Knowledge Representation and Reasoning

\(\mathcal{ALC}\)-Extensions

Prof. Dr. Hans Jürgen Ohlbach
• Unqualified Number Restrictions
• Qualified Number Restrictions
• Tableau-rules
• Role-Operations
• Role-Hierarchies
• Role-Axioms
• Concrete Domains
Language Extensions

\(\mathcal{ALC}\) is much more expressive than Propositional Logic, but for many purposes yet not expressive enough.

Therefore a number of language extensions have been developed.

Important aspects are:

- do we need new operators in the language?
- which operators can be defined in terms of other ones?
- are the different inference problems (consistency, subsumption, instancetest,...) still decidable?
- what is the complexity of the decision problem?
- what about the algorithms?
Example: Functional Roles

Example 1: CwSR = Cars with Sun Roof made of glass
In $\mathcal{ALC}$:

$$
CwSR = \text{Car} \sqcap \exists \text{has-sun-roof Glass-Objects}
$$

possible (unwanted) Interpretation: more than one sun roof.

Problem: $\text{has-sun-roof}$ is in $\mathcal{ALC}$ a relation, not a (partial) Function.

Example 2: Humans have exactly one father and one mother

$$
\text{Humans} \sqsubseteq (\exists \text{has-Father Human} \sqcap \exists \text{has-Mother Human})
$$

Problem: $\text{has-Father}$ should be a total function on Humans

Solution: Number Restrictions (see next slide)
**Example:** Car = Vehicle with at least 4 wheels.

In $\mathcal{ALC}$:

Car = Vehicle $\sqcap$

\[ \exists \text{has-Wheel} (p_1 \sqcap p_2) \sqcap \exists \text{has-Wheel} (\neg p_1 \sqcap p_2) \sqcap \\
\exists \text{has-Wheel} (p_1 \sqcap \neg p_2) \sqcap \exists \text{has-Wheel} (\neg p_1 \sqcap \neg p_2) \]

**Definition of:**

Metropolis = City with at least 1000000 inhabitants?

This is extremely expensive.

**Alternative:** Number Operators in the language
(Unqualified) Number Restrictions

\[ \mathcal{ALC} \text{ plus the following Operators:} \]

\[ \text{atleast } nr \quad \text{and} \quad \text{atmost } nr \]

\( (r \text{ is a role (relation), } n \text{ is a non-negative integer, no variable!}) \)

**Examples:**

- Car = Vehicle \( \sqcap \) at least 4 has-Wheels
- Metropolis = City \( \sqcap \) at least 1000000 has-Inhabitants
- SmallFamily = Family \( \sqcap \) atmost 2 has-Children

**Semantics:**

\[ (\text{atleast } nr)^3 = \{ x \in \mathcal{D} | |\{ y | (x, y) \in r^3\}| \geq n\} \]

\[ (\text{atmost } nr)^3 = \{ x \in \mathcal{D} | |\{ y | (x, y) \in r^3\}| \leq n\} \]

atleast \( nr \) is the set of objects with at least "r-Successors".

E.G. "atleast 1000000 has-Inhabitants" is the set of objects with at least 10000000 inhabitants.
Definable Operator: $=\,$

$=nr =: \text{atleast } nr \cap \text{atmost } nr$

defines the set of objects with exactly $n$ $r$-successors

$=nr$ is essentially syntactic sugar.

Examples:

Car $=$ Vehicle $\cap = 4$ has-wheels

(Cars are vehicles with exactly 4 wheels).

Peg-Legger $=$ Human $\cap = 1$ has-Leg
Definable: Functional Roles

Example: \( \text{CwSR} = \text{Cars with Sun-Roof} \)
\[ \text{CwSR} = \text{Car} \cap = 1 \text{ has-sun-roof} \]

has-sun-roof is still a relation, but restricted to the set CwSR it is a \textit{total} function.

Partial Functions

Example: \( \text{CwHd} = \text{Computer with at most one harddrive} \)
\[ \text{CwHd} = \text{Computer} \cap \text{atmost 1 has-harddrive} \]

has-harddrive is on the set CwHd a \textit{partial} function

Number Restriction can be used to encode \textit{partial} and \textit{total} functions, but not those mapping to a different data type (e.g. numbers)
Negation

In order to compute the negation normal form (necessary for the Tabelau calculus) one needs the following negation rules:

\[ \neg (\text{atleast } nr) \mapsto \text{atmost}(n - 1)r \]
\[ \neg (\text{atmost } nr) \mapsto \text{atleast}(n + 1)r \]

**Soundness Proofs**: via the semantics of the operators

**Example**:

1: \( \neg (\text{atleast 5 has-Children}) = \text{atmost 4 has-Children} \)
2: \( \neg (\text{atmost 5 has-Children}) = \text{atleast 6 has-Children} \)

1: The complement of the set of objects with at least 5 children is the set of objects with at most 4 children.
2: analogous
Example:
Between $x: \exists \text{has-Child blonde} \land \exists \text{has-Child tall}$

and $x: \text{atleast 2 has-Child}$

there is the following difference:
the two existential quantifiers may denote the same child (blonde and tall), whereas $\text{atleast 2 has-Child}$ guarantees the existence of two different children.

If we give the two children names:

$x \text{has-Child } y_1$ and $x \text{has-Child } y_2$

one must mark in the second case $y_1$ and $y_2$ as different (this is the Unique Name Assumption), and in the first case not.

In the subsequent tableau calculus we assume that it keeps track which constants denote different objects (UNA) and which not.
**atleast-Rule in the Tableau Calculus**

**atleast-Rule:**

\[
\begin{align*}
x & : \text{atleast } nr \\
x \ r \ y_1 \\
\ldots & \quad \text{for } n \text{ new constants, which are marked as different} \\
x \ r \ y_n \\
\end{align*}
\]  
(UNA)
**atmost-Rule**

**atmost-Rule:**

\[ x : \text{atmost } nr \]
\[ x \, r \, y_1 \]
\[ \ldots \quad \text{with } m > n \]
\[ x \, r \, y_m \]

\[ S[y_1/y_2] \quad \ldots \quad S[y_{m-1}/y_m] \]

Every possibility, to make two of the \( y_i \) **which are not marked different**, equal are tried.

**atmost-Inconsistency** Rule

\[ x : \text{atmost } nr \]
\[ x \, r \, y_1 \]
\[ \ldots \quad \text{with } m > n \text{ and all } y_i \text{ are marked different} \]
\[ x \, r \, y_m \]

inkonsistent
Example for the atmost-Rule:

We want to show:

\[(\exists r \, p \land \exists r \, \neg p) \equiv \text{atleast 2 } r\]

Proof by Contradiction:

\[x: \exists r \, p \land \exists r \, \neg p\]
\[x: \neg \text{atleast 2 } r\]

NNF: \[x: \text{atmost 1 } r\]

Start of the Tableau

\[x: \exists r \, p \quad x: \exists r \, \neg p \quad x: \text{atmost 1 } r\]
\[x \, r \, y_1 \quad y_1: p \quad \text{(2 times } \exists\text{-Regel)}\]
\[x \, r \, y_2 \quad y_2: \neg p\]
\[y_1: \neg p \quad \text{(atmost-Rule: } y_2 \text{ replaced by } y_1)\]

Contradiction
Example with at least

We want to show:

\[ \text{atmost 3 } r \subseteq \text{atmost 4 } r \]

Proof by Contradiction:

\[
\begin{align*}
\text{x: atmost 3 } r \\
\text{x: } \neg \text{atmost 4 } r & \quad \text{NNF: } \text{x: at least 5 } r
\end{align*}
\]

Start of the Tableau:

\[
\begin{align*}
\text{x } r \ y_1, \ x r y_2, \ x r y_3, \ x r y_4, \ x r y_5 \quad \text{(atleast-Regel)}
\end{align*}
\]

all are marked different

yields immediately a contradiction with the atmost-Inconsistency Rule
The calculus can be immediately applied to A-Boxes.

It must, however, always be clear, which constants are different.
Example with A-Box

Tom has seen (at most) 2 people. One of them was the murderer, and he was black-haired. It turned out that Curt and Carl were the two people. Curt is blonde and Carl is black-haired.

Who was the murderer?

The A-Box constants are

Tom, Carl, Curt, Murderer

Tom, Carl und Curt are different.
The murderer can be one of them.
A-Box Encoding

Tom : atmost 2 has-seen
Tom : has-seen Carl
Tom : has-seen Curt
Tom : has-seen Murderer
Murderer : black-haired
Curt : \neg black-haired
Carl : black-haired

Murderer \leftrightarrow Curt

atmost-Rule

Tom : has-seen Curt
Curt : black-haired
Curt : \neg black-haired
Carl : black-haired

closed

open

Carl was the Murderer
Qualified Number Restrictions

$\mathcal{ALC}$ plus the following Operators:

- atleast $n \ r \ F$ and atmost $n \ r \ F$

($r$ is a role, $n$ is a number, $F$ an arbitrary $\mathcal{ALC}$-Term.)

Examples:

Person $\cap$ atmost 2 has-Child blonde

= sets of persons with at most two blonde children.

Company $\cap$ atleast 5 has-Customer Car-Industry-Company

= Company with at least 5 customers in the car industry.
Special Cases

Unqualified Number Restrictions are now special cases:

\[
\text{atmost } n \ r = \text{atmost } n \ r \ T.
\]

\[
\text{atleast } n \ r = \text{atleast } n \ r \ T
\]

The quantifiers are also special cases

\[
\exists r F = \text{atleast } 1 r F
\]

and

\[
\forall r F = \neg \exists r \neg F = \neg (\text{atleast } 1 r \neg F) = \text{atmost } 0 r \neg F
\]

Therefore

Quantifiers are no longer necessary!
\[(\text{atleast } n \ r \ F)^3 = \{x \in S_D \mid |\{y \mid (x, y) \in r^3 \text{ and } y \in F^3\}| \geq n\}\]

\[(\text{atmost } n \ r \ F)^3 = \{x \in S_D \mid |\{y \mid (x, y) \in r^3 \text{ and } y \in F^3\}| \leq n\}\]

**Example:**

\[(\text{atleast 2 has-Child Students})^3\]

is the set of objects having at least 2 children which are students
Negation

\[ \neg \text{(atleast } n \text{ r } F) = \text{atmost } (n - 1) \text{ r } F \]
\[ \neg \text{(atmost } n \text{ r } F) = \text{atleast } (n + 1) \text{ r } F \]

Examples

\[ \neg \text{(atleast 3 has-Child Teacher)} \]
\[ = \text{atmost 2 has-Child Teacher} \]

\[ \neg \text{(atmost 3 has-Child Teacher)} \]
\[ = \text{atleast 4 has-Child Teacher} \]
S. Tobies:


For Propositional Logic + qualified Number Restrictions

Idea for $x : \text{atleast } n \land F$

As long as there are not enough $r$-successors of $x$ in $F$, generate a new variable $y$ with $x \land r \land y$, and start for $y$ immediately all relevant case distinctions. E.G. if $x : \text{atmost } m \land r \land G$ holds, generate the cases $y : G$ and $y : \neg G$

(that helps for example in the Metropolis-Example)
Example

\[ \begin{align*}
  x : \text{atleast 3 } r \ p \land \text{atmost 1 } r \ q \land \text{atmost 1 } r \ \neg q \\
  x : \text{atleast 3 } r \ p, x : \text{atmost 1 } r \ q, x : \text{atmost 1 } r \ \neg q
\end{align*} \]

\[ \begin{align*}
  xry_1 & \quad \text{Not enough } r\text{-successors} \\
  y_1 : p & \\
  y_1 : \neg q \\
  y_2 : p & \\
  y_2 : \neg q \\
  y_2 : q & \quad \text{incon.} \\
  y_3 : p & \\
  y_3 : \neg q & \quad \text{incon.}
\end{align*} \]

\[ \begin{align*}
  xry_2 & \quad \text{Not enough } r\text{-successors} \\
  y_2 : p \\
  y_2 : \neg q \\
  y_2 : q & \\
  y_3 : p \\
  y_3 : \neg q & \quad \text{incon.} \\
  y_3 : q & \quad \text{incon.}
\end{align*} \]

\[ \begin{align*}
  xry_3 & \quad \text{Not enough } r\text{-successors} \\
  y_3 : p \\
  y_3 : \neg q \\
  y_3 : q & \\
  y_3 : \neg q & \quad \text{incon. (too many in } \neg q) \\
\end{align*} \]
Rules for atleast and atmost

**atleast-Rule** for a Tableau branch $S$: If
1. $x : \text{atleast } n \ r \ F \in S$
2. $m = |\{y \mid \{xry, y \ F\} \subseteq S\}| < n$ und (not enough $r$-successors)
3. no other rule applicable

then

extend $S$ by $\{xry, y : F, y : H_1, \ldots, y : H_k\}$ ($k = n - m$)

where
1. $\{G_1, \ldots, G_k\} = \{G \mid 'x : \text{atleast } m \ r \ G' \in S \text{ or}
                             'x : \text{atmost } m \ r \ G' \in S\}$ and
   $H_i = G_i \text{ or } H_i = \neg G_i$ (all cases have to be generated)
2. $y$ is a new constant

**atmost-Inconsistency-Rule:**

$S$ is inconsistent

if $x : \text{atmost } n \ r \ F \in S$ and $|\{y \mid \{xry, y:F\} \subseteq S\}| > n$. 
Operators for Rolenames

- Rolenames denote binary relations
- Relations are sets (of tuples)
- Therefore one can apply set-operations to them.
- In addition there are operators which make only sense for relations.
- Every addition of a new operator to \(\mathcal{ALC}\), yields a new variant which requires new rules for the tableau calculus.
Role conjunktion \( \sqcap \)

Example

Persons \( \sqcap \exists (\text{has-Teacher} \sqcap \text{has-Friend}) \) Swiss

Persons having a teacher who is at the same time a friend and who is Swiss.

Companies \( \sqcap \exists (\text{has-Employee} \sqcap \text{has-Lawyer}) \) Women

Companies having an employee who is also their lawyer and who is a woman.
\[(r \cap s)^3 = r^3 \cap s^3\]
\[= \{(x, y) \in \mathcal{I}_D \mid (x, y) \in r^3 \text{ and } (x, y) \in s^3\}\]
Computation Rules

\[ \forall (r \cap s)F \equiv \forall rF \cap \forall sF \]
\[ \exists (r \cap s)F \subseteq \exists rF \cap \exists sF \]
\[
\text{atleast } n (r \cap s) F \subseteq \text{atleast } n r F \cap \text{atleast } n s F \\
\text{atmost } (n + m) (r \cap s) F \equiv \text{atleast } n r F \cap \text{atleast } m s F
\]

Proofs
with the semantics of the operators.

Example:
\[ \exists (\text{has-Friend} \cap \text{has-Neighbour}) \text{ blonde} \]
\[ \subseteq \exists \text{ has-Friend blonde} \cap \exists \text{ has-Neighbour blonde} \]

but not
\[ \exists \text{ has-Friend blond} \cap \exists \text{ has-Neighbour blonde} \]
\[ \subseteq \exists (\text{has-Friend} \cap \text{has-Neighbour}) \text{ blonde} \]
Role Disjunction $\sqcup$

**Examples**

Persons $\cap \forall(\text{has-Child} \sqcup \text{has-Friend})\text{Teacher}$

Persons whose children and friends are all teachers.

Companies $\cap \exists(\text{has-Employee} \sqcup \text{has-Advisor})\text{Member-of-Parliament}$

Companies having an employee or advisor who is a member of parliament
\[(r \uplus s)^3 = r^3 \cup s^3\]
\[= \{(x, y) \in \mathcal{D} | (x, y) \in r^3 \text{ oder } (x, y) \in s^3\}\]
Computation Rules

\[ \forall (r \cup s)F = \forall rF \cap \forall sF \]
\[ \exists (r \cup s)F = \exists rF \cup \exists sF \]

atleast \( n \) \((r \cup s) F \) ??

atmost \( n \) \((r \cup s) F \) \( \sqsubseteq \) atmost \( n \) \( r F \cap \) atmost \( n \) \( s F \)

atmost \((n + m)\) \((r \cup s) F \) \( \sqsubseteq \) atmost \( n \) \( r F \cap \) atmost \( m \) \( s F \)

atleast \( \min(n,m) \) \((r \cup s) F \) \( \sqsupseteq \) atleast \( n \) \( r F \cup \) atleast \( m \) \( s F \)

atleast \( \max(n,m) \) \((r \cup s) F \) \( \sqsupseteq \) atleast \( n \) \( r F \cap \) atleast \( m \) \( s F \)

Proofs

via the semantics of the operators.
Examples

University ⊓ ∀(has-Employee ⊓ has-Professor’) Union-Member
Universities whose employees which are not professors are union members.

House ⊓ ∃(is-Resident ⊓ is-Owner’) Academic
Houses with a resident who is not the owner and who is an academic.
Semantics

\[ r'^3 = \mathcal{S}_D \times \mathcal{S}_D \setminus r^3 = \{ (x, y) \in \mathcal{S}_D \times \mathcal{S}_D \mid (x, y) \notin r^3 \} \]

\((\cap, \cup, \neg')\) is a Boolean Algebra (Propositional Logic). Therefore one can compute role terms formed with, \((\cap, \cup, \neg')\) in the same way as with propositional formulae built with \((\land, \lor, \neg)\).

Example

\[ \forall (r \cap s)' F = \forall (r' \cup s') F \]

Further Rules:

If \( r \subseteq s \) is a tautology then

\[ \forall s F \subseteq \forall r F \]

and

\[ \exists r F \subseteq \exists s F \]
0 and 1

\[ r \cap r' = \bot \]
\[ r \cup r' = \top \]

Therefore:

"\( \bot \)" denotes the empty relation and

"\( \top \)" denotes the universal relation.

Question

What means \( \forall \top F \)?
Example

\[ \text{Person} \cap \exists (\text{has-Child} \circ \text{has-Child}) \text{ Professor} \]
(sets of persons with a grandchild who is professor)

\[ \text{Companies} \cap \forall (\text{produces} \circ \text{sells}) \text{ EU-Staates} \]
(Set of companies producing things which are sold in EU-states only.)

\[ \text{University} \cap \forall (\text{has-Student} \circ \text{has-Friend} \circ \text{lives-in}) \text{ BavarianTowns} \]
(Set of universities all whose students have a friend who lives in a Bavarian Town)
Semantics

\[(r \circ s)^3 = \{(x, z) \in \mathcal{S}_D \times \mathcal{S}_D \mid \exists y \ (x, y) \in r^3 \text{ and } (y, z) \in s^3\}\]
Converse Roles $r^{-1}$

**Example:**

$\forall \text{ has-Child}^{-1} \text{ Teacher}$

$= \{ x \mid \forall y \text{ has-Child}^{-1}(x, y) \Rightarrow \text{Teacher}(y) \}$

$= \{ x \mid \forall y \text{ has-Child}(y, x) \Rightarrow \text{Teacher}(y) \}$

$= \{ x \mid \forall y \text{ has-Parent}(x, y) \Rightarrow \text{Teacher}(y) \}$

$= \text{set of objects whose parents are teachers.}$
Semantics

\[ r^{-13} = \{(y, x) \mid (x, y) \in r^3\} \]

**Examples for converse roles:**

- \( \text{has-Child}^{-1} = \text{has-Parent} \)
- \( \text{is-Part-Of}^{-1} = \text{has-as-Part} \)
- \( \text{is-Employee}^{-1} = \text{is-Employer} \)
- \( \text{owns}^{-1} = \text{belongs} \)
- \( \text{informs}^{-1} = \text{gets-informed} \)
Tableau Calculus: $\mathcal{ALC} + \text{Roleterms}$

Difference to $\mathcal{ALC}$:
Instead of role names there can be complex roleterms.

Rule for the Existential Quantifier:

$$\frac{x: \exists r F}{x \ r \ y \ y:F} \quad (r \text{ is an arbitrary roleterm)}$$

No Difference
Modified Rule for the Universal Quantifier:

\[ S \]
\[ x: \forall r F \]
\[ x \ast y \quad \text{and} \quad S \land x \ast y \vDash x \ast r y \]
\[ y: F \]

i.e. from the actual branch \( S \) as well as \( x \ast y \) one can derive \( x \ast r y \).

**Example:**

\[ x \text{ has-Father } z \]
\[ z \text{ has-Father } y \]
\[ x: \forall (\text{has-Father} \circ \text{has-Father}) \text{ German} \]
\[ y: \text{ German} \]

(y is the grandfather of \( x \), and all his grandfathers are Germans.)
Problem: $S \vDash xry$?

Simple Case:
There is no role composition $\circ$ and no converse operator $^{-1}$.

Then we have:

$S \vDash xry$ iff. \{$s \mid "xsy" \in S\} \cup \{r'\} $ is inconsistent in propositional logic.

In general one can translate $xry$ into formulae of predicate logic and work with predicate logic proof methods.
If there is no role composition or the role complement occurs only in front of role names then $S \vDash xry$ is still decidable, otherwise not!
Role Hierarchies

Specification of subset relations between relations

Examples

has-Daughter ⊆ has-Child
has-Son ⊆ has-Child
talks-with ⊆ communicates-with
phones-with ⊆ communicates-with
buys ⊆ acquires
steals ⊆ acquires

... 

This yields a hierarchy as a tree (or dag).
(without roleterms)

Rule for the existential quantier: as in $\mathcal{ALC}$

Modified rule for the universal quantifier

\[
x: \forall r F \\
x s y \quad \text{and } s \sqsubseteq r \text{ follows from the} \\
y: F \quad \text{hierarchy graph}
\]
Properties of Relations

Binary relations can be

• **transitive**: \( r(x,y) \land r(y,z) \Rightarrow r(x,z) \)
  Example: is-successor

• **reflexive**: \( r(x,x) \)
  Example: is-smaller-than

• **symmetric**: \( r(x,y) \Rightarrow r(y,x) \)
  Example: is-Sibling-of

• **dense**: \( r(x,z) \Rightarrow \exists y \ r(x,y) \land r(y,z) \)
  Example: temporal after-relation

• ...

Some of these properties are treated in the tableau calculus.
Nominales

Nominales are Names for domain elements, used in the T-Box.

**Examples**

Students \( \sqcap \exists \) studies-in\{München,Hamburg\}
(Student studying in Munich or Hamburg)

Students \( \sqcap \exists (\text{has-Friend} \circ \text{has-Name})\)\{Eva\}
(Student having a friend named Eva.)

Persons \( \sqcap \exists \) has-phone-with\{Carl,Curt,Murderer\}
(Persons, having phoned with Carl, Curt or the Murderer)

Men \( \sqcap \) has-Friend : Eva
(Friends of Eva)

Companies \( \sqcap \forall \) has-Employee \circ has-Bank: Postbank
(Companies all whose Employees having the Postbank as their bank.)

employed-by : LMU \( \sqcap \) has-Bank:Postbank
(Employees of the LMU having the Postbank as their bank.)

Due to lack of time we do not consider this logic further
$\text{Examples}$

$\text{Personen} \sqcap \text{age} < 20$
(Persons younger than 20).
Concrete Domain: real Numbers

$\text{Companies} \sqcap (\text{has-Boss} \circ \text{has-official-car} \circ \text{Price}) > 100000$
(Companies whose boss has an official car which is more expensive than 100000).
Concrete Domain: natural numbers

$\text{Cars} \sqcap \text{hight} > \text{width}$
(Cars higher than wide)
Concrete Domain: natural numbers

$\text{Persons} \sqcap \text{first-name} < \text{surename}$
(Persons whose first name is lexicographically smaller than the surname)
Concrete Domain: strings

$\text{Personen} \sqcap \text{has-Father} \circ \text{study-time before has-Mother} \circ \text{studytime}$
(Persons whose fathers have studied before their mothers).
Concrete Domain: Temporal Interval Logic
Concrete Domain

A "concrete domain" $D$ consists of a set $\text{dom}(D)$, a set $\text{pred}(D)$ of predicate symbols for $D$. Every predicate symbol has an arity $n$ and denotes an $n$-place relation in $\text{dom}(D)$.

Examples

$D = \text{integers with predicates } =, <, \leq, >, \geq$

$D = \text{real numbers with predicates } =, <, \leq, >, \geq$

$D = \text{Allen’s Interval Logik}$
($\text{Temporal intervalls, predicates: before, after, meets etc.}$)

$D = \text{Fakts in a relational data base}$
Predicates: relations of a query language.
Concrete Domains are *admissible* if the relevant inference problems can be solved:

Let \( \{P_1, \ldots, P_n\} \subseteq \text{pred}(D) \).

We consider conjunctions \( P_1(x_1, \ldots) \wedge \ldots \wedge P_n(x_n, \ldots) \).

Such conjunctions are *satisfiable* if there is a mapping from the variables \( x_i \) to the elements of \( \text{dom}(D) \) such that the conjunction becomes true.

**Example** \( \text{dom}(D) = \text{real numbers} \)

- \( P_1(x, y) = \exists z \ (x + z^2 = y) \)
- \( P_2(x, y) = x > y \)
- \( P_1(x, y) \) is satisfiable (\( z = \sqrt{y - x} \) is a solution)
- but \( P_1(x, y) \wedge P_2(x, y) \) is not satisfiable.
Definition: admissible

Definition:

A concrete domain is admissible iff

1. The satisfiability problem of the above type is decidable;

2. The set of predicates is closed under negation (it contains for every predicate its negation) and contains a name for dom(D) (e.g. Int)
Formal Syntax of $\text{ALC}(D)$

A is a concept name

$r$ is a role name or a feature name (funktional role)

$f$ is a feature name

$P_i = f_1 \circ \ldots \circ f_n$ is a feature chain

$P$ is a concrete domain predikate.

$C, D$ are concept terms

$$C, D \rightarrow A \mid \top \mid \bot \mid C \cap D \mid C \cup D \mid \neg C \mid \exists r \ C \mid \forall r \ C \mid P(p_1, \ldots, p_n)$$
Semantics of $\mathcal{ALC}(D)$

Semantics of $\mathcal{ALC}$ plus

- $\mathcal{S}_D$ and $\text{dom}(D)$ are disjoint
- feature names are mapped to partial functions. Some (but not all) go into the concrete domain.
- feature chains $f_1 \circ \ldots \circ f_n$ are compositions of functions.
- $P(p_1, \ldots, p_n) = \{ x \in \mathcal{S}_D \mid \text{there are } \{ y_1, \ldots, y_n \} \in \text{dom}(D) \text{ with } p_1^\mathcal{S}(x) = y_1, \ldots, p_n^\mathcal{S}(x) = y_n \text{ und } (y_1, \ldots, y_n) \in P^\mathcal{S} \}$

Be aware:

$(\neg C)^\mathcal{S} = \mathcal{S}_D \setminus C^\mathcal{S}$ and $\text{not}$

$(\neg C)^\mathcal{S} = (\mathcal{S}_D \cup \text{dom}(D)) \setminus C^\mathcal{S}$
Idea of the Tableau Calculus for $\mathcal{ALC}(D)$

The calculus tries to separate the information such that pure D-problems can be extracted which can be decided with an algorithms for D.

**Examples:**

- $x: \text{height} > 2$ becomes $x: \text{height} y, y > 2$
  (\(y > 2\) can now be submitted to an algorithm for inequations)
- $x: \text{has-official-car} \circ \text{price} = 30000$ becomes
  $x \text{ has-official-car}\ y, y \text{ price}\ z, z = 30000$
  (\(z = 30000\) can now be submitted to an algorithm for equations)
- $x: \text{width} > \text{height}$ becomes
  $x \text{ width}\ y, x \text{ height}\ z, y > z$.

In addition one must exploit that partial functions have unique values.

**Example:** from $x \text{ width} y$ and $x \text{ width} z$ follows $y = z$
Example

We show (proof by contradiction)

Cars \cap \text{hight} = 2 \cap \text{width} = 1 \subseteq \text{Cars} \cap \text{hight} > \text{width}

1. \quad x : \text{Cars} \cap \text{hight} = 2 \cap \text{width} = 1
2. \quad x : \neg \text{Cars} \cup \text{hight} \leq \text{width}

3(1) \quad x : \text{Cars}
4(1) \quad x : \text{hight} = 2
5(1) \quad x : \text{width} = 1

6(2) \quad x : \neg \text{Cars} \quad \text{inc.}(6, 3)
7(4) \quad x : \text{hight} y_1
8(4) \quad y_1 = 2
9(5) \quad x : \text{width} = y_2
10(5) \quad y_2 = 1
11(6) \quad x : \text{hight} y_3
12(6) \quad x : \text{width} = y_4
13(6) \quad y_3 \leq y_4
14(7, 11\&9, 12) \quad y_1 \leq y_2 \quad \text{inc.}(14, 8, 10)
Names

There are various combinations of the different extension of $\mathcal{ALC}$. Therefore the following shortnames have been established:

$F$: functional roles

$E$: arbitrary $\exists r F$ allowed

$U$: $\sqcup$ not allowed

$C$: $\neg F$ allowed (for arbitrary $F$)

$S$: $\mathcal{ALC}$ with transitive roles

$H$: role hierarchies

$R$: $\equiv$ for roles, reflexivity, irreflexivity, disjointness of roles allowed

$O$: nominales allowed

$I$: inverse roles allowed

$N$: Unqualified number restrictions allowed

$Q$: Qualified number restrictions allowed.
**Combinations**

$SHIQ$ is $ALC$ plus general number restrictions, transitive and inverse roles.

$SHOIN(D)$ is $ALC$ plus role hierarchies, nominals, inverse roles, restricted number restrictions with the concrete Domain $D$. 
Restricted Systems

There are also systems less expressive than $\mathcal{ALC}$.

$\mathcal{AL}$ (Attributive language) has
atomic negation (only in front of concept names), $\top$, $\forall r F$
restricted number restrictions (no nesting)
(in particular no $\sqcup$)

$\mathcal{FL}$-: $\top$, $\forall r F$ and $\exists r \top$

$\mathcal{EL}$: only $\top$ and $\exists r F$

The reason is that the algorithms become more efficient
(in particular no case distinctions)
Summary

Extensions of $\mathcal{ALC}$
- number restrictions
- role terms
- role hierarchies
- nominales
- concrete domains

Tableau calculus only skimmed.
Many significant optimisations have been developed.