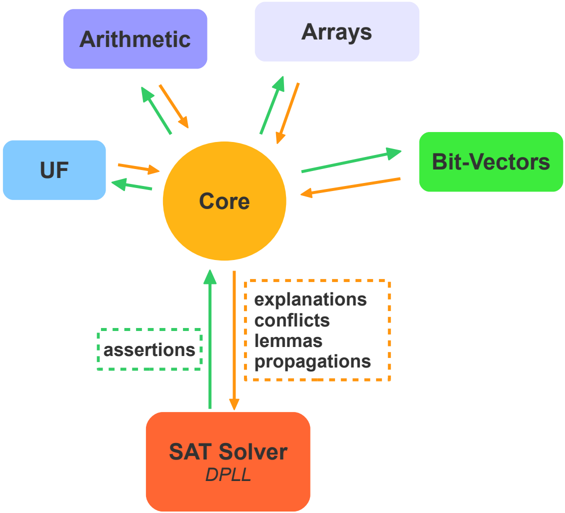
# Logical reasoning and programming

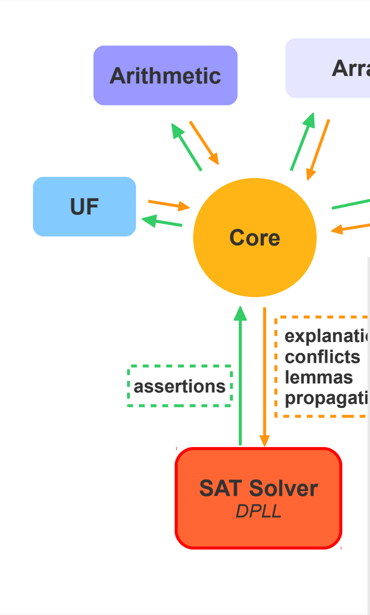
## SMT (cont'd) and quantifiers in FOL

Karel Chvalovský

CIIRC CTU

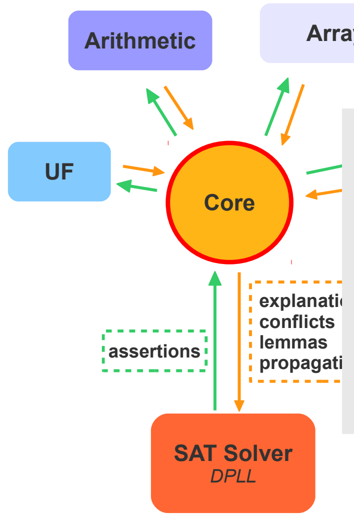
Parts of this presentation are significantly based on materials from recent SAT/SMT Summer Schools and $SC^2$ Summer School.


Arithmetic
Arrays
UF
Core
Bit-Vectors
explanations
conflicts
lemmas
propagations
assertions
SAT Solver
DPLL



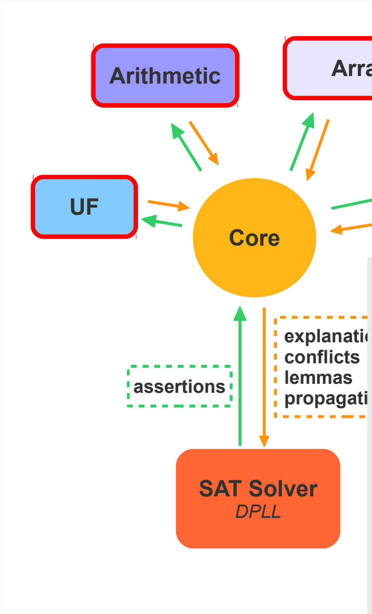

## SAT Solver

- Only sees Boolean skeleton of problem
- Builds partial model by assigning truth values to literals
- Sends these literals to the core as assertions




## Core

- Sends each assertion to the appropriate theory
- Sends deduced literals to other theories/SAT solver
- Handles theory combination




## Theory Solvers

- Decide *T*-satisfiability of a conjunction of theory literals
- Incremental
- Backtrackable
- Conflict Generation
- Theory Propagation

# Combining theories

Instead of solving satisfiability of a formula $\varphi$ in a theory $\mathcal{T}$, we can also try to solve the same problem for a theory that is created as a union of more theories $\mathcal{T}_1 \cup \cdots \cup \mathcal{T}_n$, where we already have solvers for all individual $\mathcal{T}_i$, for $1 \leq i \leq n$. For example, we can combine $\mathcal{T}_{LRA}$ and $\mathcal{T}_{UF}$.

We could develop a special solver for the new theory $\mathcal{T}_1 \cup \cdots \cup \mathcal{T}_n$, or attempt to combine individual solvers together in a uniform way by

- separating reasoning for individual theories (purification),
- exchanging entailed equalities between solvers (equality propagation).

If individual theories remain satisfiable after exchanging all entailed equalitites between solvers (propagation), then the combination is also satisfiable, otherwise it is unsatisfiable.

Consider the following set of literals over $T_{\mathrm{LRA}} \cup T_{\mathrm{UF}}$ ($T_{\mathrm{LRA}}$, linear real arithmetic):

$$\begin{aligned} f(f(x) - f(y)) &= a \\ f(0) &> a + 2 \\ x &= y \end{aligned}$$

Consider the following set of literals over $T_{\mathrm{LRA}} \cup T_{\mathrm{UF}}$ ($T_{\mathrm{LRA}}$, linear real arithmetic):

$$\begin{aligned} f(f(x) - f(y)) &= a \\ f(0) &> a + 2 \\ x &= y \end{aligned}$$

**First step:** *purify* literals so that each belongs to a single theory

Consider the following set of literals over $T_{\mathrm{LRA}} \cup T_{\mathrm{UF}}$ ($T_{\mathrm{LRA}}$, linear real arithmetic):

$$
\begin{aligned}
f(f(x) - f(y)) &= a \\
f(0) &> a + 2 \\
x &= y
\end{aligned}
$$

**First step:** *purify* literals so that each belongs to a single theory

$$
f(f(x) - f(y)) = a \implies \begin{array}{l} f(e_1) = a \\ e_1 = f(x) - f(y) \end{array} \implies \begin{array}{l} f(e_1) = a \\ e_1 = e_2 - e_3 \\ e_2 = f(x) \\ e_3 = f(y) \end{array}
$$

Consider the following set of literals over $T_{\mathrm{LRA}} \cup T_{\mathrm{UF}}$ ($T_{\mathrm{LRA}}$, linear real arithmetic):

$$\begin{aligned} f(f(x) - f(y)) &= a \\ f(0) &> a + 2 \\ x &= y \end{aligned}$$

**First step:** *purify* literals so that each belongs to a single theory

$$f(0) > a + 2 \implies \begin{array}{c} f(e_4) > a + 2 \\ e_4 = 0 \end{array} \implies \begin{array}{c} f(e_4) = e_5 \\ e_4 = 0 \\ e_5 > a + 2 \end{array}$$

**Second step:** exchange entailed *interface equalities*, equalities over shared constants $e_1, e_2, e_3, e_4, e_5, a$

| $L_1$ | $L_2$ |
|---|---|
| $f(e_1) = a$ | $e_2 - e_3 = e_1$ |
| $f(x) = e_2$ | $e_4 = 0$ |
| $f(y) = e_3$ | $e_5 > a + 2$ |
| $f(e_4) = e_5$ | |
| $x = y$ | |

**Second step:** exchange entailed *interface equalities*, equalities over shared constants $e_1, e_2, e_3, e_4, e_5, a$

| $L_1$ | $L_2$ |
|---|---|
| $f(e_1) = a$ | $e_2 - e_3 = e_1$ |
| $f(x) = e_2$ | $e_4 = 0$ |
| $f(y) = e_3$ | $e_5 > a + 2$ |
| $f(e_4) = e_5$ | |
| $x = y$ | |

$$L_1 \models_{\mathrm{UF}} e_2 = e_3$$

**Second step:** exchange entailed *interface equalities*, equalities over shared constants $e_1, e_2, e_3, e_4, e_5, a$

| $L_1$ | $L_2$ |
|---|---|
| $f(e_1) = a$ | $e_2 - e_3 = e_1$ |
| $f(x) = e_2$ | $e_4 = 0$ |
| $f(y) = e_3$ | $e_5 > a + 2$ |
| $f(e_4) = e_5$ | $e_2 = e_3$ |
| $x = y$ | |

**Second step:** exchange entailed *interface equalities*, equalities over shared constants $e_1, e_2, e_3, e_4, e_5, a$

| $L_1$ | $L_2$ |
|---|---|
| $f(e_1) = a$ | $e_2 - e_3 = e_1$ |
| $f(x) = e_2$ | $e_4 = 0$ |
| $f(y) = e_3$ | $e_5 > a + 2$ |
| $f(e_4) = e_5$ | $e_2 = e_3$ |
| $x = y$ | |

$$L_2 \models_{\mathrm{LRA}} e_1 = e_4$$

**Second step:** exchange entailed *interface equalities*, equalities over shared constants $e_1, e_2, e_3, e_4, e_5, a$

| $L_1$ | $L_2$ |
|---|---|
| $f(e_1) = a$ | $e_2 - e_3 = e_1$ |
| $f(x) = e_2$ | $e_4 = 0$ |
| $f(y) = e_3$ | $e_5 > a + 2$ |
| $f(e_4) = e_5$ | $e_2 = e_3$ |
| $x = y$ | |
| $e_1 = e_4$ | |

**Second step:** exchange entailed *interface equalities*, equalities over shared constants $e_1, e_2, e_3, e_4, e_5, a$

| $L_1$ | $L_2$ |
|---|---|
| $f(e_1) = a$ | $e_2 - e_3 = e_1$ |
| $f(x) = e_2$ | $e_4 = 0$ |
| $f(y) = e_3$ | $e_5 > a + 2$ |
| $f(e_4) = e_5$ | $e_2 = e_3$ |
| $x = y$ | |
| $e_1 = e_4$ | |

$$L_1 \models_{\mathrm{UF}} a = e_5$$

**Second step:** exchange entailed *interface equalities*, equalities over shared constants $e_1, e_2, e_3, e_4, e_5, a$

| $L_1$ | $L_2$ |
|---|---|
| $f(e_1) = a$ | $e_2 - e_3 = e_1$ |
| $f(x) = e_2$ | $e_4 = 0$ |
| $f(y) = e_3$ | $e_5 > a + 2$ |
| $f(e_4) = e_5$ | $e_2 = e_3$ |
| $x = y$ | $a = e_5$ |
| $e_1 = e_4$ | |

**Second step:** exchange entailed *interface equalities*, equalities over shared constants $e_1, e_2, e_3, e_4, e_5, a$

| $L_1$ | $L_2$ |
|---|---|
| $f(e_1) = a$ | $e_2 - e_3 = e_1$ |
| $f(x) = e_2$ | $e_4 = 0$ |
| $f(y) = e_3$ | $e_5 > a + 2$ |
| $f(e_4) = e_5$ | $e_2 = e_3$ |
| $x = y$ | $a = e_5$ |
| $e_1 = e_4$ | |

**Third step:** check for satisfiability locally

$L_1 \not\models_{\mathrm{UF}} \bot$

$L_2 \models_{\mathrm{LRA}} \bot$

**Second step:** exchange entailed *interface equalities*, equalities over shared constants $e_1, e_2, e_3, e_4, e_5, a$

| $L_1$ | $L_2$ |
|---|---|
| $f(e_1) = a$ | $e_2 - e_3 = e_1$ |
| $f(x) = e_2$ | $e_4 = 0$ |
| $f(y) = e_3$ | $e_5 > a + 2$ |
| $f(e_4) = e_5$ | $e_2 = e_3$ |
| $x = y$ | $a = e_5$ |
| $e_1 = e_4$ | |

**Third step:** check for satisfiability locally

$L_1 \not\models_{\mathrm{UF}} \bot$

$L_2 \models_{\mathrm{LRA}} \bot$

Report unsatisfiable

# Nelson–Oppen approach

A theory $\mathcal{T}$ is stably infinite, if every $\mathcal{T}$-satisfiable ground formula is $\mathcal{T}$-satisfied by an infinite $\mathcal{T}$-interpretation. For example, finite structures like (QF_BV) are not stably infinite.

A theory $\mathcal{T}$ is convex, if $\mathcal{T}$ is a finite set of literals and if $\Gamma \models_{\mathcal{T}} \varphi_1 \vee \cdots \vee \varphi_n$, then $\Gamma \models_{\mathcal{T}} \varphi_i$ for some $i \in \{1, \ldots, n\}$. For example, (QF_UF) and (QF_LRA) are convex, but (QF_LIA), (QF_AX), and (QF_BV) are not convex.

It is possible to combine theories (Nelson–Oppen method) that are

- signature-disjoint (equalities are shared),
- stably infinite, and
- convex.

It is even possible to combine non-convex theories by propagating disjunctions of equalities (splitting). From practical point of view, many optimizations are required. There are also other methods, e.g., model-based theory combinations.

Consider the following unsatisfiable set of literals over $T_{\mathrm{LIA}} \cup T_{\mathrm{UF}}$ ($T_{\mathrm{LIA}}$, linear integer arithmetic):

$$\begin{aligned} 1 \leq \quad & x \quad \leq 2 \\ f(1) \quad & = \quad a \\ f(2) \quad & = \quad f(1) + 3 \\ a \quad & = \quad b + 2 \end{aligned}$$

Consider the following unsatisfiable set of literals over $T_{\mathrm{LIA}} \cup T_{\mathrm{UF}}$ ($T_{\mathrm{LIA}}$, linear integer arithmetic):

$$
\begin{array}{rcl}
1 \leq & x & \leq 2 \\
f(1) & = & a \\
f(2) & = & f(1) + 3 \\
a & = & b + 2
\end{array}
$$

**First step:** *purify* literals so that each belongs to a single theory

Consider the following unsatisfiable set of literals over $T_{\mathrm{LIA}} \cup T_{\mathrm{UF}}$ ($T_{\mathrm{LIA}}$, linear integer arithmetic):

$$\begin{aligned} 1 \leq \quad x \quad &\leq 2 \\ f(1) &= a \\ f(2) &= f(1) + 3 \\ a &= b + 2 \end{aligned}$$

**First step:** *purify* literals so that each belongs to a single theory

$$f(1) = a \quad \Longrightarrow \quad \begin{aligned} f(e_1) &= a \\ e_1 &= 1 \end{aligned}$$

Consider the following unsatisfiable set of literals over $T_{\text{LIA}} \cup T_{\text{UF}}$ ($T_{\text{LIA}}$, linear integer arithmetic):

$$
\begin{aligned}
1 \leq \quad x \quad &\leq 2 \\
f(1) \quad &= \quad a \\
f(2) \quad &= \quad f(1) + 3 \\
a \quad &= \quad b + 2
\end{aligned}
$$

**First step:** *purify* literals so that each belongs to a single theory

$$
f(2) = f(1) + 3 \quad \Longrightarrow \quad
\begin{aligned}
e_2 &= 2 \\
f(e_2) &= e_3 \\
f(e_1) &= e_4 \\
e_3 &= e_4 + 3
\end{aligned}
$$

**Second step:** exchange entailed *interface equalities* over shared constants $x, e_1, a, b, e_2, e_3, e_4$

| $L_1$ | $L_2$ |
|---|---|
| $1 \leq x$ | $f(e_1) = a$ |
| $x \leq 2$ | $f(x) = b$ |
| $e_1 = 1$ | $f(e_2) = e_3$ |
| $a = b + 2$ | $f(e_1) = e_4$ |
| $e_2 = 2$ | |
| $e_3 = e_4 + 3$ | |
| $a = e_4$ | |

**Second step:** exchange entailed *interface equalities* over shared constants $x, e_1, a, b, e_2, e_3, e_4$

| $L_1$ | $L_2$ |
|---|---|
| $1 \leq x$ | $f(e_1) = a$ |
| $x \leq 2$ | $f(x) = b$ |
| $e_1 = 1$ | $f(e_2) = e_3$ |
| $a = b + 2$ | $f(e_1) = e_4$ |
| $e_2 = 2$ | |
| $e_3 = e_4 + 3$ | |
| $a = e_4$ | |

No more entailed equalities, but $L_1 \models_{\text{LIA}} x = e_1 \vee x = e_2$

**Second step:** exchange entailed *interface equalities* over shared constants $x, e_1, a, b, e_2, e_3, e_4$

| $L_1$ | $L_2$ |
|---|---|
| $1 \leq x$ | $f(e_1) = a$ |
| $x \leq 2$ | $f(x) = b$ |
| $e_1 = 1$ | $f(e_2) = e_3$ |
| $a = b + 2$ | $f(e_1) = e_4$ |
| $e_2 = 2$ | |
| $e_3 = e_4 + 3$ | |
| $a = e_4$ | |

Consider each case of $x = e_1 \vee x = e_2$ separately

79

**Second step:** exchange entailed *interface equalities* over shared constants $x, e_1, a, b, e_2, e_3, e_4$

| $L_1$ | $L_2$ |
|---|---|
| $1 \leq x$ | $f(e_1) = a$ |
| $x \leq 2$ | $f(x) = b$ |
| $e_1 = 1$ | $f(e_2) = e_3$ |
| $a = b + 2$ | $f(e_1) = e_4$ |
| $e_2 = 2$ | |
| $e_3 = e_4 + 3$ | |
| $a = e_4$ | |

Case 1) $x = e_1$

**Second step:** exchange entailed *interface equalities* over shared constants $x, e_1, a, b, e_2, e_3, e_4$

| $L_1$ | $L_2$ |
|---|---|
| $1 \leq x$ | $f(e_1) = a$ |
| $x \leq 2$ | $f(x) = b$ |
| $e_1 = 1$ | $f(e_2) = e_3$ |
| $a = b + 2$ | $f(e_1) = e_4$ |
| $e_2 = 2$ | $x = e_1$ |
| $e_3 = e_4 + 3$ | |
| $a = e_4$ | |
| $x = e_1$ | |

**Second step:** exchange entailed *interface equalities* over shared constants $x, e_1, a, b, e_2, e_3, e_4$

| $L_1$ | $L_2$ |
|---|---|
| $1 \leq x$ | $f(e_1) = a$ |
| $x \leq 2$ | $f(x) = b$ |
| $e_1 = 1$ | $f(e_2) = e_3$ |
| $a = b + 2$ | $f(e_1) = e_4$ |
| $e_2 = 2$ | $x = e_1$ |
| $e_3 = e_4 + 3$ | |
| $a = e_4$ | |
| $x = e_1$ | |

$L_2 \models_{\mathrm{UF}} a = b$, which entails $\bot$ when sent to $L_1$

**Second step:** exchange entailed *interface equalities* over shared constants $x, e_1, a, b, e_2, e_3, e_4$

| $L_1$ | $L_2$ |
|---|---|
| $1 \leq x$ | $f(e_1) = a$ |
| $x \leq 2$ | $f(x) = b$ |
| $e_1 = 1$ | $f(e_2) = e_3$ |
| $a = b + 2$ | $f(e_1) = e_4$ |
| $e_2 = 2$ | |
| $e_3 = e_4 + 3$ | |
| $a = e_4$ | |

**Second step:** exchange entailed *interface equalities* over shared constants $x, e_1, a, b, e_2, e_3, e_4$

| $L_1$ | $L_2$ |
|---|---|
| $1 \leq x$ | $f(e_1) = a$ |
| $x \leq 2$ | $f(x) = b$ |
| $e_1 = 1$ | $f(e_2) = e_3$ |
| $a = b + 2$ | $f(e_1) = e_4$ |
| $e_2 = 2$ | |
| $e_3 = e_4 + 3$ | |
| $a = e_4$ | |

Case 2) $x = e_2$

**Second step:** exchange entailed *interface equalities* over shared constants $x, e_1, a, b, e_2, e_3, e_4$

| $L_1$ | $L_2$ |
|---|---|
| $1 \leq x$ | $f(e_1) = a$ |
| $x \leq 2$ | $f(x) = b$ |
| $e_1 = 1$ | $f(e_2) = e_3$ |
| $a = b + 2$ | $f(e_1) = e_4$ |
| $e_2 = 2$ | $x = e_2$ |
| $e_3 = e_4 + 3$ | |
| $a = e_4$ | |
| $x = e_2$ | |

**Second step:** exchange entailed *interface equalities* over shared constants $x, e_1, a, b, e_2, e_3, e_4$

| $L_1$ | $L_2$ |
|---|---|
| $1 \leq x$ | $f(e_1) = a$ |
| $x \leq 2$ | $f(x) = b$ |
| $e_1 = 1$ | $f(e_2) = e_3$ |
| $a = b + 2$ | $f(e_1) = e_4$ |
| $e_2 = 2$ | $x = e_2$ |
| $e_3 = e_4 + 3$ | |
| $a = e_4$ | |
| $x = e_2$ | |

$L_2 \models_{\mathrm{UF}} e_3 = b$, which entails $\bot$ when sent to $L_1$

# Why is SMT useful?

- it combines (not only) propositional logic reasoning with a domain-specific reasoning in a modular way,
- it covers commonly used theories and their combinations, for example, in software verification, we reason about
  - equalities,
  - arithmetic,
  - data structures,
- many applications in
  - scheduling,
  - test generation,
  - symbolic software verification,
  - static analysis,
  - program verificaiton,
  - hardware verification,
- used by major companies like Microsoft (develops Z3), Amazon, . . .

# Quantifiers?

It is convenient to use $\forall$ and $\exists$ and modern SMT solvers can deal with quantifiers.

For some theories, it is even possible to eliminate quantifiers.

However, in SMT, it usually means that instances are produced in an ad hoc way...

Here, we continue with a systematic approach how to treat quantifiers. Hence we want to discuss provability in the full FOL.

# Example (group theory)

Assume we have axioms

$$\forall X(1 \cdot X = X)$$
$$\forall X(X^{-1} \cdot X = 1)$$
$$\forall X \forall Y \forall Z((X \cdot Y) \cdot Z = X \cdot (Y \cdot Z))$$

and we want to know whether

1. $\forall X(X \cdot 1 = X)$,
2. $\forall X(X \cdot X^{-1} = 1)$, and
3. $\forall X \forall Y(X \cdot Y = Y \cdot X)$

follow from them. These are tasks in which automated theorem provers usually outperform people.

# Two types of occurrences of variables

There is no difference between

$$\forall X p(X) \quad \text{and} \quad \forall Y p(Y)$$

when it comes to their meaning. Hence they can be freely replaced in formuale.

However, there is clearly a difference between

$$p(X) \quad \text{and} \quad p(Y),$$

because if we replace $p(X)$ by $p(Y)$ in

$$p(X) \vee \neg p(Y),$$

then we get

$$p(Y) \vee \neg p(Y).$$

# Free and bounded occurrences of variables

We distinguish two types of occurrences of variables in a formula $\varphi$

- free — not under a scope of a quantifier, denoted $FV(\varphi)$,
- bounded — under a scope of a quantifier, denoted $BV(\varphi)$.

$$FV(\varphi) = \begin{cases} \{\, X \mid X \text{ occurs in } \varphi \,\}, & \text{if } \varphi \text{ is atomic,} \\ FV(\psi), & \text{if } \varphi = \neg\psi, \\ FV(\psi) \cup FV(\chi), & \text{if } \varphi = \psi \circ \chi \text{ for } \circ \in \{\wedge, \vee, \rightarrow\}, \\ FV(\psi) \setminus \{X\}, & \text{if } \varphi = QX\psi \text{ for } Q \in \{\forall, \exists\}. \end{cases}$$

$$BV(\varphi) = \begin{cases} \emptyset, & \text{if } \varphi \text{ is atomic,} \\ BV(\psi), & \text{if } \varphi = \neg\psi, \\ BV(\psi) \cup BV(\chi), & \text{if } \varphi = \psi \circ \chi \text{ for } \circ \in \{\wedge, \vee, \rightarrow\}, \\ BV(\psi) \cup \{X\}, & \text{if } \varphi = QX\psi \text{ for } Q \in \{\forall, \exists\}. \end{cases}$$

It is possible that $FV(\varphi) \cap BV(\varphi) \neq \emptyset$.

# Tarski's definition of truth

Let $\mathcal{M} = (D, i)$ be a model for $L$, $e$ be an evaluation in $\mathcal{M}$, then we say that a formula $\varphi$ is satisfied in $\mathcal{M}$ by $e$, denoted $\mathcal{M} \models \varphi[e]$, or $e$ satisfies $\varphi$ in $\mathcal{M}$, if

- $\mathcal{M} \models p(t_1, \ldots, t_n)[e]$ iff $(t_1^{\mathcal{M}}[e], \ldots, t_n^{\mathcal{M}}[e]) \in i(p)$, where $p$ is $n$-ary predicate symbol in $L$,
- $\mathcal{M} \models (t_1 = t_2)[e]$ iff $(t_1^{\mathcal{M}}[e], t_2^{\mathcal{M}}[e]) \in \mathrm{id}_D$, (in FOL with eq.)
- $\mathcal{M} \models (\neg\psi)[e]$ iff $\mathcal{M} \not\models \psi[e]$,
- $\mathcal{M} \models (\psi \to \chi)[e]$ iff $\mathcal{M} \not\models \psi[e]$ or $\mathcal{M} \models \chi[e]$,
- $\mathcal{M} \models (\psi \wedge \chi)[e]$ iff $\mathcal{M} \models \psi[e]$ and $\mathcal{M} \models \chi[e]$,
- $\mathcal{M} \models (\psi \vee \chi)[e]$ iff $\mathcal{M} \models \psi[e]$ or $\mathcal{M} \models \chi[e]$,
- $\mathcal{M} \models (\forall X \psi)[e]$ iff for every $a \in D$ holds $\mathcal{M} \models \psi[e(X \mapsto a)]$,
- $\mathcal{M} \models (\exists X \psi)[e]$ iff exists $a \in D$ s.t. $\mathcal{M} \models \psi[e(X \mapsto a)]$.

A formula $\varphi$ is satisfiable, if there is $\mathcal{M}$ and $e$ s.t. $\mathcal{M} \models \varphi[e]$. A set of formulae $\Gamma$ is satisfiable, if there is $\mathcal{M}$ and $e$ s.t. $\mathcal{M} \models \varphi[e]$, for every $\varphi \in \Gamma$.

# Semantic consequence relation

A formula $\varphi$ is valid (or holds) in $\mathcal{M}$, denoted $\mathcal{M} \models \varphi$, if $\varphi$ is satisfied in $\mathcal{M}$ by any evaluation $e$.

A formula $\varphi$ follows from (or is a consequence of) a set of formula $\Gamma$, denoted $\Gamma \models \varphi$, if and only if for any model $\mathcal{M}$ and evaluation $e$, if for every $\psi \in \Gamma$ holds $\mathcal{M} \models \psi[e]$, then $\mathcal{M} \models \varphi[e]$. We write $\models \varphi$, if $\Gamma = \emptyset$ and say that $\varphi$ is valid (or holds).

$$\Gamma \models \varphi \quad \text{iff} \quad \forall \mathcal{M} \forall e (\forall \psi \in \Gamma (\mathcal{M} \models \psi[e]) \Rightarrow \mathcal{M} \models \varphi[e])$$

Note that

$$\Gamma \models \varphi \quad \text{iff} \quad \Gamma \cup \{\neg\varphi\} \text{ is unsatisfiable.}$$

We say that two formulae $\varphi$ and $\psi$ are (semantically) equivalent, denoted $\varphi \equiv \psi$, if $\{\varphi\} \models \psi$ and $\{\psi\} \models \varphi$.

# Basic properties

Let $\varphi$, $\psi$, and $\chi$ be formulae such that $X \notin FV(\psi)$, then

- $\neg\forall X\varphi \equiv \exists X\neg\varphi$,
- $\neg\exists X\varphi \equiv \forall X\neg\varphi$,
- $\forall X\forall Y\varphi \equiv \forall Y\forall X\varphi$,
- $\exists X\exists Y\varphi \equiv \exists Y\exists X\varphi$,
- $\forall X(\varphi \wedge \chi) \equiv \forall X\varphi \wedge \forall X\chi$,
- $\exists X(\varphi \vee \chi) \equiv \exists X\varphi \vee \exists X\chi$,
- $\exists X(\varphi \rightarrow \chi) \equiv \forall X\varphi \rightarrow \exists X\chi$,
- $(\psi \wedge \forall X\varphi) \equiv \forall X(\psi \wedge \varphi)$,
- $(\psi \wedge \exists X\varphi) \equiv \exists X(\psi \wedge \varphi)$,
- $(\psi \vee \forall X\varphi) \equiv \forall X(\psi \vee \varphi)$,
- $(\psi \vee \exists X\varphi) \equiv \exists X(\psi \vee \varphi)$,
- $(\psi \rightarrow \forall X\varphi) \equiv \forall X(\psi \rightarrow \varphi)$,
- $(\psi \rightarrow \exists X\varphi) \equiv \exists X(\psi \rightarrow \varphi)$,
- $(\forall X\varphi \rightarrow \psi) \equiv \exists X(\varphi \rightarrow \psi)$,
- $(\exists X\varphi \rightarrow \psi) \equiv \forall X(\varphi \rightarrow \psi)$.

# Equivalent formulae

We can freely replace (sub)formulae by equivalent formulae. More formally

### Lemma

*Let* $\psi$ *be a subformula of a formual* $\varphi$*, and* $\chi$ *be a formula such that* $\psi \equiv \chi$*. A formula* $\varphi'$ *is obtained by replacing* $\psi$ *in* $\varphi$ *by* $\chi$*. It holds that* $\varphi \equiv \varphi'$*.*

### Example

For example, this is useful for renaming bounded variables. Clearly $\forall X r(X, Y) \equiv \forall Z r(Z, Y)$ and hence $\forall X(p(X) \wedge \forall X r(X, Y))$ is equivalent to $\forall X(p(X) \wedge \forall Z r(Z, Y))$.

Note that $\forall X r(X, Y)$ is not equivalent to $\forall Y r(Y, Y)$!

# How to decide whether $\Gamma \models \varphi$ for FOL using computers?

First, we know that this problem is undecidable...

However, we can still use our favorite recipe

1. show that it is sufficient to deal only with a restricted class of formulae by presenting various transformations and (=clauses)
2. use techniques developed for less expressive systems (=resolution)

to create a procedure that is quite useful.

Note that this is not the only possible approach! Moreover, the other approaches may have various advantages. Similarly, using only CNFs in propositional logic may lead to various problems.

# Prenexing

We say that a formula $\varphi$ is in prenex (normal) form, if

$$\varphi = Q_1 X_1, \ldots, Q_n X_n \psi,$$

where $Q_1, \ldots, Q_n$ are quantifiers, $X_1, \ldots, X_n$ are mutually distinct variables, and $\psi$ is an open (quantifier-free) formula.

**Lemma**

*For every formula $\varphi$, there exists a formula $\psi$ in prenex (normal) form such that $\varphi \equiv \psi$.*

**Proof.**

By induction on the structure of the formula $\varphi$ using previous equivalences and renaming of bounded variables. □

# Substitutions

A substitution, $\sigma\colon \mathit{Var} \to \mathit{Term}$, is a function that assigns terms to variables. An application of a substitution $\sigma$ on a formula $\varphi$, denoted $\varphi\sigma$, is a formula $\varphi$ with all free occurrences of variables replaced simultaneously by their $\sigma$ images. We usually denote substitutions $\sigma$, $\theta$, and $\eta$.

Note that we usually provide only the non-identity part of a substitution. A substitution $\sigma\colon \mathit{Var} \to \mathit{Var}$ is called a *renaming*.

## Example

Let $\sigma = \{X \mapsto f(X, Z), Y \mapsto a\}$, then

$$((p(X, Y) \to q(Y)) \vee (\forall V r(V, X)))\sigma$$

is

$$((p(f(X, Z), a) \to q(a)) \vee (\forall V r(V, f(X, Z)))).$$

# Substitutability

A term $t$ is *substitutable* into a formula $\varphi$ for a variable $X$, if no occurrence of a variable in $t$ becomes bounded in $\varphi$ when all free occurrences of $X$ in $\varphi$ are replaced by $t$.

This directly extends to substitutions. From now on, we assume that if we apply a substitution, it is substitutable. However, we can always avoid all these potential problems by renaming bounded variables appropriately.

## Example

Let $\sigma = \{X \mapsto f(X,Y), Y \mapsto g(X), Z \mapsto g(X)\}$, then

- $(\forall Z p(X,Y,Z))\sigma = \forall Z p(f(X,Y), g(X), Z)$, and
- $(\forall Y p(X,Y,Z))\sigma$ is not substitutable, but
- $(\forall U p(X,U,Z))\sigma = \forall U (f(X,Y), U, g(X))$.

# Sentences

A term is ground (or closed), if it contains no variables. A formula $\varphi$ is a sentence (or closed), if it contains no free occurrences of variables. A formula $\varphi$ is open, if it contains no quantifiers.

### Lemma

*Let $\varphi$ be a sentence, $\sigma$ be a substitution, $\mathcal{M}$ be an interpretation, and $e$ be an evaluation, then*

1. $\varphi\sigma = \varphi$,
2. $\mathcal{M} \models \varphi[e]$ *iff* $\mathcal{M} \models \varphi[e']$ *for every evaluation* $e'$,
3. $\mathcal{M} \models \varphi$ *or* $\mathcal{M} \models \neg\varphi$.

### Example

$p(a)$ and $\forall X \forall Y (p(X, b, g(Y, X)) \rightarrow q(f(f(b)), X))$ are sentences.
$\forall Y (p(X, b, g(Y, X)) \rightarrow q(f(f(b)), X))$ is not a sentence.

# Skolem functions

It is possible to get rid of existential quantifiers by introducing Skolem functions (or Skolem constants) that behave as witnesses (or choice functions).

We know that

$$\exists X \forall Y \exists Z p(X, Y, Z) \tag{1}$$

follows from

$$\forall Y p(c, Y, f(Y)) \tag{2}$$

where $c$ and $f/1$ are fresh. Although (2) does not follow from (1), they are equisatisfiable.

# Skolemization

We say that a formula is in Skolem normal form if it is in prenex normal form and it contains no existential quantifiers.

We can obtain a formula in Skolem normal form from a formula $\varphi$ in prenex normal form by eliminating the first existential quantifier in

$$\varphi = \forall X_1 \dots \forall X_n \exists Y \psi.$$

We obtain

$$\varphi' = \forall X_1 \dots \forall X_n \psi\{Y \mapsto f(X_1, \dots, X_n)\}$$

where $f$ is a fresh $n$-ary function symbol. Then we repeat the whole process with $\varphi'$ until there is no existential quantifier in the formula. The resulting formula is equisatisfiable with $\varphi$.

We prefer Skolem functions with smaller arities.

# Usual transformations

## NNF (negation normal form)

Apply the following rewriting steps as long as possible:

$$
\begin{array}{rcl}
\neg\neg\varphi & \rightsquigarrow & \varphi \\
\varphi \to \psi & \rightsquigarrow & \neg\varphi \lor \psi \\
\neg(\varphi \land \psi) & \rightsquigarrow & \neg\varphi \lor \neg\psi \\
\neg(\varphi \lor \psi) & \rightsquigarrow & \neg\varphi \land \neg\psi \\
\neg(\forall X \varphi) & \rightsquigarrow & \exists X \neg\varphi \\
\neg(\exists X \varphi) & \rightsquigarrow & \forall X \neg\varphi
\end{array}
$$

## Rectified formulae

A formula $\varphi$ is rectified if

- no variable occurs both free and bounded in $\varphi$,
- no two quantifiers in $\varphi$ quantify over the same variable.

We obtain a rectified formula by renaming bounded variables.

# Clauses in FOL

We adapt our terminology from propositional logic.

A *literal* is an atomic formula (positive), or a negation of an atomic formula (negative).

A *clause* is a disjunction of finitely many literals. An important special case is the empty clause, denoted $\Box$.

A *formula* $\varphi$ is in conjunctive normal form (CNF) if $\varphi$ is a conjunction of clauses.

Recall two special cases:

- the empty clause $\Box$ (empty disjunction) is unsatisfiable,
- the empty conjunction is satisfiable.

# CNF

The universal closure of a formula $\varphi$, denoted $\forall\varphi$, is a formula $\forall X_1 \ldots \forall X_n \varphi$, where $\{X_1, \ldots, X_n\} = FV(\varphi)$.

We produce a CNF $\mathrm{cnf}(\varphi)$ (implicitly universally quantified) from a sentence $\varphi$ by performing the following steps

1. produce a NNF,
2. rectify,
3. skolemize (an obvious generalization for sentences not in prenex normal form),
4. remove all universal quantifiers,
5. produce a CNF as in propositional logic.

Let $\mathrm{cnf}(\varphi) = \{\chi_1, \ldots, \chi_m\}$, where $\chi_i$ are clauses. It holds that

$\varphi$ is satisf. iff $\forall \bigwedge \mathrm{cnf}(\varphi)$ is satisf. iff $\forall\chi_1 \wedge \cdots \wedge \forall\chi_m$ is satisf.

# Our problem

Let $\Gamma = \{\psi_1, \ldots, \psi_n\}$ be a set of senteces and $\varphi$ be a sentence. We know that

$$\Gamma \models \varphi$$

iff

$$\{\psi_1, \ldots, \psi_n\} \cup \{\neg\varphi\} \text{ is unsatisfiable}$$

iff

$$\forall \bigwedge \mathrm{cnf}(\psi_1 \wedge \cdots \wedge \psi_n \wedge \neg\varphi) \text{ is unsatisfiable,}$$

iff

$$\forall\chi_1 \wedge \cdots \wedge \forall\chi_m \text{ is unsatisfiable,}$$

where $\mathrm{cnf}(\psi_1 \wedge \cdots \wedge \psi_n \wedge \neg\varphi) = \{\chi_1, \ldots, \chi_m\}$. (=a set of clauses).

From now on, we always assume that a set of clauses is implicitly universally quantified.

# Used presentation

The slides 1, 3, and 5 are taken from Tinelli 2017.

# Bibliography I


Robinson, John Alan and Andrei Voronkov, eds. (2001). *Handbook of Automated Reasoning*. Vol. 1. Elsevier Science.

Tinelli, Cesare (2017). "Foundations of Satisfiability Modulo Theories". $SC^2$ Summer School 2017. URL: http://www.sc-square.org/CSA/school/lectures/SCSC-Tinelli.pdf.