Introducing first-order logic

Text: Melvin Fitting, *FOLATP*, Sections 5.1,5.3

Kamal Lodaya

March 2021

Summary so far

$$A ::= \quad P,\ P \in At \mid True \mid False \\ \mid (\neg A) \mid (A \vee B) \mid (A \wedge B) \mid (A \supset B) \mid (A \equiv B)$$

- There are many different proof/refutation systems for PL. Our proof systems are weakly sound, every theorem is a valid formula. Strongly sound, every consequence of a theory is valid modulo that theory.
- Weakly complete, every valid formula has a proof in that proof system (Emil Post 1921). Strongly complete, every valid formula modulo Th ($Th \models A$) has a derivation from Th ($Th \vdash A$).
- Martin Davis, Hilary Putnam, George Logemann and Donald Loveland 1960-61 came up with an algorithm to check propositional satisfiability/validity. Basis for modern sat solvers.

ZOL proof system for signature $\Sigma = (R, F, C)$

$$
\begin{array}{ll}
t ::= & x \in V \mid c \in C \mid f(t_1, \ldots, t_n),\ f \in F_n \\
A ::= & P(t_1, \ldots, t_n),\ P \in R_n \mid t_1 \approx t_2 \mid \mathit{True} \mid \mathit{False} \\
& \mid (\neg A) \mid (A \vee B) \mid (A \wedge B) \mid (A \supset B) \mid (A \equiv B)
\end{array}
$$

Example: If domain D is all words over alphabet $\{a, b\}$, $0^I = a, 1^I = b$, suc^I is the function appending a to the end of the word, and $+^I$ is concatenation, and if $x^s = aba$, then $\mathit{suc}(\mathit{suc}(0) + \mathit{suc}(x))^{I,s} = aaabaaa$.

ZOL proof system for signature $\Sigma = (R, F, C)$

$$
\begin{array}{ll}
t ::= & x \in V \mid c \in C \mid f(t_1, \ldots, t_n),\ f \in F_n \\
A ::= & P(t_1, \ldots, t_n),\ P \in R_n \mid t_1 \approx t_2 \mid \mathit{True} \mid \mathit{False} \\
& \mid (\neg A) \mid (A \vee B) \mid (A \wedge B) \mid (A \supset B) \mid (A \equiv B)
\end{array}
$$

Example: If domain D is all words over alphabet $\{a, b\}$, $0^I = a, 1^I = b$, suc^I is the function appending a to the end of the word, and $+^I$ is concatenation, and if $x^s = aba$, then $\mathit{suc}(\mathit{suc}(0) + \mathit{suc}(x))^{I,s} = aaabaaa$.

Proof system zHB, sound for equality (Leibniz, 17th cent.CE):

pHB + (Reflexivity) $t \approx t$ + axiom schemes of (Replacement):

$(t_1 \approx u_1) \wedge \cdots \wedge (t_n \approx u_n) \supset f(t_1, \ldots, t_n) \approx f(u_1, \ldots, u_n)$

$(t_1 \approx u_1) \wedge \cdots \wedge (t_n \approx u_n) \supset (A(t_1, \ldots, t_n) \supset A(u_1, \ldots, u_n))$

ZOL completeness for signature $\Sigma = (R, F, C)$

Earlier in Lecture 5, every atomic formula $P(t_1, \ldots, t_n)$ was made into a proposition in an uninterpreting translation to PL:

Theorem (Propositional abstraction)

For a ZOL formula A of Σ, if its propositional abstraction $Un(A)$ of signature $PL(At(\Sigma))$ is satisfiable, then A is satisfiable.

ZOL completeness for signature $\Sigma = (R, F, C)$

Earlier in Lecture 5, every atomic formula $P(t_1, \ldots, t_n)$ was made into a proposition in an uninterpreting translation to PL:

Theorem (Propositional abstraction)

For a ZOL formula A of Σ, if its propositional abstraction $Un(A)$ of signature $PL(At(\Sigma))$ is satisfiable, then A is satisfiable.

Theorem (ZOL satisfiability)

ZOL Hintikka theories are satisfiable in a ZOL model.

ZOL completeness for signature $\Sigma = (R, F, C)$

Earlier in Lecture 5, every atomic formula $P(t_1, \ldots, t_n)$ was made into a proposition in an uninterpreting translation to PL:

Theorem (Propositional abstraction)

For a ZOL formula A of Σ, if its propositional abstraction $Un(A)$ of signature $PL(At(\Sigma))$ is satisfiable, then A is satisfiable.

Theorem (ZOL satisfiability)

ZOL Hintikka theories are satisfiable in a ZOL model.

- Finite consistent ZOL theories satisfiable?
- Check: propositional abstraction of finite ZOL theory needs finite PL theory.
- DPLL for ZOL sat? Is it sufficient to run DPLL to check satisfiability of $Un(A)$?

FOL syntax for signature $\Sigma = (R, F, C)$

$$
\begin{aligned}
t ::= \quad & x \in V \mid c \in C \mid f(t_1, \ldots, t_n),\ f \in F_n \\
A ::= \quad & P(t_1, \ldots, t_n),\ P \in R_n \mid t_1 \approx t_2 \mid \mathit{True} \mid \mathit{False} \\
& \mid (\neg A) \mid (A \vee B) \mid (A \wedge B) \mid (A \supset B) \mid (A \equiv B) \\
& \mid \exists x A \mid \forall x A
\end{aligned}
$$

- Above variable x is bound to quantifiers $\exists x$ and $\forall x$ respectively.
- Variables not bound are free. $A(x)$ indicates free variable.
- Sentence is a formula with no free variables (that is, it only has variables bound to quantifiers).
  Generalizes ZOL sentence.
- $\forall x \forall y (x + y \approx y + x)$ is a sentence, x, y bound.
- $\forall y (x + y \approx y + x)$ is not, x is a free variable.
- $\forall y (3 + y \approx y + 3)$ is a sentence, y bound.
- $3 + y \approx y + 3$ is not, y is a free variable.

FOL syntax for signature $\Sigma = (R, F, C)$

$$
\begin{aligned}
t ::= \quad & x \in V \mid c \in C \mid f(t_1, \ldots, t_n),\ f \in F_n \\
A ::= \quad & P(t_1, \ldots, t_n),\ P \in R_n \mid t_1 \approx t_2 \mid \mathit{True} \mid \mathit{False} \\
& \mid (\neg A) \mid (A \vee B) \mid (A \wedge B) \mid (A \supset B) \mid (A \equiv B) \\
& \mid \exists x A \mid \forall x A
\end{aligned}
$$

- Above variable x is bound to quantifiers $\exists x$ and $\forall x$ respectively.
- Variables not bound are free. $A(x)$ indicates free variable.
- Sentence is a formula with no free variables (that is, it only has variables bound to quantifiers).
  Generalizes ZOL sentence.
- $\forall x \forall y (x + y \approx y + x)$ is a sentence, x, y bound.
- $\forall y (x + y \approx y + x)$ is not, x is a free variable.
- $\forall y (3 + y \approx y + 3)$ is a sentence, y bound.
- $3 + y \approx y + 3$ is not, y is a free variable.
- Prenex normal form: $\forall x (\exists y A(y) \vee (\exists z B(z) \supset C(x)))$ can be equivalently rewritten with quantifiers $\forall\exists\forall$ (prefix class) before ZOL formula, $\forall x \exists y \forall z (A(y) \vee (B(z) \supset C(x)))$.

FOL satisfaction

$$
\begin{array}{ll}
t ::= & x \in V \mid c \in C \mid f(t_1, \ldots, t_n),\ f \in F_n \\
A ::= & P(t_1, \ldots, t_n),\ P \in R_n \mid t_1 \approx t_2 \mid \mathit{True} \mid \mathit{False} \\
& \mid (\neg A) \mid (A \vee B) \mid (A \wedge B) \mid (A \supset B) \mid (A \equiv B) \\
& \mid \exists x A \mid \forall x A
\end{array}
$$

Definition

Given model $M = (D, I)$ and assignment $s : V \to D$.

- *Assignment r is an x-variant of s if r, s differ at most on the value assigned to variable x.*
- *Satisfaction of formula A extends that of ZOL:*

  $M, s \models \mathit{True}$ (*always*)
  $M, s \not\models \mathit{False}$ (*never*)
  $M, s \models P(t_1, \ldots, t_n)$ iff $(t_1^{I,s}, \ldots, t_n^{I,s}) \in I(P)$
  $\ldots$
  $M, s \models \forall x A$ iff $M, r \models A$ for all r *x-variant of* s
  $M, s \models \exists x A$ iff $M, r \models A$ for some r *x-variant of* s

- *(Thus r ranges over mapping x to all domain values)*

FOL examples

- $\exists y(x \approx y + y)$ is satisfied over $\mathbb{N}$ with usual interpretation iff x is even.
- The sentence $\forall x \forall y(x > y \supset \exists z(x > z \wedge z > y))$ holds over $\mathbb{Q}$ and $\mathbb{R}$, not over $\mathbb{N}$ and $\mathbb{Z}$.
- Axiom schemes of ZOL proof system BD: change $x + y \approx y + x$ to the closed sentence $\forall x \forall y(x + y \approx y + x)$.
- $D = \{u, v\}$, binary predicate E interpreted as the edge relation of a directed graph.
  Suppose $E(u, v)$ interpreted *true* and $E(v, u)$ as *false*.

  Sentence $\forall x \forall y(E(x, y) \supset E(y, x))$ does not hold, although one may think of it as an axiom in the theory of undirected graphs.