# Reasoning Procedures I

## Technical detail: the tableau algorithm

- works on a tree (semantics through viewing tree as an ABox):

  **nodes** represent elements of $\Delta^{\mathcal{I}}$, labelled with sub-concepts of $C_0$

  **edges** represent role-successorships between elements of $\Delta^{\mathcal{I}}$

- works on concepts in **negation normal form**: push negation inside using de Morgan' laws and

$$
\begin{array}{rclrcl}
\neg(\exists R.C) & \rightsquigarrow & \forall R.\neg C & \neg(\forall R.C) & \rightsquigarrow & \exists R.\neg C \\
\neg(\leq n\ R) & \rightsquigarrow & (\geq (n+1)R) & \neg(\geq n\ R) & \rightsquigarrow & (\leq (n-1)R) \quad (n \geq 0) \\
 & & & \neg(\geq 0\ R) & \rightsquigarrow & A \sqcap \neg A
\end{array}
$$

- is initialised with a tree consisting of a single (root) node $x_0$ with $\mathfrak{L}(x_0) = \{C_0\}$:

$$x_0 \bullet \{C_0\}$$

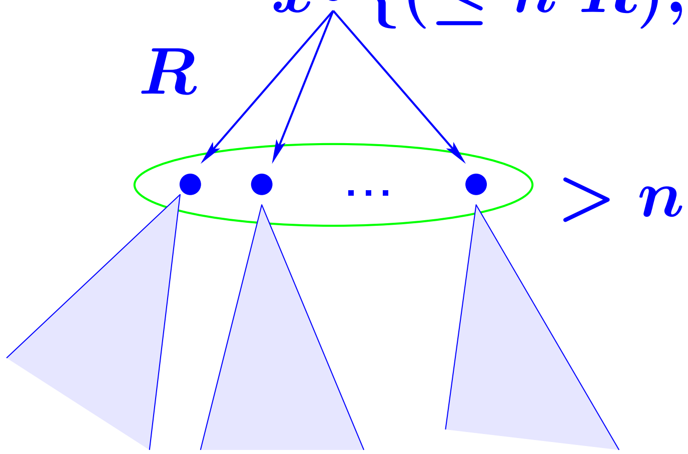
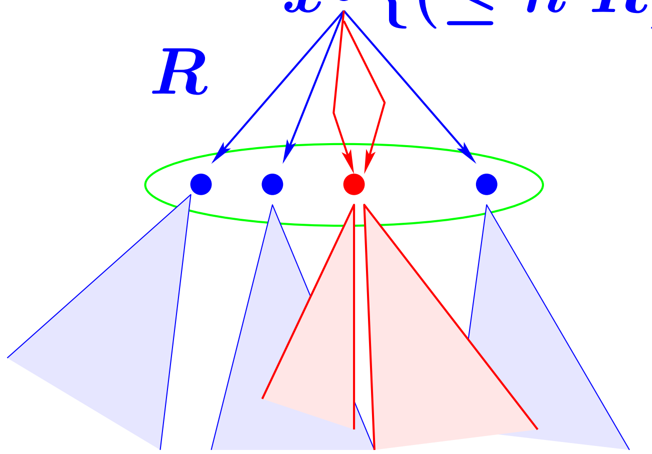
- a tree $\mathrm{T}$ contains a **clash** if, for a node $x$ in $\mathrm{T}$,

$$
\begin{array}{rcl}
\{A, \neg A\} & \subseteq & \mathfrak{L}(x) \text{ or} \\
\{(\geq m\ R), (\leq n\ R)\} & \subseteq & \mathfrak{L}(x) \text{ for } n < m
\end{array}
$$

# Reasoning Procedures: $\mathcal{ALC}$ Tableau Rules

| | | |
|---|---|---|
| $x \bullet \{C_1 \sqcap C_2, \ldots\}$ | $\rightarrow_{\sqcap}$ | $x \bullet \{C_1 \sqcap C_2, C_1, C_2, \ldots\}$ |
| $x \bullet \{C_1 \sqcup C_2, \ldots\}$ | $\rightarrow_{\sqcup}$ | $x \bullet \{C_1 \sqcap C_2, C, \ldots\}$<br>for $C \in \{C_1, C_2\}$ |
| $x \bullet \{\exists R.C, \ldots\}$ | $\rightarrow_{\exists}$ | $x \bullet \{\exists R.C, \ldots\}$<br>$R \downarrow$<br>$y \bullet \{C\}$ |
| $x \bullet \{\forall R.C, \ldots\}$<br>$R \downarrow$<br>$y \bullet \{\ldots\}$ | $\rightarrow_{\forall}$ | $x \bullet \{\forall R.C, \ldots\}$<br>$R \downarrow$<br>$y \bullet \{C, \ldots\}$ |

# Reasoning Procedures: $\mathcal{N}$ Tableau Rules

| | | |
|---|---|---|
| $x \bullet \{(\geq n\ R), \ldots\}$<br>$x$ has no $R$-succ. | $\rightarrow_{\geq}$ | $x \bullet \{(\geq n\ R), \ldots\}$<br>$\Big\downarrow R$<br>$y \bullet \{\}$ |
| $x \bullet \{(\leq n\ R), \ldots\}$<br> | $\rightarrow_{\leq}$ | $x \bullet \{(\leq n\ R), \ldots\}$<br><br>merge two $R$-succs. |

**Lemma**

Let $C_0$ be an $\mathcal{ALCN}$ concept and $\mathrm{T}$ obtained by applying the tableau rules to $C_0$. Then

1. the rule application terminates,
2. if $\mathrm{T}$ is consistent and $\longrightarrow$ is applicable to $\mathrm{T}$, then $\longrightarrow$ can be applied such that it yields consistent $\mathrm{T}'$,
3. if $\mathrm{T}$ contains a clash, then $\mathrm{T}$ has no model, and
4. if no more rules apply to $\mathrm{T}$, then $\mathrm{T}$ defines (canonical) model for $C_0$.

**Corollary**

(1) The tableau algorithm is a PSpace decision procedure for consistency (and subsumption) of $\mathcal{ALCN}$ concepts

(2) $\mathcal{ALCN}$ has the tree model property

## Proof of the Lemma

1. (Termination) The algorithm "monotonically" constructs a tree whose

   **depth** is linear in $|C_0|$: quantifier depth decreases from node to succs.

   **breadth** is linear in $|C_0|$ (even if number in NRs are coded binarily)

2. (Local Consistency) Easy to prove (by definition of the semantics) that

   if $\mathcal{I}$ is a model of $\mathrm{T}$, then $\longrightarrow$ can be applied to $\mathrm{T}$ such that

   $\mathcal{I}$ is a model of $\mathrm{T}' := \longrightarrow(\mathrm{T})$

3. Obvious: $\mathrm{T}$ with a clash has no model—recall definition of a clash:

$$\{A, \neg A\} \subseteq \mathfrak{L}(x) \text{ or}$$
$$\{(\geq m\ R), (\leq n\ R)\} \subseteq \mathfrak{L}(x) \text{ for } n < m$$

## Proof of the Lemma (ctd.)

4. (Canonical model) "Complete" tree $\mathrm{T}$ defines a (tree) pre-model $\mathcal{I}$:

| | |
|---|---|
| nodes | correspond to elements of $\Delta^{\mathcal{I}}$ |
| edges | define role-relationship |
| $x \in A^{\mathcal{I}}$ | iff $A \in \mathfrak{L}(x)$ for concept names $A$ |

Check that $C \in \mathfrak{L}(x)$ implies $x \in C^{\mathcal{I}}$—if $C$ is **no number restriction**.

For NRs, if $(\geq n\ R) \in \mathfrak{L}(x)$ and $x$ has less than $n$ $R$-successors,
**copy** some $R$-successors (including sub-trees) to obtain $n$ $R$-successors

$\leadsto$ **canonical tree model for input concept**

**To make the tableau algorithm run in PSpace:**

Recall **Savitch**: PSpace = NPSpace

① observe that branches are independent from each other

② observe that each node (label) requires linear space only

③ recall that paths are of length $\leq |C_0|$
$\rightsquigarrow$ each path can be stored in $\mathcal{O}(|C_0|^2)$

④ construct/search the tree **depth first**

⑤ re-use space from already constructed branches

# Reasoning Procedures: Extensibility

This tableau algorithm can be modified to a PSpace decision procedure for

- ✔ $\mathcal{ALC}$ with **qualifying number restrictions**
  $(\geq n\ R\ C)$ and $(\leq n\ R\ C)$
- ✔ $\mathcal{ALC}$ with **inverse roles** (e.g. `has-child`$^-$)
- ✔ $\mathcal{ALC}$ with **role conjunction**
  $\exists(R \sqcap S).C$ and $\forall(R \sqcap S).C$
- ✔ **TBoxes with acyclic concept definitions** $A \doteq C$:

| | |
|---|---|
| **unfolding** | (macro expansion) is easy, but suboptimal: may yield exponential blow-up |
| **lazy unfolding** | (unfolding on demand) is optimal, consistency in PSpace decidable |

## Reasoning Procedures: Extensibility II

Language extensions that require more elaborate techniques include

- **TBoxes with general axioms** $C_i \sqsubseteq D_i$:
  each node must be labelled with $\neg C_i \sqcup D_i$
  quantifier depth no longer decreases
  $\leadsto$ termination not guaranteed
- **Transitive closure of roles**:
  node labels $(\forall R^*.C)$ yields $C$ in all $R^n$-successor labels
  quantifier depth no longer decreases
  $\leadsto$ termination not guaranteed

Use **blocking** (cycle detection) to ensure termination
(but the right blocking to not destroy soundness or completeness)