中国科学技术大学

University of Science and Technology of China

人工智能讲义

知识库与知识推理

May 8, 2018

Outline

1. 知识库和知识推理在大数据时代的重生
2. 命题逻辑
3. 知识推理
4. 一阶逻辑
5. 其它逻辑

大数据的概念

假设输入是一个向量，不妨设 $X = (X_1, X_2, \ldots, X_n)$，第 i 个分量的值域大小记为 $|X_i|$，其第 j 个取值为 v_{ij}，则如下表格完全表示一个函数：

X_1	X_2	$\ldots$	X_n	$f(X_1, X_2, \ldots, X_n)$
v_{11}	v_{21}	$\ldots$	v_{n1}	y_1
v_{11}	v_{21}	$\ldots$	v_{n2}	y_2
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
$v_{1\|X_1\|}$	$v_{2\|X_2\|}$	$\ldots$	$v_{n\|X_n\|}$	$y_{\|X_1\|\|X_2\|\ldots\|X_n\|}$

Table: n 元函数的表示

当 n 非常大的时候，就有了大数据

- 最本质的特点“列多”，一行一个数据，大数据就是一行数据“大”，体现在“列多”，万、十万、百万、千万、亿个“列”；
- 因为列多，所以分析就需要样本多，才有意义。样本多就是行多，行多 + 列多就得到 Volume 大，这就是通常讲的大数据的特征（4、5V）中的一个；
- 因为列多，描述一个数据对象从不同角度，用不同手段得到的各种类型的数据，所以就有了 Variety;
- 因为列多，很多不相关的、不重要的数据特征被采集进来，造成价值密度低，所有就有了 Value；
- 因为数据 volume 大，所以对处理速度有要求，就有了 Velocity

大数据分析的机器学习思路

给定一个多样性属性的表格数据集："超级大表格"，解决思路：

- 假设所有的属性为同一个任务服务，
- 利用深度学习（主要是深度神经网络，自编码器等）先做特征提取，
- 然后分类或预测；

存在问题

- 假设所有属性为同一个任务服务，在现实中容易满足吗？

机器学习的局限性

谁来就某个特定的“任务”给出训练数据集?

- 包括任务目标，任务相关的特征集（哪些列）;
- 不相关的列太多了，会影响发现规律；大物理学家都是把握主要因素，忽略次要因素，物理建模的高手;
- 假设：一个 Agent M 利用现有的、将来的各种感知技术把所有相关的，无关的数据都收集来了，做成一张超级“大表格”，这是大数据的最本质特征;
- 还有一个 Agent S 能够针对任何给定的任务从“大表格”中选择某些列构成一个小表格，传递给机器学习算法，寻找一个函数 f_i，这是机器学习所擅长的工作
- Agent S 为一个任务选择出最优的小表格是 NP 难的；大数据时代我们总是有意忽略或没意识到这个缺省的约束条件，认为面临的大数据都是为某个单一任务服务的。

知识、知识库与逻辑

知识

- 一条知识就是关于客观世界的一个断言，称为一个“句子”/sentence;
- 不能从其它任何句子中推导出来的句子，称为“公理”;
- 描述知识的语言称为“知识表示语言”

知识库

- 知识的集合，断言的集合，句子集合。

逻辑

- 利用知识和知识库获得新知识的方法。

辉煌的历史

- 统治了 1990 之前的人工智能研究；

AI 中传统逻辑学派的致命缺陷

- 不能搞不确定性推理，只能做确定性推理。当 Bayes 网络出来后，AI 转向了
- 完全基于规则，不允许用数据来讲话。当机器学习流行后，其统治地位被替代了

逻辑的强大之处

- 可解释性和表达能力，机器学习望尘莫及

大数据时代，融合逻辑与机器学习

- 机器学习的结果解释称为“知识”，再运用知识推理的方式获取新知识。

逻辑带来的大数据分析的希望

知识库与知识推理

- 选取表格部分行与列，做成小表格，以此为输入用机器学习，完成一个小任务，获得一个函数 f_i
- 重复上述步骤，获得多个 $f_1, f_2, \ldots$
- 将所有的 f_i 用某种语言（包括自然语言）表达或描述出来；每个 f_i 可能是一条知识或者多条知识；
- 利用演绎推理实现更多知识的获取。

存在问题：

- 框架上的问题：谁来设置或设计“小任务”？分而治之，大任务—>小任务，迈出了求解问题的重要一步；
- 技术上的问题：如何将机器学习的输出函数 f_i 转化为一条或多条“知识”？

自然语言：好的推理的例子

- 张三比李四跑得快；
- 李四比王五跑得快；
- 所以张三比王五跑得快。

自然语言：糟糕的例子

- 1 分钱总比“没有”好；
- “没有”比世界和平更好；
- 所以 1 分钱比世界和平好。

自然语言充满二义性和模糊性；但这也是自然语言的“美”，没它就少了很多“文学”的美。

自然语言的歧义：例子

- 看到你那年才 8 岁。
- 一边站着一位同学，守卫着校门。
- 学校来了三个医院的医生。
- 他走了一个多钟头了。
- 思维科学。
- 2018 年元月 1 日前必须交货。
- 这苹果不大好吃。
- 空房间做什么用?
- 我想起来了。

语言

自然语言

- 中文
- 英文
- ...

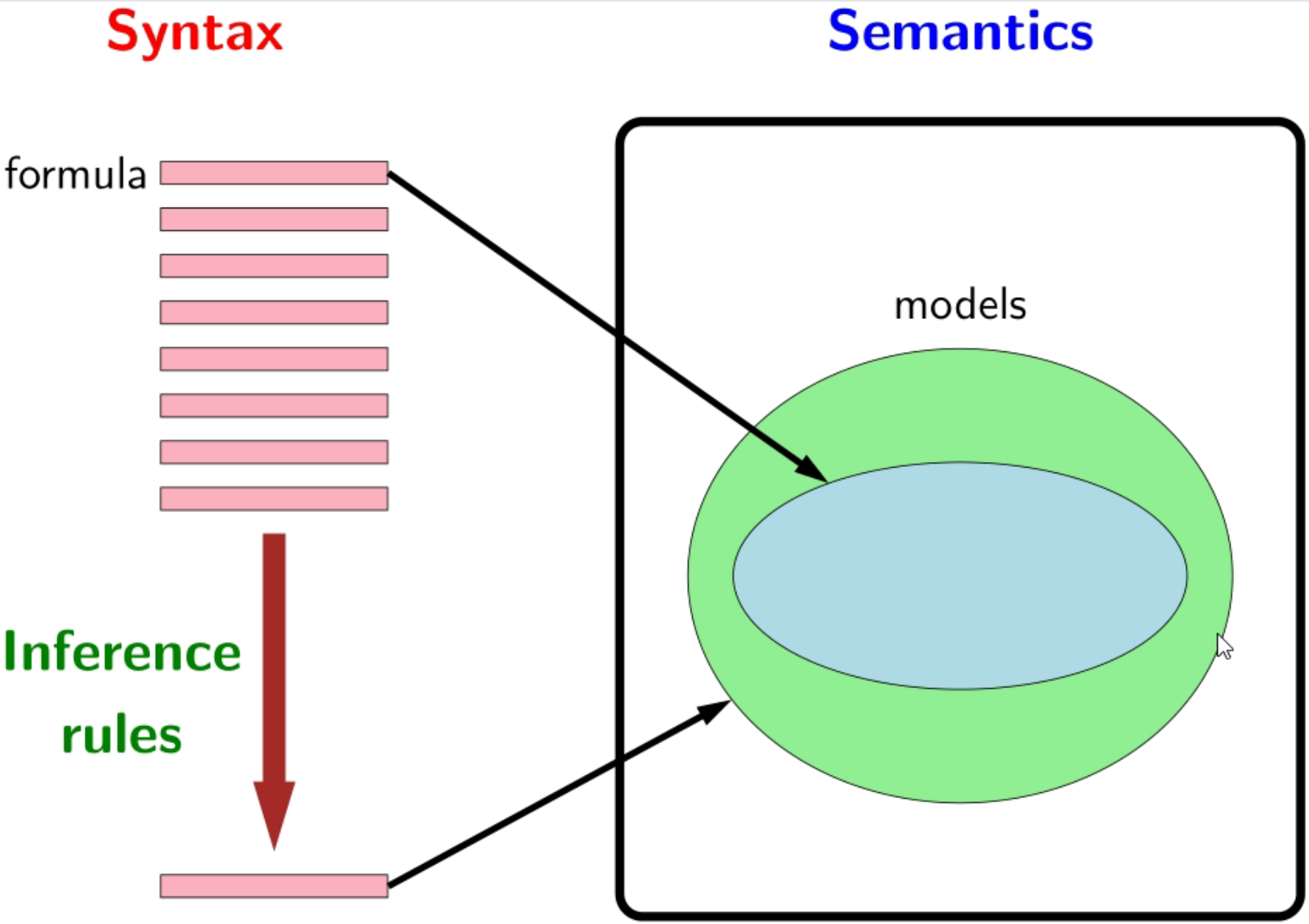
程序设计语言

- C，C++
- Java
- ...

逻辑语言

- 命题逻辑
- 一阶逻辑
- ...

语言是表达的工具和机制，有了语言就可以应用在逻辑上。逻辑两个任务：用知识描述世界，用知识进行推理。


Syntax
Semantics
formula
models
Inference
rules

逻辑的组成要素

语法/synatx

- 有效公式的集合
- 一组变量，公式描述变量间的关系，
- 例如：$x_1, x_2, \ldots, x_n, x_1 \bigwedge x_3, \ldots$

语义/semantics

- 给每个变量赋一个值，公式描述的关系可能成立也可能不成立
- 公式描述的关系，在变量赋值后（给予了物理意义），也就有了蕴含的“语义”
- 上例中 $x_1 = 1$ 表示“天黑”，$x_3 = 1$ 表示“看不见”，当 x_1, x_3 取不同的值时，$x_1 \bigwedge x_3$ 算出不同的值，具有正确或错误的“语义”

推理规则/inference rules

- 获得更多的公式，但是已有公式的语义不会发生改变；（语义是约束条件）
- 上例中，给定语义解释不变，同时语义 $x_1 \bigwedge x_3$ 维持不变（天黑看不见），那是否有公式 x_1 呢?

相同的语法，不同的语义

- x/y 在 python2 中是整除；
- x/y 在 python3 中是通常意义的除法；

不同的语法，相同的语义

- $x + y \Leftrightarrow y + x$

命题逻辑的语法：

- 命题符号/命题变量/原子公式：$A, B, C, \ldots$，千变万化，基本没有约束
- 逻辑连接符：$\wedge \neg \rightarrow \vee \leftrightarrow$，限定为这 5 种
- 公式构造方法：若 f, g 是公式，那么如下皆公式：（递归定义）
  - 逻辑非：$\neg f$
  - 逻辑与/合取式：$f \wedge g$
  - 逻辑或/析取式：$f \vee g$
  - 蕴含式：$f \rightarrow g$，if ... then ...
  - 等价式：$f \leftrightarrow g$，当且仅当
- 正确的语法：$A, \neg A, \neg B \rightarrow C, \neg\neg C$
- 错误的语法：$A \neg B, A + B$
- 理解语法：给出了符号集合以及符号间的连接方式（约束条件）

命题逻辑的语义：

- 在语法中给每个符号赋予了“初步的”物理含义：比如 x_1 表示“红色”，当 $x_1 = true$ 时，表示 x_1 就是正确表示了红色
- 语义就是在给每个符号具体的“真值指派”后，公式形成的约束条件集合。
- “模型” w 在命题逻辑中就是对命题符号的一个真值指派。

模型的例子

- 三个命题符号： A, B, C
- 一共有 $2^3 = 8$ 种可能的、不同模型

$$\{A : true, B : true, C : true\}$$
$$\{A : true, B : true, C : false\}$$
$$\{A : true, B : false, C : true\}$$
$$\{A : true, B : false, C : false\}$$
$$\{A : false, B : true, C : true\}$$
$$\{A : false, B : true, C : false\}$$
$$\{A : false, B : false, C : true\}$$
$$\{A : false, B : false, C : false\}$$

命题逻辑的解释函数

解释函数：语义通过解释函数给出

- 输入：公式 f，模型 w
- 处理：函数 $I(f, w)$
- 输出：
  - t 或 1，表示 w 满足 f
  - f 或 0，表示 w 不满足 f

例子

- A：天黑; B: 看不见；公式 $f: A \rightarrow B$：如果天黑，那么看不见
- $w = \{A = true, B = false\}$，天黑，非看不见/看见
- $I(f, w) = false$

解释函数的意义：

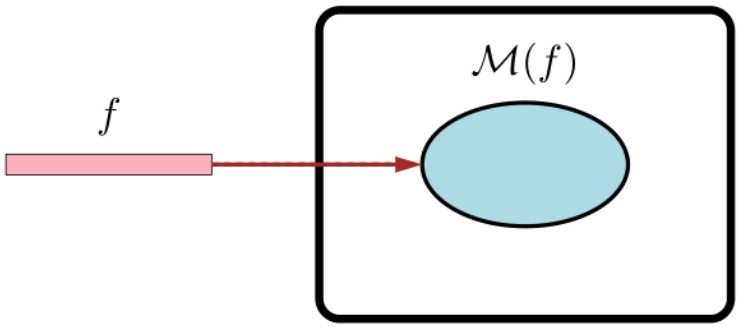
- 对任何命题符号 p: 令 $I(p, w) = w(p) \in \{true, false\}$;
- 递归定义公式的解释函数：给定两个公式 f, g，5 种连接符的语义解释如下：

$I(f, w)$	$I(g, w)$	$I(\neg f, w)$	$I(f \wedge g, w)$	$I(f \vee g, w)$	$I(f \rightarrow g, w)$	$I(f \leftrightarrow g, w)$
0	0	1	0	0	1	1
0	1	1	0	1	1	0
1	0	0	0	1	0	0
1	1	0	1	1	1	1

Table: 解释函数的定义

公式与模型：

- 给定一个模型 w:
  - 任意给定一组公式，在模型 w 下，公式被解释函数返回真或假，给出语义；
- 给定一公式 f:
  - 存在一些模型，使得公式 f 为真
  - 特别的，定义 $\mathcal{M}(f) = \{w | I(f, w) = true\}$
  - 公式是对模型的“紧凑”表示（模型视为数据集，公式是我们要找的函数，用于表示压缩的数据）



公式与模型：例子

- 公式：$f = (A \wedge B) \leftrightarrow C$，此为输入
- 能利用的东西：输入和解释函数定义的规则

使得公式 f 满足的模型

- 理解：这些模型被公式“紧凑”地表示了（4 行被描述为一行公式）

$$\begin{aligned}\mathcal{M}(f) = \{ & \\ & \{A : true, B : true, C : true\}, \\ & \{A : true, B : false, C : false\}, \\ & \{A : false, B : true, C : false\}, \\ & \{A : false, B : false, C : false\} \\ & \}\end{aligned}$$

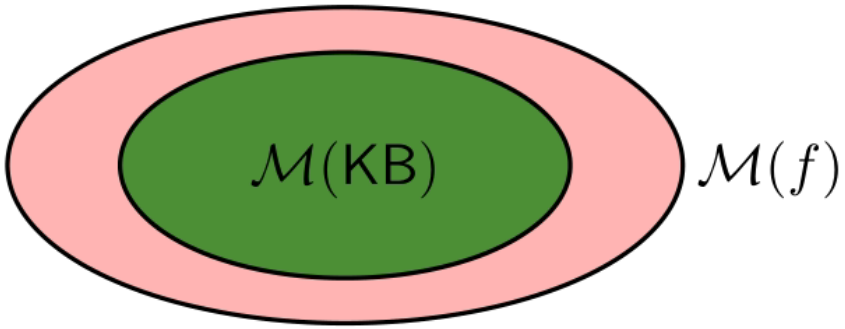
知识库/KB

- 知识库：一组公式；记为 $KB = \{f_i | i = 1, 2, \ldots\}$
- 这些公式紧凑地描述了什么数据/模型?
- 这些公式的模型的交集：$\mathcal{M}(KB) = \bigcap_i \mathcal{M}(f_i)$
- 同时满足所有公式的“真值指派”
- 每个知识库后“隐藏”这一些“模型”，这些模型让知识库都为“真”
- 知识库中的每条知识都是一条“事实”，事实会被某些数据（模型，命题符号被赋予真假值）“证实”
- 一条知识就是很多模型；知识库增加一条知识，能称为“事实”的模型就可能变少一点

例子

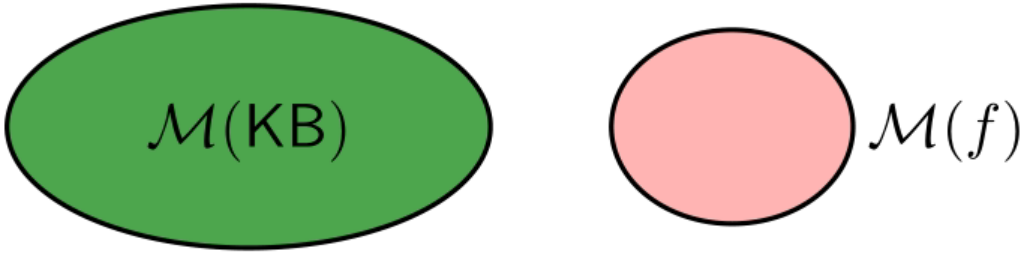
- “天黑” T，是知识库中的一个公式，记为 $KB = \{T\}$，还有一个符号 S 表示“看不见”，但 S 不是知识库中的公式。
- $\mathcal{M}(KB) = \{T = true, S = true; T = true, S = false\}$，使得知识库知识为真的模型集合，构成了“解释”或“事实数据”，
- 增加一条知识 $KB = \{T, T \to S\}$，此时得到 $\mathcal{M}(T \to S) = \{T = true, S = true; T = false, S = false; T = false, S = true\}$，
  $\mathcal{M}(KB) = \{T = true, S = true\}$

核心问题：知识库中公式的增加，如何影响模型/数据的缩减?



理解“蕴含”

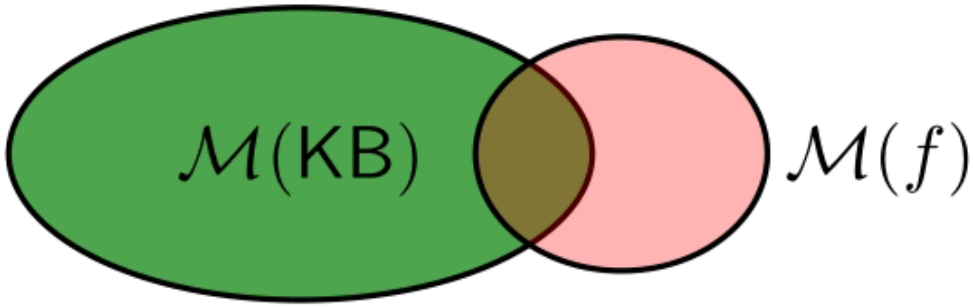
- 如果知识库 KB 蕴含公式 f，那么记作
  $KB \models f \iff \mathcal{M}(f) \subset \mathcal{M}(KB)$
- f 没有增加任何约束条件，所包括的事实/数据更多
- 例如：(天黑 T)$\wedge$(看不见 S) $\models$ (看不见 S)



理解“矛盾”

- 知识库 KB 与公式 f 冲突 $\Longleftrightarrow$ $\mathcal{M}(f) \cap \mathcal{M}(KB) = \varnothing$
- f 解释的事实（让 f 为真的真值指派）与知识库告诉我们的所有事实矛盾冲突；我们相信 KB 就不能相信 f
- 例如：(天黑 T)$\wedge$(看不见 S)$\models$ (看得见 $\neg S$)

条件满足/Contingency

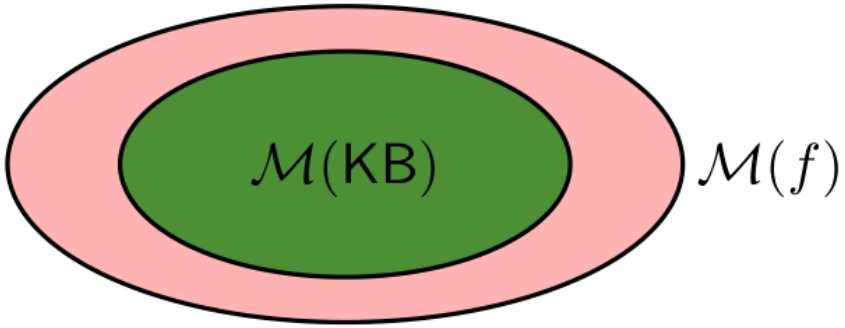


理解“条件满足”

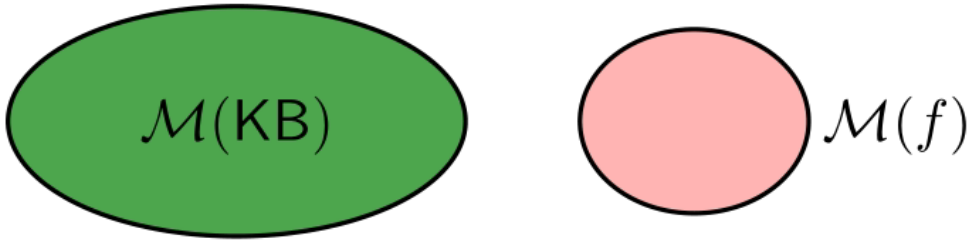
- 公式 f 条件满足知识库 $KB \iff \varnothing \subsetneq \mathcal{M}(KB) \cap \mathcal{M}(f) \subsetneq \mathcal{M}(KB)$
- 二者的模型有非空的交集，f 是否会被 KB 的模型满足，要依情况而定
- 例如：$KB = \{\text{下雨}\}, f = \text{下雪}$,
- 如果 $\mathcal{M}(f) \subset \mathcal{M}(KB)$, 怎样?

蕴含和矛盾的关系

蕴含



矛盾



蕴含与矛盾

- f 与 KB 矛盾 $\iff$ $KB \models \neg f$(KB 蕴含 $\neg f$)

知识推理问题

问题：

- 给定公式 f，如“天黑了”，问知识库 KB，f 成立吗？知识库如何回答或响应？

可能从知识库得到的三种响应

- 回答“Yes/是”：如果 $KB \models f$。公式 f 没带来新知识或数据
- 回答“No/否”：如果 $KB \models \neg f$。公式 f 带来的信息和我已知的事实矛盾，是否选择相信？
- 回答“Unknown/未知”：如果条件满足。也就是说公式 f 是对某些模型/事实的压缩描述，这些模型和 KB 描述的事实不完全一致，带来了一些新的信息

新公式 f 和知识库的三种关系 KB

- 蕴含，矛盾和条件满足
- 判断任给的公式 f 和某个知识库 KB 的关系的步骤或算法?

先定义“可满足的”

- 如果公式 $\mathcal{M}(f) \neq \varnothing$，则称 f 是可满足的
- 如果知识库 $\mathcal{M}(KB) \neq \varnothing$，则称知识库 KB 是可满足的
- 公式和知识库的关系是通过满足它们的的模型间的关系来定义的，所以“可满足的”是关键概念

判断 f 与 KB 关系步骤:

1. 检查 KB 与 $\neg f$ 是否可满足? 若不可满足，则 KB 与 $\neg f$ 矛盾，等价与 $KB \models f$
2. 检查 KB 与 f 是否可满足? 若不可满足，则 KB 与 f 矛盾
3. 否则 KB 与 f 是条件满足的，$f, \neg f$ 各有部分模型满足 KB

知识推理转变成可满足性问题求解

- 知识推理：知识库与特定公式之间的关系；
- 可满足性问题：为一组公式寻找真值指派使得公式都为真

可满足性问题可转变为 CSP 问题

- 命题符号就是逻辑变量；公式就是约束条件；模型就是真值指派
- CSP 问题是 NP-complete 的

例子: 模型检查/model checking

- 给定一个知识库，输出满足知识库的模型。
- 命题符号：A, B, C，$KB = \{A \wedge B, B \leftrightarrow \not C\}$
- 输出模型：$\{A : true, B : false, C : true\}$

命题逻辑的只是推理

知识推理

- 从公式/语法到语义，然后到模型和解释
- 通过公式的可满足性，将公式视为模型/数据的约简表示，所谓推理就是已经有了一些公式为真，判断新公式是否成立
- 求解推理问题的基本方法/根本方法：用可满足性，验证是否存在来自已有知识库的模型中，是否能满足新公式
- 能不能快一点？推理规则

命题逻辑

- 输入：若干命题符号和一组公式构成的知识库
- 输出：一组新的结论
- 例如：天黑了。如果天黑就看不见。所以“看不见”。
- 推理规则：$\dfrac{\text{天黑，天黑}\rightarrow\text{看不见}}{\text{看不见}}$，记为：$\dfrac{\text{前提}}{\text{结论}}$

命题逻辑的演绎推理规则

演绎推理规则

- 给定任意两个命题符号 p, q，演绎推理规则就是：$\frac{p, p \to q}{q}$
- 白话描述：如果 p，那么 q；现在 p 成立，所以 q 成立。if-then 推理

推理规则的一般形式

- 给定公式 $f_1, f_2, \ldots, f_k, g$，记 $\frac{f_1, f_2, \ldots, f_k}{g}$ 为推理规则
- 前提 $f_1, f_2, \ldots, f_k$ 来自 KB，新公式 g 可添加到 KB 中去

理解推理规则

- 直接作用在语法上，和语义完全无关
- 公式（语法）是数据的压缩表示，不用数据和事实，直接判决新公式的成立与否

规则推理算法

- 输入：一组推理规则，一个知识库（一组公式）
- 输出：更新的知识库
- 步骤：
  1. while KB 不再更新
     - 从 KB 中选择 k 条公式 $f_1, f_2, \ldots, f_k$
     - 验证公式 $\frac{f_1, f_2, \ldots, f_k}{g}$ 是否成立，若成立则 $KB \leftarrow KB \cup \{g\}$

理解规则推理算法

- 上述算法的时间复杂度?
- 导出/证明：记号 $\vdash$，$KB \vdash f \iff f$ 能逐步被添加到 KB 中

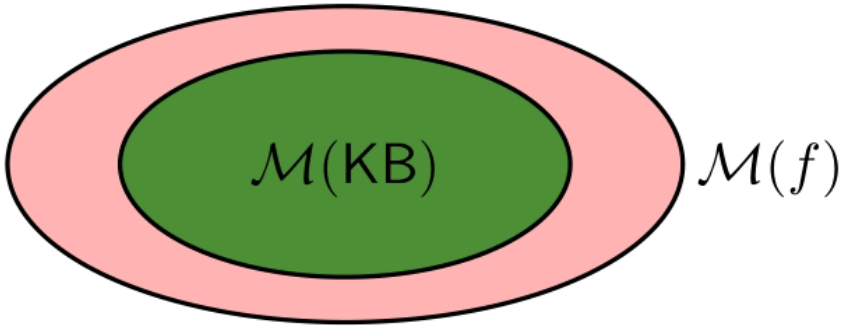
演绎推理的例子

- 已知：$KB = \{$下雨，下雨 $\rightarrow$ 地湿，地湿 $\rightarrow$ 地滑$\}$，推理规则 $\frac{p, p \rightarrow q}{q}$
- 应用一次演绎推理规则得到
  $KB = \{$下雨，下雨 $\rightarrow$ 地湿，地湿 $\rightarrow$ 地滑，地湿$\}$
- 再次演绎推理规则得到
  $KB = \{$下雨，下雨 $\rightarrow$ 地湿，地湿 $\rightarrow$ 地滑，地湿，地滑$\}$

新的推理问题：成立吗?

- 地不湿/地干?
- 地又湿又滑

命题逻辑：两种推理方法



两个增加新公式/知识到知识库的方法：

- 利用可满足性，从模型满足公式的角度，得到 $KB \models f$，如图
- 利用推理规则，得到 $KB \vdash f$

存在问题：

- 二者等价吗？有什么关系？

命题逻辑中推理规则的性质

蕴含$\models$和导出$\vdash$

- 蕴含 $\models$：来自语义对公式的解释；导出 $\vdash$：对语法的直接作用得到
- 一组推理规则的可靠性/soundness：$\{f : KB \vdash f\} \subseteq \{f : KB \models f\}$，也就是说推理规则导出的公式全都是“正确的”或者“可满足的”
- 一组推理规则的完备性/completeness:$\{f : KB \vdash f\} \supseteq \{f : KB \models f\}$，也就是所有知识库中蕴含的公式都可以被推理规则导出

任给一组推理规则：可靠性？完备性？

推理规则例子：可靠性 1

演绎推理规则: $\frac{p, p \to q}{q}$

- 演绎推理规则的可靠性？举例子说明：$KB = \{$下雨，下雨 $\to$ 地湿$\}$。

解释过程：

编号	下雨	地湿
m_1	true	true
m_2	true	false
m_3	false	true
m_4	false	false

Table: 所有可能的模型 $\mathcal{M}$

- $\mathcal{M}($下雨$) = \{m_1, m_2\}$
- $\mathcal{M}($下雨 $\to$ 地湿$) = \{m_1, m_3, m_4\}$

证明推理规则的可靠性：

- 即证明：if $KB \vdash f$, then $KB \models f$
- 由演绎规则能导出公式：$f =$ 地湿，要证明 $KB \models$ 地湿
- $KB \models$ 地湿 $\Leftrightarrow \mathcal{M}(KB) \subseteq \mathcal{M}($地湿$)$
- $\mathcal{M}(KB) = \mathcal{M}($下雨$) \cap \mathcal{M}($下雨 $\to$ 地湿$) = \{m_1\}$
- $\mathcal{M}(KB) = \{m_1\} \subseteq \{m_1, m_3\} = \mathcal{M}($地湿$)$，成立

该例子中“演绎规则”是可靠的

推理规则例子：可靠性 2

另一个推理规则：$\frac{q, p \to q}{p}$

- 该推理规则的可靠性？举例子说明：$KB = \{$地湿，下雨 $\to$ 地湿$\}$。

解释过程：

编号	下雨	地湿
m_1	true	true
m_2	true	false
m_3	false	true
m_4	false	false

Table: 所有可能的模型 $\mathcal{M}$

- $\mathcal{M}($地湿$) = \{m_1, m_3\}$
- $\mathcal{M}($下雨 $\to$ 地湿$) = \{m_1, m_3, m_4\}$

证明推理规则的可靠性：

- 即证明：if $KB \vdash f$, then $KB \models f$
- 由演绎规则能导出公式：$f =$ 下雨，要证明 $KB \models$ 下雨
- $KB \models$ 下雨 $\Leftrightarrow \mathcal{M}(KB) \subseteq \mathcal{M}($下雨$)$
- $\mathcal{M}(KB) = \mathcal{M}($地湿$) \cap \mathcal{M}($下雨 $\to$ 地湿$) = \{m_1, m_3\}$
- $\mathcal{M}(KB) = \{m_1, m_3\} \subsetneq \{m_1, m_2\} = \mathcal{M}($下雨$)$

该例子中“推理规则”是不可靠的

推理规则的例子：完备性 1

演绎推理规则: $\frac{p, p \to q}{q}$

- 该推理规则的完备性?

证明推理规则的完备性:

- 即证明：for all f, if $KB \models f$, then $KB \vdash f$
- 然而演绎推理规则是不完备的:
- 举例子说明：$KB = \{$下雨, 下雨 $\vee$ 下雪 $\to$ 地湿$\}$。$f =$ 地湿
- 从语义的角度，得到蕴含 $KB \models$ 地湿，但是语义上通过演绎推理规则无法得到 f，无法确定是下雨还是下雪

因此，演绎推理规则是不完全的

推理规则的例子：完备性 2

将不完备的推理规则改进为完备的推理规则

- 方法一：对知识库中的公式添加某种约束；即在特殊要求的知识库上应用推理规则
- 方法二：设计功能更强大的推理规则

具备完备性的演绎推理规则

规定知识库中公式的形式

- 肯定句：现实生活中，是对事物作出肯定判断的句子。
- 肯定句：逻辑中，$(p_1 \wedge p_2 \wedge \ldots \wedge p_n) \rightarrow q$，其中 $p_1, p_2, \ldots, p_n, q$ 都是命题符号。
- 直觉的理解：如果 $p_1, p_2, \ldots, p_n$ 都成立，那么 q 成立

例子

- 肯定句：下雨 $\wedge$ 下雪 $\rightarrow$ 交通拥堵；交通拥堵；
- 不是肯定句：下雨 $\wedge$ 下雪；$\neg$ 交通拥堵；下雨 $\wedge$ 下雪 $\rightarrow$ 交通拥堵 $\vee$ 世界和平；

具备完备性的演绎推理规则

Horn 句子：包括以下两种

- 肯定句：形如 $(p_1 \wedge p_2 \wedge \ldots \wedge p_n) \rightarrow q$
- 目标句：形如 $(p_1 \wedge p_2 \wedge \ldots \wedge p_n) \rightarrow false$

由 Horn 句子构成的知识库

- 下雨；周三；
- 下雨 to 地湿；
- 周三 $\wedge$ 下雨 $\rightarrow$ 交通拥堵；
- 交通拥堵 $land$ 不小心 $\rightarrow$ 发生事故；

公式被限制为 Horn 句子的知识库中，演绎推理规则：完备吗？

- $\frac{p_1, p_2, \ldots, p_n, (p_1 \wedge p_2 \wedge \ldots \wedge p_n) \rightarrow q}{q}$
- 如果 q 是一个命题符号，那么演绎推理规则是完备的！

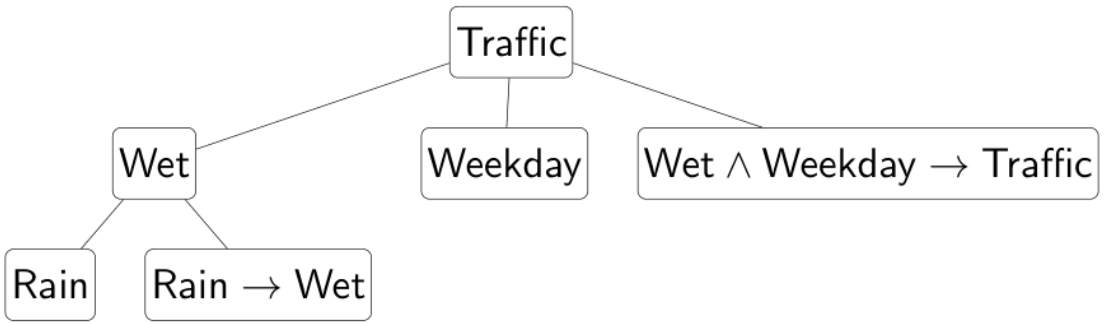
具备完备性的演绎推理规则

如果 f 不是命题符号，$f = p \wedge q$，那么

- 如果 $KB \models f$，有 $KB \vdash f$ 吗?
- 如果 $KB \vdash f$，那么 $KB \models f$ 吗?

引入目标句

- 将目标句 $q \wedge q \rightarrow false$ 加入知识库 KB，判断命题符号 $false$ 是否能用演绎推理规则导出或用语义蕴涵导出即可。
- 语义蕴涵推导的关键技巧：$KB \models f$ 等价于 $KB \cup \{\neg f\}$ 是不可满足的
- 推理规则推到的关键技巧：连续使用推理规则，构建如图所示的树（演绎推理规则)。树叶是知识库中的公式，非叶节点是用演绎推理规则获得的公式。



知识推理/逻辑推理

- 基于给定的知识库: 一组为 $true$ 的公式
- 目标：判定一个查询公式 f 是否被知识库语义蕴涵？还是和知识库相矛盾？还是不能由知识库给出判断，即未知（依情况而定)？
- 存在的问题：检查模型是否为空来判定查询公式是否被知识库语义蕴含，过程非常复杂。有多复杂?

推理规则的作用

- 不检查模型，如果推理规则即可靠又完备，那么直接利用推理规则进行公式的判定会更简单
- 注意：解释函数实现从语法到语义的连接
- 直观的理解：公式是规律，模型是数据，满足规律的数据。设想一下：知识库的公式是世界的运行规律，模型是世界运行时，某个时刻的状态数据，这些数据都满足描述规律的公式；

推理规则不完备时：

- 命题逻辑在演绎推理规则下，并不完备
- 将知识库中的公式限制为只能是 Horn 句子，那么演绎推理规则既可靠又完备
- 能够开发新的完备的推理规则?

几个概念

- Literal：若 p 是命题符号，那么 $p, \neg p$ 分别被叫做正 Literal 和负 Literal
- 句子：Literal 的析取表达式
- Horn 句子：至多只有一个正 Literal
- 任何一个蕴含式 $p \to q$ 都可以写成析取式 $\neg p \vee q$，为什么？(把解释函数写出来)。$A \wedge B \to C$ 可以写成 $\neg A \vee \neg B \vee C$

演绎推理规则的新表示方法

- $\frac{A, \neg A \vee C}{C}$
- 形式上看，有种“对消”的“感觉”，将 A 和 $\neg A$ 成对从前提条件中消除，得到 C

归结推理规则

- 一般通用句子：$A \vee \neg B \vee C \vee \neg D \vee \neg E \vee F$，包含任意多个 Literal
- 归结推理规则：$\frac{f_1 \vee \ldots \vee f_n \vee p, \neg p \vee g_1 \vee \ldots \vee g_n}{f_1 \vee \ldots \vee f_n \vee g_1 \vee \ldots \vee g_n}$
- 归结推理规则是对演绎推理规则的一种扩展，将两个句子中的正负 Literal 成对消去，并析取链接两个句子
- 既可靠又完备的推理规则

归结推理规则如何应用与知识库推理?

- 此时，知识库中的知识或公式，没有任何的限制

知识库的“规范化”：合取范式/Conjunctive Normal Form,CNF

- 知识库中知识表示方式包括：$\neg, \vee, \wedge, \rightarrow, \leftrightarrow$，太多了，能不能少一点？表示更简洁一些？
- 合取范式：句子是析取式，句子的合取式构成合取范式，例如 $(A \vee \neg B \vee C) \wedge (\neg D \vee E) \wedge (\neg F)$
- 知识库中每条公式是一个句子，知识库就是一个合取范式
- 任何一个公式/知识库都可以转化为合取范式

转换为合取范式的例子

- 初始公式：$(rain \rightarrow wet) \rightarrow trumble$
- 消除蕴含：$\neg(\neg rain \vee wet) \vee trumble$
- 非运算进入括号内：$(\neg\neg rain \wedge \neg wet) \vee trumble$
- 消除双重非运算：$(rain \wedge \neg wet) \vee trumble$
- 分配：$(rain \vee trumble) \wedge (\neg wet \vee trumble)$

知识库转换为合取范式的转换规则

- 消除等价：$\frac{f \leftrightarrow g}{(f \rightarrow g) \wedge (g \rightarrow f)}$
- 消除蕴含：$\frac{f \rightarrow g}{\neg f \vee g}$
- 非运算进入括号内：$\frac{\neg(f \vee g)}{\neg f \wedge \neg g}, \frac{\neg(f \wedge g)}{\neg f \vee \neg g}$
- 消除双重非运算：$\frac{\neg\neg f}{f}$
- 在 $\wedge$ 上分配 $\vee$：$\frac{f \wedge (g \vee h)}{(f \vee g) \wedge (f \vee h)}$

理解

- 转换规则看成是推理规则，可靠与完备的
- 但是没有增加任何新的公式，只是重写公式，所以本质上不是推理规则

知识库归结推理

推理算法

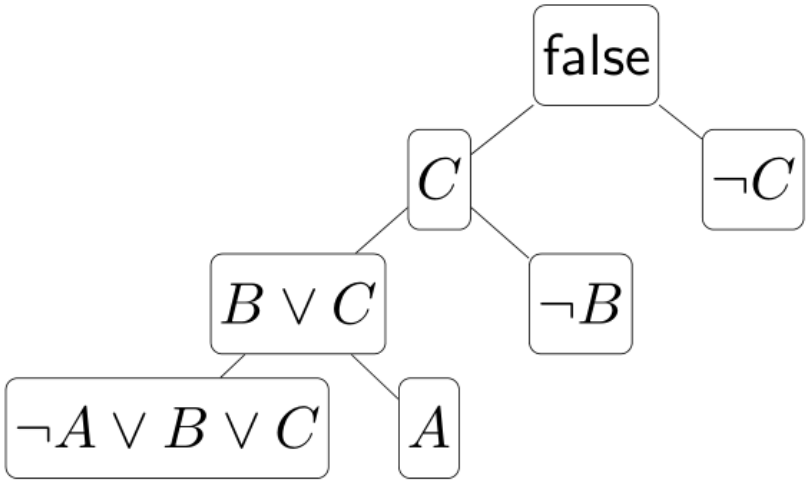
- 目标：$KB \models f$ 吗?
- 令 $KB \cup \{\neg f\}$ 所有公式转换为合取范式
- 连续在合取范式上应用归结推理规则
- 如果由规则导出 $false$，那么表示 $KB \cup \{\neg f\}$ 不可被满足，$KB \models f$ 成立
- 令 $KB = KB \cup \{f\}$，然后将 KB 中所有公式转换为合取范式
- 连续在合取范式上应用归结推理规则
- 如果由规则导出 $false$，那么表示知识库不可被满足，$KB \models \neg f$ 成立

理解

- 当且仅当归结推理规则导出 $false$ 的时候，知识库是不可被满足的
- 如果不能导出 $false$，那么知识库是可满足的（在推理过程中，这一点事实没办法使用）

例子：

- 知识库：$KB = \{A \to (B \vee C), A, \neg B, \neg C\}$
- 转换为知识库的合取范式：$KB = \{\neg A \vee B \vee C, A, \neg B, \neg C\}$
- 连续利用规则推理规则，得到



- 结论：知识库不可被满足

知识库推理的复杂性

演绎推理规则

- $\frac{p_1, p_2, \ldots, p_n, (p_1 \wedge p_2 \wedge \ldots \wedge p_n) \rightarrow q}{q}$
- 演绎推理规则的每次应用就获得一个包含单个命题符号的句子，时间复杂度是关于命题符号的个数线性阶
- 演绎推理规则要保证可靠与完备，那么就限制了知识库的公式形式，即降低了表达知识的能力，知识库只能描述 Horn 句子，但是优点突出：推理过程高效

归结推理

- $\frac{f_1 \vee \ldots \vee f_n \vee p, \neg p \vee g_1 \vee \ldots \vee g_n}{f_1 \vee \ldots \vee f_n \vee g_1 \vee \ldots \vee g_n}$
- 归结推理规则的每次应用会产生可能包括多个命题符号的句子，时间复杂度是命题符号个数的指数阶
- 归结推理规则既可靠又完备，知识库中的句子也没有任何的约束限制，表达能力强，但是缺点是付出更多，甚至不可接受的推理过程中的时间开销。

命题逻辑的不足

逻辑的目的

- 用简单的方式表示复杂的事物

命题逻辑表达时的臃肿和“啰嗦”

- 张三和李四要参加考试：张三要参加考试 $\wedge$ 李四要参加考试
- 所有的学生要参加考试：张三是学生 $\rightarrow$ 张三要参加考试，李四是学生 $\rightarrow$ 李四要参加考试，...

命题逻辑表达不力的原因

- 缺少“对象”和“关系”：比如“张三要参加考试”命题中有对象和对象间的关系，命题具有结构；世界中除了最基本的原子事实，还有原子事实通过特定结构符合出来的“复杂事实”
- 缺少“量词”和“变量”：比如缺少量词“所有/all”，就只能枚举“所有”的学生；量词和变量能用一个公式紧凑地描述大量的事实

目的：提升命题逻辑的表达能力

- 例子 1：要参加（张三，考试）$\wedge$ 要参加（李四，考试），引入对象和关系；
- 例子 2: $\forall x, student(x) \rightarrow$ 要参加（x，考试），引入量词、变量和函数

一阶逻辑的语法

- 三种“项”：常数项（特定对象，如“考试”，“张三”等）；变量（将被量词指明的非特定对象（一组对象），如 x）；函数（如 要参加（ $\cdot$ ），$student(x)$ 等

一阶逻辑的公式

- 原子公式，其公式的项全是常数，例如：要参加（张三，考试）
- 应用连接符得到的公式，例如，$student(x) \rightarrow$ 要参加（x，考试）
- 应用量词得到的公式，例如，$\forall x, student(x) \rightarrow$ 要参加（x，考试）

一阶逻辑的量词

量词

- 全称量词：$\forall$，可理解为合取式，如 $\forall x, P(x)$ 就是 $P(A) \wedge P(B) \wedge P(C) \wedge \ldots$
- 存在量词：$\exists$，可理解为析取式，如 $\exists x, P(x)$ 就是 $P(A) \vee P(B) \vee P(C) \vee \ldots$

量词的一些属性

- $\neg \forall x P(x) \Leftrightarrow \exists x \neg P(x)$，用全称量词的合取式理解来分析即可
- $\forall x \exists y P(x, y)$ 和 $\exists y \forall x P(x, y)$ 二者不一样（与或操作不具备交换性），用全称量词的合取式理解和存在量词的析取式来分析即可

自然语言中的量词

- 全称量词：每一个，每个，所有，任意一个，例如：所有学生要参加考试，
  $\forall x, student(x) \rightarrow attends(x, exam)$
- 存在量词：一些，有的，部分，例如：有的学生要参加考试，
  $\exists x, student(x) \wedge attends(x, exam)$
- 注意上述例子中连接符号的差异！

一些例子

- 这里有些学生选的课程。
  $\exists x, course(x) \wedge (\forall y, student(y) \rightarrow takes(y, x))$
- 每个比 2 大的偶数都可以写成两个质数之和。
  $\forall x, evenInt(x) \wedge greater(x, 2) \rightarrow$
  $\exists y \exists z, equals(x, sum(y, z)) \wedge prime(y) \wedge prime(z)$
- 如果一个学生学了一门课，且课上讲了某个概念，那么这个学生应该就知道这个概念。
  $\forall x \forall y \forall z, student(x) \wedge course(y) \wedge takes(x, y) \wedge covers(y, z) \rightarrow$
  $knows(x, z)$

一阶逻辑的语义

语义

- 命题逻辑中，语义是指模型，能让公式满足的模型，能让公式成立的、各种可能在现实世界中呈现出来的数据，更直接的说法就是命题符号的真值指派（给所有命题符号赋予 $true, false$）
- 一阶逻辑中，给所有的原子公式一个真值指派?

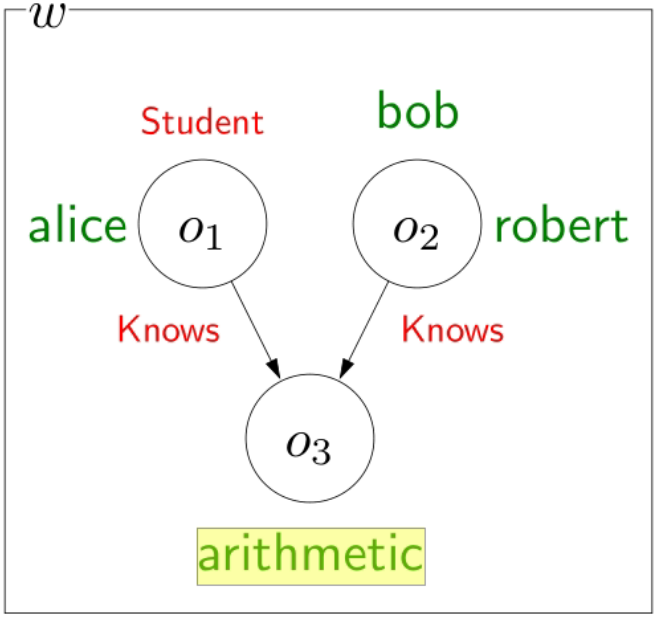
存在的问题

- 大多数的时候，用原子公式的真值指派来定义一个模型是没问题的
- 但是，原子公式中的常数项之间的关系没办法描述出来，比如，张三参加考试，李四是张三的别名，但是我们不知道李四是张三的别人，我们就无法得到李四参加考试这个事实。

一阶逻辑的语义模型

模型 w：有向图描述

- 对象：表示为一个顶点，每个顶点添加若干标签/权值，每个标签是对象的一个特征，特征可以是一阶逻辑中的常数项。
- 有向边：表示二元关系，添加“谓语/动词”作为边的标签/权值，表示两个对象相互作用
- 一元关系：可理解为一种顶点的标签。
- 如图：对象集：o_1, o_2, o_3；常数项：$Alice, Bob, Robert, arithmetic$，一元关系：$Student$；二元关系：$Knows$



模型的定义

- 将常数项映射为对象：
  $w(Alice) = O_1, w(Bob) = O_2, w(arithmetic) = O_3$，常数项属于语法范畴，对象属于语义范畴
- 将谓语/动词映射为对象多元组集合：
  $w(Knows) = \{(O_1, O_3), (O_2, O_3), \ldots\}$

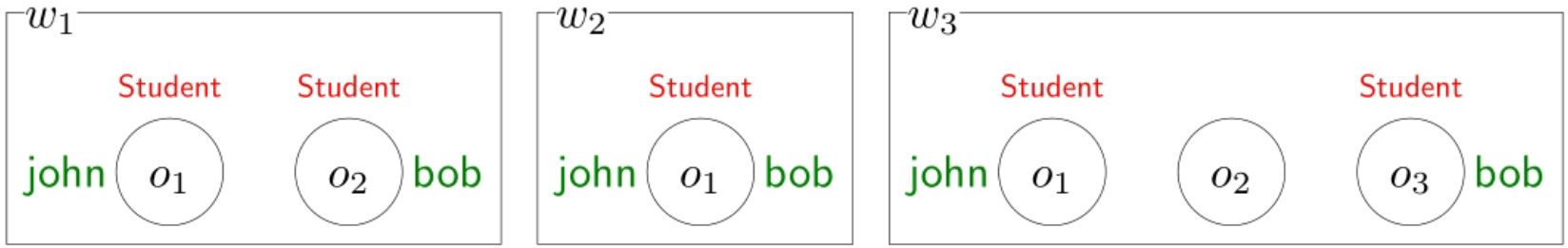
存在的问题：

- 常数项之间的关系如何处理？
- 图描述中，顶点对象可能有多个标签（包括一元关系）
- 两个不同的标签可能描述的是同一个对象

关于模型的一些限制

- 例子：John 和 Bob 都是学生，如图，有三个（更多）模型 w_1, w_2, w_3 满足公式
- 唯一性假设：每个对象至多只有一个常数项符号标签，删除 w_2
- 域封闭假设：每个对象至少有一个常数项符号标签，删除 w_3
- 这些限制条件导致的结果就是：对象 $\Leftrightarrow$ 常数项，建立了语法单元和语义单元的一一对应

$$\text{Student}(\text{john}) \wedge \text{Student}(\text{bob})$$



肯定句

- 一阶逻辑的肯定句：形如 $\forall x_1 \ldots \forall x_n (a_1 \wedge \ldots \wedge a_k) \rightarrow b$，其中 $a_1, \ldots, a_n, b$ 是原子公式，$x_1, \ldots, x_n$ 是变量
- 由命题逻辑中的肯定句和全称量词构成

演绎推理规则

- 命题逻辑中仅包括 Horn 句子的知识库用演绎推理是可靠又完备的
- 一阶逻辑中仅包括上述肯定句的知识库用演绎推理是可靠又完备的
- 一节逻辑中的演绎推理规则是怎样的?

一阶逻辑推理中量词的处理

推理规则能应用到量词上

- 给定 $P(alice), \forall x P(x) \rightarrow Q(x)$，能推理出 $Q(alice)$ 吗?
- 不一定！应为我们不知道 x 是否包括 $alice$，即变量和常数项符号之间的关系没有!

解决办法:

- 替换/substitution：结果是将公式变形为另一公式;
- 统一/unifiction：将两个公式变为同一个公式

替换/substituation

- 目的：把公式变形为另一种形式
- 做法：把变量变换为其他变量或常数项，
- 记作 $Subst[\theta, f]$，在公式 f 上做替换 θ，
- 例 1：$Subst[\{x/alice\}, P(x)] = P(alice)$
- 例 2：$Subst[\{x/alice, y/z\}, P(x) \wedge K(x, y)] = P(alice) \wedge K(x, z)$

统一/unification

- 目的：把两个公式统一为一个公式
- 做法：输入两个公式 f, g，对它们都实施一个替换 θ 使得替换后 f, g 公式的新形式一样，即：$Unify[f, g] = \theta$ 满足 $Subst[\theta, g] = Subst[\theta, f]$，或者当没有这样的 θ 存在时返回失败；
- 期中 θ 称为统一算子
- 例子如下

$$\text{Unify}[\text{Knows}(\text{alice}, \text{arithmetic}), \text{Knows}(x, \text{arithmetic})] = \{x/\text{alice}\}$$

$$\text{Unify}[\text{Knows}(\text{alice}, y), \text{Knows}(x, z)] = \{x/\text{alice}, y/z\}$$

$$\text{Unify}[\text{Knows}(\text{alice}, y), \text{Knows}(\text{bob}, z)] = \text{fail}$$

$$\text{Unify}[\text{Knows}(\text{alice}, y), \text{Knows}(x, F(x))] = \{x/\text{alice}, y/F(\text{alice})\}$$

演绎推理规则

定义

- $\frac{a'_1,\ldots,a'_k \forall x_1 \ldots \forall x_n (a_1 \wedge \ldots \wedge a_k) \rightarrow b)}{b'}$
- 在前提条件中应用统一算子 $\theta = Unify[a'_1 \wedge \ldots \wedge a'_k, a_1 \wedge \ldots \wedge a_k]$
- 在结论中应用替换：$Subst[\theta, b] = b'$

应用演绎推理规则在一阶逻辑中的例子

- 前提：$Takes(alice, AAI), Covers(AAI, ML), \forall x \forall y \forall z Takes(x, y) \wedge Covers(y, z) \rightarrow Knows(x, z)$
- 替换和统一算子：$\theta = \{x/alice, y/AAI, Z/ML\}$
- 结论：$Knows(alice, ML)$

时间复杂度

- 每次应用演绎推理规则得到一个原子公式
- 如果没有函数符号，那么原子公式的个数至多为 常数项符号的个数的 谓语/动词个数最大值 次方，是指数个，例如 $\forall x \forall y \forall z P(x, y, z)$
- 如果有函数符号，那么可能造成无限多个公式，例如：$Q(a), Q(F(a)), Q(F(F(a))), \ldots$

完备性

- 仅包括 Horn 句子的一阶逻辑中演绎推理规则是完备的

半可判定性

- 对于仅包含 Horn 句子的一阶逻辑是半可判定的，也就是说：
- 如果 $KB \models f$，那么用推理规则在有限的时间内能导出 f
- 如果 $KB \not\models f$，没有算法能在有限的时间内导出，也就是说公式 f 的判定过程不知道什么时候能停下来！

归结推理规则

一阶逻辑包括不是 Horn 句子的公式

- 例如：$\forall x Stuent(x) \rightarrow \exists y Knows(x, y)$

归结推理框架

- 将所有知识库中的公式转换为合取范式/CNF
- 重复应用归结推理规则到知识库中的公式上

例子：

- 转换为 CNF：输入公式/$\forall x(\forall y Animal(y) \rightarrow Loves(x, y)) \rightarrow \exists y Loves(y, x)$；输出 CNF/$(Animal(\,Y(x)) \vee Loves(Z(z), x)) \wedge (\neg Loves(x,\, Y(x)) \vee Loves(Z(z), x))$
- 注意：上式中 CNF 有个扩展，每个变量缺省带有全称量词，变量 x 的 Skolem 函数 $Y(x), Z(z)$ 表示存在量词
- 喜欢动物的人会被其他人所喜欢

归结推理规则

推理规则的改变：

- 一阶逻辑中的归结推理规则：$\frac{f_1 \vee \ldots \vee f_n \vee h_1, \neg h_2 \vee g_1 \vee \ldots \vee g_n}{Subst[\theta, f_1 \vee \ldots \vee f_n \vee g_1 \vee \ldots \vee g_n]}$
- 其中 $\theta = Unifiy[h_1, h_2]$

例子：

- $\frac{Animal(Y(x)) \vee Loves(Z(x), x)), \neg Loves(u, v) \vee Feeds(u, v)}{Animal(Y(x)) \vee Feeds(Z(x), x)}$
- $\theta = \{u/Z(x), v/x\}$

命题逻辑和一阶逻辑的异同

模型检查

- 命题逻辑有，一阶逻辑没有。(什么是模型?)

基于语法的推理

- 演绎推理和归结推理类似，但是一阶逻辑需要扩充两个操作：替换和统一；
- 一阶逻辑的推理时间复杂度上升

表达能力

- 命题逻辑中公式紧凑地描述了大量的复杂的模型集合；
- 一阶逻辑中的变量用一阶逻辑的语法描述了大量复杂的命题逻辑中的公式；

发展其它逻辑的动机

表达能力的扩展：

- 时序逻辑：表达时间
- 模态逻辑：表达信念
- 高阶逻辑：量词的一般化

便捷和计算的高效性

- 描述逻辑

例子：

- 川普是美国总统，$President(DonaldTrump, US)$
- 奥巴马是美国总统，$President(BarackObama, US)$
- 问题：逻辑描述了对象及其间的关系，总是假设这些对象及关系是不随时间变化的

改进：描述时间

- 引入 $\mathbf{P}, \mathbf{F}$ 分别表示 Past 和 Future
- $\mathbf{P}$ $President(BarackObama, US)$，奥巴马曾经是美国总统
- $\mathbf{F}$ $\exists Female(x) \wedge President(x, US)$，美国将来会出现女总统

考察每个时间点 t 时刻的公式 f 的语义

- 解释函数 $I(f, w)$ 增加时间变量 $I(f, w, t)$
- 如果时刻 t，用模型 w 解释公式 f 为真，则 $I(f, w, t) = 1$
- 如果 $I(f, w, s) = 1$，则 $I(\mathbf{P}\ f, w, t) = 1, t > s$

时序逻辑中的算子

- $\mathbf{P}\ f$: 表示在过去某些时间点公式 f 成立
- $\mathbf{F}\ f$: 表示在将来某些时间点公式 f 成立
- $\mathbf{H}\ f$: 表示在过去所有时间点公式 f 成立
- $\mathbf{P}\ f$: 表示在将来所有时间点公式 f 成立
- 例子：$\forall x Student(x) \rightarrow \mathbf{FG}\neg Student(x)$，每个学生将来都不再会是学生。

概念

- 逻辑的一个分支，它研究必然、可能及其相关概念的逻辑性质

理解

- 处理用模态如“可能”、“或许”、“可以”、“一定”、“必然”等限定的句子的逻辑；
- 从语义方面理解：复杂公式的真值不能由子公式的真值来决定的
- “所有逻辑上可能的世界”。如果一个陈述在所有可能世界中是真的，则它是必然的真理。如果一个陈述碰巧在我们的世界中是真的，但不是在所有可能世界中是真的，则它是偶然的真理。

例子

- 亚里斯多德把命题分为三类：(I) 实然命题，”a 是 b”；(II) 必然命题，”a 必然是 b”；(III) 偶然命题，”a 偶然是 b”。
- (I) 和 (III) 是模态逻辑命题

模态算子

- 必然的 A（A 必然成立）：$\Box A$
- 偶然的 B（B 偶然成立）：$\Diamond B$

模态逻辑中的语义

- 用解释函数来解释每个公式的时候，仅限定在给定的“某个世界中”，比如“张三的世界“
- 定义算子 $B_{zs}f$，表示在 zs 的世界中解释公式 f
- $I(B_{zs}f, w) = I(f, w_{zs})$
- $B_{zs}Rain(hefei, today)$，张三认为合肥今天下雨。（偶然的，张三的看法）

例子

- Alice 访问了几个城市。$\exists x City(x) \wedge Visited(Alice, x)$
- 如果要清楚表达访问的城市的个数，怎么办?

Lambda 演算

- Alice 访问了至少 10 个城市。
- $GreatThan(Count(\lambda x City(x) \wedge Visited(Alice, x)), 10)$
- Lambda 算子：$\lambda x City(x) \wedge Visited(Alice, x)$ 表示 Alice 访问过的城市的集合。
- 高阶逻辑增加了 Lambda 算子