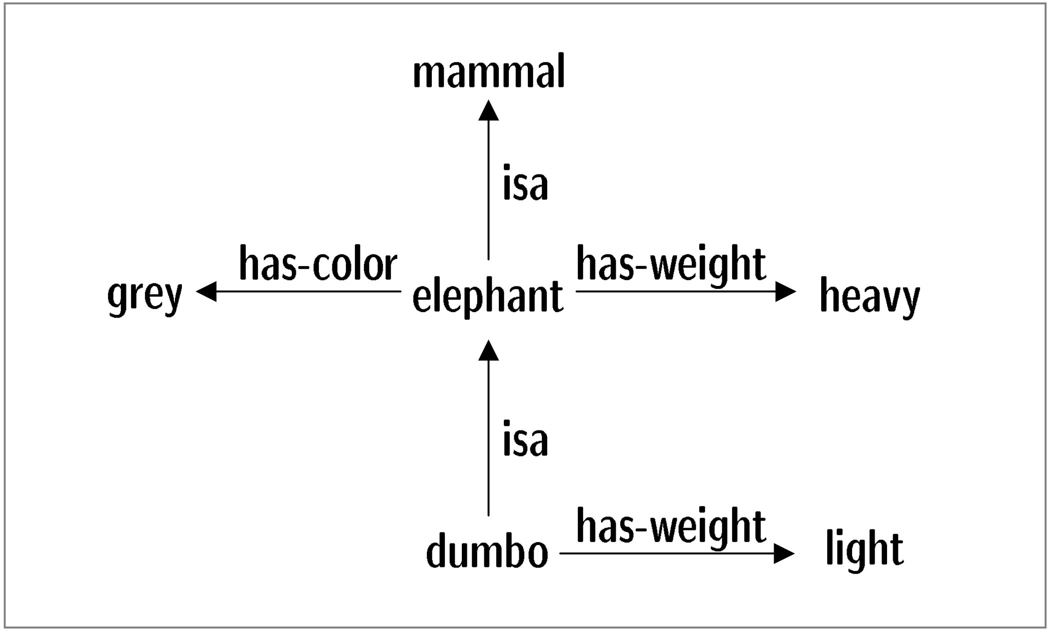
Description Logics

Carsten Lutz and Ulrike Sattler

TU Dresden, Germany

Origins of Description Logics

- Representation of conceptual knowledge is subfield of Artificial Intelligence
- Early days of AI: KR through obscure pictures (semantic networks)



Problems: missing semantics (reasoning!), complex pictures

Remedy: Use a logical formalism for KR rather than pictures

Representing Conceptual Knowledge with DLs

Defining elephants using DLs:

- $\mathsf{Mammal} \sqcap \exists \mathsf{bodypart}.\mathsf{Trunk} \sqcap \forall \mathsf{color}.\mathsf{Grey}$
- $\mathsf{Mammal} \sqcap \exists \mathsf{bodypart}.\mathsf{Trunk} \sqcap (= 1\ \mathsf{color}) \sqcap \forall \mathsf{color}.\mathsf{Grey}$
  $\sqcap\ (= 1\ \mathsf{weight}) \sqcap ((\forall \mathsf{weight}.\mathsf{Heavy}) \sqcup (\mathsf{Dumbo} \sqcap \forall \mathsf{weight}.\mathsf{Light}))$

A concept language does not solve all problems...

Do these concepts describe necessary or sufficient conditions?

How can we describe specific elephants such as Dumbo?

How do I avoid losing track when constructing large knowledge bases?

Modern Description Logics

Foci of “modern” DL research:

1. Identify interesting Description Logics and study their properties

   Main topics: decidability, computational complexity, expressivity

2. Implement Description Logics in highly-optimized reasoning systems

   Fast and powerful systems available: e.g. FaCT and RACER

3. Apply Description Logics in several application areas

   - Reasoning about Entity Relationship (ER) diagrams
   - Representation of Ontologies for the Semantic Web

The Description Logic $\mathcal{ALC}$: Syntax

$\mathcal{ALC}$ is the smallest propositionally closed Description Logic.

Atomic types: concept names $A, B, \ldots$ (unary predicates)

role names $R, S, \ldots$ (binary predicates)

Constructors:

- $\neg C$ (negation)
- $C \sqcap D$ (conjunction)
- $C \sqcup D$ (disjunction)
- $\exists R.C$ (existential restriction)
- $\forall R.C$ (universal restriction)

For example: $\neg(A \sqcup \exists R.(\forall S.B \sqcap \neg A))$

Mammal ⊓ ∃bodypart.trunk ⊓ ∀color.Grey

Semantics of $\mathcal{ALC}$

Semantics based on interpretations $(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\cdot^{\mathcal{I}}$ maps

- each concept name A to a subset $A^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$.
- each role name R to a binary relation $R^{\mathcal{I}}$ over $\Delta^{\mathcal{I}}$.

Semantics of complex concepts:

$$(\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \qquad (C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}} \qquad (C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$$

$$(\exists R.C)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid (d, e) \in R^{\mathcal{I}} \text{ and } e \in C^{\mathcal{I}}\}$$

$$(\forall R.C)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid (d, e) \in R^{\mathcal{I}} \text{ implies } e \in C^{\mathcal{I}}\}$$

An interpretation $\mathcal{I}$ is a model for a concept C if $C^{\mathcal{I}} \neq \emptyset$.

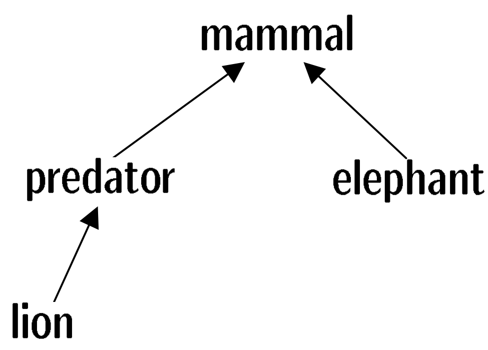
Reasoning Tasks

Two main reasoning tasks:

1. Concept satisfiability — does there exist a model of C?
2. Concept subsumption — does $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ hold for all $\mathcal{I}$? (written $C \sqsubseteq D$)

Why subsumption?

$\Longrightarrow$ Can be used to compute a concept hierarchy:



In propositionally closed DLs, these can be mutually reduced to one another.

Expressive Power vs. Computational Complexity

In many cases, the expressive power of $\mathcal{ALC}$ does not suffice:

- an elephant has precisely four legs
- every elephant has a bodypart which is a trunk and every trunk is a bodypart of an elephant

Many extensions of $\mathcal{ALC}$ have been developed, for example:

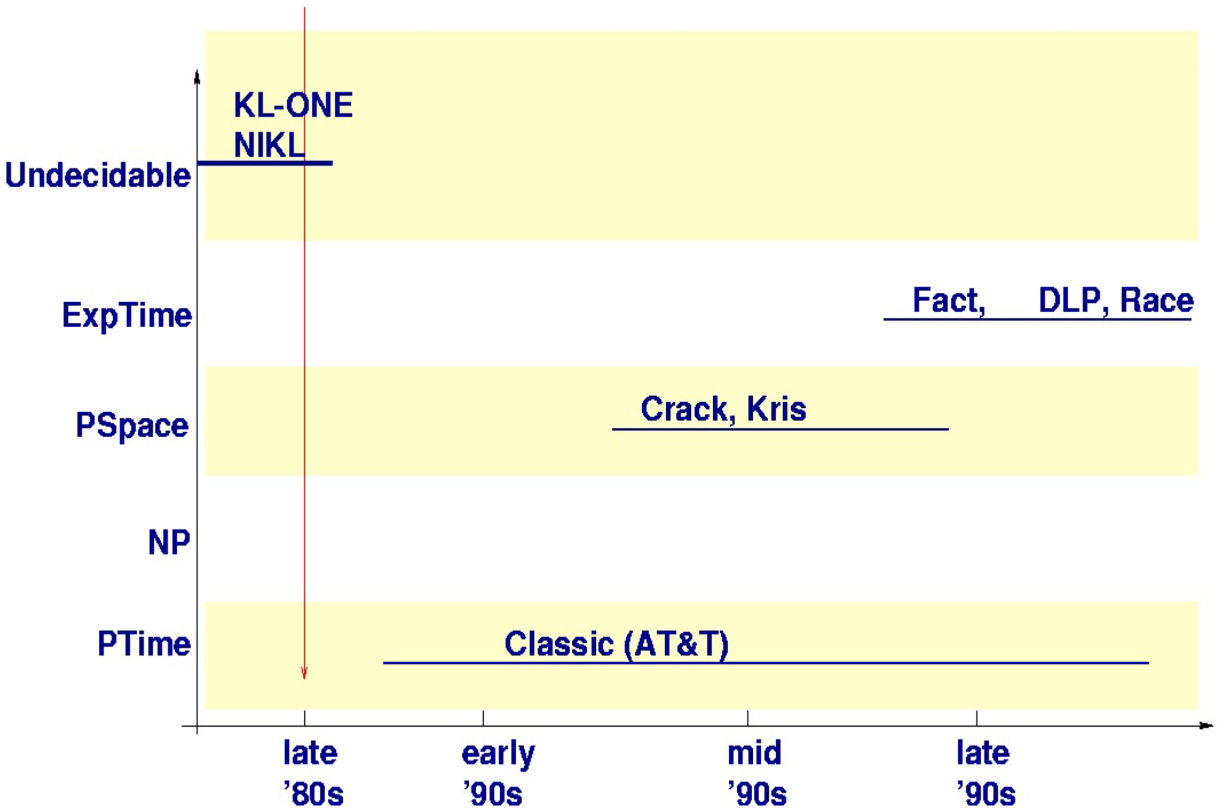
- qualified number restrictions $(\leq n\ R\ C)$ and $(\geq n\ R\ C)$
- inverse roles R^- to be used in existential and universal restriction

But: Increasing expressivity also increases computational complexity

$\Longrightarrow$ **!! tradeoff between expressivity and computational complexity !!**

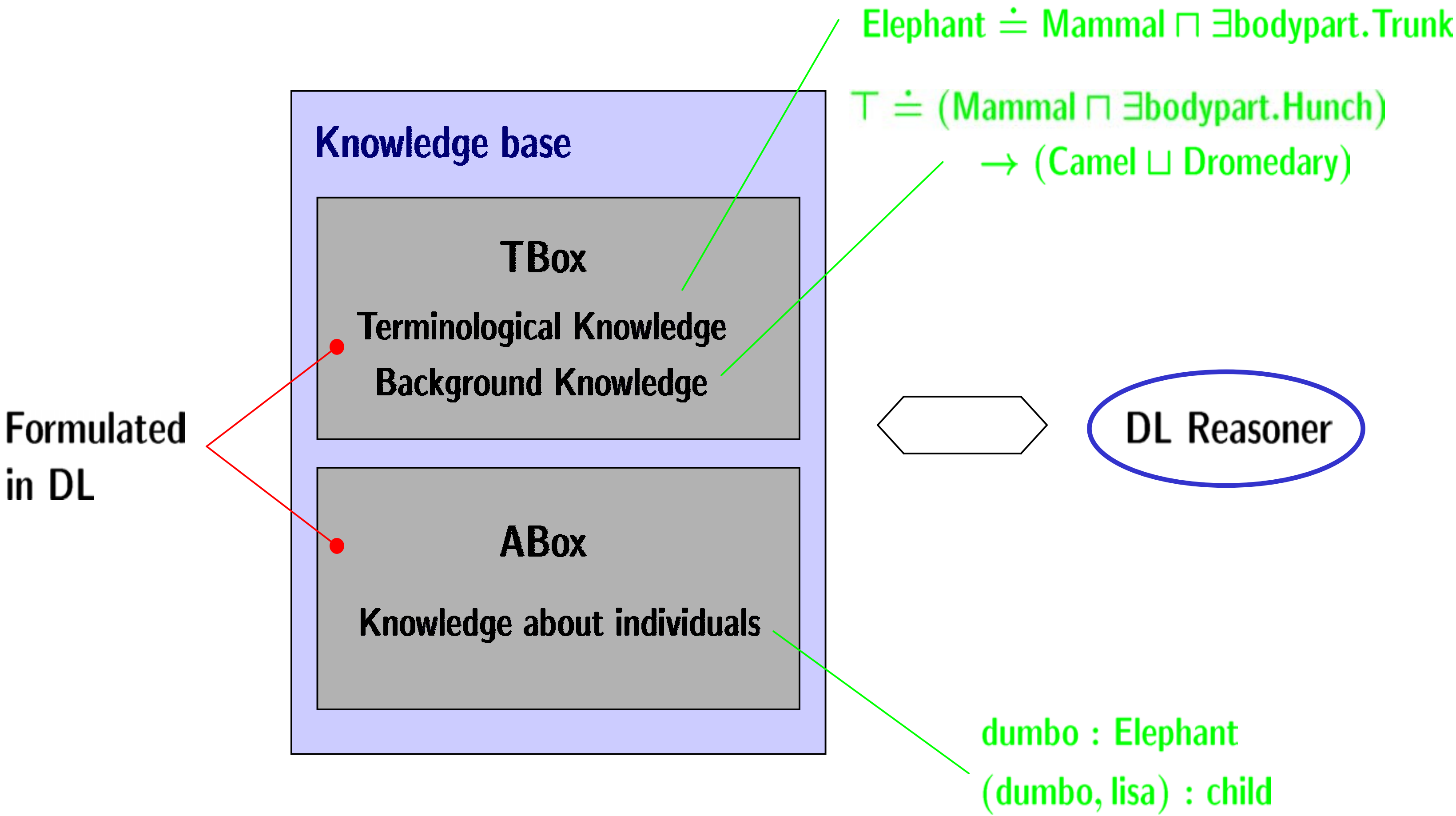
Development of DL Systems

Description Logics should be decidable. But what complexity is "ok"?



TU Dresden

DLs are more than a Concept Language



General TBoxes

There exist several kinds of TBoxes.

General TBox: finite set of concept equations $C \doteq D$

An interpretation $\mathcal{I}$ is a model of a TBox $\mathcal{T}$ if

$$C^{\mathcal{I}} = D^{\mathcal{I}} \text{ for all } C \doteq D \in \mathcal{T}.$$

$$\{\top \doteq (\text{Mammal} \sqcap \exists \text{bodypart.Hunch}) \rightarrow (\text{Camel} \sqcup \text{Dromedary})\}$$

Reasoning tasks with TBoxes:

1. Concept satisfiability w.r.t. TBoxes

   Given C and $\mathcal{T}$, does there exist a common model of C and $\mathcal{T}$?

2. Concept subsumption w.r.t. TBoxes

   Given C,D, and $\mathcal{T}$, does $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ hold in all models of $\mathcal{T}$?

   (written $C \sqsubseteq_{\mathcal{T}} D$)

TU Dresden

Acyclic TBoxes

Concept definition: expression $A \doteq C$ with A concept name and C concept

$$\text{Elephant} \doteq \text{Mammal} \sqcap \exists \text{bodypart.Trunk}$$

A finite set $\mathcal{T}$ of concept definitions is an acyclic TBox if

a) the left-hand sides of concept definitions in $\mathcal{T}$ are unique

b) it contains no "cycles"

not an acyclic TBox:

$$\{A_0 \doteq A_1 \sqcap C \quad A_1 \doteq \exists R.A_2 \quad A_2 \doteq A_0\}$$

Acyclic TBoxes can be conceived as macro definitions.

ABoxes

Fix a set of individual names.

An ABox is a finite set of assertions

$$a : C \qquad (a \text{ individual name, } C \text{ concept})$$

$$(a, b) : R \qquad (a, b \text{ individual names, } R \text{ role name})$$

{dumbo : Elephant , (dumbo, lisa) : child}

Interpretations $\mathcal{I}$ map each individual name a to an element of $\Delta^{\mathcal{I}}$.

$\mathcal{I}$ satisfies an assertion

$$a : C \quad \text{iff} \quad a^{\mathcal{I}} \in C^{\mathcal{I}}$$

$$(a, b) : R \quad \text{iff} \quad (a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$$

$\mathcal{I}$ is a model for an ABox $\mathcal{A}$ if $\mathcal{I}$ satisfies all assertions in $\mathcal{A}$.

Reasoning with ABoxes

Reasoning tasks with ABoxes:

1. **ABox consistency**

   Given an ABox $\mathcal{A}$ and a TBox $\mathcal{T}$, do they have a common model?

2. **Instance checking**

   Given an ABox $\mathcal{A}$, a TBox $\mathcal{T}$, an individual name a, and a concept C does $a^{\mathcal{I}} \in C^{\mathcal{I}}$ hold in all models of $\mathcal{A}$ and $\mathcal{T}$?

   (written $\mathcal{A}, \mathcal{T} \models a : C$)

Instance checking can be reduced to ABox consistency.

Concept satisfiability can be reduced to ABox consistency.

Description Logics and First-order Logic

concept names A	$\Longleftrightarrow$	unary predicates P_A
role names R	$\Longleftrightarrow$	binary predicates P_R
concepts	$\Longleftrightarrow$	formulas with one free variable

$$
\begin{aligned}
\varphi^x(A) &= P_A(x) \\
\varphi^x(\neg C) &= \neg\varphi^x(C) \\
\varphi^x(C \sqcap D) &= \varphi^x(C) \land \varphi^x(D) \\
\varphi^x(C \sqcup D) &= \varphi^x(C) \lor \varphi^x(D) \\
\varphi^x(\exists R.C) &= \exists y.P_R(x,y) \land \varphi^y(C) \\
\varphi^x(\forall R.C) &= \forall y.P_R(x,y) \to \varphi^y(C)
\end{aligned}
$$

φ^y symmetric with x and y exchanged

Note:
- two variables suffices (no "$=$", no constants, no function symbols)
- formulas obtained by translation have "guarded" structure
- not all DLs are purely first-order (transitive closure, etc.)

TBoxes:

Let C be a concept and $\mathcal{T}$ a (general or acyclic) TBox.

$$\varphi(C,\mathcal{T}) = \varphi^x(C) \wedge \forall x. \bigwedge_{D \doteq E \in \mathcal{T}} \varphi^x(D) \leftrightarrow \varphi^x(E)$$

ABoxes:

individual names a $\Longleftrightarrow$ constants c_a

$$\begin{aligned} \varphi(a : C) &= \varphi^x(C)[c_a] \\ \varphi((a,b) : R) &= P_R(c_a, c_b) \\ \varphi(\mathcal{A}) &= \bigwedge_{\beta \in \mathcal{A}} \varphi(\beta) \end{aligned}$$

Obvious translation:

concept names	$\Longleftrightarrow$	propositional variables
role names	$\Longleftrightarrow$	modal parameters
concepts $\exists R.C$	$\Longleftrightarrow$	formulas $\Diamond\psi$
concepts $\forall R.C$	$\Longleftrightarrow$	formulas $\Box\psi$

Notes:

- Interpretations can be viewed as Kripke structures
- $\mathcal{ALC}$ is a notational variant of modal K_ω
- TBoxes related to universal modality: $\Box_u \bigwedge_{D \doteq E \in \mathcal{T}} D \leftrightarrow E$
- ABoxes related to nominals / hybrid modal logic
- Extensions of $\mathcal{ALC}$ are related to graded modal logic, PDL, etc.

Overview of the Course

- Introduction and Tableau Algorithm for $\mathcal{ALCN}$
- Tableau algorithms for expressive Description Logics
- Automata-based decision procedures for expressive Description Logics
- Computational complexity
- Applications, System demonstration, other topics of DL research

The Description Logic $\mathcal{ALCN}$

$\mathcal{ALCN}$: $\mathcal{ALC}$

+ unqualified number restrictions $(\leqslant n\ R)$ and $(\geqslant n\ R)$

Semantics:

$$(\leqslant n\ R)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \#\{(d,e) \mid (d,e) \in R^{\mathcal{I}}\} \leq n\}$$
$$(\geqslant n\ R)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \#\{(d,e) \mid (d,e) \in R^{\mathcal{I}}\} \geq n\}$$

Mother of many children: Female $\sqcap$ $\forall$has-children.Human $\sqcap$ $(\geqslant 4$ has-children)

Chinese mother: Female $\sqcap$ (($\leqslant 1$ has-children) $\sqcup$ $\exists$pays-tax.Expensive)

Note:

Less expressive than qualified number restrictions $(\leqslant n\ R\ C)$ and $(\geqslant n\ R\ C)$

$\Longrightarrow$ decidability/complexity of $\mathcal{ALCN}$-concept satisfiability (without TBoxes)

Tableau Algorithms

Appropriate tool: Tableau Algorithms

- Frequently used to prove decidability/complexity bounds of DLs
- All state-of-the-art DL reasoners are based on tableau algorithms

Strategy:

- Try to construct a model for the input concept C_0
- Represent models by completion trees
- To decide satisfiability of C_0, start with initial completion tree T_{C_0}
- Repeatedly apply completion rules and check for obvious contradictions
- Return "satisfiable" iff a complete and contradiction-free completion tree was found

Negation Normal Form

A concept C is in negation normal form (NNF) if

negation occurs only in front of concept names.

Transformation rules:

$$\neg\neg C \rightsquigarrow C$$

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

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

$$\neg(\exists R.C) \rightsquigarrow \forall R.\neg C$$

$$\neg(\forall R.C) \rightsquigarrow \exists R.\neg C$$

$$\neg(\leqslant n\ R) \rightsquigarrow (\geqslant n+1\ R)$$

$$\neg(\geqslant 0\ R) \rightsquigarrow \bot$$

$$\neg(\geqslant n\ R) \rightsquigarrow (\leqslant n-1\ R) \quad \text{if } n > 0$$

TU Dresden

Completion Tree

Completion tree:

Finite tree $T = (V, E, \mathcal{L})$ where $\mathcal{L}$ labels

- each node $x \in V$ with a set $\mathcal{L}(x) \subseteq \mathsf{sub}(C_0)$
- each edge $(x, y) \in E$ with a role $\mathcal{L}(x, y)$ occurring in C_0.

Initial completion tree for concept C_0: $(\{x_0\}, \emptyset, \mathcal{L})$ where $\mathcal{L}(x_0) = \{C_0\}$

Apply completion rules until

- the completion tree is **complete**.

or - there exists a node $x \in V$ such that

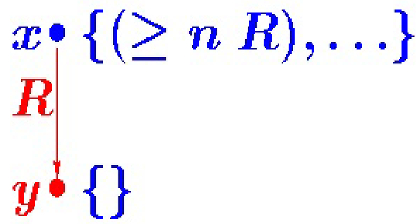
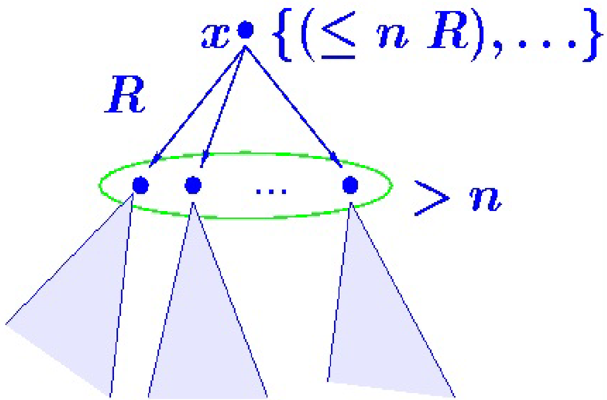
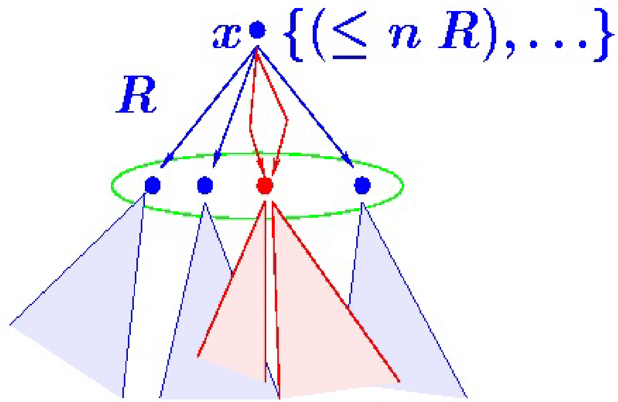
Clash {
1. $\{A, \neg A\} \subseteq \mathcal{L}(x)$ for some concept name A

or 2. $\{(\leqslant n\ R), (\geqslant m\ R)\} \subseteq \mathcal{L}(x)$ with $m > n$.

Completion Rules I

$x \bullet \{C_1 \sqcap C_2, \ldots\}$	$\rightarrow_{\sqcap}$	$x \bullet \{C_1 \sqcap C_2, C_1, C_2, \ldots\}$
$x \bullet \{C_1 \sqcup C_2, \ldots\}$	$\rightarrow_{\sqcup}$	$x \bullet \{C_1 \sqcup C_2, C, \ldots\}$ for $C \in \{C_1, C_2\}$
$x \bullet \{\exists R.C, \ldots\}$	$\rightarrow_{\exists}$	$x \bullet \{\exists R.C, \ldots\}$ R ↓ $y \bullet \{C\}$
$x \bullet \{\forall R.C, \ldots\}$ R ↓ $y \bullet \{\ldots\}$	$\rightarrow_{\forall}$	$x \bullet \{\forall R.C, \ldots\}$ R ↓ $y \bullet \{\ldots, C\}$

Completion Rules II

$x \bullet \{(\geq n\ R), \ldots\}$ x has no R-succ.	$\rightarrow_{\geq}$	
	$\rightarrow_{\leq}$	 merge two R-succs.

Example: blackboard

Correctness of the Algorithm

Lemma

1. The algorithm terminates on any input
2. If the algorithm returns "satisfiable", then the input concept has a model.
3. If the input concept has a model, then the algorithm returns "satisfiable".

Corollary

1. $\mathcal{ALCN}$-concept satisfiability and subsumption are decidable
2. $\mathcal{ALCN}$ has the tree model property
3. $\mathcal{ALCN}$ has the finite model property

Termination

Role depth of concepts:

$$d(A) = d(\leq n\ R) = 0 \qquad d(\geq n\ R) = 1$$

$$d(\neg C) = d(C)$$

$$d(C \sqcap D) = d(C \sqcup D) = \max\{d(C), d(D)\}$$

$$d(\exists R.C) = d(\forall R.C) = d(C) + 1$$

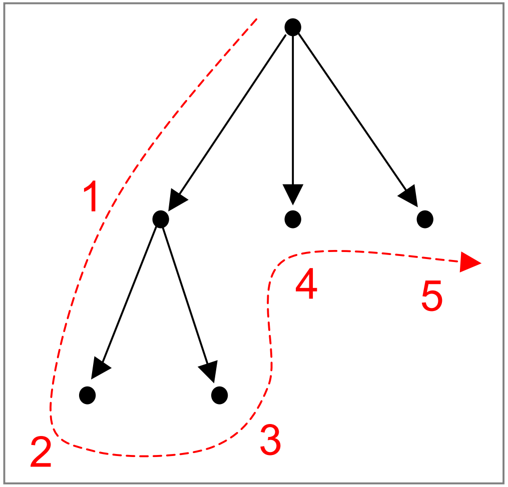
The algorithm terminates since:

1. depth of the completion tree bounded by $d(C_0)$.
2. for each node, at most $\#\mathrm{sub}(C_0)$ successors are generated
3. each node label contains at most $\#\mathrm{sub}(C_0)$ concepts
4. concepts are never deleted from node labels
5. nodes may be deleted (via identification), but 1 and 2 is independent from this

A PSPACE upper bound for $\mathcal{ALCN}$

Modify EXPSPACE tableau algorithm:

1. Construct completion tree in a depth-first manner:



2. Keep only paths of the tree in memory!

Yields a PSPACE algorithm:
- paths are of length polynomial in $|C_0|$
- the outdegree is polynomial in $|C_0|$.

PSPACE lower bound will be proved later!

Reasoning with Acyclic TBoxes

Naive approach: unfolding

$\Longrightarrow$ reduce satisfiability w.r.t. TBoxes to satisfiability without TBoxes

Let C_0 be concept, $\mathcal{T}$ acyclic TBox

1. replace concept names on right hand sides of definitions $A \doteq C$ with their defining concept
2. replace each concept name in C_0 defined in $\mathcal{T}$ with its definition.

Terminates due to acyclicity!

But: exponential blowup in the worst case

$$\begin{aligned} A_0 &\doteq \forall R.A_1 \sqcap \forall S.A_1 \\ A_1 &\doteq \forall R.A_2 \sqcap \forall S.A_2 \\ &\vdots \\ A_{k-1} &\doteq \forall R.A_k \sqcap \forall S.A_k \end{aligned}$$

Unfolding on the Fly

Idea:

Modify existing tableau algorithm to directly deal with acyclic TBoxes

Roadmap:

- convert concept definitions into one of the forms

$$A \doteq \neg X,\ A \doteq B_1 \sqcap B_2,\ A \doteq B_1 \sqcup B_2,\ A \doteq \forall R.B,\ A \doteq \exists R.B$$

with A, B, B_1, B_2 concept names and X primitive concept name

- restrict node labels to concept names

- make on the fly TBox lookups for rule application

Result: Satisfiability of $\mathcal{ALC}$-concepts w.r.t. acyclic TBoxes is PSPACE-complete.

That's it

More on tableau algorithms tomorrow!