一阶逻辑初步

哲学数学计算机中的逻辑课程 (2016 年秋)

王彦晶
北大哲学系

2016 年 12 月 22 日

一阶逻辑的语言和语义

一些重要的元定理

一阶逻辑的语言和语义

一阶逻辑的语言 (有很多个)

先确定一个符号表 (signature, non-logical symbols):

- 变元 (variable) 符号 $x, y \ldots$
- 常元 (constant) 符号 $c, d \ldots$
- 谓词 (predicate) 符号 $P, Q \ldots$ 每个有一个相应的元数 (arity)
- 函数 (function) 符号 $f, g \ldots$ 每个有一个相应的元数

给定这个符号表, (带等词的) 一阶逻辑的语言定义为:

$$\begin{aligned} \text{项 (term) } t &::= x \mid c \mid f(\vec{t}) \\ \text{公式 (formula) } \varphi &::= t \approx t \mid P\vec{t} \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \forall x\varphi \end{aligned}$$

可以定义 $\exists$ 为 $\neg\forall\neg$

例如 $x \approx f(c)$, $\exists x f(c) \approx x$, $\forall x \exists y P(x, y)$ 等等. 有时也用 $\exists! x\varphi$ 表示 $\exists x \varphi(x) \wedge \forall y \forall z(\varphi(y) \wedge \varphi(z) \to y \approx z)$. $\forall x \forall y(F(x) \wedge F(y) \to (((H(x) \wedge H(y) \to Sim(x, y) \wedge (\neg H(x) \wedge \neg H(y) \to \neg Sim(x, y))$ 托尔斯泰.

变元的约束

一个公式里, 某个变元的某次出现不在任何一个量词的辖域 (scope) 里, 则称其是自由的 (free). $\forall x P(x, f(y)) \wedge \exists y Q(x, y)$ 里第二个 x 和第一个 y 是自由的. 直观上讲, 该公式可以等价的写成 $\forall z P(z, f(y)) \wedge \exists u Q(x, u)$ (变元易字).

- 开公式 (open formulas): 有变元自由出现的.
- 闭公式 (closed formulas, sentences): 没有自由出现的.

4 年前为真的例子 (为了简化), 怎么让计算机理解和推理?

周迅的前男友窦鹏是窦唯的堂弟；窦唯是王菲的前老公；周迅的前男友宋宁是高原的表弟；高原是窦唯的前任老婆；周迅的前男友李亚鹏是王菲的现任老公；周迅的前男友朴树的音乐制作人是张亚东；张亚东是王菲的前老公窦唯的妹妹窦颖的前老公，也是王菲的音乐制作人；张亚东是李亚鹏前女友瞿颖的现男友。请问下列说法不正确的是:

- 王菲周迅是情敌关系;
- 瞿颖王菲是情敌关系;
- 窦颖周迅是情敌关系;
- 瞿颖周迅是情敌关系.

谓词: 前亲密关系 EX, 现亲密关系 NOW, 一种情敌关系可被定义为: $QD(x, y) := \exists z((NOW(x, z) \wedge EX(y, z)) \vee (NOW(y, z) \wedge EX(x, z)))$.

4 年前的例子 (为了简化): 贵圈真乱

只能用父子关系谓词 FZ, 兄弟关系谓词 XD, 前任函数 ex, 父亲函数 father, 年纪函数 age, 以及大小谓词 $<$, 怎么写周迅的前男友是王菲的前老公的堂弟:

$\exists x\,((FZ(x,(ex(c_{zx}))) \wedge XD(x, father(ex(c_{wf}))) \wedge age(ex(c_{zx})) < age(ex(c_{wf}))$

进一步可以采用更基本的性别谓词, 手足 (sibling) 谓词, 父母子女 (parent) 谓词, 年龄函数及大小谓词定义所有的关系: 如爸爸是你父母中的男性等.

例子: 一阶算数的语言符号表: $(0, <, S, +, \cdot)$:

- 0 是一个常元,
- $<$ 是二元谓词符号 “小于”
- S 是 “后继”(successor) 函数符号, 表示自然数, 如 $S0$ 表示 1.
- $+$ 是二元函数符号 “加”, $\cdot$ 是二元函数符号 “乘”

能表达的举例:

- 0 不是任何自然数的后继: $\forall x \neg Sx \approx 0$.
- 任何数加 0 还是其本身: $\forall x(x + 0 \approx x)$
- 一个数加另外一个数的后继等于这两个数的和的后继: $\forall x \forall y((x + Sy) \approx S(x + y))$
- x 是素数: $S0 < x \wedge \forall y \forall z(y < x \wedge z < x \to \neg y \cdot z \approx x))$
- 对 $\varphi(x)$ 的数学归纳法: $\varphi(0) \wedge \forall x(\varphi(x) \to \varphi(S(x))) \to \forall x \varphi(x)$.

有谓词变元, 并对其做量化的是二阶语言, 可表达更一般的数学归纳法: $\forall P(P(0) \wedge \forall x(P(x) \to P(S(x)) \to \forall x P(x))$.

ZEMELO 定理的简单证明

两个人 的有穷深度 的完美信息 的总有输赢 的博弈必有一个玩家有必胜策略.

我们之前是通过画出有穷树, 从底下一点点做标注证明的. 其实还有更简单的证明: 假设游戏深度为 n, 什么叫玩家 1 有必胜策略?

$\exists x_1 \forall y_1 \exists x_2 \forall y_2 \cdots \exists x_n \forall y_n WIN_1(x_1, y_1, \cdots x_n, y_n)$.

什么叫第一个游戏者没有 必胜策略?

$\neg \exists x_1 \forall y_1 \exists x_2 \forall y_2 \cdots \exists x_n \forall y_n WIN_1(x_1, y_1, \cdots x_n, y_n)$.

根据 $\exists$ 和 $\forall$ 的关系, 等价于:

$\forall x_1 \exists y_1 \cdots \forall x_n \exists y_n \neg WIN_1(x_1, y_1, \cdots x_n, y_n)$.

因为这是输赢游戏 (游戏结束时不是你赢就是我赢), 所以:

$\forall x_1 \exists y_1 \cdots \forall x_n \exists y_n WIN_2(x_1, y_1, \cdots x_n, y_n)$. 这就在说玩家 2 有必胜策略.

一阶逻辑的结构与模型

给定具体的一阶逻辑语言, 它的一个结构 (structure) $\mathfrak{A} = \langle D, I \rangle$:

- D 是一个非空的个体域或论域 (Domain) (也记为 $|\mathfrak{A}|$)
- I 是对常元符号, 谓词符号, 函数符号的解释 (Interpretation):
  - I 给每个常元符号 c 一个 D 里的个体 (也记为 $c^{\mathfrak{A}}$)
  - I 给每个 n 元谓词符号 P 一个 D 上的 n 元关系 (也记为 $P^{\mathfrak{A}}$)
  - I 给每个 n 元函数符号一个 D 上的 n 元函数 (也记为 $f^{\mathfrak{A}}$)

一个模型 $\mathfrak{M}$ 是一个结构加一个指派 (assignment) $\langle \mathfrak{A}, \sigma \rangle$, 指派 σ 告诉我们变元 x 们指的是个体域里的哪个东西.

如果我们只关心没有自由变元的句子的真假, 为啥还要指派? 用于语义定义.

一阶逻辑的语义

给定一个模型 $\mathfrak{M} = \langle \mathfrak{A}, \sigma \rangle$, 可以给所有的项解释了 (解释成个体)
可以推广 σ 的定义域到所有项 (本来只对 x 们):

- $\sigma(c) = c^{\mathfrak{A}}$
- $\sigma(f(t_1 \cdots t_n)) = f^{\mathfrak{A}}(\sigma(t_1), \ldots \sigma(t_n))$.

语义定义在模型 $\mathfrak{M} = \langle \mathfrak{A}, \sigma \rangle$ 上:

$$
\begin{array}{rcl}
\mathfrak{A}, \sigma \vDash t \approx t' & \Leftrightarrow & \sigma(t) = \sigma(t') \\
\mathfrak{A}, \sigma \vDash P(t_1 \cdots t_n) & \Leftrightarrow & (\sigma(t_1), \cdots, \sigma(t_n)) \in P^{\mathfrak{A}} \\
\mathfrak{A}, \sigma \vDash \neg\varphi & \Leftrightarrow & \mathfrak{A}, \sigma \nvDash \varphi \\
\mathfrak{A}, \sigma \vDash \varphi \wedge \psi & \Leftrightarrow & \mathfrak{A}, \sigma \vDash \varphi \text{ 且} \mathfrak{A}, \sigma \vDash \psi \\
\mathfrak{A}, \sigma \vDash \forall x \varphi & \Leftrightarrow & \text{对所有的} a \in |\mathfrak{A}|\ \mathfrak{A}, \sigma[x \mapsto a] \vDash \varphi
\end{array}
$$

$\sigma[x \mapsto a]$ 就是改变 σ 对 x 的解释为 a, 其他不变.

可以定义有效式 为在任何模型上都为真的公式, 如:
$\exists x \forall y \varphi \to \forall y \exists x \varphi$. 但是反过来可不有效.

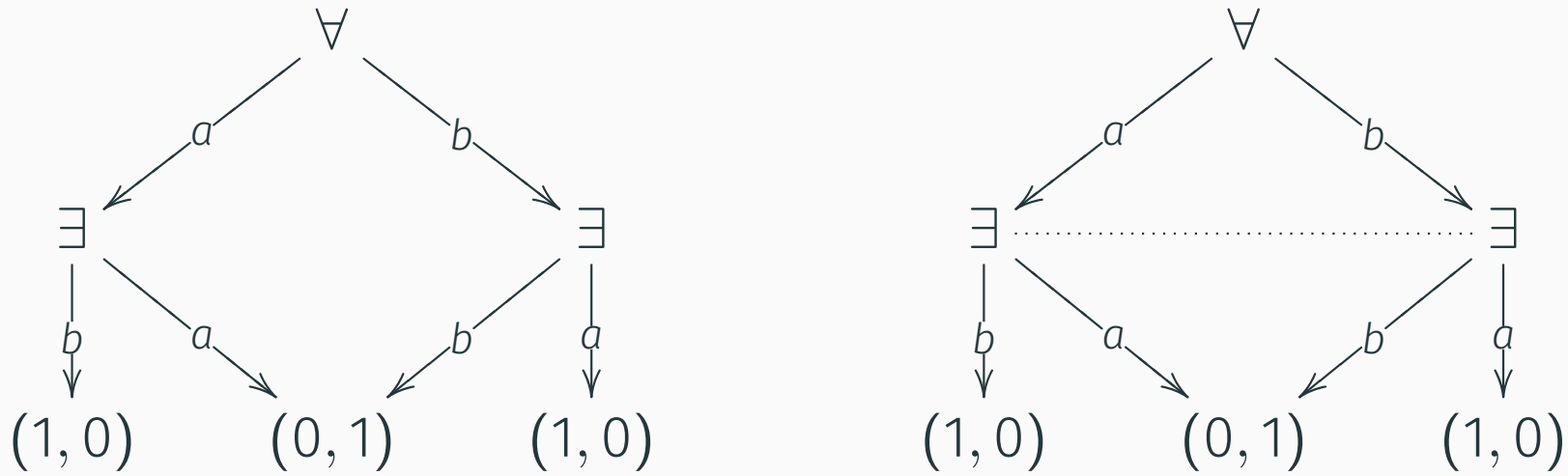
给定结构和指派, 对一阶逻辑公式 φ 我们可以玩一个游戏

给定结构 $\mathfrak{A}$ 和指派 σ:

- 游戏者: 支持者 (Proponent) 和反对者 (Opponent)
- 游戏的状态 $\langle \eta, \psi \rangle$, 其中 η 是一个指派, ψ 是 φ 的子公式, 初始状态是 $\langle \sigma, \varphi \rangle$.
- 游戏的规则:
  - P 首先断言公式 φ 在 $\mathfrak{A}, \sigma$ 上是真的.
  - 如果当前公式是 $\psi \vee \psi'$ 则 P 可以选择 ψ 或 ψ 其中一个继续
  - 如果当前公式是 $\neg\psi$ 则 P 和 O 的角色互换从 ψ 继续
  - 如果当前公式是 $\forall x \psi$ 则 O 选择一个 $a \in |\mathfrak{A}|$ 并且改变游戏状态为 $\langle \sigma[x \mapsto a], \psi \rangle$ 继续.
- 游戏的输赢条件: 状态是 $\langle \eta, \psi \rangle$, ψ 是原子公式且 $\mathfrak{A}, \eta \vDash \psi$. 则 (当前的)P 赢. 反之则 (当前的)O 赢.

可以证明: $\mathfrak{A}, \sigma \vDash \varphi \iff$ 在相应的游戏里 P 有一个必胜策略.

非完美信息 (Imperfect Information Games): 玩家不一定知道其所在的游戏状态:



$\forall x \exists y\ x = y$ 有效 *v.s.* $\forall x \exists y_{/\forall x} x = y$ 不有效

量词为啥一定要线性排序? $\forall x \exists y_{/\forall x} x = y$ 说 y 的选择不依赖于 x.

可以用这样的游戏给这样的公式 (三值) 语义: Player 1 有必胜策略就是真, Player 2 有必胜策略就是假, 也可能两个人都没有必胜策略, 这时公式就既不真也不假.

模态逻辑的真值游戏

给定点模型M,w 和 φ:

- 游戏者: 支持者 (Proponent) 和反对者 (Opponent)
- 游戏的状态 $\langle v, \psi \rangle$, 其中 v 在 $\mathcal{M}$ 中, ψ 是 φ 的子公式, 初始状态是 $\langle w, \varphi \rangle$.
- 游戏的规则:
  - P 首先断言公式 φ 在 $\mathcal{M}, w$ 上是真的.
  - 如果当前公式是 $\psi \vee \psi'$ 则 P 可以选择 ψ 或 ψ 其中一个继续
  - 如果当前公式是 $\neg\psi$ 则 P 和 O 的角色互换从 ψ 继续
  - 如果当前公式是 $\Box\psi$ 则 O 选择一个 v' 使得 vRv', 并且改变游戏状态为 $\langle v', \psi \rangle$ 继续.
- 游戏的输赢条件: 状态是 $p \in \mathbf{PV}$, 若 $V(p) = 1$ 则 (当前的)P 赢, 反之则 (当前的)O 赢.

可以证明: $\mathcal{M}, w \vDash \varphi \iff$ 在相应的游戏里 P 有一个必胜策略.

从一阶逻辑的角度看模态逻辑

$$\mathcal{M}, w \vDash \Box\varphi \quad \Leftrightarrow \quad \text{对所有的} v, \text{如果} wRv \text{则} \mathcal{M}, v \vDash \varphi$$

按照模态词的语义, $\Box p$(相对于 w) 可以看成一阶公式: $\forall v(w\mathsf{R}v \to \mathsf{P}v)$ 其中 R 是一个二元谓词, P 是一个一元谓词.

克里普克模型 $\{W, R, V\}$ 可以看成是特殊的一阶结构: W 是个体域, R 这个二元关系是 R 的解释, $V(p)$ 是 P 的解释.

可以证明, 所有模态逻辑公式 φ 都可以翻译成这样语言下的一阶逻辑公式 $t(\varphi)$ 使得对任意克里普克点模型 $\mathcal{M}, s$:

$$\mathcal{M}, s \vDash_{ML} \varphi \iff \mathcal{M}, \sigma[x \mapsto x] \vDash_{FOL} t(\varphi)$$

van Benthem 证明了: 模态逻辑恰好是一阶逻辑在互模拟关系下保持真值的片段. 不过在谈论框架的意义上二者的表达能力不可比较 (为什么?).

一些重要的元定理

重要的元理论结果

- 演绎定理
- 紧致性定理
- 相对于特定推理系统的可靠性与完全性定理

与模态逻辑的区别:

- 没有有穷模型性 只有无穷模型, 如:
  $\forall x \exists y(xRy) \wedge \forall x \forall y \forall z(xRy \wedge yRz \rightarrow xRz) \wedge \forall x \neg xRx$.
- 同构 Isomorphism (比互模拟严格得多) 保持一阶逻辑闭公式的真值, 但反过来很不对, 例如有与自然数算数结构 $(\mathbb{N}, 0, S, <, +, \cdot)$ 不同构的但满足同样一阶闭公式的 “非标准模型”.

Löwenheim-Skolem 定理 (大致): 语句集有无穷模型则有任意大的无穷模型. 总可以加些东西 $\{\neg c_i \approx c_j \mid j \not\approx j, i, j \in \mathbb{N}\}$

一阶逻辑的推理系统, 规则就是一条分离规则 MP

公理:

- 命题逻辑的三条公理模式
- $\forall x\varphi \to \varphi[t/x]$ (用 t 统一替代 φ 的 x, t 要对 x 在 φ 中代入自由, 见下面的直观想法.)
- $\varphi \to \forall x\varphi$ (x 在 φ 中不自由的出现)
- $\forall x(\varphi \to \psi) \land \forall x\varphi \to \forall x\psi$ (类似于模态逻辑 K 公理)
- $t \approx t$
- $(t_1 \approx t_1' \land \cdots t_n \approx t_n') \to f(t_1 \cdots t_n) \approx f(t_1' \cdots t_n'))$
- $(t_1 \approx t_1' \land \cdots t_n \approx t_n') \to (P(t_1 \cdots t_n) \to P(t_1' \cdots t_n'))$
- 对上面的公理模式 φ, 做任意全称概括得到的 $\forall x_1 \cdots \forall x_n \varphi$ (类似模态逻辑必然化规则), 这与上面的 $\varphi \to \forall \varphi$ 不一样.

$\forall x \exists y \neg(x \approx y)$, 这时不能用 y 去替换 $\exists y \neg(x \approx y)$ 里的 x, 会得到 $\forall x \exists y \neg(x \approx y) \to \exists y \neg(y \approx y)$. y 对 x 在 $\exists y x \not\approx y$ 里代入不自由.

完全性定理的思路 (Henkin method)

完全性: $\Gamma \vDash \varphi \implies \Gamma \vdash \varphi$.

- 把完全性转化为一致的公式集可满足 (语形到语义).
- 给个一致的公式集, 补完它的信息 (极大一致集), 然后根据这个集合去造模型.
  - 对 $\exists x \varphi(x)$ 这样的公式找到实例, 在语言里加一堆新的常项 c, 补上公式 $\exists x \varphi(x) \to \varphi(c)$ 扩充成极大一致集 Δ(相对于新的大语言). 这本质上消去了不好处理的量词.
  - 根据这个极大一致集造模型 (结构 + 指派):
    - 个体域是大语言里所有项的集合
    - 谓词 P 解释成 $\{(t_1, \cdots t_n) \mid Pt_1 \cdots t_n \in \Delta\}$, 类似处理函数符号 f
    - 常项 c 解释成自己, 指派变元 x 给自己.
  - 处理等词相关的麻烦 (考虑项的等价类作为新的个体域).
  - 证明我们造的东西是新语言极大一致集 Δ 的模型.
  - 证明新语言的极大一致集中的老语言的部分还是一致的.
  - 忽略新语言相关的部分得到老语言一致集的模型.

判定性问题

问题: 任给一个公式, 是不是有一种一步步的 “机械” 可操作的办法, 一定在有穷步之内判定这个公式是不是有效式 (放之四海模型皆准).

对于有经典否定词 $\neg$ 的逻辑, 问这个问题等价于问任意一个公式是不是可以被满足: φ 有效 iff $\neg\varphi$ 不可被满足而且 φ 可满足 iff $\neg\varphi$ 不有效.

命题逻辑和模态逻辑公式的有效性 (或可满足性) 都是可判定的.

那一阶逻辑呢? 如果可以判定, 那很大多数数学家理论上就可以休息了: 用有穷多个公理表示数学理论, 然后判断这些公理是否蕴含某个特定结论.

可惜一阶逻辑是不可判定的. 但怎么证明不可判定性?

要定义机械可操作的边界

最直观的图灵机: 一个抽象的机器, 它有有穷多个内在的状态, 可以读写一个 (两边无穷长) 的纸带上的 (有穷多种) 符号, 并且根据一些规则该变读写头的位置改变内部状态等. 抽象的说 $\langle Q, \Sigma, \delta \rangle$

- Q 是有穷多个状态的集合
- Σ 有穷多个能读写的符号的集合
- δ 是有穷多个状态转换的指令集, 具有如下形式 ($q \in Q, a \in \Sigma$):
  - $qaa'q'm$ 内在状态 q 的时候如果读 a 就擦掉写 a', 改变状态到 q', 以 m 方式动作读写头
  - m 可以是不动, 可以左移 L, 或者右移 R.

  可要求至多只有一个指令以 qa 开头 (确定性图灵机).

除此之外有一个初始状态 q_0, 开始时读写头在左起第一个有字符的格子上. 如果到某个时候没有指令可做了就停了. 如果最后停在一些特殊状态 (称为接收状态) 上, 则称接收纸带上最开始的字符串.

例子(自己试试看)

https://turingmachinesimulator.com/

祝大家圣诞快乐!