数理逻辑

杨睿之

复旦大学哲学学院

2019 年秋季

一阶逻辑的语言

符号：

- 逻辑符号
  - 括号：(,)
  - 命题联词：$\neg, \rightarrow$
  - 量词：$\forall$
  - 变元：$v_1, v_2, \ldots$

一阶逻辑的语言

符号:

- 非逻辑符号（参数符号）
  - 常数符号: $c_1, c_2, \ldots$ $(*)$
  - 函数符号: $f_1, f_2, \ldots$ $(*)$
  - 谓词符号: $P_1, P_2, \ldots$ $(*)$
  - 等词: $\approx$ （可有可无）
- $(*)$ 可以没有，也可以有无穷多
- 存在能行的函数 $g, h : \mathbb{N} \to \mathbb{N}^+$ 告诉我们每个 f_i 是 $g(i)$-元函数符号，每个 P_i 是 $h(i)$-元谓词符号

一阶逻辑的语言

各种一阶逻辑语言

- 集合论语言：$\{\approx, \in\}$
- 初等数论的语言：$\{\approx, 0, <, S, +, \cdot\}$
- 序关系的语言：$\{\approx, R\}$

我们说"给定一个一阶逻辑语言"就是规定各类参数符号的集合。

一阶逻辑的语言

各种一阶逻辑语言

- 集合论语言: $\{\approx, \in\}$
- 初等数论的语言: $\{\approx, 0, <, S, +, \cdot\}$
- 序关系的语言: $\{\approx, R\}$

我们说“给定一个一阶逻辑语言”就是规定各类参数符号的集合。

一阶逻辑的语言

各种一阶逻辑语言

- 集合论语言: $\{\approx, \in\}$
- 初等数论的语言: $\{\approx, 0, <, S, +, \cdot\}$
- 序关系的语言: $\{\approx, R\}$

我们说“给定一个一阶逻辑语言”就是规定各类参数符号的集合。

一阶逻辑的语言

各种一阶逻辑语言

- 集合论语言: $\{\approx, \in\}$
- 初等数论的语言: $\{\approx, 0, <, S, +, \cdot\}$
- 序关系的语言: $\{\approx, R\}$

我们说“给定一个一阶逻辑语言”就是规定各类参数符号的集合。

一阶逻辑的语言

各种一阶逻辑语言

- 集合论语言: $\{\approx, \in\}$
- 初等数论的语言: $\{\approx, 0, <, S, +, \cdot\}$
- 序关系的语言: $\{\approx, R\}$

我们说"给定一个一阶逻辑语言"就是规定各类参数符号的集合。

一阶逻辑的语言

项（term）

给定一个一阶逻辑语言 $\mathcal{L}$，定义 $\mathcal{L}$ 中的 **项** 为：

- 每个变元 v_i 是项

  令 $\mathcal{V}$ 是所有变元组成的集合

- 每个 $\mathcal{L}$ 中的常数符号是项

- 如果 $t_1, \ldots, t_n$ 是项并且 f 是 $\mathcal{L}$ 中 n 元函数符号，那么 $ft_1 \ldots t_n$ 也是项

令 $\mathcal{T}$ 是所有项组成的集合

一阶逻辑的语言

项（term）

给定一个一阶逻辑语言 $\mathcal{L}$，定义 $\mathcal{L}$ 中的 **项** 为：

- 每个变元 v_i 是项

  令 $\mathcal{V}$ 是所有变元组成的集合

- 每个 $\mathcal{L}$ 中的常数符号是项
- 如果 $t_1, \ldots, t_n$ 是项并且 f 是 $\mathcal{L}$ 中 n 元函数符号，那么 $ft_1 \ldots t_n$ 也是项

令 $\mathcal{T}$ 是所有项组成的集合

一阶逻辑的语言

项（term）

给定一个一阶逻辑语言 $\mathcal{L}$，定义 $\mathcal{L}$ 中的 **项** 为：

- 每个变元 v_i 是项

  令 $\mathcal{V}$ 是所有变元组成的集合

- 每个 $\mathcal{L}$ 中的常数符号是项

- 如果 $t_1, \ldots, t_n$ 是项并且 f 是 $\mathcal{L}$ 中 n 元函数符号，那么 $ft_1 \ldots t_n$ 也是项

令 $\mathcal{T}$ 是所有项组成的集合

一阶逻辑的语言

项（term）

给定一个一阶逻辑语言 $\mathcal{L}$，定义 $\mathcal{L}$ 中的 项 为：

- 每个变元 v_i 是项

  令 $\mathcal{V}$ 是所有变元组成的集合

- 每个 $\mathcal{L}$ 中的常数符号是项

- 如果 $t_1, \ldots, t_n$ 是项并且 f 是 $\mathcal{L}$ 中 n 元函数符号，那么 $f t_1 \ldots t_n$ 也是项

令 $\mathcal{T}$ 是所有项组成的集合

一阶逻辑的语言

例:

初等数论语言 $\{\approx, 0, <, S, +, \cdot\}$ 中的项

- v_3
- $S0$
- $+Sv_1SS0$，常记作 $Sv_1 + SS0$

一阶逻辑的语言

例:

初等数论语言 $\{\approx, 0, <, S, +, \cdot\}$ 中的项

- v_3
- $S0$
- $+Sv_1SS0$ ，常记作 $Sv_1 + SS0$

一阶逻辑的语言

合式公式（well-formed formula）

给定一个一阶逻辑语言 $\mathcal{L}$，定义 $\mathcal{L}$ 的 合式公式 （$\mathcal{F}$）如下：

- 如果 t_1, t_2 是项，那么 $t_1 \approx t_2$ 是公式（若 $\mathcal{L}$ 中有等词）
- 如果 $t_1, \ldots, t_n$ 是项且 P 是一个 n 元谓词符号，那么 $Pt_1 \ldots t_n$ 是公式

  称上述公式是 原子公式
- 如果 α, β 是合式公式，那么 $(\neg\alpha), (\alpha \to \beta)$ 也是
- 如果 α 是合式公式，x 是变元，那么 $\forall x\alpha$ 也是

一阶逻辑的语言

合式公式（well-formed formula）

给定一个一阶逻辑语言 $\mathcal{L}$，定义 $\mathcal{L}$ 的 合式公式 （$\mathscr{F}$）如下：

- 如果 t_1, t_2 是项，那么 $t_1 \approx t_2$ 是公式（若 $\mathcal{L}$ 中有等词）
- 如果 $t_1, \ldots, t_n$ 是项且 P 是一个 n 元谓词符号，那么 $Pt_1 \ldots t_n$ 是公式

  称上述公式是 原子公式
- 如果 α, β 是合式公式，那么 $(\neg\alpha), (\alpha \to \beta)$ 也是
- 如果 α 是合式公式，x 是变元，那么 $\forall x\alpha$ 也是

一阶逻辑的语言

合式公式（well-formed formula）

给定一个一阶逻辑语言 $\mathcal{L}$，定义 $\mathcal{L}$ 的 合式公式 （$\mathcal{F}$）如下：

- 如果 t_1, t_2 是项，那么 $t_1 \approx t_2$ 是公式（若 $\mathcal{L}$ 中有等词）
- 如果 $t_1, \ldots, t_n$ 是项且 P 是一个 n 元谓词符号，那么 $Pt_1 \ldots t_n$ 是公式

  称上述公式是 原子公式
- 如果 α, β 是合式公式，那么 $(\neg\alpha), (\alpha \to \beta)$ 也是
- 如果 α 是合式公式，x 是变元，那么 $\forall x\alpha$ 也是

一阶逻辑的语言

缩写规定：

- $\alpha \vee \beta =_{\text{abbr}} \neg\alpha \rightarrow \beta$
- $\alpha \wedge \beta =_{\text{abbr}} \neg(\alpha \rightarrow \neg\beta)$
- $\alpha \leftrightarrow \beta =_{\text{abbr}} (\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$
- $\exists x\alpha =_{\text{abbr}} \neg\forall x\neg\alpha$

我们习惯称 $\forall x$ 为 全称量词， 称 $\exists x$ 为 存在量词

一阶逻辑的语言

缩写规定：

- $\alpha \vee \beta =_{\text{abbr}} \neg\alpha \to \beta$
- $\alpha \wedge \beta =_{\text{abbr}} \neg(\alpha \to \neg\beta)$
- $\alpha \leftrightarrow \beta =_{\text{abbr}} (\alpha \to \beta) \wedge (\beta \to \alpha)$
- $\exists x\alpha =_{\text{abbr}} \neg\forall x\neg\alpha$

我们习惯称 $\forall x$ 为 全称量词，称 $\exists x$ 为 存在量词

自由出现与约束出现

例

给定 $\{a_i \mid i \in \mathbb{N}\}$

$$b = \sum_{i=0}^{k} a_i$$

自由出现与约束出现

例

给定 $\{a_i \mid i \in \mathbb{N}\}$

$$b = \sum_{i=0}^{k} a_i$$

自由出现与约束出现

例

给定 $\{a_i \mid i \in \mathbb{N}\}$

$$b = \sum_{j=0}^{k} a_j$$

自由出现与约束出现

对公式 α 递归定义 x 在 α 中自由出现：

- 当 α 是原子公式： x 在 α 中出现
- 当 $\alpha = \neg\beta$： x 在 β 中自由出现
- 当 $\alpha = \beta \to \gamma$： x 在 β 或 γ 中自由出现
- 当 $\alpha = \forall y\beta$： x 在 β 中自由出现且 $x \neq y$

x 在 $\forall x\alpha$ 中的出现称作 约束出现

自由出现与约束出现

对公式 α 递归定义 x 在 α 中自由出现：

- 当 α 是原子公式： x 在 α 中出现
- 当 $\alpha = \neg\beta$： x 在 β 中自由出现
- 当 $\alpha = \beta \to \gamma$： x 在 β 或 γ 中自由出现
- 当 $\alpha = \forall y\beta$： x 在 β 中自由出现且 $x \neq y$

x 在 $\forall x\alpha$ 中的出现称作 约束出现

自由出现与约束出现

我们称一个合式公式 α 是 **语句** （sentence），当且仅当没有变元在 α 中自由出现

代入

我们定义元语言中的一个表达方式 α^x_t

首先对项 u 递归定义 u^x_t

- 当 $u = y$:

$$u^x_t = \begin{cases} t & \text{若 } x = y \\ y & \text{否则} \end{cases}$$

- 当 $u = ft_1 \ldots t_n$: $u^x_t = f(t_1)^x_t \ldots (t_n)^x_t$

代入

例

- $(v_1)^{v_1}_{f v_1 v_2} = f v_1 v_2$
- $(v_2)^{v_1}_{c} = v_2$
- $(f v_1 g v_2)^{v_2}_{gc} = f v_1 g g c$

代入

例

- $(v_1)^{v_1}_{fv_1v_2} = fv_1v_2$
- $(v_2)^{v_1}_{c} = v_2$
- $(fv_1gv_2)^{v_2}_{gc} = fv_1ggc$

代入

例

- $(v_1)^{v_1}_{fv_1v_2} = fv_1v_2$
- $(v_2)^{v_1}_{c} = v_2$
- $(fv_1gv_2)^{v_2}_{gc} = fv_1ggc$

代入

其次对公式 α 递归定义 α_t^x

- 当 $\alpha = t_1 \approx t_2$: $\alpha_t^x = (t_1)_t^x \approx (t_2)_t^x$
- 当 $\alpha = Pt_1 \dots t_n$: $\alpha_t^x = P(t_1)_t^x \dots (t_n)_t^x$
- 当 $\alpha = \neg\beta$: $\alpha_t^x = (\neg\beta)_t^x = \neg\beta_t^x$
- 当 $\alpha = \beta \to \gamma$: $\alpha_t^x = (\beta \to \gamma)_t^x = \beta_t^x \to \gamma_t^x$
- 当 $\alpha = \forall y\beta$:

$$\alpha_t^x = (\forall y\beta)_t^x = \begin{cases} \alpha & \text{若 } x = y \\ \forall y\beta_t^x & \text{否则} \end{cases}$$

代入

例

- $(Pv_1 f v_2 \to \exists v_2 P v_1 v_2)^{v_1}_{c}$
- $(Pv_1 f v_2 \to \exists v_2 P v_1 v_2)^{v_2}_{c}$
- $((Pv_1 f v_2 \to \exists v_2 P v_1 v_2)^{v_1}_{v_2})^{v_2}_{v_1}$

代入

例

- $(Pv_1 f v_2 \to \exists v_2 P v_1 v_2)^{v_1}_{c}$
- $(Pv_1 f v_2 \to \exists v_2 P v_1 v_2)^{v_2}_{c}$
- $((Pv_1 f v_2 \to \exists v_2 P v_1 v_2)^{v_1}_{v_2})^{v_2}_{v_1}$

代入

例

- $(Pv_1 f v_2 \to \exists v_2 P v_1 v_2)^{v_1}_{c}$
- $(Pv_1 f v_2 \to \exists v_2 P v_1 v_2)^{v_2}_{c}$
- $((Pv_1 f v_2 \to \exists v_2 P v_1 v_2)^{v_1}_{v_2})^{v_2}_{v_1}$

一阶谓词逻辑的希尔伯特系统

一阶谓词逻辑的希尔伯特系统

定义 (全称概括)

称公式 φ 是公式 ψ 的 全称概括 ，当且仅当存在自然数 n 和变元 $x_1, \ldots, x_n$ 使得

$$\varphi = \forall x_1 \ldots \forall x_n \psi$$

一阶谓词逻辑的希尔伯特系统

定义 (素公式)

我们称原子公式或形如 $\forall x\beta$ 的公式为 素公式

令 $\langle\beta_1,\beta_2,\ldots\rangle$ 是对所有素公式的枚举。我们定义一个一阶逻辑公式 α 的命题逻辑公式对应 α^P ：

- 当 $\alpha=\beta_i$ 是一个素公式： $\alpha^P=\beta_i^P=A_i$
- 当 $\alpha=\neg\beta$： $\alpha^P=\neg\beta^P$
- 当 $\alpha=\beta\to\gamma$： $\alpha^P=\beta^P\to\gamma^P$

一阶谓词逻辑的希尔伯特系统

公理：一阶逻辑希尔伯特系统的公理是由下列公式的全称概括组成的

1. 对应的命题逻辑公式 α^P 是重言式的一阶逻辑公式 α
2. $\forall x\alpha \to \alpha_t^x$，其中项 t 可以在 α 中替代 x
3. $\forall x(\alpha \to \beta) \to (\forall x\alpha \to \forall x\beta)$
4. $\alpha \to \forall x\alpha$，其中变元 x 不在 α 中自由出现

一阶谓词逻辑的希尔伯特系统

公理：一阶逻辑希尔伯特系统的公理是由下列公式的全称概括组成的

1. 对应的命题逻辑公式 α^P 是重言式的一阶逻辑公式 α
2. $\forall x\alpha \rightarrow \alpha_t^x$，其中项 t 可以在 α 中替代 x
3. $\forall x(\alpha \rightarrow \beta) \rightarrow (\forall x\alpha \rightarrow \forall x\beta)$
4. $\alpha \rightarrow \forall x\alpha$，其中变元 x 不在 α 中自由出现

一阶谓词逻辑的希尔伯特系统

公理：一阶逻辑希尔伯特系统的公理是由下列公式的全称概括组成的

1. 对应的命题逻辑公式 α^P 是重言式的一阶逻辑公式 α
2. $\forall x\alpha \rightarrow \alpha_t^x$，其中项 t 可以在 α 中替代 x
3. $\forall x(\alpha \rightarrow \beta) \rightarrow (\forall x\alpha \rightarrow \forall x\beta)$
4. $\alpha \rightarrow \forall x\alpha$，其中变元 x 不在 α 中自由出现

一阶谓词逻辑的希尔伯特系统

公理：一阶逻辑希尔伯特系统的公理是由下列公式的全称概括组成的

1. 对应的命题逻辑公式 α^P 是重言式的一阶逻辑公式 α
2. $\forall x\alpha \to \alpha_t^x$，其中项 t 可以在 α 中替代 x
3. $\forall x(\alpha \to \beta) \to (\forall x\alpha \to \forall x\beta)$
4. $\alpha \to \forall x\alpha$，其中变元 x 不在 α 中自由出现

一阶谓词逻辑的希尔伯特系统

公理：一阶逻辑希尔伯特系统的公理是由下列公式的全称概括组成的

若语言中含有等词，则还有

5. $x \approx x$
6. $x \approx y \to \alpha \to \alpha'$，其中 α 为原子公式，且 α' 是将 α 中若干个 x 的出现替换为 y 所得到的公式

一阶谓词逻辑的希尔伯特系统

不可替代的例子

令 $\alpha = \exists y x \not\approx y$。分别考虑

- $\forall x\alpha \to \alpha_z^x$
- $\forall x\alpha \to \alpha_y^x$

如果我们希望定义 项 t 可以在 α 中替代 x 为 “替换后 t 中的变元不会被 α 中已有的量词抓住”，我们该怎样严格地给出定义?

一阶谓词逻辑的希尔伯特系统

不可替代的例子

令 $\alpha = \exists y x \not\approx y$。分别考虑

- $\forall x\alpha \to \alpha^x_z$
- $\forall x\alpha \to \alpha^x_y$

如果我们希望定义 项 t 可以在 α 中替代 x 为 “替换后 t 中的变元不会被 α 中已有的量词抓住”，我们该怎样严格地给出定义？

习题

- 分别写出一阶谓词逻辑 合式公式 自上而下与自下而上的定义
- 定义递归函数 $F : \mathcal{F} \times \mathcal{V} \to P(\mathbb{N})$ 使得 $F(\alpha, x) = \{i_1, \ldots, i_k\}$ 当且仅当 x 在 α 中第 $i_1, \ldots, i_k$ 个出现是自由的出现
- 3.1.2, 3.1.3, 3.2.2（对每个 3.2.1 中公式）
- 4.1.1

下期预告

一阶逻辑希尔伯特公理系统及其元定理