Description Logics

Carsten Lutz and Ulrike Sattler

TU Dresden, Germany

Part III: Automata-based Decision Procedures
Automata vs. Tableaux

Tableau algorithms
+ based on a simple idea (model construction)
+ amenable to optimization techniques
+ basis for all state-of-the-art DL reasoners
  - bad for proving deterministic upper time bounds
  - termination proofs can become very hard

Automata-based algorithms
+ elegant and simple
+ well-suited for proving \( \text{ExpTime} \) upper bounds
+ no termination proofs
  - no optimized implementations exist (?)
  - not very natural for upper space bounds and
    nondeterministic upper time bounds
(Infinite) $k$-ary $M$-tree $T$: mapping $\{1, \ldots, k\}^* \rightarrow M$

Looping tree automaton $A$: tuple $(Q, M, I, \Delta)$ with
- $Q$ a finite set of states
- $M$ an alphabet
- $I \subseteq Q$ a set of initial states
- $\Delta \subseteq Q \times M \times Q^k$ a transition relation

Run of $A$ on $T$: mapping $r : \{1, \ldots, k\}^* \rightarrow Q$ such that
1. $r(\epsilon) \in I$
2. $(r(\alpha), T(\alpha), r(\alpha 1), \ldots, r(\alpha k)) \in \Delta$ for all $\alpha \in \{1, \ldots, k\}^*$

$L(A) = \{T \mid$ there is a run of $A$ on $T\}$

$A$ accepts $T$

$A$ recognizes $L(A)$
Examples

Looping tree automaton on alphabet \( \{a, b\} \)

\[ Q = \{q_a, q_b\} \]

\[ M = \{a, b\} \]

\[ I = \{q_a, q_b\} \]

\[ \Delta = \{(q_a, a, q_a, q_a), (q_b, b, q_a, q_a) \]

\[ (q_b, b, q_b, q_a) \]

\[ (q_b, b, q_a, q_b) \]

\[ (q_b, b, q_b, q_b) \}\]

A tree language that is not recognizable:
Idea for deciding concept satisfiability w.r.t. TBoxes:

1. for concept $C_0$ and TBox $\mathcal{T}$, define automaton $A_{C_0,\mathcal{T}}$ such that $A_{C_0,\mathcal{T}}$ accepts precisely the tree models of $C_0$ and $\mathcal{T}$

2. use emptiness test for looping automata to decide satisfiability of $C_0$ w.r.t. $\mathcal{T}$

(emptyness test: given $A$, is $L(A) = \emptyset$?)

Note: this presupposes the tree model property!

Can be used to establish $\text{ExpTime}$ upper bound:

- size of automata is usually exponential in size of $C_0$ and $\mathcal{T}$.
- emptyness can be decided in deterministic polynomial time [VardiWolper]
The Description Logic $\text{ALCIO}$

$\text{ALCIO} = \text{ALC}$

+ inverse roles $R^{-}$

+ nominals $N_1, N_2, \ldots$

Nominals $N$ are used like concept names but interpreted in singleton sets $N^I$.

Example:

RealCatholic $\vdash$ Religious $\sqcap \exists$follows.Pope

$\text{ALCIO}$-ABox consistency can be reduced to $\text{ALCIO}$-concept satisfiability:

$\{a : C, \ b : D, \ (a, b) : R\}$ consistent

iff

$\exists L. (N_a \sqcap C) \sqcap \exists L. (N_b \sqcap D) \sqcap \exists L. (N_a \sqcap \exists R \cdot N_b)\text{ is satisfiable}$
\textbf{ALCIO} lacks the tree model property: $N \sqcap \exists R. \exists R. N$

But then: How can we use automata?

1. abstract interpretations into \textit{Hintikka trees} such that $C_0$ and $\mathcal{T}$ have a model iff $C_0$ and $\mathcal{T}$ have a Hintikka tree

2. define automata for Hintikka trees rather than for tree models.
Let $C_0$, $\mathcal{T}$ be in NNF and $\vdash$ and $\text{cl}(C_0, \mathcal{T})$ be defined as before.

Hintikka sets are used as node labels in Hintikka trees.

**Hintikka set for $C_0$ and $\mathcal{T}$:** subset $\Psi \subseteq \text{cl}(C_0, \mathcal{T})$ such that

(S1) if $C \cap D \in \Psi$, then $\{C, D\} \subseteq \Psi$
(S2) if $C \cup D \in \Psi$, then $\{C, D\} \cap \Psi \neq \emptyset$
(S3) $\{C, \neg C\} \not\subseteq \Psi$ for all $C \in \text{cl}(C_0, \mathcal{T})$
(S4) $C \vdash D \in \mathcal{T}$ implies $C \leftrightarrow D \in \Psi$

$\mathcal{H}_{C_0,\mathcal{T}}$: set of all Hintikka-sets for $C_0$ and $\mathcal{T}$
Hintikka Trees

Let $k$ be the number of concepts $\exists R.C$ and $\exists R^- . C$ in $\text{cl}(C_0, \mathcal{T})$

These concepts are linearly ordered with $\mathcal{E}_i$ denoting the $i$’th one.

A $k + 1$-tuple of Hintikka-sets $S, S_1, \ldots, S_k$ is matching iff

(M1) if $\mathcal{E}_i = \exists R . C \in S$, then $C \in S_i$

(M2) if $\forall R . C \in S$ and $\mathcal{E}_i = \exists R . D \in S$, then $C \in S_i$

(M3) if $\forall R . C \in S_i$ and $\mathcal{E}_i = \exists \text{Inv}(R) . D \in S$, then $C \in S$

Hintikka tree for $C_0$ and $\mathcal{T}$:

$k$-ary $\mathcal{H}_{C_0, \mathcal{T}}$-tree $T$ such that, for all $\alpha, \beta \in \{1, \ldots, k\}^*$:

(T1) $C_0 \in T(\epsilon)$

(T2) the tuple $(T(\alpha), T(\alpha 1), \ldots, T(\alpha k))$ is matching

(T3) if $N \in T(\alpha) \cap T(\beta)$, then $T(\alpha) = T(\beta)$

Lemma. $C_0$ and $\mathcal{T}$ have a model iff there exists a Hintikka-tree for $C_0$ and $\mathcal{T}$. 
Basic idea:

- use Hintikka sets as states and define $\Delta$ such that

$$q_0 = \ell \text{ for all } (q_0, \ell, q_1, \ldots, q_k) \in \Delta$$

(recall that $\Delta \subseteq Q \times M \times Q^k$)

- use initial states to ensure that $C_0 \in T(\varepsilon)$

- Check matching properties via transition relation:

  e.g. (M1) if $E_i = \exists R.C \in S^*$, then $C \in S_i$

**But:** (T3) is of **global** character

(T3) if $N \in T(\alpha) \cap T(\beta)$, then $T(\alpha) = T(\beta)$
(T3) if $N \in T(\alpha) \cap T(\beta)$, then $T(\alpha) = T(\beta)$

**Additional effort needed:**

States are pairs $(S, F)$ with

- $S$ Hintikka set (as before)
- $F$ nominal flock:

  mapping from $\text{Nom}(C_0, \mathcal{T})$ to $\mathcal{H}_{C_0, \mathcal{T}}$

set of nominals occurring in $C_0$ and $\mathcal{T}$.

**Intuition:** second component of states used for *bookkeeping* (bb)

- initial state “guesses” a nominal flock
- every state in a run has same nominal flock
- if nominal $N$ occurs in a node label $T(\alpha)$, then $T(\alpha) = F(N)$
Constructing Automata III

Automaton for $C_0$ and $\mathcal{T}$:

$\mathcal{A}_{C_0,\mathcal{T}} = (Q, M, I, \Delta)$ where

$Q = \{(S, F) \mid S \in \mathcal{H}_{C_0,\mathcal{T}} \text{ and } F \text{ nominal flock}\}$

$M = \mathcal{H}_{C_0,\mathcal{T}}$

$I = \{(S, F) \in Q \mid C_0 \in S \}$

$((S_0, F_0), S, (S_1, F_1), \ldots, (S_k, F_k)) \in \Delta$ iff

- $S_0 = S$
- $F_0 = F_1 = \cdots = F_k$
- if $N \in S$, then $S = F_0(N)$
- the tuple $(S_0, S_1, \ldots, S_k)$ is matching

**Lemma.** $T \in L(\mathcal{A}_{C_0,\mathcal{T}})$ iff $T$ is a Hintikka-tree for $C_0$ and $\mathcal{T}$
Results

Size of \( \mathcal{A}_{C_0,T} \):

1. number of hintikka-sets exponential in \( |C_0, T| \)
2. number of nominals in \( C_0 \) and \( T \) linear in \( |C_0, T| \)

\[ 1 + 2 \implies 3. \quad |Q| \text{ and } |I| \text{ exponential in } |C_0, T|. \]

\[ 1 \implies 4. \quad |M| \text{ exponential in } |C_0, T|. \]

\[ 3 + 4 \implies 5. \quad |\Delta| \text{ exponential in } |C_0, T|. \]

\[ 3 - 5 \implies \text{size of } |\mathcal{A}_{C_0,T}| \text{ exponential in } |C_0, T|. \]

Since the emptiness problem for looping automata is in \( \text{PTime} \):

**Theorem.** \( \text{ALCIO} \)-concept satisfiability and subsumption w.r.t. TBoxes

is in \( \text{ExpTime} \).

\( \text{ExpTime} \) lower bound for \( \text{ALCIO} \)-concept satisfiability without TBoxes

will be proved later!
The Emptiness Problem of Looping Automata

Determine in $|Q|$ rounds the set of blocking states $B \subseteq Q$:

**Initialization:**

Set $B_0 := \{ q \in Q \mid \text{there is no } (q, a, q_1, \ldots, q_k) \in \Delta \}$

**Round $i$:**

Set $B_i := B_{i-1} \cup \{ q \in Q \mid \text{for all } (q, a, q_1, \ldots, q_k) \in \Delta$

there is an $1 \leq i \leq k$ such that $q_i \in B_{i-1} \}$

**Lemma.** $L(A) = \emptyset$ iff $I \subseteq B$.

Computation of the set $B = B_{|Q|}$ is clearly in PTime.
\( \text{\textit{ALC}}^+ = \text{\textit{ALC}} \)

+ transitive closure operator on roles \( R^+ \)

Semantics: \( (R^+)^I = (R^I)^+ \)

Example: \( \text{Car} \models \exists \text{has-part}. \text{Engine} \sqcap \exists \text{has-part}^+. \text{ShaftSeal} \)

Reflexive-transitive closure can be “simulated”. 

Diagram: 
- Car
  - has-part: Engine
  - has-part: Piston
- Engine
  - has-part: Piston
  - has-part: ShaftSeal
- Piston
  - has-part: ShaftSeal
- ShaftSeal
- Chassis
\[\text{Idea:}\]

\[\{\ldots, \forall R^+.C, \ldots\} \quad \text{or} \quad \{\ldots, \exists R^+.C, \ldots\}\]

\[R\]

\[\{\ldots, C, \forall R^+.C, \ldots\} \quad \text{or} \quad \{\ldots, C, \ldots\} \quad \{\ldots, \exists R^+.C, \ldots\}\]

\[\text{Problem: Satisfaction of existential restriction may be delayed forever}\]

\[\implies \text{Looping tree automata do not suffice!}\]
Büchi Tree Automata

Büchi tree automata: \( \mathcal{A} = (Q, M, I, \Delta, F) \)

like looping tree automata with additional set of accepting states \( F \subseteq Q \)

A run \( r : \{1, \ldots, k\}^* \rightarrow Q \) is accepting iff

for each (infinite) branch \( B \in \{1, \ldots, k\}^\omega \),

\[
\inf(B) \cap F \neq \emptyset.
\]

\( L(\mathcal{A}) = \{T \mid \text{there is an accepting run of } \mathcal{A} \text{ on } T\} \)

Lemma. The emptyness problem of Büchi tree automata is in \( \text{PTIME} \).
Generalized Büchi Tree Automata

**Generalized Büchi tree automata:** possibly many sets of accepting states

\[ \mathcal{A} = (Q, M, I, \Delta, F_1, \ldots, F_\ell) \]

A run \( r : \{1, \ldots, k\}^* \rightarrow Q \) is accepting iff

for each \( i \in \{1, \ldots, \ell\}, \)

for each (infinite) branch \( B \in \{1, \ldots, k\}^\omega, \)

\[ \inf(B) \cap F_i \neq \emptyset. \]

**Lemma.** For every generalized Büchi tree automaton, an equivalent Büchi tree automaton can be constructed.
Let $k$ be the number of concepts $\exists R.C$ or $\exists R^+.C$ in $\text{cl}(C_0, T)$.

These concepts are linearly ordered with $E_i$ denoting the $i$'th one ($1 \leq i \leq k$).

A $k + 1$-tuple of Hintikka-sets $S, S_1, \ldots, S_k$ is matching iff

(M1) if $E_i = \exists R.C \in S$, then $C \in S_i$

(M2) if $E_i = \exists R^+.C \in S$, then $\{\exists R^+.C, C\} \cap S_i \neq \emptyset$

(M3) if $\forall R.C \in S$ and $E_i = \exists R.D \in S$ or $E_i = \exists R^+.D \in S$, then $C \in S_i$

(M4) if $\forall R^+.C \in S$ and $E_i = \exists R.D \in S$ or $E_i = \exists R^+.D \in S$, then $\{\forall R^+.C, C\} \subseteq S_i$
Hintikka tree for $C_0$ and $\mathcal{T}$:

$k$-ary $\mathcal{H}_{C_0,\mathcal{T}}$-tree $T$ such that, for all $\alpha, \beta \in \{1, \ldots, k\}^*$:

(T1) $C_0 \in T(\epsilon)$

(T2) The tuple $(T(\alpha), T(\alpha 1), \ldots, T(\alpha k))$ is matching

(T3) if $\mathcal{E}_i = \exists R^+. C \in T(\alpha)$, then there exists an $n \in \mathbb{N}_+$ s.t. $C \in T(\alpha i^n)$.

Lemma. $C_0$ and $\mathcal{T}$ have a model iff there exists a Hintikka-tree for $C_0$ and $\mathcal{T}$. 
Büchi acceptance condition needed for:

(T3) if $\mathcal{E}_i = \exists R^+.C \in T(\alpha)$, then there exists an $n \in \mathbb{N}_+$ s.t. $C \in T(\alpha i^n)$.

Idea:

States are pairs $(S, f)$ with
- $S$ Hintikka set
- $f \in \{0, \ldots, k\}$ a flag.

Intuition:

If $r(\alpha) = (S, f)$, then

the existential restriction $\mathcal{E}_f = \exists R^+.C$ has been "seen" but not yet satisfied

For each $\mathcal{E}_i = \exists R^+.C \in \text{cl}(C_0, T)$, there is a set of accepting states $F_i$:

$$F_i = \{(S, j) \in Q \mid j \neq i\}$$
Constructing Automata II

Automaton for \( C_0 \) and \( \mathcal{T} \) (excerpt):

\[
\mathcal{A}_{C_0, \mathcal{T}} = (Q, M, I, \Delta, F_1, \ldots, F_i) \quad \text{where}
\]

\[
\# \text{ concepts } \exists R^+. C \in \text{ cl}(C_0, \mathcal{T})
\]

\[
I = \{(S, f) \in Q \mid C_0 \in S \text{ and } f = 0\}
\]

\[
F_i = \{(S, j) \in Q \mid j \neq i\}
\]

\[
((S_0, f_0), S, (S_1, f_1), \ldots, (S_k, f_k)) \in \Delta \text{ iff}
\]

- \( S_0 = S \)
- the tuple \( (S_0, S_1, \ldots, S_k) \) is matching

\[
f_i = \begin{cases} 
  i & \text{if } \mathcal{E}_i = \exists R^+. C \in S_0 \text{ and } C \notin S_i \\
  0 & \text{otherwise}
\end{cases}
\]

Lemma. \( T \in L(\mathcal{A}_{C_0, \mathcal{T}}) \) iff \( T \) is a Hintikka-tree for \( C_0 \) and \( \mathcal{T} \)

Theorem. \( \mathcal{ALC}^+ \)-concept satisfiability and subsumption w.r.t. TBoxes is in \( \text{ExpTime} \).
- Automata-based approach also works for logics from Part II

\begin{equation}
\implies \text{they are in } \text{ExpTime}.
\end{equation}

- There exist other acceptance conditions than Büchi

- There exist other automata models, for example alternating automata:

  Idea:
  - states are formulas, not sets of formulas
  - size of automaton is polynomial in the size of the input concept + TBox
  - emptiness check is \( \text{ExpTime}\)-complete