# Decision Procedures for Verification

First-Order Logic (2)

29.11.2016

Viorica Sofronie-Stokkermans

sofronie@uni-koblenz.de

# Until now:

**Syntax** (one-sorted signatures vs. many-sorted signatures)

**Semantics** $\Sigma$-structures and valuations

# Conventions

In what follows we will use the following conventions:

**constants** (0-ary function symbols) are denoted with $a, b, c, d, \ldots$

**function symbols** with arity $\geq 1$ are denoted

- $f, g, h, \ldots$ if the formulae are interpreted into arbitrary algebras
- $+, -, s, \ldots$ if the intended interpretation is into numerical domains

**predicate symbols** with arity 0 are denoted $P, Q, R, S, \ldots$

**predicate symbols** with arity $\geq 1$ are denoted

- $p, q, r, \ldots$ if the formulae are interpreted into arbitrary algebras
- $\leq, \geq, <, >$ if the intended interpretation is into numerical domains

**variables** are denoted $x, y, z, \ldots$

## 2.2 Semantics

To give semantics to a logical system means to define a notion of truth for the formulas. The concept of truth that we will now define for first-order logic goes back to Tarski.

As in the propositional case, we use a two-valued logic with truth values "true" and "false" denoted by 1 and 0, respectively.

# Structures

A $\Sigma$-algebra (also called $\Sigma$-interpretation or $\Sigma$-structure) is a triple

$$\mathcal{A} = (U,\ (f_{\mathcal{A}} : U^n \to U)_{f/n \in \Omega},\ (p_{\mathcal{A}} \subseteq U^m)_{p/m \in \Pi})$$

where $U \neq \emptyset$ is a set, called the universe of $\mathcal{A}$.

Normally, by abuse of notation, we will have $\mathcal{A}$ denote both the algebra and its universe.

By $\Sigma$-Alg we denote the class of all $\Sigma$-algebras.

# Structures

A Σ-algebra (also called Σ-interpretation or Σ-structure) is a triple

$$\mathcal{A} = (U,\ (f_{\mathcal{A}} : U^n \to U)_{f/n \in \Omega},\ (p_{\mathcal{A}} \subseteq U^m)_{p/m \in \Pi})$$

where $U \neq \emptyset$ is a set, called the universe of $\mathcal{A}$.

Normally, by abuse of notation, we will have $\mathcal{A}$ denote both the algebra and its universe.

By Σ-Alg we denote the class of all Σ-algebras.

A many-sorted Σ-algebra (also called Σ-interpretation or Σ-structure), where $\Sigma = (S, \Omega, \Pi)$ is a triple

$$\mathcal{A} = (\{U_s\}_{s \in S},\ (f_{\mathcal{A}} : U_{s_1} \times \ldots \times U_{s_n} \to U_s)_{\substack{f \in \Omega, \\ a(f) = s_1 \ldots s_n \to s}}\ (p_{\mathcal{A}} : U_{s_1} \times \ldots \times U_{s_m} \to \{0, 1\})_{\substack{p \in \Pi \\ a(p) = s_1 \ldots s_m}})$$

where $U_s \neq \emptyset$ is a set, called the universe of $\mathcal{A}$ of sort $s$.

# Assignments

A variable has no intrinsic meaning. The meaning of a variable has to be defined externally (explicitly or implicitly in a given context) by an assignment.

A (variable) assignment, also called a valuation (over a given $\Sigma$-algebra $\mathcal{A}$), is a map $\beta : X \to \mathcal{A}$.

Variable assignments are the semantic counterparts of substitutions.

# Assignments

A variable has no intrinsic meaning. The meaning of a variable has to be defined externally (explicitly or implicitly in a given context) by an assignment.

A (variable) assignment, also called a valuation (over a given $\Sigma$-algebra $\mathcal{A}$), is a map $\beta : X \to \mathcal{A}$.

Variable assignments are the semantic counterparts of substitutions.

**Many-sorted case:**

$\beta = \{\beta_s\}_{s \in S}, \beta_s : X_s \to U_s$

# Value of a Term in $\mathcal{A}$ with Respect to $\beta$

By structural induction we define

$$\mathcal{A}(\beta) : \mathsf{T}_\Sigma(X) \to \mathcal{A}$$

as follows:

$$\mathcal{A}(\beta)(x) = \beta(x), \qquad x \in X$$
$$\mathcal{A}(\beta)(f(s_1, \ldots, s_n)) = f_{\mathcal{A}}(\mathcal{A}(\beta)(s_1), \ldots, \mathcal{A}(\beta)(s_n)), \qquad f/n \in \Omega$$

# Value of a Term in $\mathcal{A}$ with Respect to $\beta$

In the scope of a quantifier we need to evaluate terms with respect to modified assignments. To that end, let $\beta[x \mapsto a] : X \to \mathcal{A}$, for $x \in X$ and $a \in \mathcal{A}$, denote the assignment

$$\beta[x \mapsto a](y) := \begin{cases} a & \text{if } x = y \\ \beta(y) & \text{otherwise} \end{cases}$$

# Truth Value of a Formula in $\mathcal{A}$ with Respect to $\beta$

$\mathcal{A}(\beta) : \mathsf{F}_\Sigma(X) \to \{0, 1\}$ is defined inductively as follows:

$$\begin{aligned}
\mathcal{A}(\beta)(\bot) &= 0 \\
\mathcal{A}(\beta)(\top) &= 1 \\
\mathcal{A}(\beta)(p(s_1, \dots, s_n)) = 1 &\quad\Leftrightarrow\quad (\mathcal{A}(\beta)(s_1), \dots, \mathcal{A}(\beta)(s_n)) \in p_\mathcal{A} \\
\mathcal{A}(\beta)(s \approx t) = 1 &\quad\Leftrightarrow\quad \mathcal{A}(\beta)(s) = \mathcal{A}(\beta)(t) \\
\mathcal{A}(\beta)(\neg F) = 1 &\quad\Leftrightarrow\quad \mathcal{A}(\beta)(F) = 0 \\
\mathcal{A}(\beta)(F \rho G) &= \mathsf{B}_\rho(\mathcal{A}(\beta)(F), \mathcal{A}(\beta)(G)) \\
&\phantom{=}\ \text{with } \mathsf{B}_\rho \text{ the Boolean function associated with } \rho \\
\mathcal{A}(\beta)(\forall x F) &= \min_{a \in U} \{\mathcal{A}(\beta[x \mapsto a])(F)\} \\
\mathcal{A}(\beta)(\exists x F) &= \max_{a \in U} \{\mathcal{A}(\beta[x \mapsto a])(F)\}
\end{aligned}$$

# Example

The "Standard" Interpretation for Peano Arithmetic:

$$
\begin{array}{rcl}
U_{\mathbb{N}} & = & \{0, 1, 2, \ldots\} \\
0_{\mathbb{N}} & = & 0 \\
s_{\mathbb{N}} & : & n \mapsto n + 1 \\
+_{\mathbb{N}} & : & (n, m) \mapsto n + m \\
*_{\mathbb{N}} & : & (n, m) \mapsto n * m \\
\leq_{\mathbb{N}} & = & \{(n, m) \mid n \text{ less than or equal to } m\} \\
<_{\mathbb{N}} & = & \{(n, m) \mid n \text{ less than } m\}
\end{array}
$$

Note that $\mathbb{N}$ is just one out of many possible $\Sigma_{PA}$-interpretations.

# Example

Values over $\mathbb{N}$ for Sample Terms and Formulas:

Under the assignment $\beta : x \mapsto 1, y \mapsto 3$ we obtain

$$
\begin{aligned}
\mathbb{N}(\beta)(s(x) + s(0)) &= 3 \\
\mathbb{N}(\beta)(x + y \approx s(y)) &= 1 \\
\mathbb{N}(\beta)(\forall x, y(x + y \approx y + x)) &= 1 \\
\mathbb{N}(\beta)(\forall z\ z \leq y) &= 0 \\
\mathbb{N}(\beta)(\forall x \exists y\ x < y) &= 1
\end{aligned}
$$

## 2.3 Models, Validity, and Satisfiability

$F$ is valid in $\mathcal{A}$ under assignment $\beta$:

$$\mathcal{A}, \beta \models F \quad :\Leftrightarrow \quad \mathcal{A}(\beta)(F) = 1$$

$F$ is valid in $\mathcal{A}$ ($\mathcal{A}$ is a model of $F$):

$$\mathcal{A} \models F \quad :\Leftrightarrow \quad \mathcal{A}, \beta \models F, \text{ for all } \beta \in X \to U_{\mathcal{A}}$$

$F$ is valid (or is a tautology):

$$\models F \quad :\Leftrightarrow \quad \mathcal{A} \models F, \text{ for all } \mathcal{A} \in \Sigma\text{-alg}$$

$F$ is called satisfiable iff there exist $\mathcal{A}$ and $\beta$ such that $\mathcal{A}, \beta \models F$.
Otherwise $F$ is called unsatisfiable.

# Substitution Lemma

The following propositions, to be proved by structural induction, hold for all $\Sigma$-algebras $\mathcal{A}$, assignments $\beta$, and substitutions $\sigma$.

**Lemma 2.3:**

For any $\Sigma$-term $t$

$$\mathcal{A}(\beta)(t\sigma) = \mathcal{A}(\beta \circ \sigma)(t),$$

where $\beta \circ \sigma : X \to \mathcal{A}$ is the assignment $\beta \circ \sigma(x) = \mathcal{A}(\beta)(x\sigma)$.

**Proposition 2.4:**

For any $\Sigma$-formula $F$, $\mathcal{A}(\beta)(F\sigma) = \mathcal{A}(\beta \circ \sigma)(F)$.

# Substitution Lemma

**Corollary 2.5:**

$\mathcal{A}, \beta \models F\sigma \quad \Leftrightarrow \quad \mathcal{A}, \beta \circ \sigma \models F$

These theorems basically express that the syntactic concept of substitution corresponds to the semantic concept of an assignment.

# Entailment and Equivalence

$F$ entails (implies) $G$ (or $G$ is a consequence of $F$), written $F \models G$

$:\Leftrightarrow$ for all $\mathcal{A} \in \Sigma$-alg and $\beta \in X \to U_{\mathcal{A}}$,
whenever $\mathcal{A}, \beta \models F$ then $\mathcal{A}, \beta \models G$.

$F$ and $G$ are called equivalent

$:\Leftrightarrow$ for all $\mathcal{A} \in \Sigma$-alg und $\beta \in X \to U_{\mathcal{A}}$ we have
$\mathcal{A}, \beta \models F \quad \Leftrightarrow \quad \mathcal{A}, \beta \models G$.

# Entailment and Equivalence

**Proposition 2.6:**
$F$ entails $G$ iff $(F \rightarrow G)$ is valid

**Proposition 2.7:**
$F$ and $G$ are equivalent iff $(F \leftrightarrow G)$ is valid.

Extension to sets of formulas $N$ in the "natural way", e.g., $N \models F$

$:\Leftrightarrow$ for all $\mathcal{A} \in \Sigma$-alg and $\beta \in X \rightarrow U_{\mathcal{A}}$:
if $\mathcal{A}, \beta \models G$, for all $G \in N$, then $\mathcal{A}, \beta \models F$.

# Validity vs. Unsatisfiability

Validity and unsatisfiability are just two sides of the same medal as explained by the following proposition.

**Proposition 2.8:**

$$F \text{ valid} \quad \Leftrightarrow \quad \neg F \text{ unsatisfiable}$$

Hence in order to design a theorem prover (validity checker) it is sufficient to design a checker for unsatisfiability.

$Q$: In a similar way, entailment $N \models F$ can be reduced to unsatisfiability. How?

# Theory of a Structure

Let $\mathcal{A} \in \Sigma$-alg. The (first-order) theory of $\mathcal{A}$ is defined as

$$Th(\mathcal{A}) = \{G \in \mathsf{F}_\Sigma(X) \mid \mathcal{A} \models G\}$$

Problem of axiomatizability:

For which structures $\mathcal{A}$ can one axiomatize $Th(\mathcal{A})$, that is, can one write down a formula $F$ (or a recursively enumerable set $F$ of formulas) such that

$$Th(\mathcal{A}) = \{G \mid F \models G\}?$$

Analogously for sets of structures.

# Two Interesting Theories

Let $\Sigma_{Pres} = (\{0/0, s/1, +/2\},\ \emptyset)$ and $\mathbb{Z}_+ = (\mathbb{Z}, 0, s, +)$ its standard interpretation on the integers.
$Th(\mathbb{Z}_+)$ is called Presburger arithmetic (M. Presburger, 1929).
(There is no essential difference when one, instead of $\mathbb{Z}$, considers the natural numbers $\mathbb{N}$ as standard interpretation.)

Presburger arithmetic is decidable in 3EXPTIME (D. Oppen, JCSS, 16(3):323–332, 1978), and in 2EXPSPACE, using automata-theoretic methods (and there is a constant $c \geq 0$ such that $Th(\mathbb{Z}_+) \notin \text{NTIME}(2^{2^{cn}})$).

# Two Interesting Theories

However, $\mathbb{N}_* = (\mathbb{N}, 0, s, +, *)$, the standard interpretation of $\Sigma_{PA} = (\{0/0, s/1, +/2, */2\}, \emptyset)$, has as theory the so-called Peano arithmetic which is undecidable, not even recursively enumerable.

*Note:* The choice of signature can make a big difference with regard to the computational complexity of theories.

# Logical theories

## Syntactic view

first-order theory: given by a set $\mathcal{F}$ of (closed) first-order $\Sigma$-formulae.

the models of $\mathcal{F}$: $\quad \mathsf{Mod}(\mathcal{F}) = \{\mathcal{A} \in \Sigma\text{-alg} \mid \mathcal{A} \models G, \text{ for all } G \text{ in } \mathcal{F}\}$

## Semantic view

given a class $\mathcal{M}$ of $\Sigma$-algebras

the first-order theory of $\mathcal{M}$: $\mathsf{Th}(\mathcal{M}) = \{G \in F_\Sigma(X) \text{ closed} \mid \mathcal{M} \models G\}$

# Theories

$\mathcal{F}$ set of (closed) first-order formulae

$$\mathrm{Mod}(\mathcal{F}) = \{A \in \Sigma\text{-alg} \mid \mathcal{A} \models G, \text{ for all } G \text{ in } \mathcal{F}\}$$

$\mathcal{M}$ class of $\Sigma$-algebras

$$\mathrm{Th}(\mathcal{M}) = \{G \in F_\Sigma(X) \text{ closed} \mid \mathcal{M} \models G\}$$

$\mathrm{Th}(\mathrm{Mod}(\mathcal{F}))$ the set of formulae true in all models of $\mathcal{F}$
represents exactly the set of consequences of $\mathcal{F}$

# Theories

$\mathcal{F}$ set of (closed) first-order formulae

$$\mathrm{Mod}(\mathcal{F}) = \{A \in \Sigma\text{-alg} \mid \mathcal{A} \models G, \text{ for all } G \text{ in } \mathcal{F}\}$$

$\mathcal{M}$ class of $\Sigma$-algebras

$$\mathrm{Th}(\mathcal{M}) = \{G \in F_\Sigma(X) \text{ closed} \mid \mathcal{M} \models G\}$$

$\mathrm{Th}(\mathrm{Mod}(\mathcal{F}))$ the set of formulae true in all models of $\mathcal{F}$
represents exactly the set of consequences of $\mathcal{F}$

**Note:** $\mathcal{F} \subseteq \mathrm{Th}(\mathrm{Mod}(\mathcal{F}))$ (typically strict)

$\mathcal{M} \subseteq \mathrm{Mod}(\mathrm{Th}(\mathcal{M}))$ (typically strict)

# Examples

## 1. Groups

Let $\Sigma = (\{e/0, */2, i/1\}, \emptyset)$

Let $\mathcal{F}$ consist of all (universally quantified) group axioms:

$$\begin{array}{ll} \forall x, y, z & x * (y * z) \approx (x * y) * z \\ \forall x & x * i(x) \approx e \quad \wedge \quad i(x) * x \approx e \\ \forall x & x * e \approx x \quad \wedge \quad e * x \approx x \end{array}$$

Every group $\mathcal{G} = (G, e_G, *_G, i_G)$ is a model of $\mathcal{F}$

Mod($\mathcal{F}$) is the class of all groups

$\mathcal{F} \subset \mathrm{Th}(\mathrm{Mod}(\mathcal{F}))$

# Examples

## 2. Linear (positive)integer arithmetic

Let $\Sigma = (\{0/0, s/1, +/2\}, \{\leq /2\})$

Let $\mathbb{Z}_{+} = (\mathbb{Z}, 0, s, +, \leq)$ the standard interpretation of integers.

$\{\mathbb{Z}_{+}\} \subset \mathsf{Mod}(\mathsf{Th}(\mathbb{Z}_{+}))$

## 3. Uninterpreted function symbols

Let $\Sigma = (\Omega, \Pi)$ be arbitrary

Let $\mathcal{M} = \Sigma$-alg be the class of all $\Sigma$-structures

The theory of uninterpreted function symbols is Th($\Sigma$-alg) the family of all first-order formulae which are true in all $\Sigma$-algebras.

# Examples

## 4. Lists

Let $\Sigma = (\{\text{car}/1, \text{cdr}/1, \text{cons}/2\}, \emptyset)$

Let $\mathcal{F}$ be the following set of list axioms:

$$\begin{aligned} \text{car}(\text{cons}(x, y)) &\approx x \\ \text{cdr}(\text{cons}(x, y)) &\approx y \\ \text{cons}(\text{car}(x), \text{cdr}(x)) &\approx x \end{aligned}$$

$\text{Mod}(\mathcal{F})$ class of all models of $\mathcal{F}$

$\text{Th}_{\text{Lists}} = \text{Th}(\text{Mod}(\mathcal{F}))$ theory of lists (axiomatized by $\mathcal{F}$)

## 2.4 Algorithmic Problems

**Validity($F$):** $\models F$ ?

**Satisfiability($F$):** $F$ satisfiable?

**Entailment($F$,$G$):** does $F$ entail $G$?

**Model($\mathcal{A}$,$F$):** $\mathcal{A} \models F$?

**Solve($\mathcal{A}$,$F$):** find an assignment $\beta$ such that $\mathcal{A}, \beta \models F$

**Solve($F$):** find a substitution $\sigma$ such that $\models F\sigma$

**Abduce($F$):** find $G$ with "certain properties" such that $G$ entails $F$

# Decidability/Undecidability

In 1931, Gödel published his incompleteness theorems in

> "Über formal unentscheidbare Sätze der
> Principia Mathematica und verwandter Systeme"

(in English "On Formally Undecidable Propositions of Principia Mathematica and Related Systems").

He proved for any computable axiomatic system that is powerful enough to describe the arithmetic of the natural numbers (e.g. the Peano axioms or Zermelo-Fraenkel set theory with the axiom of choice), that:

- If the system is consistent, it cannot be complete.
- The consistency of the axioms cannot be proven within the system.

# Decidability/Undecidability

These theorems ended a half-century of attempts, beginning with the work of Frege and culminating in Principia Mathematica and Hilbert's formalism, to find a set of axioms sufficient for all mathematics.

The incompleteness theorems also imply that not all mathematical questions are computable.

## Consequences of Gödel's Famous Theorems

1. For most signatures $\Sigma$, validity is undecidable for $\Sigma$-formulas. (One can easily encode Turing machines in most signatures.)
2. For each signature $\Sigma$, the set of valid $\Sigma$-formulas is recursively enumerable.
   (We will prove this by giving complete deduction systems.)
3. For $\Sigma = \Sigma_{PA}$ and $\mathbb{N}_* = (\mathbb{N}, 0, s, +, *)$, the theory $Th(\mathbb{N}_*)$ is not recursively enumerable.

These undecidability results motivate the study of subclasses of formulas (fragments) of first-order logic

$Q$: Can you think of any fragments of first-order logic for which validity is decidable?

# Some Decidable Fragments/Problems

**Validity/Satisfiability/Entailment:** Some decidable fragments:

- Variable-free formulas without equality:
  satisfiability is NP-complete. (why?)
- Variable-free Horn clauses (clauses with at most one positive atom): entailment is decidable in linear time.
- Monadic class: no function symbols, all predicates unary; validity is NEXPTIME-complete.
- Q: Other decidable fragments of FOL (with variables)?
  Which methods for proving decidability?

**Decidable problems.**

Finite model checking is decidable in time polynomial in the size of the structure and the formula.

# Goals

### Identify:

- decidable fragments of first-order logic
- fragments of FOL for which satisfiability checking is easy

### Methods:

- Theoretical methods (automata theory, finite model property)
- Adjust automated reasoning techniques
  (e.g. to obtaining efficient decision procedures)

  Extend methods for automated reasoning in propositional logic?

    Instantiation/reduction to propositional logic

    Extend the resolution calculus for first-order logic

# Goals

Extend methods for automated reasoning in propositional logic?

Instantiation/reduction to propositional logic

Extend the resolution calculus for first-order logic

**Ingredients:**

- Give a method for translating formulae to clause form
- Regard formulae with variables as a set of all their instances (where variables are instantiated with ground terms)
    - Show that only certain instances are needed

      $\mapsto$ reduction to propositional logic
    - Finite encoding of infinitely many inferences

      $\mapsto$ resolution for first-order logic

## 2.5 Normal Forms and Skolemization

Study of normal forms motivated by

- reduction of logical concepts,
- efficient data structures for theorem proving.

The main problem in first-order logic is the treatment of quantifiers. The subsequent normal form transformations are intended to eliminate many of them.

# Prenex Normal Form

Prenex formulas have the form

$$Q_1 x_1 \ldots Q_n x_n \; F,$$

where $F$ is quantifier-free and $Q_i \in \{\forall, \exists\}$;
we call $Q_1 x_1 \ldots Q_n x_n$ the quantifier prefix and $F$ the matrix of the formula.

# Prenex Normal Form

Computing prenex normal form by the rewrite relation $\Rightarrow_P$:

$$
\begin{array}{rcll}
(F \leftrightarrow G) & \Rightarrow_P & (F \rightarrow G) \wedge (G \rightarrow F) & \\
\neg Q x F & \Rightarrow_P & \overline{Q} x \neg F & (\neg Q) \\
(Q x F \ \rho \ G) & \Rightarrow_P & Q y (F[y/x] \ \rho \ G), \ y \text{ fresh}, \ \rho \in \{\wedge, \vee\} & \\
(Q x F \rightarrow G) & \Rightarrow_P & \overline{Q} y (F[y/x] \rightarrow G), \ y \text{ fresh} & \\
(F \ \rho \ Q x G) & \Rightarrow_P & Q y (F \ \rho \ G[y/x]), \ y \text{ fresh}, \ \rho \in \{\wedge, \vee, \rightarrow\} &
\end{array}
$$

Here $\overline{Q}$ denotes the quantifier dual to $Q$, i.e., $\overline{\forall} = \exists$ and $\overline{\exists} = \forall$.

# Example

$$F := (\forall x((p(x) \vee q(x, y)) \wedge \exists z\, r(x, y, z))) \rightarrow ((p(z) \wedge q(x, z)) \wedge \forall z\, r(z, x, y))$$

# Example

$$F := (\forall x((p(x) \vee q(x,y)) \wedge \exists z\, r(x,y,z))) \rightarrow ((p(z) \wedge q(x,z)) \wedge \forall z\, r(z,x,y))$$

$$\Rightarrow_P \ \exists x' \, ((p(x') \vee q(x',y)) \wedge \exists z\, r(x',y,z)) \rightarrow ((p(z) \wedge q(x,z)) \wedge \forall z\, r(z,x,y))$$

# Example

$$F := (\forall x((p(x) \vee q(x,y)) \wedge \exists z\, r(x,y,z))) \rightarrow ((p(z) \wedge q(x,z)) \wedge \forall z\, r(z,x,y))$$

$$\Rightarrow_P \;\; \exists x'((p(x') \vee q(x',y)) \wedge \exists z\, r(x',y,z)) \rightarrow ((p(z) \wedge q(x,z)) \wedge \forall z\, r(z,x,y))$$

$$\Rightarrow_P \;\; \exists x'(\exists z'((p(x') \vee q(x',y)) \wedge r(x',y,z'))) \rightarrow ((p(z) \wedge q(x,z)) \wedge \forall z\, r(z,x,y$$

# Example

$$F := (\forall x((p(x) \vee q(x,y)) \wedge \exists z\, r(x,y,z))) \rightarrow ((p(z) \wedge q(x,z)) \wedge \forall z\, r(z,x,y))$$

$$\Rightarrow_P \;\; \exists x'((p(x') \vee q(x',y)) \wedge \exists z\, r(x',y,z))) \rightarrow ((p(z) \wedge q(x,z)) \wedge \forall z\, r(z,x,y))$$

$$\Rightarrow_P \;\; \exists x'(\exists z'((p(x') \vee q(x',y)) \wedge r(x',y,z'))) \rightarrow ((p(z) \wedge q(x,z)) \wedge \forall z\, r(z,x,y$$

$$\Rightarrow_P \;\; \exists x' \forall z' (((p(x') \vee q(x',y)) \wedge r(x',y,z'))) \rightarrow ((p(z) \wedge q(x,z)) \wedge \forall z\, r(z,x,y$$

# Example

$$F := (\forall x((p(x) \vee q(x,y)) \wedge \exists z\, r(x,y,z))) \rightarrow ((p(z) \wedge q(x,z)) \wedge \forall z\, r(z,x,y))$$

$$\Rightarrow_P \;\; \exists x'((p(x') \vee q(x',y)) \wedge \exists z\, r(x',y,z))) \rightarrow ((p(z) \wedge q(x,z)) \wedge \forall z\, r(z,x,y))$$

$$\Rightarrow_P \;\; \exists x'(\exists z'((p(x') \vee q(x',y)) \wedge r(x',y,z'))) \rightarrow ((p(z) \wedge q(x,z)) \wedge \forall z\, r(z,x,y$$

$$\Rightarrow_P \;\; \exists x' \forall z'\, ((p(x') \vee q(x',y)) \wedge r(x',y,z')) \rightarrow ((p(z) \wedge q(x,z)) \wedge \forall z\, r(z,x,y))$$

$$\Rightarrow_P \;\; \exists x' \forall z'\, ((p(x') \vee q(x',y)) \wedge r(x',y,z')) \rightarrow \forall z''((p(z) \wedge q(x,z)) \wedge r(z'',x,$$

# Example

$$F := (\forall x((p(x) \vee q(x,y)) \wedge \exists z\, r(x,y,z))) \to ((p(z) \wedge q(x,z)) \wedge \forall z\, r(z,x,y))$$

$$\Rightarrow_P \quad \exists x'((p(x') \vee q(x',y)) \wedge \exists z\, r(x',y,z))) \to ((p(z) \wedge q(x,z)) \wedge \forall z\, r(z,x,y))$$

$$\Rightarrow_P \quad \exists x'(\exists z'((p(x') \vee q(x',y)) \wedge r(x',y,z'))) \to ((p(z) \wedge q(x,z)) \wedge \forall z\, r(z,x,y$$

$$\Rightarrow_P \quad \exists x'\forall z'\,((p(x') \vee q(x',y)) \wedge r(x',y,z')) \to ((p(z) \wedge q(x,z)) \wedge \forall z\, r(z,x,y))$$

$$\Rightarrow_P \quad \exists x'\forall z'\,((p(x') \vee q(x',y)) \wedge r(x',y,z')) \to \forall z''((p(z) \wedge q(x,z)) \wedge r(z'',x,$$

$$\Rightarrow_P \quad \exists x'\forall z'\forall z''(((p(x') \vee q(x',y)) \wedge r(x',y,z')) \to ((p(z) \wedge q(x,z)) \wedge r(z'',x,$$

# Skolemization

**Intuition:** remove $\exists y$.

For this:

- we introduce a concrete choice function $\mathrm{sk}_y$ computing $y$ from all the arguments $y$ depends on (i.e. from all variables $x_1, \dots, x_n$ which occur universally quantified before $y$ in the quantifier prefix)
- We replace $y$ with $\mathrm{sk}_y(x_1, \dots, x_n)$ everywhere in the scope of $\exists y$.

Transformation $\Rightarrow_S$ (to be applied outermost, *not* in subformulas):

$$\forall x_1, \dots, x_n \exists y F \quad \Rightarrow_S \quad \forall x_1, \dots, x_n F[\mathrm{sk}_y(x_1, \dots, x_n)/y]$$

where $\mathrm{sk}_y/n$ is a new function symbol (Skolem function).

# Skolemization

**Together:** $F \overset{*}{\Rightarrow}_P \underbrace{G}_{\text{prenex}} \overset{*}{\Rightarrow}_S \underbrace{H}_{\text{prenex, no } \exists}$

**Theorem 2.9:**

Let $F$, $G$, and $H$ as defined above and closed. Then

(i) $F$ and $G$ are equivalent.

(ii) $H \models G$ but the converse is not true in general.

(iii) $G$ satisfiable (wrt. $\Sigma$-alg) $\Leftrightarrow$ $H$ satisfiable (wrt. $\Sigma'$-Alg) where $\Sigma' = (\Omega \cup SKF, \Pi)$, if $\Sigma = (\Omega, \Pi)$.

# Clausal Normal Form (Conjunctive Normal Form)

$$
\begin{array}{rcl}
(F \leftrightarrow G) & \Rightarrow_K & (F \rightarrow G) \wedge (G \rightarrow F) \\
(F \rightarrow G) & \Rightarrow_K & (\neg F \vee G) \\
\neg(F \vee G) & \Rightarrow_K & (\neg F \wedge \neg G) \\
\neg(F \wedge G) & \Rightarrow_K & (\neg F \vee \neg G) \\
\neg\neg F & \Rightarrow_K & F \\
(F \wedge G) \vee H & \Rightarrow_K & (F \vee H) \wedge (G \vee H) \\
(F \wedge \top) & \Rightarrow_K & F \\
(F \wedge \bot) & \Rightarrow_K & \bot \\
(F \vee \top) & \Rightarrow_K & \top \\
(F \vee \bot) & \Rightarrow_K & F
\end{array}
$$

These rules are to be applied modulo associativity and commutativity of $\wedge$ and $\vee$. The first five rules, plus the rule $(\neg Q)$, compute the negation normal form (NNF) of a formula.

# The Complete Picture

$$
\begin{array}{llll}
F & \Rightarrow^*_P & Q_1 y_1 \ldots Q_n y_n \; G & (G \text{ quantifier-free}) \\
& \Rightarrow^*_S & \forall x_1, \ldots, x_m \; H & (m \leq n,\; H \text{ quantifier-free}) \\
& \Rightarrow^*_K & \underbrace{\underbrace{\forall x_1, \ldots, x_m}_{\text{leave out}} \; \bigwedge_{i=1}^{k} \underbrace{\bigvee_{j=1}^{n_i} L_{ij}}_{\text{clauses } C_i}}_{F'} &
\end{array}
$$

$N = \{C_1, \ldots, C_k\}$ is called the clausal (normal) form (CNF) of $F$.
*Note:* the variables in the clauses are implicitly universally quantified.

**Theorem 2.10:**
Let $F$ be closed. Then $F' \models F$. (The converse is not true in general.)

**Theorem 2.11:**
Let $F$ be closed. Then $F$ is satisfiable iff $F'$ is satisfiable iff $N$ is satisfiable

# Example

**Given:** $\exists u \forall w (\exists x (p(w, x, u) \lor \forall y (q(w, x, y) \land \exists z\, r(y, z))))$

# Example

**Given:** $\exists u \forall w (\exists x (p(w, x, u) \vee \forall y\, (q(w, x, y) \wedge \exists z\, r(y, z))))$

**Prenex Normal Form:**

$$\Rightarrow^{*}_{P} \; \exists u \forall w \exists x \forall y \exists z ((p(w, x, u) \vee (q(w, x, y) \wedge r(y, z))))$$

# Example

**Given:** $\exists u \forall w (\exists x (p(w, x, u) \vee \forall y \, (q(w, x, y) \wedge \exists z \, r(y, z))))$

**Prenex Normal Form:**

$$\Rightarrow^*_P \ \exists u \forall w \exists x \forall y \exists z ((p(w, x, u) \vee (q(w, x, y) \wedge r(y, z))))$$

**Skolemisation:**

$$\Rightarrow^*_S \ \forall w \forall y ((p(w, \mathrm{sk}_x(w), \mathrm{sk}_u) \vee (q(w, \mathrm{sk}_x(w), y) \wedge r(y, \mathrm{sk}_z(w, y)))))$$

# Example

**Given:** $\exists u \forall w (\exists x (p(w,x,u) \vee \forall y\, (q(w,x,y) \wedge \exists z\, r(y,z))))$

**Prenex Normal Form:**

$$\Rightarrow^*_P \ \exists u \forall w \exists x \forall y \exists z ((p(w,x,u) \vee (q(w,x,y) \wedge r(y,z))))$$

**Skolemisation:**

$$\Rightarrow^*_S \ \forall w \forall y ((p(w,\mathrm{sk}_x(w),\mathrm{sk}_u) \vee (q(w,\mathrm{sk}_x(w),y) \wedge r(y,\mathrm{sk}_z(w,y)))))$$

**Clause normal form:**

$$\Rightarrow^*_K \ \forall w \forall y [(p(w,\mathrm{sk}_x(w),\mathrm{sk}_u) \vee q(w,\mathrm{sk}_x(w),y)) \wedge (p(w,\mathrm{sk}_x(w),\mathrm{sk}_u) \vee r(y,\mathrm{sk}_y(w,y)))]$$

**Set of clauses:**

$$\{p(w,\mathrm{sk}_x(w),\mathrm{sk}_u) \vee q(w,\mathrm{sk}_x(w),y), \quad p(w,\mathrm{sk}_x(w),\mathrm{sk}_u) \vee r(y,\mathrm{sk}_y(w,y))\}$$

# Optimization

Here is lots of room for optimization since we only can preserve satisfiability anyway:

- size of the CNF exponential when done naively;
- want to preserve the original formula structure;
- want small arity of Skolem functions.