Logical Aspects of Artificial Intelligence
Tableaux Calculi and Complexity

Stéphane Demri    demri@lsi.fr

December 16th, 2019
Plan of the lecture

- Recapitulation of the previous lecture.
- Decision procedures using model-theoretical properties.
- Tableaux proof system for $\mathcal{ALC}$.
- Complexity of decision problems for $\mathcal{ALC}$ and variants.
- Exercises session.
Recapitulation of the previous lecture
ALC in a nutshell

\[ C ::= \top \mid \bot \mid A \mid \neg C \mid C \cap C \mid C \cup C \mid \exists r . C \mid \forall r . C \]

- Interpretation \( \mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I}) \).

- TBox \( \mathcal{T} = \{ C \sqsubseteq D, \ldots \} \).

- ABox \( \mathcal{A} = \{ a : C, (b, b') : r, \ldots \} \).

- Knowledge base \( \mathcal{K} = (\mathcal{T}, \mathcal{A}) \). (a.k.a. ontology)

- Decision problems include concept satisfiability, knowledge base consistency, and other problems for classification.
\[
\top^\mathcal{I} \quad \text{def} \quad \Delta^\mathcal{I}
\]
\[
\bot^\mathcal{I} \quad \text{def} \quad \emptyset
\]
\[
(\neg C)^\mathcal{I} \quad \text{def} \quad \Delta^\mathcal{I} \setminus C^\mathcal{I}
\]
\[
(C_1 \sqcup C_2)^\mathcal{I} \quad \text{def} \quad C_1^\mathcal{I} \cup C_1^\mathcal{I}
\]
\[
(C_1 \sqcap C_2)^\mathcal{I} \quad \text{def} \quad C_1^\mathcal{I} \cap C_1^\mathcal{I}
\]
\[
(\exists r. C)^\mathcal{I} \quad \text{def} \quad \{ a \in \Delta^\mathcal{I} \mid r^\mathcal{I}(a) \cap C^\mathcal{I} \neq \emptyset \}
\]
\[
(\forall r. C)^\mathcal{I} \quad \text{def} \quad \{ a \in \Delta^\mathcal{I} \mid r^\mathcal{I}(a) \subseteq C^\mathcal{I} \}
\]
A few properties about $\mathcal{ALC}$

- Concept satisfiability problem is PSPACE-complete.

- Knowledge base consistency problem is EXPSPACE-complete.

- $\mathcal{ALC}$ has many well-known fragments and extensions, some of them to deal with
  - inverse roles,
  - number restrictions,
  - properties on the role interpretations,
  - inclusions between the composition of roles,
  - etc..
DLs and ontologies

- W3C’s OWL 2 is based on the queen description logic $\text{SROIQ}$.

- OWL reasoners: implement decision procedures for consistency and ontology classification.

- Open-source ontology editor Protégé.
  - Interaction with DL reasoners (FaCT++, Pellet, Racer) via the OWL API.
  - Show results about ontology classification.
Tree interpretations

- $\mathcal{I}$ is a tree interpretation for $C$ with respect to $\mathcal{T}$ iff the conditions below hold:
  - $\mathcal{I} = (\Delta^\mathcal{I}, \bigcup_r r^\mathcal{I})$ is a tree,
  - the root of $\mathcal{I}$ belongs to $C^\mathcal{I}$,
  - $\mathcal{I} \models \mathcal{T}$.

- $\mathcal{ALC}$ has the tree interpretation property and the finite interpretation property.

- **Path** in $\mathcal{I}$: finite sequence $(a_1, \ldots, a_n) \in (\Delta^\mathcal{I})^+$ such that for all $i \in [1, n - 1]$, we have $(a_i, a_{i+1}) \in \bigcup_r r^\mathcal{I}$.

- **α-path**: path such that $a_1 = α$. 
Unravelling an interpretation with a single role

- Unravelling of $\mathcal{I}$ at $a \in \Delta^\mathcal{I}$: $\mathcal{U} = (\Delta^\mathcal{U}, \mathcal{U})$ with
  - $\Delta^\mathcal{U}$ is the set of $a$-paths in $\mathcal{I}$.
  - For all $A$, $A^\mathcal{U} \overset{\text{def}}{=} \{(a_1, \ldots, a_n) \in \Delta^\mathcal{U} \mid a_n \in A^\mathcal{I}\}$,
  - For all role names $r$, we have

$$r^\mathcal{U} \overset{\text{def}}{=} \{((a_1, \ldots, a_n), (a_1, \ldots, a_n, a_{n+1})) \mid (a_n, a_{n+1}) \in r^\mathcal{I}\}$$

- $C$ is satisfiable with respect to a TBox $\mathcal{T}$ implies $C$ has a tree interpretation with respect to $\mathcal{T}$. 
Small interpretation property
Subconcepts

Set of subconcepts $\text{sub}(C)$ and size $\text{size}(C)$:

<table>
<thead>
<tr>
<th>Concept $C$</th>
<th>$\text{sub}(C)$</th>
<th>$\text{size}(C)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>${A}$</td>
<td>1</td>
</tr>
<tr>
<td>$\top/\bot$</td>
<td>${\top}/{\bot}$</td>
<td>1</td>
</tr>
<tr>
<td>$\neg C_1$</td>
<td>$\text{sub}(C_1) \cup {\neg C_1}$</td>
<td>$1 + \text{size}(C_1)$</td>
</tr>
<tr>
<td>$C_1 \cap C_2$</td>
<td>$\text{sub}(C_1) \cup \text{sub}(C_2) \cup {C_1 \cap C_2}$</td>
<td>$1 + \text{size}(C_1) + \text{size}(C_2)$</td>
</tr>
<tr>
<td>$C_1 \cup C_2$</td>
<td>$\text{sub}(C_1) \cup \text{sub}(C_2) \cup {C_1 \cup C_2}$</td>
<td>$1 + \text{size}(C_1) + \text{size}(C_2)$</td>
</tr>
<tr>
<td>$\exists r. C_1$</td>
<td>$\text{sub}(C_1) \cup {\exists r. C_1}$</td>
<td>$1 + \text{size}(C_1)$</td>
</tr>
<tr>
<td>$\forall r. C_1$</td>
<td>$\text{sub}(C_1) \cup {\forall r. C_1}$</td>
<td>$1 + \text{size}(C_1)$</td>
</tr>
</tbody>
</table>

\[
\text{sub}(T) = \bigcup_{C \subseteq D \in T} \text{sub}(C) \cup \text{sub}(D) \quad \text{sub}(A) = \bigcup_{a: C \in A} \text{sub}(C)
\]

- $\text{size}(T)$, $\text{size}(A)$: sum of the sizes of its elements.
- $\text{card}(\text{sub}(T) \cup \text{sub}(A)) \leq \text{size}(T) + \text{size}(A)$.
Type

- A set of concepts $X$ is **closed under subconcepts** iff $\text{sub}(X) = X$.

- $\text{sub}(C)$, $\text{sub}(\mathcal{T})$, $\text{sub}(\mathcal{A})$ are closed under subconcepts.

- **$X$-type** of $a$ in $\mathcal{I}$:

  $$\text{type}_X(a) \overset{\text{def}}{=} \{ C \in X \mid a \in C^\mathcal{I} \}$$

- If $X$ is finite, then

  $$\text{card}(\{ \text{type}_X(a) \mid a \in \Delta^\mathcal{I} \}) \leq 2^{\text{card}(X)}$$

- Small interpretation property is established by showing that no need to keep too many individuals with the same $X$-type with $X = \text{sub}(C) \cup \text{sub}(\mathcal{T}) \cup \text{sub}(\mathcal{A})$. 
Filtration

- Set $X$ closed under subconcepts, interpretation $\mathcal{I}$.

- $a \approx_X b \iff \text{type}_X(a) = \text{type}_X(b)$, and equivalence class $[a]_X$.

- $X$-filtration $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$:
  - $\Delta^{\mathcal{J}} \overset{\text{def}}{=} \{[a]_X \mid a \in \Delta^{\mathcal{I}}\}$.
  - $A^{\mathcal{J}} \overset{\text{def}}{=} \{[a]_X \mid \text{there is } a' \in [a]_X \text{ such that } a' \in A^{\mathcal{I}}\}$. ($A \in X$)
  - $r^{\mathcal{J}} \overset{\text{def}}{=} \{([a]_X, [b]_X) \mid \text{there is } a' \in [a]_X, b' \in [b]_X \text{ such that } (a', b') \in r^{\mathcal{I}}\}$.

- $\text{card}(\Delta^{\mathcal{J}}) \leq 2^{\text{card}(X)}$.

- For all $C \in X$ and $a \in \Delta^{\mathcal{I}}$, we have $a \in C^{\mathcal{I}}$ iff $[a]_X \in C^{\mathcal{J}}$. 
Induction step for $\exists r.C \in X$

- First, suppose that $a \in \exists r.C^\mathcal{I}$.
  - By definition of $\mathcal{I}$, there is $b$ such that $(a, b) \in r^\mathcal{I}$ and $b \in C^\mathcal{I}$.
  - By definition of $\mathcal{J}$, $([a]_X, [b]_X) \in r^\mathcal{J}$.
  - By $X$ closed under subconcepts and (IH), $[b]_X \in C^\mathcal{J}$.
  - By definition of $\mathcal{J}$, $[a]_X \in (\exists r.C)^\mathcal{J}$.

- Suppose that $[a]_X \in (\exists r.C)^\mathcal{J}$.
  - By definition of $\mathcal{J}$, there is $[b]_X$ such that $([a]_X, [b]_X) \in r^\mathcal{J}$ and $[b]_X \in C^\mathcal{J}$.
  - By definition of $r^\mathcal{J}$, there is $(a', b') \in r^\mathcal{I}$ such that $a' \in [a]_X$ and $b' \in [b]_X$.
  - By $X$ closed under subconcepts and (IH), $b \in C^\mathcal{I}$ and as $b' \in [b]_X$, $b' \in C^\mathcal{I}$ too.
  - By definition of $\mathcal{I}$, $a' \in (\exists r.C)^\mathcal{I}$.
  - As $a' \in [a]_X$, $a \in (\exists r.C)^\mathcal{I}$. 
Small interpretation property

- $\mathcal{C}, \mathcal{K} = (\mathcal{T}, \mathcal{A})$, $X$ closed under subconcepts with $(\text{sub} (\mathcal{C}) \cup \text{sub} (\mathcal{T}) \cup \text{sub} (\mathcal{A})) \subseteq X$.

- If $\mathcal{C}$ is satisfiable w.r.t. $\mathcal{K}$, then there is an interpretation $\mathcal{J}$ such that $\mathcal{J} \models \mathcal{K}$, $C^\mathcal{J} \neq \emptyset$ and $\text{card}(\Delta^\mathcal{J}) \leq 2^{\text{card}(X)}$.

- $\mathcal{J}$ is an $X$-filtration based on some interpretation $\mathcal{I}$ such that for all $a : C \in \mathcal{A}$, we have $a^\mathcal{J} \overset{\text{def}}{=} [a^\mathcal{I}]_X$.

- The equivalence “$a \in C^\mathcal{I}$ iff $[a]_X \in C^\mathcal{J}$” leads to the satisfaction of $\mathcal{K}$ in $\mathcal{J}$.
Generalities about decision procedures

- A decision problem $P$ is a subset of $\Sigma^*$.
  ($\Sigma$ is a finite alphabet)

- Alternatively, given $w \in \Sigma^*$, is $w$ in $P$?

- An algorithm for $P$ is sound if whenever it answers “$w \in P$”, then $w \in P$.

- An algorithm for $P$ is complete if whenever $w \in P$, it answers “$w \in P$”.

- An algorithm for $P$ is terminating if it stops after finitely many steps for all $w \in \Sigma^*$.

- Decision procedure: sound, complete and terminating.
A brute force decision procedure

- Input: $C, \mathcal{K} = (\mathcal{T}, \mathcal{A})$.

- Guess an interpretation $\mathcal{I}$ such that $\text{card}(\Delta^\mathcal{I}) \leq 2^{\text{card}(X)}$ with $X = \text{sub}(C) \cup \text{sub}(\mathcal{T}) \cup \text{sub}(\mathcal{A})$.

- Compute the set $C^\mathcal{I}$ using a labelling algorithm based on the definition of $\cdot^\mathcal{I}$ for complex concepts.

- Check the satisfaction of $\mathcal{I} \models \mathcal{K}$ using again a labelling algorithm.

- Checking $C^\mathcal{I} \neq \emptyset$ and $\mathcal{I} \models \mathcal{K}$ can be done in $\text{NEXPTime}$.
A PSPACE algorithm for $\mathcal{ALC}$ satisfiability
Closure

- $\sim C \overset{\text{def}}{=} D$ if $C = \neg D$, otherwise $\sim C \overset{\text{def}}{=} \neg C$.

- **Closure** $\text{cl}(C)$ of a concept $C$: least set closed under subconcepts containing $\text{sub}(C)$ and closed under $\sim$.

- $\text{card}(\text{cl}(C)) \leq 2 \times \text{card}(\text{sub}(C))$.

- A set $X$ is **closed** $\iff X = \bigcup_{C \in X} \text{cl}(C)$.

- A set $Y$ is **patently inconsistent** $\iff Y$ contains $\perp$, $\neg \top$, or a pair of concepts of the form either $C$ and $\neg C$ or $C$ and $\sim C$. 
Maximally consistent sets

- Given a closed set of concepts $X$, the set $Y \subseteq X$ is maximally consistent $\iff$
  - $Y$ is not patently inconsistent.
  - For all $C \in X$, either $C \in Y$ or $\sim C \in Y$.
  - For all $\neg\neg C \in X$, $C \in Y$ iff $\neg\neg C \in Y$.
  - For all $C_1 \cap C_2 \in X$, $C_1 \cap C_2 \in Y$ iff $\{C_1, C_2\} \subseteq Y$.
  - For all $C_1 \cup C_2 \in X$, $C_1 \cup C_2 \in Y$ iff $\{C_1, C_2\} \cap Y \neq \emptyset$.

- Consequently, for all $\neg(C_1 \cap C_2) \in Y$, we have $\sim C_1 \in Y$ or $\sim C_2 \in Y$.

- $\{C \mid a \in C^T, C \in X\}$ is maximally consistent.
Extended closure

- $\text{ecl}(n, C)$: subconcepts occurring in $\exists/\forall$-depth at least $n$.

- The $\text{ecl}(n, C)$’s are the least sets of concepts satisfying the conditions below:
  - $\text{ecl}(n, C)$ is closed.
  - $\text{ecl}(0, C) \overset{\text{def}}{=} \text{cl}(C)$.
  - If $\exists r.D$ or $\forall r.D$ occurs in some concept of $\text{ecl}(n, C)$, then $D \in \text{ecl}(n + 1, C)$.

- $Y \subseteq \text{cl}(C)$ is \textit{n-maximally consistent} $\overset{\text{def}}{=} Y$ is maximally consistent w.r.t. $\text{ecl}(n, C)$.

- $\text{ecl}(\text{size}(C), C) = \emptyset$. 


Example

- $C = \exists r. T \sqcup (\forall r. \exists s. A)$.

- $ecl(0, C) = \{ C, \neg C, \exists r. T, \neg \exists r. T, \forall r. \exists s. A,$
  $\neg \forall r. \exists s. A, T, \neg T, \exists s. A, \neg \exists s. A, A, \neg A \}$

- $ecl(1, C) = \{ T, \neg T, \exists s. A, \neg \exists s. A, A, \neg A \}$.

- $ecl(2, C) = \{ A, \neg A \}$.

- $ecl(3, C) = \emptyset$. 
Nondeterministic algorithm for concept satisfiability

1: procedure \( \text{SAT}_{\mathbb{ALC}}(Y, d) \)
2: \hspace{1em} if \( Y \) is not \( d \)-maximally consistent then abort
3: \hspace{1em} end if
4: \hspace{1em} if \( Y \) contains only \( \exists / \forall \)-free concepts then return \( \text{true} \)
5: \hspace{1em} end if
6: \hspace{1em} for \( \exists r. D \in Y \) do
7: \hspace{1em} \hspace{1em} Guess \( Z \subseteq \text{ecl}(d + 1, C) \) such that \( D \in Z \) and \( \{ \neg D : \neg \exists r. D' \in Y \} \cup \{ D' : \forall r. D' \in Y \} \subseteq Z \)
8: \hspace{1em} \hspace{1em} if not SAT\(_{\mathbb{ALC}}(Z, d + 1) \) then abort
9: \hspace{1em} \hspace{1em} end if
10: \hspace{1em} end for
11: \hspace{1em} for \( \neg \forall r. D \in Y \) do
12: \hspace{1em} \hspace{1em} Guess \( Z \subseteq \text{ecl}(d + 1, C) \) such that \( \neg D \in Z \) and \( \{ \neg D : \neg \exists r. D' \in Y \} \cup \{ D' : \forall r. D' \in Y \} \subseteq Z \)
13: \hspace{1em} \hspace{1em} if not SAT\(_{\mathbb{ALC}}(Z, d + 1) \) then abort
14: \hspace{1em} \hspace{1em} end if
15: \hspace{1em} end for
16: end procedure
Computational properties and correctness

- As $\text{ecl}(|C|, C) = \emptyset$, the recursive depth of $\text{SAT}_{\mathcal{ALC}}$ is bounded by $|C|$.

- Each call requires linear space in $|C|$.

- If $Y \subseteq \text{cl}(C)$, then $\text{SAT}_{\mathcal{ALC}}(Y, d)$ runs in nondeterministic polynomial space. ($\text{PSPACE} = \text{NPSPACE}$)

- $C$ is satisfiable iff there is a 0-maximally consistent set $Y$ such that $C \in Y$ and $\text{SAT}_{\mathcal{ALC}}(Y, 0)$ has an accepting computation.

- Concept satisfiability problem for $\mathcal{ALC}$ is in $\text{PSPACE}$. 
Miscellaneous remarks

- The correctness proof establishes a tree interpretation property as an accepting computation leads to a tree interpretation.

- The algorithm can be easily extended to admit GCIs but the recursive depth becomes exponential.

- The algorithm does not assume any proof system but it has brutal nondeterministic steps.

- The forthcoming tableaux-style proof systems are able to better control the nondeterministic steps and admit strategies leading to optimal complexity upper bounds.
Tableaux for $\text{ALC}$
Automated reasoning for non-classical logics

- Direct methods:
  - Analytical calculi: tableaux, sequents, hypersequents, etc.
  - Resolution.
  - Automata-based decision procedures.

- Translation into
  - other modal logics (PDL, modal $\mu$-calculus, . . .)
  - decidable fragments of first-order logic (FO2, GF, . . .)
  - second-order monadic logics (S2S, . . .)
Tableaux-based proof systems

▶ Analytical method: the rules of the calculi perform a syntactic decomposition of the concepts (formulae, etc.).

▶ Method developed initially by R. Smullyan for first-order logic (circa 1968).

▶ Close relationships with sequent-style proof systems.

▶ Modular approach as new conditions or new ingredients may correspond to the addition of new rules.

▶ Labels are sometimes used in such proof systems.
  ▶ Labels are interpreted as entities of the domain under construction.
  ▶ Expressions external to the original logical language.
  ▶ Labels can be also used as control data structures.
Methodology

- To design a sound and complete proof system for knowledge base consistency, we start by designing a calculus when the TBox $\mathcal{T}$ is empty.

- The extension with a TBox is treated in a second part.

- This will lead to a decision procedure (terminating) for knowledge base consistency in which a few optimisations leads to \textsc{ExpTime} (not presented today).

- The proof system works with ABoxes and rewrite it with the intention to build an interpretation from the ABoxes.

- The presentation follows the usual way to present tableaux-style proof systems for description logics but other presentations exist for modal and temporal logics.
Having \( \sim \) in the algorithm for concept satisfiability was fine but an alternative way to proceed it to use concepts in negation normal form.

C is in **negation normal form (NNF)** \( \defeq \) the negation \( \neg \) occurs only in front of concept names.

Every concept has an equivalent concept in NNF:

\[
\neg(C \sqcup D) \equiv \neg C \sqcap \neg D \quad \neg(C \sqcap D) \equiv \neg C \sqcup \neg D
\]

\[
\neg \exists r. C \equiv \forall r. \neg C \quad \neg \forall r. C \equiv \exists r. \neg C \quad \neg \neg C \equiv C \quad \neg \top \equiv \bot \quad \neg \bot \equiv \top
\]

Transforming a concept into an equivalent concept in NNF takes polynomial time (only).

NNFs are not a must but this simplifies forthcoming developments.
Principles of the tableaux-style proof systems

- The calculus is made of rewriting rules that transform an ABox $\mathcal{A}$ into another ABox $\mathcal{A}'$ nondeterministically. (ex: $\cap$-rule)

- The order of rule applications is irrelevant, except when optimal strategies are designed.

- To guarantee termination, provisos are added to the application of the rewriting rules. (ex. blocking technique)

- Modular approach as new ingredients in the logic leads to new rules, and the provisos are refined (when possible).

- ABoxes with no contradiction and for which no rule application adds value correspond to interpretations.
Example: the $\cap$-rule

$\cap$-rule: If $a : C \cap D \in A$ and $\{a : C, a : D\} \not\subseteq A$ then

$$A \rightarrow A \cup \{a : C, a : D\}$$

- Applying the rule can be viewed as repairing locally the non maximal consistency.

- Satisfaction of $\{a : C, a : D\} \not\subseteq A$ avoids void rule applications.

- The other rules are designed on the same pattern.
Expansion rules for $\mathcal{ALC}$ ABox consistency

$\sqcap$-rule: If $a : C \sqcap D \in \mathcal{A}$ and $\{a : C, a : D\} \not\subseteq \mathcal{A}$ then

$$\mathcal{A} \rightarrow \mathcal{A} \cup \{a : C, a : D\}$$

$\sqcup$-rule: If $a : C \sqcup D \in \mathcal{A}$ and $\{a : C, a : D\} \cap \mathcal{A} = \emptyset$ then

$$\mathcal{A} \rightarrow \mathcal{A} \cup \{a : E\} \quad \text{for some } E \in \{C, D\}$$

$\exists$-rule: If $a : \exists r. C \in \mathcal{A}$ and there is no $b$ such that $\{(a, b) : r, b : C\} \subseteq \mathcal{A}$ then

$$\mathcal{A} \rightarrow \mathcal{A} \cup \{(a, c) : r, c : C\} \quad \text{where } c \text{ is fresh}$$

$\forall$-rule: If $\{(a, b) : r, a : \forall r. C\} \subseteq \mathcal{A}$ and $b : C \not\in \mathcal{A}$, then

$$\mathcal{A} \rightarrow \mathcal{A} \cup \{b : C\}$$
Complete and clash-free ABox

- An ABox $\mathcal{A}$ contains a clash if $\{ a : A, a : \neg A \} \subseteq \mathcal{A}$ or $a : \bot \in \mathcal{A}$.

⚠️ Notion of clash to be extended if the concepts are not in NNF.

- An ABox $\mathcal{A}$ is clash-free if it does not contain a clash.

- An ABox $\mathcal{A}$ is complete if it contains a clash or if no rule is applicable.

Objective: to show that $\mathcal{A}$ is consistent iff $\mathcal{A} \rightarrow^{*} \mathcal{A}'$ for some complete and clash-free ABox $\mathcal{A}'$.

- The only nondeterministic rule is the $\sqcup$-rule.
Example

\[ A = \{(a, b) : s, (a, c) : r\} \cup \]
\{ a : A_1 \cap \exists s. A_5, a : \forall s. \neg A_5 \cup \neg A_2, b : A_2, c : A_3 \cap \exists s. A_4 \}\]

\[ A \xrightarrow{*} A \cup \{ a : A_1, a : \exists s. A_5, a_{\text{new}} : A_5, (a, a_{\text{new}}) : s \]
\{ b : \neg A_5 \cup \neg A_2, a_{\text{new}} : \neg A_5 \cup \neg A_2, b : \neg A_5, a_{\text{new}} : \neg A_2, \]
\{ c : A_3, c : \exists s. A_4, c_{\text{new}} : A_4, (c, c_{\text{new}}) : s \}\]
Termination

- The \( \exists \text{-weight} \) of \( C \) is the number of its subconcepts of the form \( \exists r.D \).

\[
w_{\exists}(C) \overset{\text{def}}{=} \text{card}(\{\exists r.D \mid \exists r.D \in \text{sub}(C)\})
\]

- The definition assumes that \( C \) is in NNF.

- \( w_{\exists}(A) \overset{\text{def}}{=} \sum_{a : C \in A} w_{\exists}(C) \).

- The \( \forall \exists \text{-depth} \) of \( C \), written \( d_{\forall \exists}(C) \), is the maximal number of imbrications of \( \exists r. \) and \( \forall s. \) in \( C \).

- \( d_{\forall \exists}(\exists r. T \sqcup \forall r. \exists s. A) = 2 \)

- \( d_{\forall \exists}(A) = \max\{d_{\forall \exists}(C) \mid a : C \in A\} \).
Labelling the individual names

Let \( \mathcal{A} \) be an ABox with \( W = w_\exists(\mathcal{A}) \), \( D = d_\forall(\mathcal{A}) \) and \( N \) is the number of distinct individual names in \( \mathcal{A} \).

Let \( \mathcal{A}^0 \) be the variant of \( \mathcal{A} \) where \( a : C \) is replaced by \( a^0 : C \).

\( \cap \)-rule: If \( a^i : C \cap D \in \mathcal{A} \) and \( \{ a^i : C, a^j : D \} \not\subseteq \mathcal{A} \) then

\[ \mathcal{A} \rightarrow \mathcal{A} \cup \{ a^i : C, a^j : D \} \]

\( \cup \)-rule: If \( a^i : C \cup D \in \mathcal{A} \) and \( \{ a^i : C, a^j : D \} \cap \mathcal{A} = \emptyset \) then

\[ \mathcal{A} \rightarrow \mathcal{A} \cup \{ a^i : E \} \quad \text{for some } E \in \{ C, D \} \]

\( \exists \)-rule: If \( a^i : \exists r. C \in \mathcal{A} \) and there is no \( b^j \) such that \( \{ (a^i, b^j) : r, b^j : C \} \subseteq \mathcal{A} \) then

\[ \mathcal{A} \rightarrow \mathcal{A} \cup \{ (a^i, c^{i+1}) : r, c^{i+1} : C \} \quad \text{where } c \text{ is fresh} \]

\( \forall \)-rule: If \( \{ (a^i, b^j) : r, a^i : \forall r. C \} \subseteq \mathcal{A} \) and \( b^j : C \not\in \mathcal{A} \), then

\[ \mathcal{A} \rightarrow \mathcal{A} \cup \{ b^j : C \} \]
Quantities about $\mathcal{A}^0 \rightarrow^* \mathcal{A}'$

- If $a^i : C \in \mathcal{A}'$, then $i + \text{d}_{\forall\exists}(C) \leq D$. 
  
  Trees from individual names labelled by zero have depth at most $D$.

- $a^i : C \in \mathcal{A}'$ implies 
  \[ \text{card}(\{(a^i, b^j) | (a^i, b^j) : r \in \mathcal{A}'\}) \leq N + W. \]
  
  The maximum branching degree of nodes in the trees is at most $N + W$.

- $a^i : C \in \mathcal{A}'$ implies $C \in \text{sub}(\mathcal{A})$.

- The length of the derivation $\mathcal{A}^0 \rightarrow^* \mathcal{A}'$ is at most 
  \[ N \times (D + 1) \times (N + W)^D \times \text{card}(\text{sub}(\mathcal{A})) \]
  
  (why?)
The auxiliary function exp

- Expansion function \( \text{exp}(\mathcal{A}, R, X) \) taking as arguments
  - an ABox \( \mathcal{A} \),
  - an expansion rule \( R \),
  - a subset \( X \) of \( \mathcal{A} \) (with one or two elements) allowing the application of \( R \)

- \( \ldots \) and returning the set of ABoxes obtained from \( \mathcal{A} \) by applying the rule \( R \) with main assertions in \( X \).

- \( \text{exp}(\{ a : E, a : C \sqcup D \}, \sqcup\text{-rule}, a : C \sqcup D) \) is equal to

\[
\{ \{ a : E, a : C \sqcup D, a : C \}, \{ a : E, a : C \sqcup D, a : D \} \}
\]
Main algorithm

We shall show that $\mathcal{A}$ is consistent iff $\mathcal{A} \xrightarrow{\ast} \mathcal{A}'$ for some complete and clash-free ABox $\mathcal{A}'$.

Existence of $\mathcal{A}'$ amounts to explore a finite tree of bounded depth and bounded degree.

1: procedure $\text{EXPAND}(\mathcal{A})$
2: \hspace{1em} if $\mathcal{A}$ has a clash then return $\emptyset$
3: \hspace{1em} end if
4: \hspace{1em} if $\mathcal{A}$ is clash-free and complete then return $\mathcal{A}$
5: \hspace{1em} end if
6: \hspace{1em} for applicable $R, X$ on $\mathcal{A}$ and $\mathcal{A}' \in \exp(\mathcal{A}, R, X)$ do
7: \hspace{2em} if $\text{expand}(\mathcal{A}') \neq \emptyset$ then return $\text{expand}(\mathcal{A}')$
8: \hspace{2em} end if
9: \hspace{1em} end for
10: \hspace{1em} return $\emptyset$
11: end procedure
1: procedure EXPAND(\(\mathcal{A}\))
2: \hspace{1em} if \(\mathcal{A}\) has a clash then return \(\emptyset\)
3: \hspace{1em} end if
4: \hspace{1em} if \(\mathcal{A}\) is clash-free and complete then return \(\mathcal{A}\)
5: \hspace{1em} end if
6: \hspace{1em} for applicable \(R, X\) on \(\mathcal{A}\) and \(\mathcal{A}' \in \text{exp}(\mathcal{A}, R, X)\) do
7: \hspace{2em} if \(\text{expand}(\mathcal{A}') \neq \emptyset\) then return \(\text{expand}(\mathcal{A}')\)
8: \hspace{2em} end if
9: \hspace{1em} end for
10: \hspace{1em} return \(\emptyset\)
11: end procedure
Root individuals and tree individuals

- **Tree individuals** are generated by application of the $\exists$-rule.
- If $(a, b) : r$ is added by application of the $\exists$-rule, $b$ is an $r$-successor of $a$.
- Root individuals have no predecessors or ancestors.
Soundness

- Let $\mathcal{A}$ be a finite ABox with at least one concept assertion, complete, clash-free and all the concepts in NNF. Then, $\mathcal{A}$ is consistent.

- For each individual name $a$ occurring in $\mathcal{A}$, we write $\text{con}_\mathcal{A}(a)$ to denote the set $\{C \mid a : C \in \mathcal{A}\}$.

- Let us define $\mathcal{I} \overset{\text{def}}{=} (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ as follows.
  - $\Delta^\mathcal{I} \overset{\text{def}}{=} \{a \mid a : C \in \mathcal{A}\}$.
  - $a^\mathcal{I} \overset{\text{def}}{=} a$ for all individual names $a$ in $\mathcal{A}$.
  - $A^\mathcal{I} \overset{\text{def}}{=} \{a \mid A \in \text{con}_\mathcal{A}(a)\}$ for all concept names $A \in \text{sub}(\mathcal{A})$.
  - $r^\mathcal{I} \overset{\text{def}}{=} \{(a, b) \mid (a, b) : r \in \mathcal{A}\}$.

- Let us show that for all $a : C \in \mathcal{A}$, we have $a^\mathcal{I} \in C^\mathcal{I}$. 
Proof by structural induction

- The base case with concept assertions $a : A$ is immediate by definition of $A^\mathcal{I}$.

- The base case with concept assertions $a : \neg A$ is immediate by definition of $A^\mathcal{I}$ as $\mathcal{A}$ is clash-free.

- Case $a : C \sqcup D$ in the induction step.
  - As $\mathcal{A}$ is complete, $a : C \in \mathcal{A}$ or $a : D \in \mathcal{A}$.
  - W.l.o.g., suppose $a : C \in \mathcal{A}$. By (IH), $a^\mathcal{I} \in C^\mathcal{I}$.
  - By definition of $\cdot^\mathcal{I}$, we conclude $a^\mathcal{I} \in (C \sqcup D)^\mathcal{I}$.

- Case $a : \exists r.C$ in the induction step.
  - As $\mathcal{A}$ is complete, $\{(a, b) : r, b : C\} \subseteq \mathcal{A}$ for some $b$.
  - By definition of $r^\mathcal{I}$, $(a, b) \in r^\mathcal{I}$.
  - By (IH), $b^\mathcal{I} \in C^\mathcal{I}$.
  - By definition of $\cdot^\mathcal{I}$, we conclude $a^\mathcal{I} \in (\exists r.C)^\mathcal{I}$.
The cases in the induction step for $\cap$-concept assertions and $\forall$-concept assertions are similar.

If $\text{expand}(\mathcal{A}) \neq \emptyset$, then $\mathcal{A}$ is consistent.

Indeed, $\text{expand}(\mathcal{A}) \neq \emptyset$ if there is some $\mathcal{A}'$ with $\mathcal{A} \subseteq \mathcal{A}'$ such that $\mathcal{A}'$ is complete and clash-free.

Consistency of $\mathcal{A}'$ leads to the consistency of $\mathcal{A}$. 
Completeness

- If $\mathcal{A}$ is consistent, then $\mathcal{A} \rightarrow^* \mathcal{A}'$ for some complete and clash-free ABox $\mathcal{A}'$.

- Let $\mathcal{I} \overset{\text{def}}{=} (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ be such that $\mathcal{I} \models \mathcal{A}$.

- If $\mathcal{A}$ is complete, we are done. Otherwise, at least one rule is application to $\mathcal{A}$ preserving consistency.

- Otherwise, if $\mathcal{A}$ is not complete, we show that there is $\mathcal{A}'$ such that $\mathcal{A} \rightarrow \mathcal{A}'$ and $\mathcal{A}'$ is consistent.

- As the length of a derivation from $\mathcal{A}$ is bounded by an exponential in the size of $\mathcal{A}$, there is $\mathcal{A}'$ such that $\mathcal{A} \rightarrow^* \mathcal{A}'$ and $\mathcal{A}'$ is complete, clash-free (and consistent).

- It remains to prove that non-completeness implies the existence of one expansion preserving consistency.
Single steps in the completeness proof

- If the $\sqcup$-rule is applicable on $a : C \sqcup D$, then there is $E \in \{C, D\}$ such that $\mathcal{I} \models A \cup \{a : E\}$.

- $A \rightarrow A \cup \{a : E\}$ and $\mathcal{I} \models A \cup \{E\}$.

- If the $\exists$-rule is applicable on $a : \exists r.C$, then we use the fact that $a^\mathcal{I} \in (\exists r.C)^\mathcal{I}$.

- There is $\alpha \in \Delta^\mathcal{I}$ such that $\alpha \in C^\mathcal{I}$ and $(a^\mathcal{I}, \alpha) \in r^\mathcal{I}$.

- Let $\mathcal{I}'$ be equal to $\mathcal{I}$ except that $\mathcal{I}'(c) = \alpha$ for some fresh $c$.

- Then, $A \rightarrow A \cup \{c : C, (a, c) : r\}$ and $\mathcal{I}' \models A \cup \{c : C, (a, c) : r\}$
Decision procedure of ABox consistency

- $\mathcal{A}$ is consistent iff $\mathcal{A} \xrightarrow{*} \mathcal{A}'$ for some complete and clash-free ABox $\mathcal{A}'$.

- Derivations $\mathcal{A} \xrightarrow{*} \mathcal{A}'$ have length bounded by an exponential in size($\mathcal{A}$).

- Existence of $\mathcal{A}'$ amounts to explore a tree of bounded depth and bounded degree.
Adding a TBox – First properties

- \( \mathcal{I} \models C \subseteq D \) iff \( \mathcal{I} \models T \subseteq \neg C \cup D \).

- \( \mathcal{I} \models C \equiv D \) iff \( \mathcal{I} \models T \subseteq (\neg C \cup D) \cap (\neg D \cup C) \).

- In the sequel, GCIs are of the form \( T \subseteq E \) with \( E \) in NNF.

\( \sqsubseteq \)-rule: If \( a : C \in \mathcal{A}, T \subseteq D \in \mathcal{T} \) and \( a : D \not\in \mathcal{A} \), then

\[
\mathcal{A} \rightarrow \mathcal{A} \cup \{a : D\}
\]

- The termination argument for ABox consistency does not work anymore. (Why?)
Blocking

* Given \( A \xrightarrow{\ast} A' \), \( a \) is an **ancestor** of \( b \) in \( A' \) iff

\[
\{(a_1, a_2) : r_1, \ldots, (a_k, a_{k+1}) : r_k\} \subseteq A'
\]

with \( a_1 = a \), \( a_{k+1} = b \) and \( b \) is a tree individual.

⚠️ The notion of ancestor assumes that one can distinguish the **root individuals** (individual names from \( A \)) from the **tree individuals** (those introduced by applying the \( \exists \)-rule).

* Termination can be regained thanks to the blocking technique.

* An individual name \( b \) in \( A' \) is **blocked by** \( a \) if

  * \( a \) is an ancestor of \( b \),
  * \( \text{con}_{A'}(b) \subseteq \text{con}_{A'}(a) \).
Expansion rules with blocking

□-rule: If \( a : C \cap D \in \mathcal{A} \) and \( \{a : C, a : D\} \not\subseteq \mathcal{A} \) then \( \mathcal{A} \rightarrow \mathcal{A} \cup \{a : C, a : D\} \).

⊔-rule: If \( a : C \sqcup D \in \mathcal{A} \) and \( \{a : C, a : D\} \cap \mathcal{A} = \emptyset \) then \( \mathcal{A} \rightarrow \mathcal{A} \cup \{a : E\} \) for some \( E \in \{C, D\} \).

∃-rule: If \( a : \exists r. C \in \mathcal{A} \) and there is no \( b \) such that \( \{(a, b) : r, b : C\} \subseteq \mathcal{A} \) then

\[
\mathcal{A} \rightarrow \mathcal{A} \cup \{(a, c) : r, c : C\}
\]
where \( c \) is fresh.

∀-rule: If \( \{(a, b) : r, a : \forall r. C\} \subseteq \mathcal{A} \) and \( b : C \not\in \mathcal{A} \), then \( \mathcal{A} \rightarrow \mathcal{A} \cup \{b : C\} \).

⊑-rule: If \( a : C \in \mathcal{A} \), \( \top \sqsubseteq D \in \mathcal{T} \), and \( a : D \not\in \mathcal{A} \), then \( \mathcal{A} \rightarrow \mathcal{A} \cup \{a : D\} \).
Termination

- $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ with concepts in NNF, $a : C \in \mathcal{A}$, and CGIs of the form $\top \subseteq D$.

- $N$: number of root individuals in $\mathcal{A}$, $M = \text{card}(\text{sub}(\mathcal{K}))$, $W = w_\exists(\mathcal{K})$.

- $\mathcal{A} \overset{*}{\rightarrow} \mathcal{A}'$ and $a : C \in \mathcal{A}'$ imply $\text{card}(\{(a, b) : (a, b) : r \in \mathcal{A}'\}) \leq N + W$.

- $\mathcal{A} \overset{*}{\rightarrow} \mathcal{A}'$ and $a : C \in \mathcal{A}'$ imply $C \in \text{sub}(\mathcal{K})$.

- $\{(a_1, a_2) : r_1, \ldots, (a_k, a_{k+1}) : r_k\} \subseteq \mathcal{A}'$ and $a_2$ is a tree individual imply $k \leq 2^M$.

- The length of the derivation $\mathcal{A} \overset{*}{\rightarrow} \mathcal{A}'$ is at most
  
  $$N \times (2^M + 1) \times (N + W)^{2^M} \times M$$
Soundness

- $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ with concepts in NNF, $a : C \in \mathcal{A}$, and CGIs of the form $\top \subseteq D$.

- $\mathcal{A} \xrightarrow{*} \mathcal{A}'$ with $\mathcal{A}'$ complete and clash-free.

- We construct $\mathcal{A}''$ as the ABox made of the following assertions

\[
\{ a : C \mid a : C \in \mathcal{A}', \ a \text{ is not blocked} \} \cup \\
\{ (a, b) : r \mid (a, b) : r \in \mathcal{A}', \ b \text{ is not blocked} \} \cup \\
\{ (a, b') : r \mid (a, b) : r \in \mathcal{A}', \ a \text{ is not blocked and } b \text{ is blocked by } b' \} 
\]
Properties of $\mathcal{A''}$

- $\mathcal{A} \subseteq \mathcal{A''}$ as root individual cannot be blocked and $\mathcal{A} \subseteq \mathcal{A'}$.
- None of the individual names occurring in $\mathcal{A''}$ is blocked.
- For all $a$ in $\mathcal{A'}$, we have $\text{con}_{\mathcal{A''}}(a) = \text{con}_{\mathcal{A'}}(a)$.
- $\mathcal{A''}$ is complete and clash-free.
More about the soundness proof

- $\mathcal{A} \rightarrow^* \mathcal{A}'$ with $\mathcal{A}'$ complete and clash-free and $\mathcal{A}''$ computed as above.

- Let us define $\mathcal{I} \overset{\text{def}}{=} (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ as follows.
  - $\Delta^{\mathcal{I}} \overset{\text{def}}{=} \{ a \mid a : C \in \mathcal{A}'' \}$.
  - $a^{\mathcal{I}} \overset{\text{def}}{=} a$ for all individual names $a$ in $\mathcal{A}''$.
  - $A^{\mathcal{I}} \overset{\text{def}}{=} \{ a \mid A \in \text{con}_A(a) \}$ for all concept names $A \in \text{sub}(\mathcal{A}'')$.
  - $r^{\mathcal{I}} \overset{\text{def}}{=} \{ (a, b) \mid (a, b) : r \in \mathcal{A}'' \}$.

- One can show that for all $a : C \in \mathcal{A}''$, we have $a^{\mathcal{I}} \in C^{\mathcal{I}}$. 
Completeness (bis)

- If $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is consistent, then $\mathcal{A} \rightarrow^* \mathcal{A}'$ for some complete and clash-free ABox $\mathcal{A}'$.

- Let $\mathcal{I} \overset{\text{def}}{=} (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ be such that $\mathcal{I} \models \mathcal{A}$.

- If $\mathcal{A}$ is complete, we are done. Otherwise, at least one rule is application to $\mathcal{A}$ preserving consistency.

- Otherwise ($\mathcal{A}$ is not complete), we show there is $\mathcal{A}'$ such that $\mathcal{A} \rightarrow \mathcal{A}'$ and $\mathcal{A}'$ is consistent.

- As the length of a derivation from $\mathcal{A}$ is bounded by a double-exponential in the size of $\mathcal{A}$, there is $\mathcal{A}'$ such that $\mathcal{A} \rightarrow^* \mathcal{A}'$ and $\mathcal{A}'$ is complete, clash-free (and consistent).

- One can prove that non-completeness implies the existence of one expansion preserving consistency.
Complexity issues

- $\mathcal{ALC}$ concept satisfiability in $\text{PSPACE}$, knowledge base consistency in $\text{EXPTIME}$.

- The algorithm for ABox consistency runs in exponential space:
  - Because of the nondeterministic $\sqcup$-rule, exponentially many ABoxes may be generated.
  - Complete ABoxes may be exponentially large.

- $\text{PSPACE}$ bound for ABox consistency can be regained by exploring the tree-like interpretations in a depth-first manner having only one path at a time.
Tableaux for $\mathcal{ALCI}$ ($\mathcal{ALC}$ + inverse)

$\{(a, b) : r, b : \forall r^- . D, a : \forall r . C\} \models_{\mathcal{ALCI}} \{b : C, a : D\}$

- $b$ is an $r$-neighbour of $a$ if $(a, b) : r$ or $(b, a) : r^-$. 
- $b$ is an $r^-$-neighbour of $a$ if $(a, b) : r^-$ or $(b, a) : r$. 
- Below, $R$ is either some $r$ or some $r^-$. 

\textit{∃-rule:} If $a : \exists R . C \in A$ and there is no $b$ such that $b : C \in A$ and $b$ is an $R$-neighbour of $a$ then

$$A \rightarrow A \cup \{(a, c) : R, c : C\} \quad \text{where $c$ is fresh}$$

\textit{∀-rule:} If $a : \forall R . C \in A$ and $b$ is an $R$-neighbour of $a$, then

$$A \rightarrow A \cup \{b : C\}$$
Equality blocking

- We need to strenghten the blocking (inclusion of sets of concepts is not anymore sufficient).

- An individual name $b$ in $\mathcal{A}'$ is **blocked by** $a$ if
  - $a$ is an ancestor of $b$,
  - $\text{con}_{\mathcal{A}'}(b) = \text{con}_{\mathcal{A}'}(a)$ (equality blocking).

- Termination, soundness and completeness can be established in a similar fashion, though adaptations are needed.
Recapitulation: 
Tableaux for $\mathcal{ALC}$ knowledge base consistency

- Tableaux-based algorithm to decide $\mathcal{ALC}$ knowledge base consistency.

- All other standard decision problems can be handled too.

- Termination is guaranteed thanks to the blocking technique.

- In the worst-case, exponential space is needed but optimisations exist to meet the optimal complexity upper bounds.
Complexity of problems for $\mathcal{ALC}$ and variants
Recapitulation of upper bounds

- The concept satisfiability problem for $\mathcal{ALC}$ is in $\text{PSpace}$.
- The knowledge base consistency problem for $\mathcal{ALC}$ is in $\text{ExpTime}$.
- Same upper bounds for $\mathcal{ALCI}$.
- In the sequel, we focus on complexity lower bounds.
Why hardness results using tiling problems?

- Ideally, master reductions from Turing machines.

- Tiling problems form a family of problems complete for numerous complexity classes.

- Tiling an arena with tile types naturally corresponds to computations in Turing machines.

- Tiling problems have little structure.
Tiling system

- **Tiling system**: $\mathbf{(T, H, V, t_0)}$ where
  - $T$ is a finite set of **tiles** and $t_0 \in T$,
  - $H, V \subseteq T \times T$ are two relations referred to as the **horizontal**, resp. **vertical matching relation**.

- A set of tiles

  
  \[
  \begin{align*}
  t_1 &= \begin{array}{c|c}
    1 & 2 \\
    \hline
    2 & 0 \\
  \end{array} &
  t_2 &= \begin{array}{c|c|c}
    1 & 2 \\
    \hline
    2 & & 1 \\
  \end{array} &
  t_3 &= \begin{array}{c|c|c|c}
    0 & & & \\
    \hline
    & 2 & & \\
  \end{array} &
  t_4 &= \begin{array}{c|c|c|c}
    2 & 0 & & \\
    \hline
    & & 2 & 1 \\
  \end{array}
  \end{align*}
  \]

- ...with its matching relations

  - $H = \{(t_1, t_3), (t_1, t_4), (t_2, t_1), (t_3, t_2), (t_4, t_1)\}$,
  - $V = \{(t_1, t_2), (t_1, t_4), (t_2, t_3), (t_4, t_1), (t_4, t_2)\}$.
A tiling for the \(([0, 3] \times [0, 2])\)-arena
A \textsc{PSPACE}-complete tiling game problem

- Let \((T, H, V, t_0)\) be a tiling system.

- \((n \times n)\)-tiling game on \((T, H, V, t_0)\) is played on a finite number of rounds between Player 1 and Player 2 to construct a tiling \(\tau : [0, n-1] \times [0, n-1] \rightarrow T\).

- At the \(j\)th, Player 1 chooses \(\tau(0, j)\), and then Player 2 chooses \(\tau(1, j), \ldots, \tau(n-1, j)\).

- Player 1 loses immediately in
  - round 0, if \(\tau(0, 0) \neq t_0\);
  - round \(j > 0\), if \((\tau(0, j-1), \tau(0, j)) \notin V\).

- Player 2 loses immediately in
  - round \(j \geq 0\), if there is an \(i < n - 1\) such that \((\tau(i, j), \tau(i+1, j)) \notin H\);
  - round \(j > 0\), if there is an \(i\) with \(0 < i < n\) and \((\tau(i, j-1), \tau(i, j)) \notin V\).

- A player wins if the opponent loses.
Complexity of the \((n \times n)\)-tiling game problem

- Tiling game problems are closely related to alternating Turing machines where Player 1/Player 2, corresponds to universal/existential states.

- Tiling problems are closely related to nondeterministic Turing machines but of no help to characterise complexity classes with deterministic Turing machines.

- The \((n \times n)\)-tiling game problem is \(\text{APTIME}\)-complete.

- \ldots and therefore the \((n \times n)\)-tiling game problem is \(\text{PSPACE}\)-complete as \(\text{PSPACE} = \text{APTIME}\).
An \textit{EXPTIME}-complete tiling game problem

- \((n \times \infty)\)-tiling game on \((T, H, V, t_0)\) is played on an \textit{infinite} number of rounds between Player 1 and Player 2 to construct a tiling \(\tau : [0, n - 1] \times \mathbb{N} \to T\).

- At the \(j\)th, Player 1 chooses \(\tau(0, j)\), and then Player 2 chooses \(\tau(1, j), \ldots, \tau(n - 1, j)\).

- Player 1 loses immediately in
  - round 0, if \(\tau(0, 0) \neq t_0\);
  - round \(j > 0\), if \((\tau(0, j - 1), \tau(0, j)) \notin V\).

- Player 2 loses immediately in
  - round \(j \geq 0\), if there is an \(i < n - 1\) such that \((\tau(i, j), \tau(i + 1, j)) \notin H\);
  - round \(j > 0\), if there is an \(i\) with \(0 < i < n\) and \((\tau(i, j - 1), \tau(i, j)) \notin V\).

- A player wins if the opponent loses.
Complexity of the \((n \times \infty)\)-tiling game problem

- Given \((T, H, V, t_0)\) and \(n \geq 1\) in unary, has Player 2 has a winning strategy on the \(([0, n - 1] \times \mathbb{N})\)-arena?

- Tiling game problems are closely related to alternating Turing machines where Player 1/Player 2, corresponds to universal/existential states.

- The \((n \times \infty)\)-tiling game problem is \text{APSPACE}-complete.

- \ldots and therefore the \((n \times \infty)\)-tiling game problem is \text{EXPTIME}-complete as \text{EXPTIME} = \text{APSPACE}.
An undecidable tiling problem

- The \((\infty \times \infty)-\)tiling problem.

**Input:** A tiling system \((T, H, V, t_0)\).

**Question:** Is there a tiling \(\tau : \mathbb{N} \times \mathbb{N} \rightarrow T\) such that for all \(i, j \in \mathbb{N}\),
- (hori) if \(\tau(i, j) = t\) and \(\tau(i + 1, j) = t'\), then \((t, t') \in H\),
- (verti) if \(\tau(i, j) = t\) and \(\tau(i, j + 1) = t'\), then \((t, t') \in V\),

- The \((\infty \times \infty)-\)tiling problem is undecidable.
Concept satisfiability for $\mathcal{ALC}$ is PSPACE-hard

- Reduction from the $(n \times n)$-tiling game problem.
- Tiling system $\mathcal{T} = (T, H, V, t_0)$ and $n \in \mathbb{N}$.
- We construct an $\mathcal{ALC}$ concept $C^n_T$ such that $C^n_T$ is satisfiable iff Player 2 has a winning strategy for the tiling game on $\mathcal{T}$ on the $([0, n-1]^2)$-arena.
Strategies are finite trees

\[ t_0 = \begin{array}{c c c}
2 & 1 & 1 \\
2 & 2 & 0
\end{array} \quad t_1 = \begin{array}{c c c}
2 & 1 & 1 \\
2 & 2 & 0
\end{array} \quad t_2 = \begin{array}{c c c}
0 & 0 & 0 \\
0 & 2 & 1
\end{array} \quad t_3 = \begin{array}{c c c}
2 & 1 & 1 \\
2 & 2 & 0
\end{array} \]

Player 2 has a winning strategy with initial tile \( t_0 \) on the \(([0, 1] \times [0, 2])\)-arena.
The reduction

- Every individual belongs to a unique tile type.

\[\text{uni} \overset{\text{def}}{=} \bigsqcup_{t \in T} (t \cap \prod_{t' \neq t} \neg t') \quad \text{(tile types understood as concept names)}\]

- Each individual “at distance” at most \(n^2\) has a unique tile type (arbitrary role name \(r\)).

\[C^n_{\text{uni}} \overset{\text{def}}{=} t_0 \cap \prod_{t \neq t_0} \neg t \cap \forall r. (\text{uni} \cap \forall r. (\text{uni} \cap \ldots \forall r. (\text{uni} \cap \forall r. \text{uni}) \ldots ))\]

- Local horizontal/vertical matching relation:

\[\text{hm} \overset{\text{def}}{=} \prod_{(t,t') \notin H} \neg t \cup \forall r. \neg t' \quad \text{vm} \overset{\text{def}}{=} \prod_{(t,t') \notin V} \neg t \cup (\forall r)^n \neg t' \quad (\forall r)^{i+1} D \overset{\text{def}}{=} (\forall r)^i \forall r. D\]
Satisfying the matching relations everywhere!

- The horizontal matching relation is respected everywhere:

\[
hm \sqcap \forall r (hm \sqcap \ldots \forall r (hm \sqcap \forall r (hm \sqcap \ldots \forall r (hm \sqcap \forall r hm) \ldots)))
\]

\(n-2\) occurrences of \(\forall r\)

\(n^2-1\) occurrences of \(\forall r\)

- The vertical matching relation is respected everywhere:

\[
vm \sqcap \forall r \cdot (vm \sqcap \forall r \cdot (vm \sqcap \ldots \forall r \cdot (vm \sqcap \forall r vm) \ldots))
\]

\(n^2-n-1\) occurrences of \(\forall r\)
Avoiding a single individual interpretation

Every first individual of a row has a chain of \( n - 1 \) \( r \)-successors and the last one has an \( r \)-successor for all possible matching choices of Player 1.

\[
\text{chain} \overset{\text{def}}{=} \bigcap_{t \in T} \neg t \sqcup (\exists r)^{n-1} \cdot \left( \bigcap_{(t,t') \in V} \exists r \cdot t' \right)
\]

\[
C^n_{\text{struct}} \overset{\text{def}}{=} \text{chain} \sqcap (\forall r)^n (\text{chain} \sqcap \ldots (\forall r)^n (\text{chain} \sqcap (\forall r)^n \text{chain}) \ldots)
\]

\( n-2 \) occurrences of \( (\forall r)^n \)
The properties

- $C^n_T$ defined as the conjunction of the above concepts is constructed in logarithmic space in $n$ and $\text{card}(T)$.

- A winning strategy for Player 2 on the $([0, n-1]^2)$-arena induces an interpretation $\mathcal{I}$ such that $(C^n_T)^{\mathcal{I}}$ is non-empty.

- Similarly, every interpretation $\mathcal{I}$ such that $(C^n_T)^{\mathcal{I}}$ is non-empty, leads to a winning strategy for Player 2.

- Consequently, the concept satisfiability problem for $\mathcal{ALC}$ is $\text{PSPACE}$-hard.
Conclusion

- Lecture 1 (last week): Introduction to description logics
- Lecture 2: Tableaux proof systems and complexity.
  - Model-theoretical properties.
  - Complete calculi for $\mathcal{ALC}$ and variants.
  - Complexity results, a bit of undecidability.
- Lecture 3: Introduction to temporal logics for multi-agents systems.