Special Topics on Applied Mathematical Logic

Spring 2012

Lecture 03

Jie-Hong Roland Jiang

National Taiwan University

March 10, 2012

Outline

First-Order Logic

First-Order Languages (Syntax)

First-Order Logic

First-order logic provides

1. a syntax capable of expressing detailed mathematical statements
2. semantics that identify a sentence with its intended mathematical application
3. a generic and comprehensive proof system

Metalanguage and Metamathematics

- Metalanguage vs. object language
  - We study an object language in terms of a metalanguage
  - English will be our metalanguage to study the object languages, such as the language of sentential logic, first-order languages, etc.
- Metamathematics vs. mathematics
  - We study mathematics in terms of metamathematics
  - Mathematical logic will be our metamathematics to study mathematics, such as number theory, set theory, etc.

First-Order Logic

E.g., first-order language of number theory:

- Symbols:
  - Constant symbol 0 (meaning "zero"); function symbol S (meaning successor of); predicate symbol $<$ (meaning less than); quantifier symbol $\forall$ (meaning for every natural number); equality symbol $=$
- Formulas:
  - E.g., $\forall v_1(0 < v_1 \Rightarrow \neg(v_1 = 0))$, $\exists v_1 \forall v_2(v_1 = v_2)$, ...

First-Order Languages

symbol
logical symbols
parenthesis: (,)
sentential connective symbols: $\Rightarrow$, $\neg$
variables: $v_1, v_2, \ldots$
equality symbol: $=$ (optional)
parameters
quantifier symbol: $\forall$
predicate symbols (possibly empty)
constant symbols (possibly empty)
function symbols (possibly empty)

- $\{\Rightarrow, \neg\}$ is functionally complete
- Quantifier $\exists$ is unnecessary since $\exists x \phi$ equals $\neg \forall x \neg \phi$

First-Order Languages

- Equality symbol "="
  - can be seen as a two-place predicate symbol, but distinguished (to consider English translation)
  - coincides with "$\Leftrightarrow$" in sentential logic
- Constant symbols
  - can be seen as a 0-place function symbol
- Quantifier $\forall$
  - not necessary in sentential logic, but necessary in first-order logic (why?)

First-Order Languages

To specify a language, we need to specify

1. Presence of "="
2. Parameters

E.g.,

pure predicate language:

1. No
2. n-place predicate symbols A_1^n, A_2^n, $\ldots$; constant symbols a_1, a_2, $\ldots$

language of set theory:

1. Yes
2. 2-place predicate symbol $\in$; optionally a constant symbol $\emptyset$

language of elementary number theory:

1. Yes
2. 2-place predicate symbol $<$; constant symbol 0; function symbols S, $+$, $\cdot$, E

Translation into Formulas

Example

Language of set theory (ST)

- There is no set of which every set is a member
  $\neg\exists v_1 \forall v_2 (v_2 \in v_1)$; equivalently, $\forall v_1 \neg\forall v_2 (v_2 \in v_1)$
- For any two sets, there is a set whose members are exactly the two given sets (pair-set axiom)
  $\forall v_1 \forall v_2 \exists v_3 \forall v_4 ((v_4 \in v_3) \Leftrightarrow ((v_4 = v_1) \vee (v_4 = v_2)))$

Language of elementary number theory (NT)

- Any nonzero natural number is the successor of some number
  $\forall v_1 \exists v_2 (\neg(v_1 = 0) \wedge v_1 = S v_2)$ or
  $\forall v_1 \exists v_2 (\neg(v_1 = 0) \Rightarrow v_1 = S v_2)$?
- There is a smallest prime
  ...

Translation into Formulas (cont'd)

Example

Language of analysis

- f converges to L as x approaches to a
  $\forall \epsilon ((\epsilon > 0) \Rightarrow \exists \delta ((\delta > 0) \wedge \forall x (|x - a| < \delta \Rightarrow |fx - L| < \epsilon)))$

Ad hoc language

- All apples are bad $\forall v_1 (A v_1 \Rightarrow B v_1)$
- Some apple is bad $\exists v_1 (A v_1 \wedge B v_1)$
- How about $\forall v_1 (A v_1 \wedge B v_1)$ and $\exists v_1 (A v_1 \Rightarrow B v_1)$?

Translation into Formulas (cont'd)

Observations:

- No free variables in the translated formulas
  - A variable in a formula is **free** if it is not quantified
  - Formulas without free variables are called **sentences**
- Common patterns

$$\forall v((\ldots) \Rightarrow (\ldots)) \text{ and}$$
$$\exists v((\ldots) \wedge (\ldots))$$

Formulas

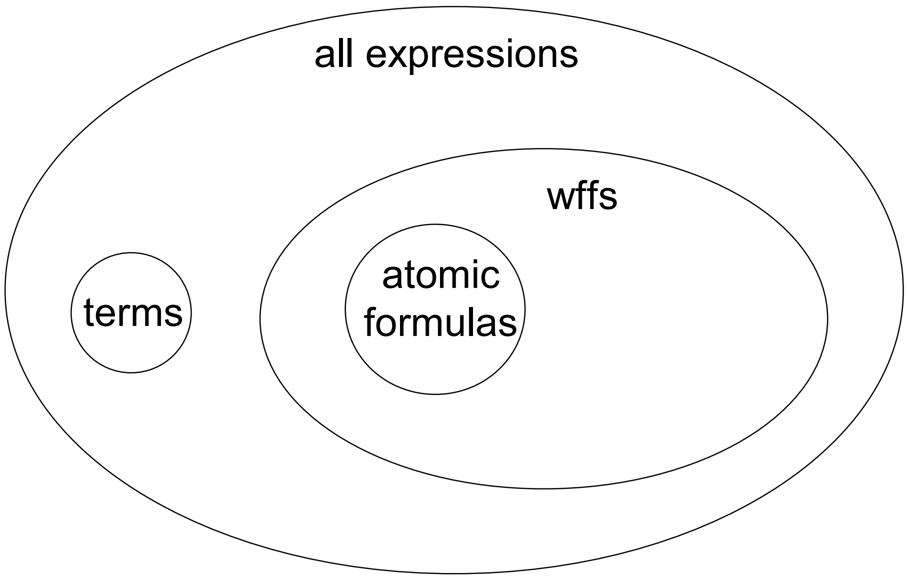
- Expression: any finite sequence of symbols
  - Meaningful expressions: terms and wffs
- Term: noun/pronoun (object name)
  - expression built up from constant symbols and variables by applying (zero or more times) the $\mathcal{F}_f$ operations with $\mathcal{F}_f(\epsilon_1, \ldots, \epsilon_n) = f\epsilon_1, \ldots, \epsilon_n$
- Atomic formula: $Pt_1, \ldots, t_n$ (not inductive definition)
  - wff having neither connective nor quantifier symbols
- Wff:
  - expression built up from atomic formulas by applying (zero or more times) the operations $\mathcal{E}_\neg$, $\mathcal{E}_\Rightarrow$, $\mathcal{Q}_i$ with $\mathcal{E}_\neg(\alpha) = (\neg\alpha)$, $\mathcal{E}_\Rightarrow(\alpha, \beta) = (\alpha \Rightarrow \beta)$, $\mathcal{Q}_i(\alpha) = \forall v_i\alpha$

Formulas

E.g.,

- $SS0$, $+SS0S0$ are terms
- $= v_1 v_2$, $\in v_1 v_2$ are atomic formulas
- $\forall v_1((\neg\forall v_3(\neg \in v_3 v_1)) \Rightarrow (\neg\forall v_2(\in v_1 v_2 \Rightarrow (\neg\forall v_4(\in v_4 v_2) \Rightarrow (\neg \in v_4 v_1)))))$ is a wff
- $\neg v_1$ is NOT a wff

Formulas



Free Variables

- $\forall v_1 \exists v_2 ((\neg v_1 = \emptyset) \Rightarrow (v_2 \in v_1))$ — sentence
- $\forall v_1 (\neg (v_1 = \emptyset) \Rightarrow (v_2 \in v_1))$ — v_2 occurs free
- $\exists v_2 (\neg (v_1 = \emptyset) \Rightarrow (v_2 \in v_1))$ — v_1 occurs free

Formulas

Two ways to define free variables:

1. By recursion, for each wff α, x **occurs free** in α if
   1.1 for atomic α, x occurs free in α iff x occurs in α
   1.2 x occurs free in $(\neg\alpha)$ iff x occurs free in α
   1.3 x occurs free in $(\alpha \Rightarrow \beta)$ iff x occurs free in α or β
   1.4 x occurs free in $\forall v_i \alpha$ iff x occurs free in α and $x \neq v_i$
2. Define $h(\alpha)$ as the set of all variables, if any, in the *atomic formula* α. Extend h to

$$\begin{aligned}
\overline{h}(\mathcal{E}_{\neg}(\alpha)) &= \overline{h}(\alpha), \\
\overline{h}(\mathcal{E}_{\Rightarrow}(\alpha, \beta)) &= \overline{h}(\alpha) \cup \overline{h}(\beta), \\
\overline{h}(\mathcal{Q}_i(\alpha)) &= \overline{h}(\alpha) \backslash v_i.
\end{aligned}$$

   Then x occurs free in α (x is a **free variable** of α) iff $x \in \overline{h}(\alpha)$.

A **sentence** is a wff without free variables (usually the most interesting wff)

Scope of Quantification

E.g.,

$$\forall v_1 \exists v_2((v_3 \in v_4) \Leftrightarrow (\forall v_1(v_1 = v_2) \vee v_1 = v_3))$$

Formula Simplification

For readability, we write

- $\forall v_1(v_1 \neq 0 \Rightarrow \exists v_2 v_1 = Sv_2)$ for $\forall v_1((\neg = v_1 0) \Rightarrow (\neg \forall v_2(\neg = v_1 S v_2)))$
- $(\alpha \vee \beta)$ for $((\neg\alpha) \Rightarrow \beta)$
- $(\alpha \wedge \beta)$ for $(\neg(\alpha \Rightarrow (\neg\beta)))$
- $(\alpha \Leftrightarrow \beta)$ for $(\neg((\alpha \Rightarrow \beta) \Rightarrow (\neg(\beta \Rightarrow \alpha))))$
- $\exists x \alpha$ for $(\neg \forall x(\neg\alpha))$
- $u = t$ for $= ut$
- $2 < 3$ for < 23
- $2 + 2$ for $+22$
- $u \neq t$ for $(\neg = ut)$
- $u \not< t$ for $(\neg < ut)$

Also we may use $[,]$ besides $(,)$

Formula Simplification

We use the following convention (in order)

1. drop outermost parentheses
   E.g., $\alpha \Rightarrow \beta$ for $(\alpha \Rightarrow \beta)$
2. $\neg$, $\forall$, $\exists$ apply to as little as possible
   E.g., $\neg\alpha \vee \beta$ for $(\neg\alpha) \vee \beta$
3. $\wedge$, $\vee$, apply to as little as possible
4. Grouping is to the right for a repeated connective
   E.g., $\alpha \Rightarrow \beta \Rightarrow \gamma$ for $\alpha \Rightarrow (\beta \Rightarrow \gamma)$

Notational Convention

- Predicates: uppercase letters (also $\in, <$)
- Variables: v_i, u, v, x, y, z
- Functions: f, g, h (also S, $+$)
- Constants: a, b, ... (also 0)
- Terms: u, t
- Formulas: lowercase Greek letters, e.g., α, β
- Sentences: σ, τ
- Sets of formulas: uppercase Greek letters, e.g., Γ
- Structures: uppercase German letters, e.g., $\mathfrak{A}$, $\mathfrak{B}$