Using an Expressive Description Logic: FaCT or Fiction

Ian R. Horrocks

presented by Sebastian Mieth

March 9, 2006

Description Logics

Problems

- limited expressiveness
- intractability of subsumption algorithms

Paper's contribution

- sound and complete tableaux subsumption testing algorithm for expressive DL
- good performance on realistic applications

Terminologies

A terminology $\mathcal{T}$ defines the knowledge base's vocabulary and consists of

- concept names (sets of individuals)
- role names (binary relationships between individuals)
- complex concepts / roles (axioms)

DL systems differ w.r.t. the DL they use

Expressive Requirements for DLs

What: transitive relations (roles in DL)

- part-whole
  - (component-object, member-collection, ...)
  - $hasPart(Bob, Leg) \wedge hasPart(Leg, Ankle) \supset hasPart(Bob, Ankle)$
- causal
  - $caused(crash, injury) \wedge caused(injury, dead) \supset caused(crash, dead)$

Why: reasoning about parts, not knowing

- the level of decomposition
- the part-whole relation the part belongs to

The Language $\mathcal{ALC}$ I

Atomic concepts

- A, B (e.g., `Mother`, `Person`)

Atomic roles

- R, S (e.g., `hasChild`, `offspring`)

Concept descriptions

- $C, D \rightarrow \mathbf{CN} \mid \top \mid \bot \mid \neg C \mid C \sqcap D \mid \forall R.C \mid \exists R.T$
- **CN** is a set of concept names

The Language $\mathcal{ALC}$ II

Possible concept descriptions

- DL: $Male \sqcap (\exists Son.\neg Male)$

  FOL: $Male(x) \wedge (\exists y).[Son(x, y) \wedge \neg Male(y)]$

- DL: $\forall Child.[Male \sqcup \exists Offspring.Male) \sqcup \neg Male$

  FOL: $(\forall y).[Child(x, y) \vee$
  $(\exists z).(Offspring(y, z) \wedge Male(z))] \vee$
  $\neg Male(x)$

$\mathcal{ALC}$ extended: $\mathcal{ALCH}_{\mathcal{R}+}$

Axioms

- $C \sqsubseteq D$, $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$

  e.g., $hasPhD \sqsubseteq hasDegree$
- $C \doteq D$, $C^{\mathcal{I}} = D^{\mathcal{I}}$

  e.g., $Father \doteq Male \sqcap \exists Child.\top$
- $R \sqsubseteq S$, $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$

  e.g., $Father \sqsubseteq Parent$
- $R \in \mathbf{R}_+$, $R^{\mathcal{I}} = (R^{\mathcal{I}})^+$

  e.g., $Offspring \in \mathbf{R}_+$
- $\mathbf{R}_+$ is a set of transitive role names

Assumptions

- $\mathbf{R}_+$ and relation $\sqsubseteq$ have been defined in $\mathcal{T}, s.t.$
  - $\mathbf{R}_+ = \{R \mid R \in \mathbf{R}_+ \text{ is an axiom in } \mathcal{T}\}$
  - $R \sqsubseteq S$ iff $[R \sqsubseteq S \text{ is an axiom in } \mathcal{T}] \vee$ $[(\exists R').R \sqsubseteq R' \text{ is an axiom of } \mathcal{T} \text{ and } R' \sqsubseteq S]$
- concept expressions are in NNF
  - negations are only applied to concept names
- no axioms like $C \sqsubseteq D$ and $C \doteq D$ in $\mathcal{T}$
  - all concept names are atomic primitives

Tableau: Idea

Given a concept expression D

- try to construct a model for D
- represent model by a tree
- let nodes correspond to model's individuals
- label nodes with $\mathcal{ALCH}_{\mathcal{R}+}$-concepts being satisfied by them
- label arcs by dependencies between nodes (individuals)

Tableau: Example

$$D = Blond \sqcup E, \;\; E = \exists Father.(\exists Son.Male)$$

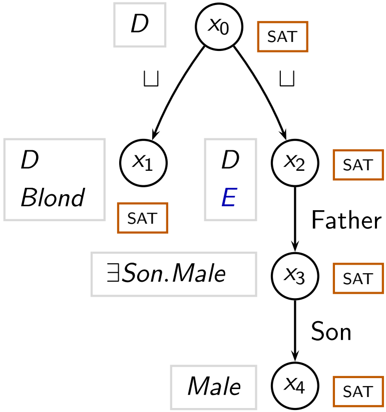


Tableau: Formal I

A tableau T for D is a triple $(\mathbf{S}, \mathcal{L}, \mathcal{E})$, where

- $\mathbf{S}$ is a set of individuals
- $\mathcal{L} : \mathbf{S} \longrightarrow 2^{sub(D)}$
- $\mathcal{E} : \mathbf{R}_D \longrightarrow 2^{\mathbf{S} \times \mathbf{S}}$
- $(\exists s \in \mathbf{S}).D \in \mathcal{L}(s)$

$sub(D)$ is the closure of subexpressions of D,

- e.g., $D = C \sqcup \exists R.(E \sqcap F)$,
  $sub(D) = \{D, C, \exists R.(E \sqcap F), E \sqcap F, E, F\}$

$\mathbf{R}_D$ is the set of rolenames occurring in D

Tableau: Formal II

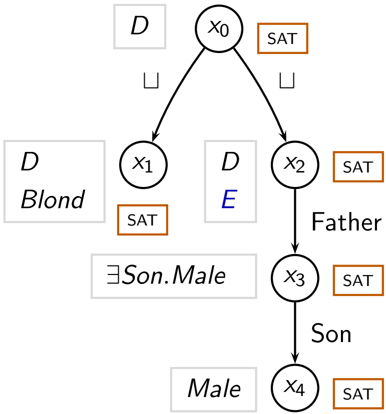
$D = Blond \sqcup E,\ \ E = \exists Father.(\exists Son.Male)$

$\mathbf{S} = \{x_1\}$

$\mathcal{L}(x) = \boxed{\cdot}$ next to x

$\mathcal{E}_{Father} = \{\}$

$\mathcal{E}_{Son} = \{\}$



$\mathbf{S} = \{x_2, x_3, x_4\}$

$\mathcal{L}(x) = \boxed{\cdot}$ next to x

$\mathcal{E}_{Father} = \{(x_2, x_3)\}$

$\mathcal{E}_{Son} = \{(x_3, x_4)\}$

Tableau: Formal III

Furthermore, we need to ensure that we construct a valid model w.r.t. D, thus all $s \in \mathbf{S}$ need to satisfy:

- *consistency*: $\perp \notin \mathcal{L}(s)$,
  $C \in \mathcal{L}(s) \supset \neg C \notin \mathcal{L}(s)$
- $\sqcap$: $C_1 \sqcap C_2 \in \mathcal{L}(s) \supset$
  $C_1 \in \mathcal{L}(s) \wedge C_2 \in \mathcal{L}(s)$
- $\sqcup$: $C_1 \sqcup C_2 \in \mathcal{L}(s) \supset$
  $C_1 \in \mathcal{L}(s) \vee C_2 \in \mathcal{L}(s)$

Tableau: Formal IV

- $\forall$: $\forall R.C \in \mathcal{L}(s) \wedge \langle s, t\rangle \in \mathcal{E}(R) \supset$
  $C \in \mathcal{L}(t)$
- $\exists$: $\exists R.C \in \mathcal{L}(s) \supset$
  $(\exists t \in \mathbf{S}).[\langle s, t\rangle \in \mathcal{E}(R) \wedge C \in \mathcal{L}(t)]$
- $\mathbf{R}_+$: $\forall R.C \in \mathcal{L}(s) \wedge S \in \mathbf{R}_+ \wedge S \sqsubseteq R \wedge$
  $\langle s, t\rangle \in \mathcal{E}(S) \supset \forall S.C \in \mathcal{L}(t)$
- $\sqsubseteq$: $R \sqsubseteq S \supset$
  $\mathcal{E}(R) \subseteq \mathcal{E}(S)$

Tableau: Formal V

Lemma 1

- an $\mathcal{ALCH}_{\mathcal{R}+}$-concept D is satisfiable iff there exists a tableau T for D

Given a tableau T, the corresponding model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ is constructed as follows:

- $\Delta^{\mathcal{I}} = \mathbf{S}$
- for all $A \in \mathbf{CN}$, $A^{\mathcal{I}} = \{s \mid A \in \mathcal{L}(s)\}$
- for all $R \in \mathbf{R}_D$,

$$R^{\mathcal{I}} = \begin{cases} \mathcal{E}(R)^+ & \text{if } R \in \mathbf{R}_+ \\ \mathcal{E}(R) \cup \bigcup_{S \sqsubseteq R} S^{\mathcal{I}} & \text{otherwise} \end{cases}$$

Tableau: Formal VI

$\mathbf{S} = \{x_2, x_3, x_4\}$

$\mathcal{L}(x) = \boxed{\cdot}$ next to x

$\mathcal{E}_{Father} = \{(x_2, x_3)\}$

$\mathcal{E}_{Son} = \{(x_3, x_4)\}$

$\Delta^{\mathcal{I}} = \{x_2, x_3, x_4\}$

$Male^{\mathcal{I}} = \{x_4\}$

$Father^{\mathcal{I}} = \{(x_2, x_3)\}$

$Son^{\mathcal{I}} = \{(x_3, x_4)\}$

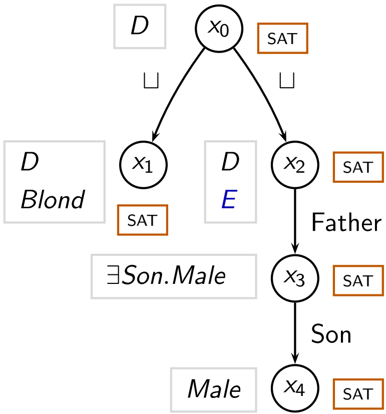


Tableau: Construction I

Initialization

- create a tree T with a single node x_0 and set $\mathcal{L}(x_0) = \{D\}$

Step

- expand the tree gradually according to the following rules

Termination

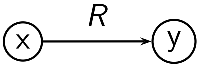
- if the root node x_0 is marked *satisfiable* or none of the rules is applicable, terminate

Tableau: first, some Terminology I

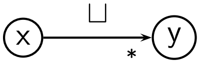
Some terminology

- *clash:* $\mathcal{L}(s)$ contains a clash iff $\bot \in \mathcal{L}(s) \vee \{C, \neg C\} \subseteq \mathcal{L}(s)$
- *pre-tableau:* $\mathcal{L}(s)$ is clash-free and contains no unexpanded con-/disjunctions

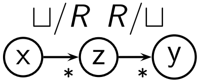
Tableau: first, some Terminology II



- *R-successor:* y is a R-successor of x if there is an edge $\langle x, y \rangle$ labelled R



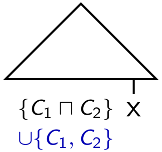
- *$\sqcup$-successor:* y is a $\sqcup$-successor of x if there is a path using $\sqcup$-edges from x to y



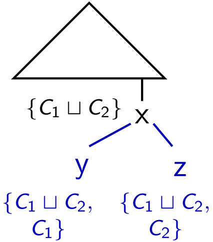
- *ancestor:* x is an ancestor of y if there is some path from x to y

the relations $\sqcup$-successor and ancestor are reflexive

Tableau: Rules 1 & 2



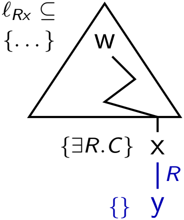
if x is a leaf of T, $\mathcal{L}(x)$ is clash-free,
$C_1 \sqcap C_2 \in \mathcal{L}(x)$
then $\mathcal{L}(x) = \mathcal{L}(x) \cup \{C_1, C_2\}$

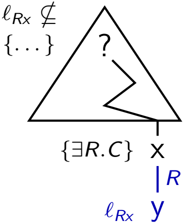


if x is a leaf of T, $\mathcal{L}(x)$ is clash-free,
$C_1 \sqcup C_2 \in \mathcal{L}(x)$
then create two new $\sqcup$-successors y,z of x with

$$\mathcal{L}(y) = \mathcal{L}(x) \cup \{C_1\}$$
$$\mathcal{L}(z) = \mathcal{L}(x) \cup \{C_2\}$$

Tableau: Rule 3

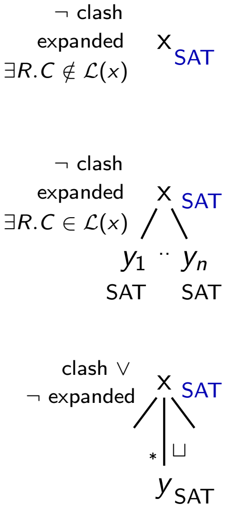




if x is a leaf of T, $\mathcal{L}(x)$ is a pre-tableau, $\exists R.C \in \mathcal{L}(x)$ **then** ($\forall$such $\exists R.C$)

1. $\ell_{Rx} = \{C\}$
   $\cup \{D \mid \forall S.D \in \mathcal{L}(x), R \sqsubseteq S\}$
   $\cup \{\forall S.D \mid \forall P.D \in \mathcal{L}(x),$
   $S \in \mathbf{R}_+, R \sqsubseteq S, S \sqsubseteq P\}$
2. if $\ell_{Rx} \subseteq \mathcal{L}(w)$, w is ancestor of x, then create a R-successor y of x with $\mathcal{L}(y) = \emptyset$
3. otherwise, create a R-successor y of x with $\mathcal{L}(y) = \ell_{Rx}$

Tableau: Rule 4



if x is a node of T, and either

- $\mathcal{L}(x)$ is a pre-tableau, $\exists R.C \notin \mathcal{L}(x)$
- $\mathcal{L}(x)$ is a pre-tableau, has successors, all successors are marked with *satisfiable*
- $\mathcal{L}(x)$ is not a pre-tableau, some $\sqcup$-successor is marked *satisfiable*

then mark x *satisfiable*

Tableau: Construction II

Termination (cont.)

use the tree to construct a tableau:

- $\mathbf{S} = \{x \mid x$ is a node in T, marked satisfiable, $\mathcal{L}(x) \neq \emptyset$ and is a pre-tableau$\}$
- $x, y \in \mathbf{S}$, $\langle x, y \rangle \in \mathcal{E}(R)$ if one of
  - y is a $\sqcup$-successor of an R-successor of x
  - x has an R-successor z, $\mathcal{L}(z) = \emptyset$, y is ancestor of x, $\ell_{Rx} \subseteq \mathcal{L}(y)$
  - $S \sqsubseteq R$, $\langle x, y \rangle \in \mathcal{E}(S)$

Example I

Concept names / Axioms

- $\mathbf{CN} = \{Male, Blond\}$
- $Child \notin \mathbf{R}_+$
- $Offspring \in \mathbf{R}_+$
- $Child \sqsupseteq Offspring$
- $(Child \sqsupseteq Child,\ Offspring \sqsupseteq Offspring)$

$$D = \forall Offspring.\ (\exists Child.Male \sqcap \exists Child.\neg Male \sqcap Blond) \sqcap \exists Child.\neg Male$$

Example II

$$C = \exists Child.Male \sqcap \exists Child.\neg Male \sqcap Blond$$
$$D = \forall Offspring.(C) \sqcap \exists Child.\neg Male$$

D

x_0

Example II

$C = \exists Child.Male \sqcap \exists Child.\neg Male \sqcap Blond$
$D = \forall Offspring.(C) \sqcap \exists Child.\neg Male$

D

$\sqcap$-*rule*

$\forall Offspring.(C)$

$\exists Child.\neg Male$

x_0

Example II

$C = \exists Child.Male \sqcap \exists Child.\neg Male \sqcap Blond$
$D = \forall Offspring.(C) \sqcap \exists Child.\neg Male$

ℓ_{Child,x_0}: $\neg Male$, C, $\forall Offspring.(C)$

D

$\sqcap$-rule: $\forall Offspring.(C)$, $\exists Child.\neg Male$

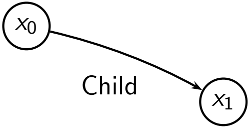
x_0

Example II

$$C = \exists Child.Male \sqcap \exists Child.\neg Male \sqcap Blond$$
$$D = \forall Offspring.(C) \sqcap \exists Child.\neg Male$$

D

$\sqcap$-rule:
- $\forall Offspring.(C)$
- $\exists Child.\neg Male$

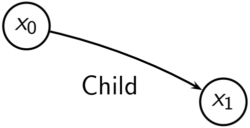


ℓ_{Child,x_0}:
- $\neg Male$
- C
- $\forall Offspring.(C)$

Example II

$$C = \exists Child.Male \sqcap \exists Child.\neg Male \sqcap Blond$$
$$D = \forall Offspring.(C) \sqcap \exists Child.\neg Male$$

	D
$\sqcap$-*rule*	$\forall Offspring.(C)$
	$\exists Child.\neg Male$



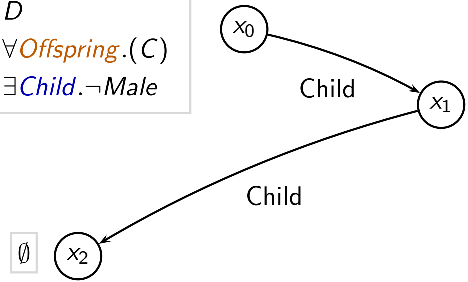
ℓ_{Child,x_0}	$\neg Male$
	C
	$\forall Offspring.(C)$
$\sqcap$-*rule*	$\exists Child.Male$
	$\exists Child.\neg Male$
	$Blond$

Example II

$C = \exists Child.Male \sqcap \exists Child.\neg Male \sqcap Blond$
$D = \forall Offspring.(C) \sqcap \exists Child.\neg Male$

D

$\sqcap$-rule:
- $\forall Offspring.(C)$
- $\exists Child.\neg Male$



ℓ_{Child,x_0}:
- $\neg Male$
- C
- $\forall Offspring.(C)$

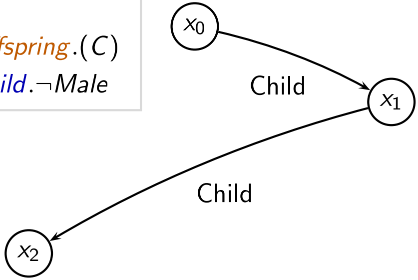
$\sqcap$-rule:
- $\exists Child.Male$
- $\exists Child.\neg Male$
- $Blond$

Example II

$C = \exists Child.Male \sqcap \exists Child.\neg Male \sqcap Blond$
$D = \forall Offspring.(C) \sqcap \exists Child.\neg Male$

D

$\sqcap$-rule: $\forall Offspring.(C)$, $\exists Child.\neg Male$



$\emptyset$

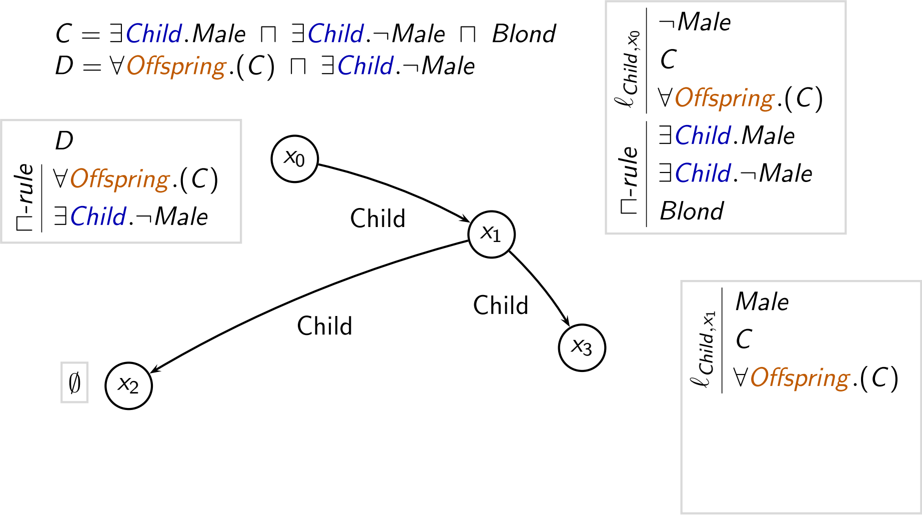
ℓ_{Child,x_0}: $\neg Male$, C, $\forall Offspring.(C)$

$\sqcap$-rule: $\exists Child.Male$, $\exists Child.\neg Male$, $Blond$

ℓ_{Child,x_1}: $Male$, C, $\forall Offspring.(C)$

Example II

$$C = \exists Child.Male \sqcap \exists Child.\neg Male \sqcap Blond$$
$$D = \forall Offspring.(C) \sqcap \exists Child.\neg Male$$

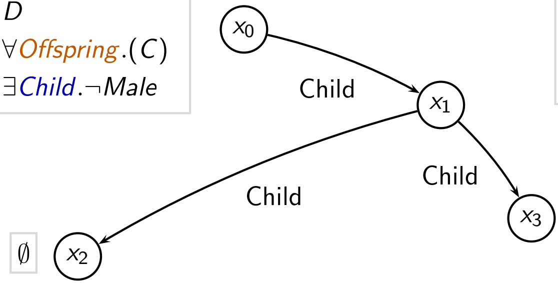


Example II

$C = \exists Child.Male \sqcap \exists Child.\neg Male \sqcap Blond$
$D = \forall Offspring.(C) \sqcap \exists Child.\neg Male$

x_0:
- D
- $\sqcap$-rule: $\forall Offspring.(C)$, $\exists Child.\neg Male$

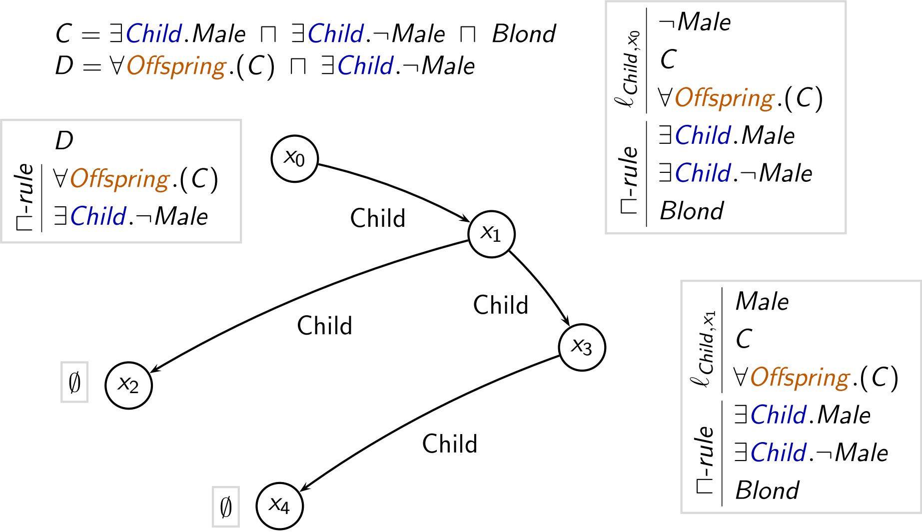
x_1:
- ℓ_{Child,x_0}: $\neg Male$, C, $\forall Offspring.(C)$
- $\sqcap$-rule: $\exists Child.Male$, $\exists Child.\neg Male$, $Blond$



x_3:
- ℓ_{Child,x_1}: $Male$, C, $\forall Offspring.(C)$
- $\sqcap$-rule: $\exists Child.Male$, $\exists Child.\neg Male$, $Blond$

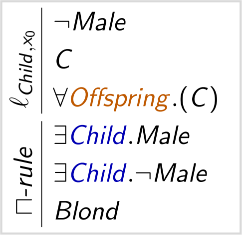
Example II

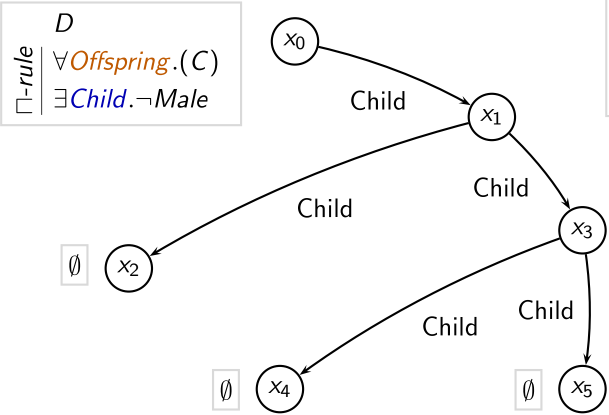
$$C = \exists Child.Male \sqcap \exists Child.\neg Male \sqcap Blond$$
$$D = \forall Offspring.(C) \sqcap \exists Child.\neg Male$$

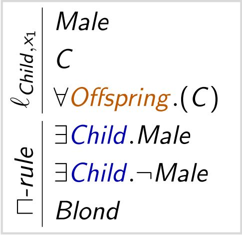


Example II

$C = \exists Child.Male \sqcap \exists Child.\neg Male \sqcap Blond$
$D = \forall Offspring.(C) \sqcap \exists Child.\neg Male$

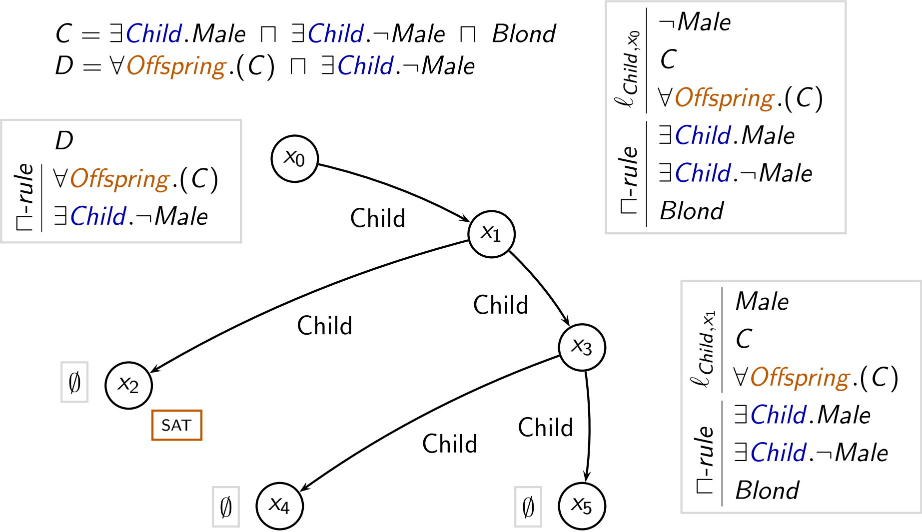






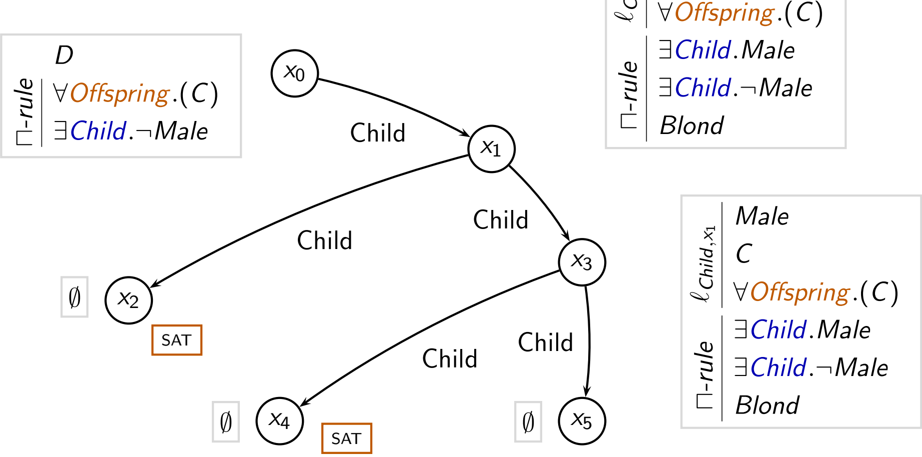
Example II

$C = \exists Child.Male \sqcap \exists Child.\neg Male \sqcap Blond$
$D = \forall Offspring.(C) \sqcap \exists Child.\neg Male$



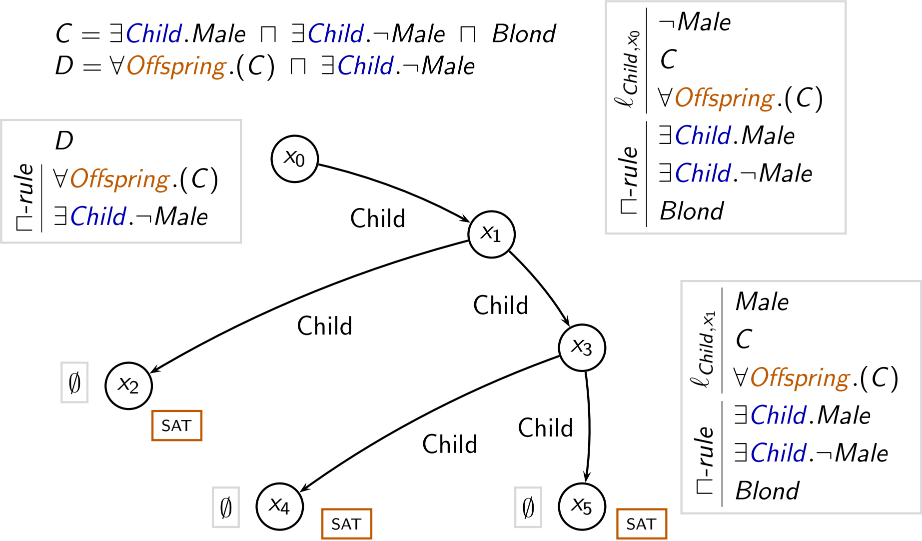
Example II

$$C = \exists Child.Male \sqcap \exists Child.\neg Male \sqcap Blond$$
$$D = \forall Offspring.(C) \sqcap \exists Child.\neg Male$$



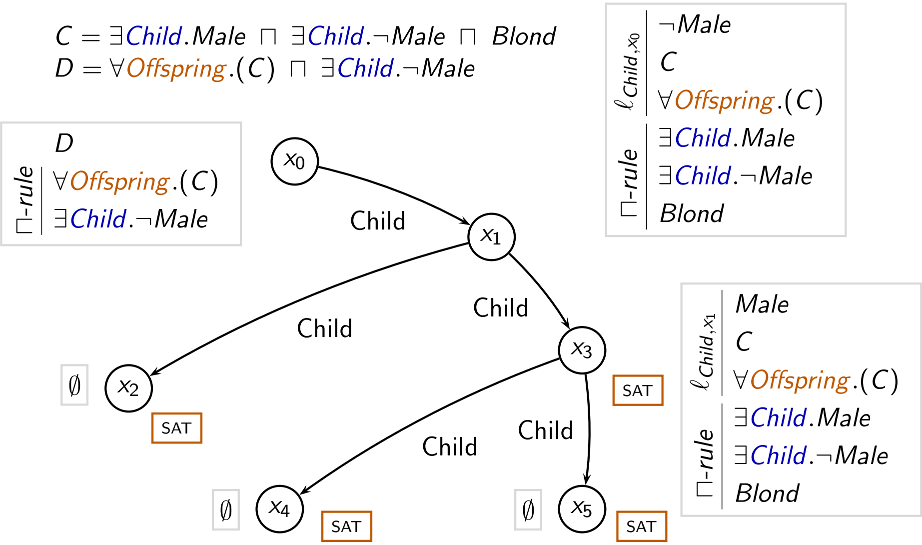
Example II

$C = \exists Child.Male \sqcap \exists Child.\neg Male \sqcap Blond$
$D = \forall Offspring.(C) \sqcap \exists Child.\neg Male$



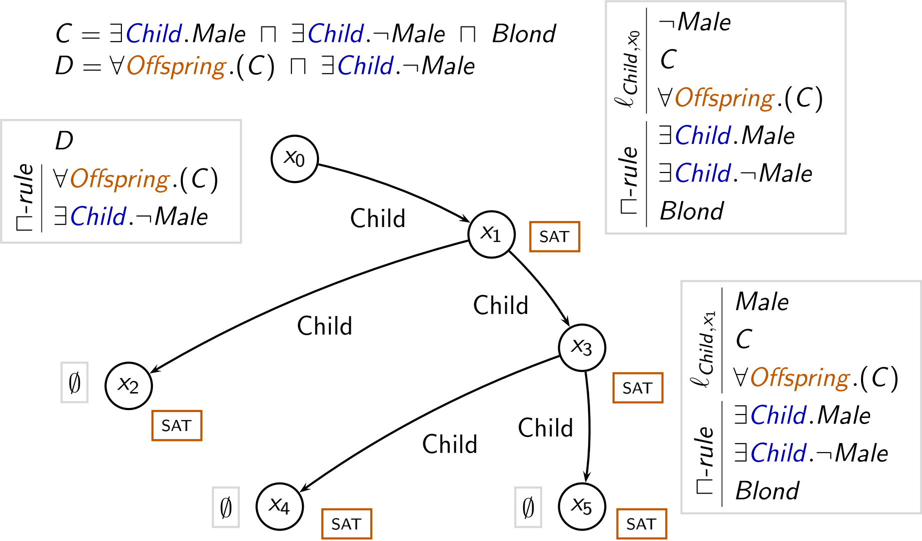
Example II

$C = \exists Child.Male \sqcap \exists Child.\neg Male \sqcap Blond$
$D = \forall Offspring.(C) \sqcap \exists Child.\neg Male$



Example II

$$C = \exists Child.Male \sqcap \exists Child.\neg Male \sqcap Blond$$
$$D = \forall Offspring.(C) \sqcap \exists Child.\neg Male$$

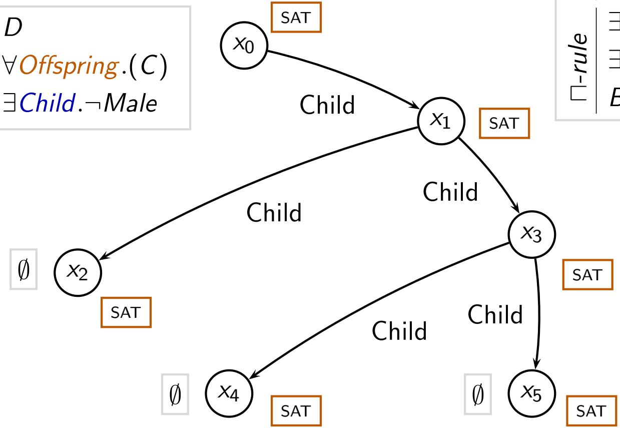


Example II

$C = \exists Child.Male \sqcap \exists Child.\neg Male \sqcap Blond$

$D = \forall Offspring.(C) \sqcap \exists Child.\neg Male$

	D
$\sqcap$-rule	$\forall Offspring.(C)$
	$\exists Child.\neg Male$



ℓ_{Child,x_0}	$\neg Male$
	C
	$\forall Offspring.(C)$
$\sqcap$-rule	$\exists Child.Male$
	$\exists Child.\neg Male$
	$Blond$

ℓ_{Child,x_1}	$Male$
	C
	$\forall Offspring.(C)$
$\sqcap$-rule	$\exists Child.Male$
	$\exists Child.\neg Male$
	$Blond$

Tableau: Some Nice Properties

Tableau construction terminates

- since $(\forall x).|\mathcal{L}(x)| < m$, rules cannot be applied infinitely (blocking done by rule 3.2)
- therefore tableau algorithm is decidable

Tableau algorithm is sound and complete (for $\mathcal{ALCH}_{\mathcal{R}+}$)

- sound since tableau construction adheres to constraints (slides 13 & 14)
- algorithm will explore all possible tableau (models)
- algorithm returns SAT for D iff a tableau exists for D

$\mathcal{ALCH}_{\mathcal{R}+}$ extended: $\mathcal{ALCH}f_{\mathcal{R}+}$

Additional support for functional roles (attributes)

New axiom

- $A \in \mathbf{F}$
- $\mathbf{F}$ is the set of attributes

New condition for interpretations $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$

- for all $A \in \mathbf{F}$, $A^{\mathcal{I}}$ is a partial function $A^{\mathcal{I}} : \mathit{dom}\, A^{\mathcal{I}} \rightarrow \Delta^{\mathcal{I}}$

$\mathcal{ALCH}f_{\mathcal{R}+}$- Tableau I

- *constrained* relation is transitive closure of *directly constrained*
- B is *directly constrained* by A if

  $(\exists A.C \in \mathcal{L}(x) \wedge A \sqsubseteq B) \vee (\exists B.C \in \mathcal{L}(x) \wedge B \sqsubseteq A)$

$\mathcal{ALCH}f_{\mathcal{R}+}$- Tableau II

Addition to rule 3 ($\exists R.C$)

- for each $\exists R.C \in \mathcal{L}(x)$ with $R \notin \mathbf{F}$ use previous algorithm
- for each $\exists A.C \in \mathcal{L}(x)$ with $A \in \mathbf{F}$ do

  if for some **A**-successor y of x, $A \in \mathbf{A}$ then do nothing; otherwise:

$\mathcal{ALCH}f_{\mathcal{R}+}$- Tableau III

1. $\mathbf{A} = \mathbf{A}_{Ax} = \{B \in \mathbf{F} \mid B \text{ is constrained by } A \text{ in } x\}$
2. $\ell_{Ax} = \bigcup_{B \in \mathbf{A}}($

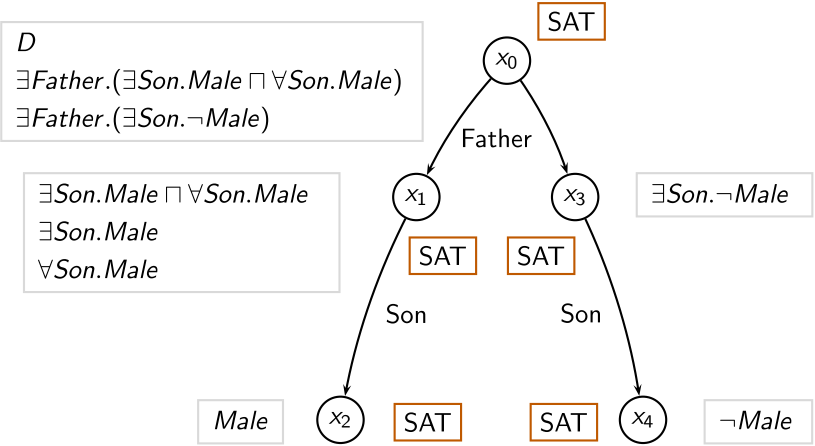
$$\{C \mid \exists B.C \in \mathcal{L}(x)\}$$
$$\cup \{C \mid \forall S.D \in \mathcal{L}(x), B \sqsubseteq S\}$$
$$\cup \{\forall S.D \mid \forall P.D \in \mathcal{L}(x), S \in \mathbf{R}_+, S \sqsubseteq P, R \sqsubseteq S\})$$

3. if $\ell_{Ax} \subseteq \mathcal{L}(w)$, w is ancestor of x, then create a **A**-successor y of x with $\mathcal{L}(y) = \emptyset$
4. otherwise, create a **A**-successor y of x with $\mathcal{L}(y) = \ell_{Ax}$

Non-Functional vs. Functional Roles

$D = \exists Father.(\exists Son.Male \sqcap \forall Son.Male) \sqcap \exists Father.(\exists Son.\neg Male)$

assume $\{Father, Son\} \not\subseteq$ **F**

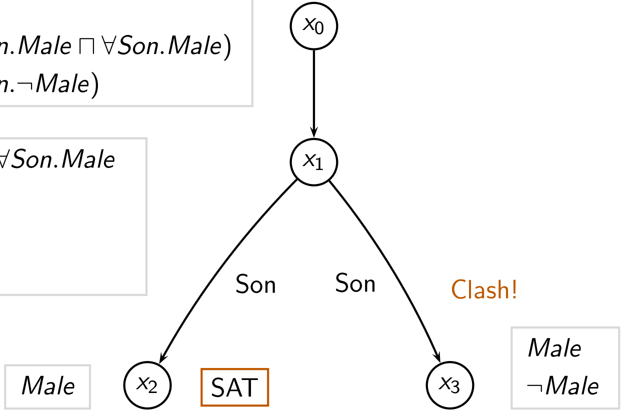


Non-Functional vs. Functional Roles

$D = \exists Father.(\exists Son.Male \sqcap \forall Son.Male) \sqcap \exists Father.(\exists Son.\neg Male)$
assume $Father \in \mathbf{F}$, $Son \notin \mathbf{F}$

D
$\exists Father.(\exists Son.Male \sqcap \forall Son.Male)$
$\exists Father.(\exists Son.\neg Male)$

$\exists Son.Male \sqcap \forall Son.Male$
$\exists Son.\neg Male$
$\exists Son.Male$
$\forall Son.Male$



Concept Axioms in $\mathcal{T}$

- general terminologies $\mathcal{T}$ contain concept axioms ($C \sqsubseteq D$ or $C \doteq D$)
- these axioms might be cyclic

Concept Axioms: Solution I

Acyclic concept definitions D

- unfold D until solely atomic primitives remain

Primitive cyclic concept definitions

- lazy unfolding

Non-primitive cyclic concept definitions

- lazy unfolding $+$ transform axioms into primitive definitions

Concept Axioms: Solution II

General concept inclusion axioms (GCI)

- *internalisation:* say, we have the GCIs

$$\{A_1 \sqsubseteq B1, \ldots, A_n \sqsubseteq B_n\} \subseteq \mathcal{T}$$

Test the satifiablitiy of

$$D \sqcap \mathcal{M} \sqcap \forall U.\mathcal{M}$$

where

$$\mathcal{M} \doteq (B_1 \sqcup \neg A_1) \sqcap \cdots \sqcap (B_n \sqcup \neg A_n)$$

$U \in \mathbf{R}_+$ and U subsumes all other rules in $\mathcal{T}$

FaCT (Summary)

- represents terminological classifier (TBox)
- reasons about concept, role and attribute descriptions
- outperforms significantly all other tested systems (Crack, KSAT, Kris)
- works as well with the propositional modal logics $\mathbf{K}_{(m)}$, $\mathbf{KT}_{(m)}$, $\mathbf{K4}_{(m)}$ and $\mathbf{S4}_{(m)}$
- incorporates a range of optimizations