, \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))\} \\
\mathbf{L}(v_1) &= \{A \sqcup \exists R.B, \neg A \sqcap \forall R.(\neg B \sqcup A), \neg A, \forall R.(\neg B \sqcup A), \xcancel{A}, \\
&\qquad \exists R.B\} \\
\mathbf{L}(v_2) &= \{\neg A, \neg A \sqcap \forall R.(\neg B \sqcup A)\} \\
\mathbf{L}(v_3) &= \{B, \neg B \sqcup A, \xcancel{\neg B}, A\}
\end{aligned}$$

Example

Input: $\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))$



$$\begin{aligned}
\mathbf{L}(v_0) &= \{\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A)) \\
&\quad \exists R.(A \sqcup \exists R.B), \exists R.\neg A, \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))\} \\
\mathbf{L}(v_1) &= \{A \sqcup \exists R.B, \neg A \sqcap \forall R.(\neg B \sqcup A), \neg A, \forall R.(\neg B \sqcup A), \cancel{A}, \\
&\quad \exists R.B\} \\
\mathbf{L}(v_2) &= \{\color{red}{\neg A, \neg A \sqcap \forall R.(\neg B \sqcup A)}, \color{red}{\forall R.(\neg B \sqcup A)}\} \\
\mathbf{L}(v_3) &= \{B, \neg B \sqcup A, \cancel{\neg B}, A\}
\end{aligned}$$

Since the complete tableau is clash-free, the output is "satisfiable"
$\rightsquigarrow$ the input concept is satisfiable, and we can construct a model (next slide)

Model Construction

A model $\mathcal{I}$ for $C \coloneqq \exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A))$ is as follows:

$$\Delta^{\mathcal{I}} = \{v_0, v_1, v_2, v_3\}$$
$$A^{\mathcal{I}} = \{v_3\}$$
$$B^{\mathcal{I}} = \{v_3\}$$
$$R^{\mathcal{I}} = \{\langle v_0, v_1\rangle, \langle v_0, v_2\rangle, \langle v_1, v_3\rangle\}$$

The following are easy to verify by the semantics:

$$(\neg A)^{\mathcal{I}} = (\neg B)^{\mathcal{I}} = \{v_0, v_1, v_2\} \qquad (\exists R.B)^{\mathcal{I}} = \{v_1\} \qquad (\exists R.\neg A)^{\mathcal{I}} = \{v_0\}$$
$$(\neg B \sqcup A)^{\mathcal{I}} = \{v_0, v_1, v_2, v_3\} \qquad (\forall R.(\neg B \sqcup A))^{\mathcal{I}} = \{v_0, v_1, v_2, v_3\}$$
$$(\neg A \sqcap \forall R.(\neg B \sqcup A))^{\mathcal{I}} = \{v_0, v_1, v_2\} \qquad (\forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A)))^{\mathcal{I}} = \{v_0, v_2, v_3\}$$
$$(A \sqcup \exists R.B)^{\mathcal{I}} = \{v_1, v_3\} \qquad (\exists R.(A \sqcup \exists R.B))^{\mathcal{I}} = \{v_0, v_1\}$$
$$(\exists R.(A \sqcup \exists R.B) \sqcap \exists R.\neg A \sqcap \forall R.(\neg A \sqcap \forall R.(\neg B \sqcup A)))^{\mathcal{I}} = \{v_0\}$$

Since $C^{\mathcal{I}} \neq \emptyset$, C is thus satisfiable.

Correctness of the Algorithm I

- termination:
  - the number of nested quantifiers decrease in every node generated
  - every node is labeled only with subformulas of the input concept
  - the input concept has only polynomially many subformulas
- soundness:
  - if the output is "satisfiable", then we can construct a model of the input concept, which implies that the input concept is indeed satisfiable
- completeness:
  - if the input concept is satisfiable, then it has a model, and this model can be used to construct a clash-free tableau for the concept.

Correctness of the Algorithm II

Theorem

1. The tableau algorithm for $\mathcal{ALC}$ concepts terminates for every input
2. If the output is "satisfiable", then the input concept is satisfiable
3. If the input concept is satisfiable, then the output is "satisfiable"

Corollary

Every $\mathcal{ALC}$ concept C has the following properties:

1. **finite model property**: if C has a model, then it also has a finite model (i.e., has only finitely many universe elements)
2. **tree model property**: if C has a model, then it also has a tree-shaped model

- the finite and tree-shaped model above can be obtained by the model construction from a clash-free tableau
- finiteness and/or tree-shapedness may no longer hold in the presence of knowledge bases (i.e., not just concepts)

Example for Unsatisfiable Concept

Input: $(\exists R.A \sqcup \exists R.\neg B) \sqcap \forall R.(\neg A \sqcap B)$

Note: Formulas due to picking a choice point are marked with underscore.

v_0

$$\mathbf{L}(v_0) = \{(\exists R.A \sqcup \exists R.\neg B) \sqcap \forall R.(\neg A \sqcap B), \exists R.A \sqcup \exists R.\neg B \underline{\exists R.A} \sqcup \exists R.\neg$$

$$\forall R.(\neg A \sqcap B), \underline{\exists R.A}, \underline{\exists R.\neg B}\}$$

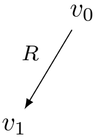
$$\mathbf{L}(v_1) = \{\underline{A}\}\mathbf{L}(v_1)$$

All choice points lead to a clash $\rightsquigarrow$ the concept is unsatisfiable.

Example for Unsatisfiable Concept

Input: $(\exists R.A \sqcup \exists R.\neg B) \sqcap \forall R.(\neg A \sqcap B)$

Note: Formulas due to picking a choice point are marked with underscore.



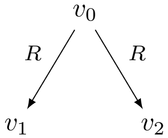
$$\mathbf{L}(v_0) = \{(\exists R.A \sqcup \exists R.\neg B) \sqcap \forall R.(\neg A \sqcap B), \exists R.A \sqcup \exists R.\neg B\underline{\exists R.A} \sqcup \exists R.\neg$$
$$\forall R.(\neg A \sqcap B), \underline{\exists R.A}, \underline{\exists R.\neg B}\}$$
$$\mathbf{L}(v_1) = \{\underline{A}\}\mathbf{L}(v_1)$$

All choice points lead to a clash $\rightsquigarrow$ the concept is unsatisfiable.

Example for Unsatisfiable Concept

Input: $(\exists R.A \sqcup \exists R.\neg B) \sqcap \forall R.(\neg A \sqcap B)$

Note: Formulas due to picking a choice point are marked with underscore.



$$\mathbf{L}(v_0) = \{(\exists R.A \sqcup \exists R.\neg B) \sqcap \forall R.(\neg A \sqcap B), \exists R.A \sqcup \exists R.\neg B\underline{\exists R.A} \sqcup \exists R.\neg$$
$$\forall R.(\neg A \sqcap B), \underline{\exists R.A}, \underline{\exists R.\neg B}\}$$

$$\mathbf{L}(v_1) = \{\underline{A}\}\mathbf{L}(v_1)$$

All choice points lead to a clash $\rightsquigarrow$ the concept is unsatisfiable.

Outline

1. Basic Idea: Example from Propositional Logic
2. Satisfiability of $\mathcal{ALC}$ Concepts
3. Satisfiability of $\mathcal{ALC}$ Knowledge Bases

Instead of concept satisfiability, we consider knowledge base satisfiability.

Knowledge Base Satisfiability

Given a knowledge base $\mathcal{K}$, is $\mathcal{K}$ satisfiable?

Note that a knowledge base is the union of a TBox, an ABox, and an RBox. For $\mathcal{ALC}$, RBox is always empty.

If we have a decision procedure (i.e., algorithm) for KB satisfiability, then we could use it to solve other DL basic reasoning problems.

Below, $\mathcal{K}$ is a knowledge base,
$c, c_0, \ldots, c_n$ are fresh individual names not occurring in $\mathcal{K}$,
U is the universal role (usable if the logic allows it – $\mathcal{ALC}$ does not!),
a, b are individual names (may or may not occur in $\mathcal{K}$),
C, D are concepts, $R, R_1, \ldots, R_n$ are roles/properties.

1. **Axiom entailment**:
   - $\mathcal{K} \models C \sqsubseteq D$ iff $\mathcal{K} \cup \{(C \sqcap \neg D)(c)\}$ is unsatisfiable
   - $\mathcal{K} \models C \sqsubseteq D$ iff $\mathcal{K} \cup \{\top \sqsubseteq \exists U.(C \sqcap \neg D)\}$ is unsatisfiable
   - $\mathcal{K} \models C(a)$ iff $\mathcal{K} \cup \{\neg C(a)\}$ is unsatisfiable
   - $\mathcal{K} \models R(a, b)$ iff $\mathcal{K} \cup \{\neg R(a, b)\}$ is unsatisfiable
   - $\mathcal{K} \models \neg R(a, b)$ iff $\mathcal{K} \cup \{R(a, b)\}$ is unsatisfiable
   - $\mathcal{K} \models \mathsf{Dis}(R_1, R_2)$ iff $\mathcal{K} \cup \{R_1(c_1, c_2), R_2(c_1, c_2)\}$ is unsatisfiable
   - $\mathcal{K} \models R_1 \circ \cdots \circ R_n \sqsubseteq R$ iff $\mathcal{K} \cup \{\neg R(c_0, c_n), R_1(c_0, c_1), \ldots, R_n(c_{n-1}, c_n)\}$ is unsatisfiable

2. **Concept (un)satisfiability**:
   C is unsatisfiable w.r.t. $\mathcal{K}$ iff $\mathcal{K} \models C \sqsubseteq \bot$ iff $\mathcal{K} \cup \{C(c)\}$ is unsatisfiable.
   $\leadsto$ Thus, C is satisfiable w.r.t $\mathcal{K}$ iff $\mathcal{K} \cup \{C(c)\}$ is satisfiable.
3. **Concept subsumption**:
   C is subsumed by D w.r.t. $\mathcal{K}$ iff $\mathcal{K} \models C \sqsubseteq D$ iff $\mathcal{K} \cup \{(C \sqcap \neg D)(c)\}$ is unsatisfiable iff $\mathcal{K} \cup \{\top \sqsubseteq \exists U.(C \sqcap \neg D)\}$ is unsatisfiable
4. **Instance checking**:
   An individual a is an instance of a concept C w.r.t $\mathcal{K}$ iff $\mathcal{K} \models C(a)$

Tableau algorithm for deciding knowledge base satisfiability is obtained by modifying/extending the tableau algorithm for deciding concept satisfiability as follows:

- Accommodating ABox $\rightsquigarrow$ modify the initialization phase by using information from the ABox
- Accommodating TBox $\rightsquigarrow$ internalize/compress the TBox and add a tableau rule special for TBox

Other tableau rules ($\sqcap$, $\sqcup$, $\exists$, $\forall$) as well as the definition of clash stay the same.

Accommodating ABox

We accommodate the ABox by modifying the initialization:

For ABox $\mathcal{A}$ part of the input, initialize the tableau graph $G = \langle \mathbf{V}, \mathbf{E}, \mathbf{L} \rangle$:

- Initialize the set of nodes $\mathbf{V}$ to contain a node v_a for every individual name a occurring in $\mathcal{A}$
- Initialize node labels $\mathbf{L}(v_a) := \{C \mid C(a) \in \mathcal{A}\}$
- For every role assertion $R(a, b)$, initialize the set of edges $\mathbf{E}$ to contain an edge $\langle v_a, v_b \rangle$ and the edge label $\mathbf{L}(v_a, v_b)$ to contain R.

If $\mathcal{A}$ is empty, we set $\mathbf{V} := \{v_0\}$ for a fresh node v_0 and $\mathbf{E} := \emptyset$ and $\mathbf{L}(v_0) := \emptyset$.

Afterwards, the tableau rules can be applied to the graph initialized as above.

- Concept equivalence $C \equiv D$ are replaced with $C \sqsubseteq D$ and $D \sqsubseteq C$
- Every GCI $C \sqsubseteq D$ is equivalent to $\top \sqsubseteq \neg C \sqcup D$

The TBox containing n GCIs:

$$\mathcal{T} = \{C_i \sqsubseteq D_i \mid 1 \leq i \leq n\}$$

can be compressed/**internalized** into the following equivalent TBox containing only a single axiom:

$$\mathcal{T}' = \{\top \sqsubseteq \bigsqcap_{1 \leq i \leq n} (\neg C_i \sqcup D_i)\}$$

Denote the NNF of the right-hand side of the GCI in $\mathcal{T}'$ as the concept $C_{\mathcal{T}}$.

Accommodating TBox

- Assuming the TBox is internalized, we could use the $\mathcal{T}$-rule:

> $\mathcal{T}$-rule: For an arbitrary node v such that $C_\mathcal{T} \notin \mathbf{L}(v)$, set $\mathbf{L}(v) := \mathbf{L}(v) \cup \{C_\mathcal{T}\}$

But there is a potential problem ...

Consider TBox $\mathcal{T} = \{\top \sqsubseteq A, A \sqsubseteq \exists R.A\}$. Is A satisfiable given $\mathcal{T}$? (That is, is there a model of both A and $\mathcal{T}$?)

Termination is not guaranteed!
Reason: the quantifier depth does not necessarily decrease for newly introduced child nodes.

What do we do? $\leadsto$ we should recognize "cycles" (repeated node labelings)

Node Blocking

For detecting repeated node labelings

Let $G = \langle \mathbf{V}, \mathbf{E}, \mathbf{L} \rangle$ be the tableau graph/tree.

A node $v \in \mathbf{V}$ **directly blocks** a node $v' \in \mathbf{V}$, if:

1. v' is reachable from v,
2. $\mathbf{L}(v') \subseteq \mathbf{L}(v)$, and
3. there is no directly blocking node v'' such that v' is reachable from v''

A node v' is **blocked** if either v' is directly blocked node or there is a directly blocked node w such that v' is reachable from w.

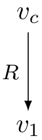
The $\exists$-rule can only be applied to nodes that are **NOT blocked**.

Example

Is A satisfiable with respect to the TBox $\mathcal{T} = \{A \sqsubseteq \exists R.A\}$?

Answer: First, $C_{\mathcal{T}} = \neg A \sqcup \exists R.A$. Also, A is satisfiable w.r.t $\mathcal{T}$ iff $\mathcal{T} \cup \{A(c)\}$ is satisfiable.

The clash-free tableau is:



$$\mathbf{L}(v_c) = \{A, \neg A \sqcup \exists R.A, \exists R.A\}$$
$$\mathbf{L}(v_1) = \{A, \neg A \sqcup \exists R.A, \exists R.A\}$$

note: v_1 is directly blocked by v_c

Model Construction with Blocked Nodes

- Blocked nodes do not represent elements in the model.
- For each edge from v to v', if v' is directly blocked (by some node, say w), then the model would have an "edge" from v to w instead.
- This model is finite $\rightsquigarrow$ finite model property holds.
- But the model may not be tree-shaped.

The tableau from the previous slide gives us the following model of A and $\mathcal{T}$.

$$\Delta^{\mathcal{I}} = \{v_0\}$$
$$A^{\mathcal{I}} = \{v_0\}$$
$$R^{\mathcal{I}} = \{\langle v_0, v_0\rangle\}$$

More Examples (on the whiteboard)

- Is A satisfiable with respect to $\mathcal{T} = \{A \sqsubseteq \exists R.A \sqcap \exists S.B\}$?
- Is A satisfiable with respect to $\mathcal{T} = \{A \sqsubseteq \exists R.B, B \sqsubseteq D \sqcap \forall S.B, D \sqsubseteq \exists S.C, B \sqcap C \sqsubseteq \bot\}$?
- Is A satisfiable with respect to $\mathcal{T} = \{A \sqsubseteq B \sqcap \exists R.C, B \equiv C \sqcup D, C \sqsubseteq \exists R.D, \exists R.B \sqsubseteq A\}$?

For each of the above example, if the answer is yes, give a model of $\mathcal{T}$ that satisfies A.