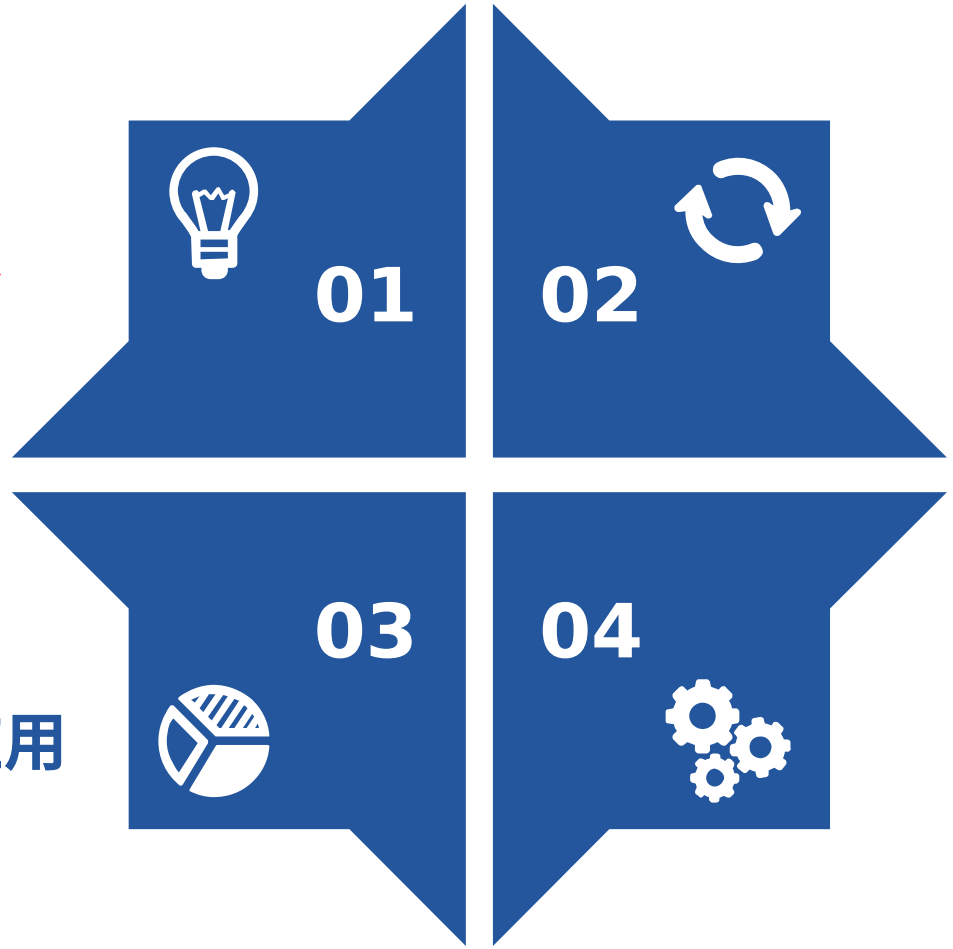
课程《人工智能原理与技术》

同济大学 TONGJI UNIVERSITY

第二章 知识表达与推理

同济大学 TONGJI UNIVERSITY

目录



知识表示方法

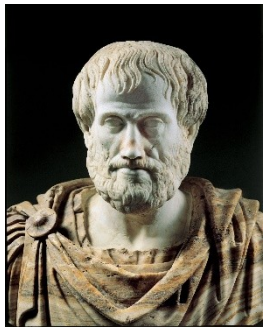
命题逻辑与谓词逻辑

归结原理及应用

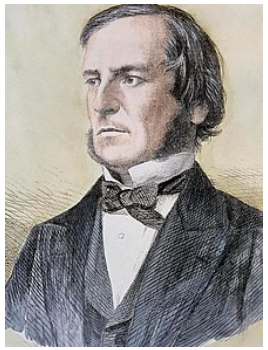
逻辑推理案例

什么是逻辑：进行正确推理和充分论证的研究

符号主义人工智能 (symbolic AI) 就是脱胎于逻辑推理



亚里士多德
三段论
Syllogism
deductive reasoning



布尔
数理逻辑
algebraic logic
the law of thought

墨翟
名、辞、说

逻辑关心的问题

- 从一个或若干前提出发，是否存在一个有效的论证或推理来支持所得到的结论，也就是说在前提和结论之间架构逻辑结构的桥梁。
- 逻辑是研究推理的一门科学，“逻辑”这一单词起源于希腊单词“**logos**”，表示原因、话语和演讲，其内涵非常类似汉语的“道”，本意都和说话有关，延伸出“道理”之义。
- 逻辑与推理是人工智能的核心问题。人类思维活动的一个重要功能就是设定一些逻辑规则，然后进行分析，如通过归纳和演绎等手段对现有观测现象由果溯因（归纳）或由因溯果（推理），从观测现象中得到结论。

认知基石和燃料是规范化的逻辑知识

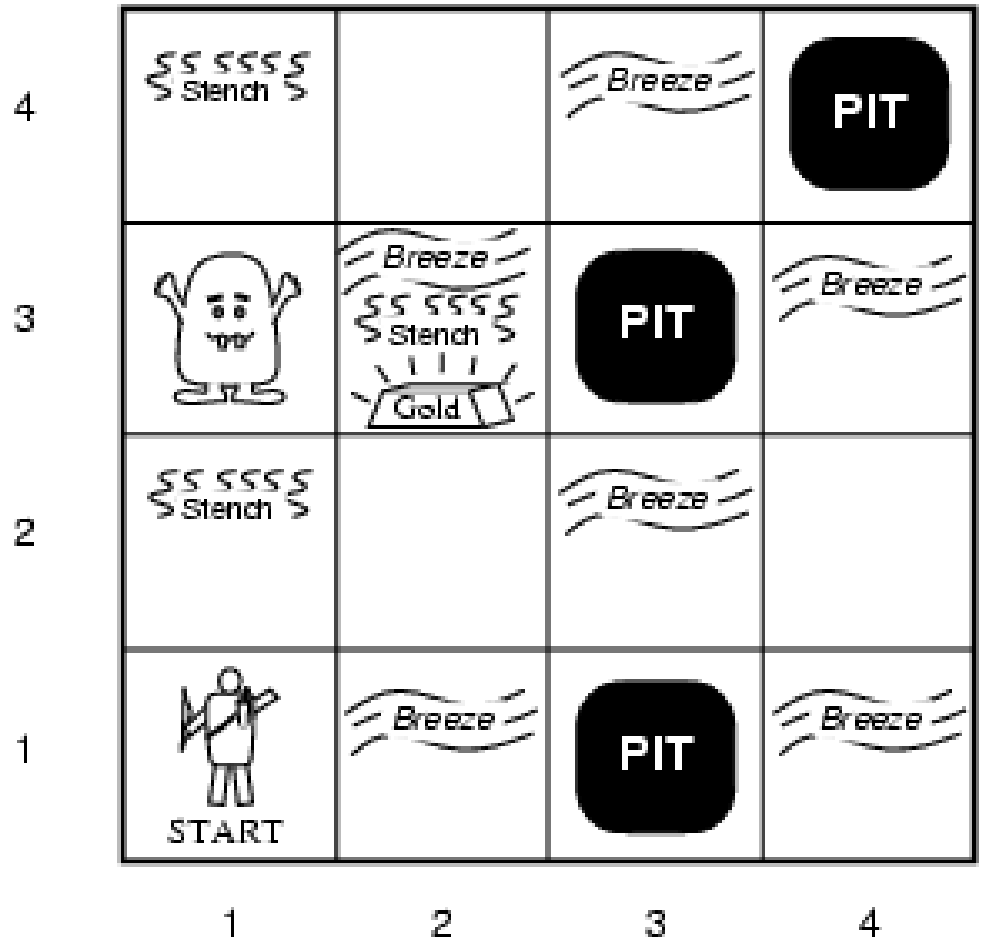
同济大学 TONGJI UNIVERSITY

- 认知这个单词“ cognition”来源于拉丁语“ cognitio”，表示学习和知识。古希腊哲学认为人类理解世界的途径源于**“学习技能”和“解释已得知识”**两种途径，从而“开始知道”。

- 早在 13 世纪末，加泰罗尼（现西班牙境内）诗人、哲学家、逻辑学家雷蒙·卢尔（Ramon Llull）提出了对知识进行规范化描述的**“知识树（Tree of Knowledge）或科学树（Tree of Science）”**，这是目前最早的一种对知识进行规范化表示的努力。这种知识树的表示方法深刻影响了莱布尼兹（Leibnitz）所提出的**“人类思想字母表（alphabet of human thought）”**，即对规范化表达知识按照规则进行组合等操作，就可构造思想机器，实现思维的计算。

目前代表性知识表达方法有命题逻辑、谓词逻辑、产生式规则（production rule）、框架 (frame) 表示法以及知识图谱等。

Wumpus world



Wumpus 世界环境：

由多个房间组成并连接起来的**山洞**。

在洞穴某处隐藏着一只 **Wumpus**，会吃掉进入它房间的任何人。 Wumpus 周围有臭气 **Stench**

某些房间是**无底洞 PIT**，任何人漫游到这些房间都会被无底洞吞噬。无底洞周围有微风 **Breeze**

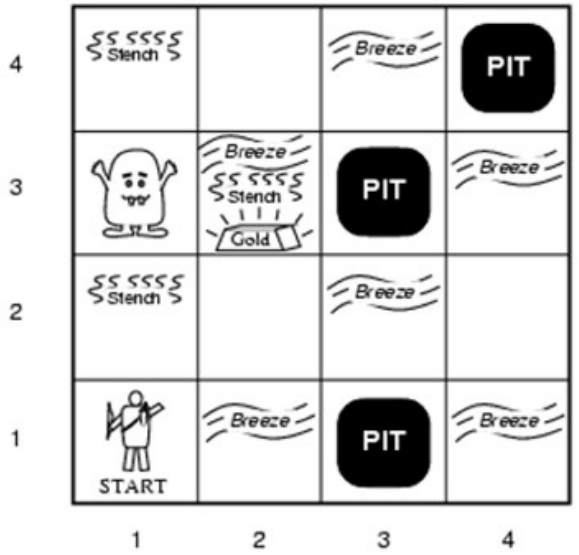
生活在该环境的唯一希望是存在**发现一堆金子**的可能性。

Agent：

可以移动到相邻房间，可以射杀 Wumpus，但是只有一支箭。

Wumpus world PEAS description

- Performance measure
  - 黄金 +1000, 死亡 -1000
  - 每步 -1, 使用箭 -10
- Environment
  - 与怪兽相邻的方块是臭的
  - 无底黑洞旁边的广场有微风
  - 亮光和金子在同一个方块中
  - 如果你面对怪兽射击，会杀死它
  - 射击会消耗唯一的箭
  - 如果和金子在同一个方块中，抓取可以获得金子
  - 如果和金子在同一个方块中，放手可以扔掉金子
- Sensors: 臭味，风，发光，撞击，尖叫
- Actuators: 向左转，向右转，向前，抓取，放手，射击



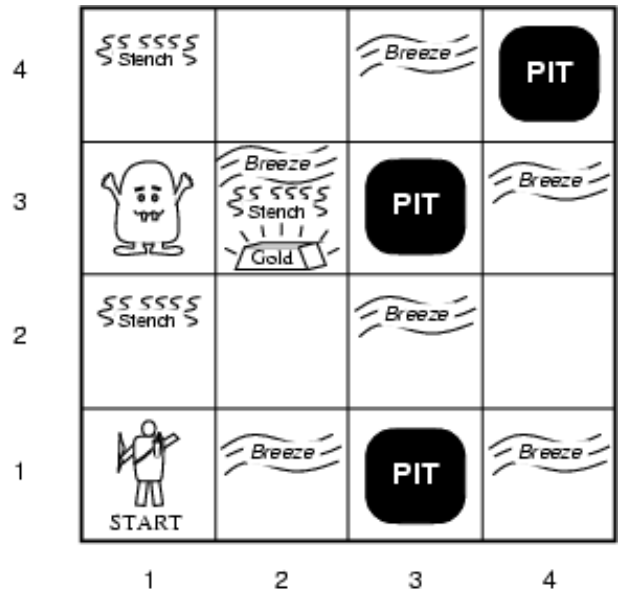
a percept
[Stench, Breeze, Glitter, Crash, Yell]

Exploring a wumpus world

首先，定义感知 (状态描述)： [Stench, Breeze, Glitter, Crash, Yell]

1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2	3,2	4,2
1,1 A OK	2,1 OK	3,1	4,1

A = *Agent*
B = *Breeze*
G = *Glitter, Gold*
OK = *Safe square*
P = *Pit*
S = *Stench*
V = *Visited*
W = *Wumpus*

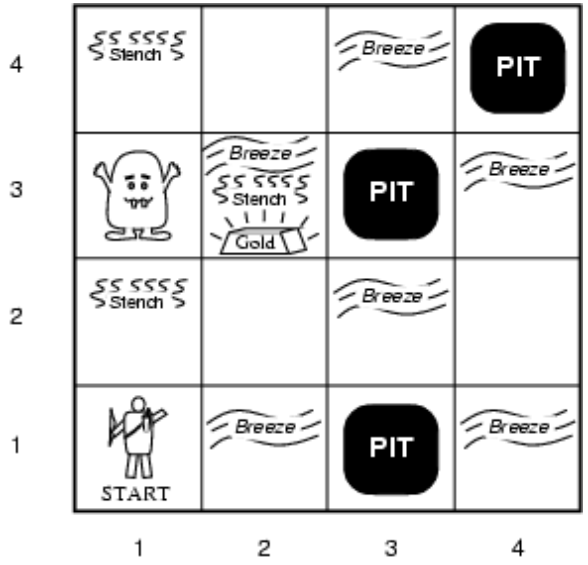


The first percept is [None,None,None,None,None]
conclude [1,2] and [2,1] are OK

Action： [2,1]

Exploring a wumpus world

1,4	2,4	3,4	4,4
1,3	2,3	3,3	4,3
1,2 OK	2,2 P?	3,2	4,2
1,1 V OK	2,1 A B OK	3,1 P?	4,1



there must be a pit in [2,2] or [3,1] or both

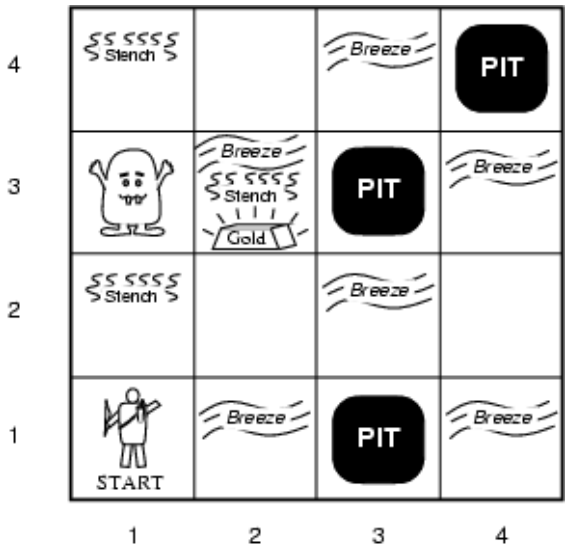
percept [None,Breeze,None,None,None] in [2,1]

Action： [1,2]

Exploring a wumpus world

1,4	2,4	3,4	4,4
1,3 W!	2,3	3,3	4,3
1,2 A S OK	2,2 OK	3,2	4,2
1,1 V OK	2,1 B V OK	3,1 P!	4,1

percept [Stench,None,None,None,None] in [1,2]

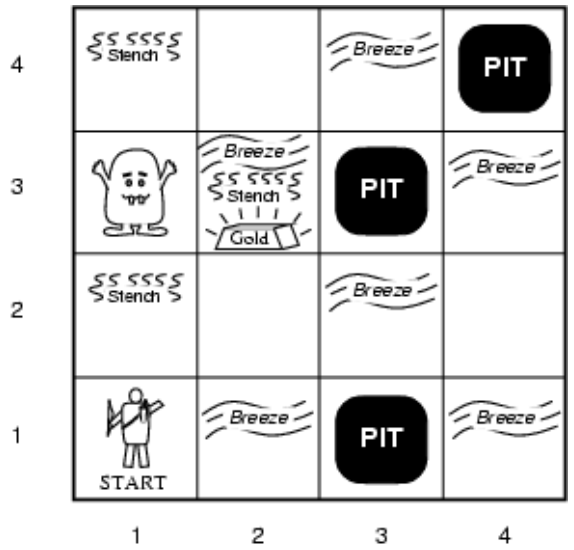


wumpus is in [1,3].
the lack of a breeze in [1,2] implies no pit in [2,2]
neither a pit nor a wumpus in [2,2], so it is OK
pit in [3,1]

Action： [2,2]

Exploring a wumpus world

1,4	2,4	3,4	4,4
1,3 W!	2,3 OK	3,3	4,3
1,2 S OK	2,2 A OK	3,2 OK	4,2
1,1 V OK	2,1 B V OK	3,1 P!	4,1



[2,3] is OK

[3,2] is OK

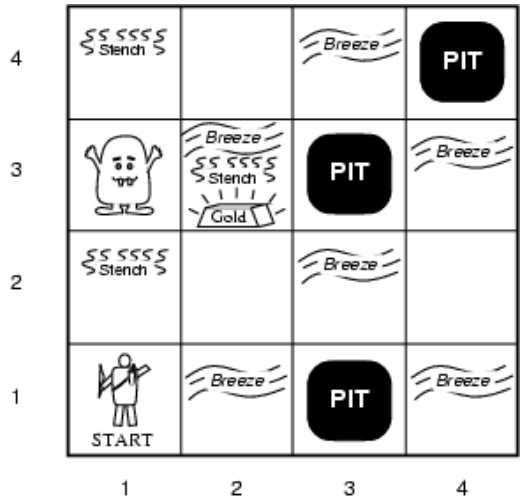
percept [None,None,None,None,None] in [2,2]

Action： [2,3]

Exploring a wumpus world

1,4	2,4 P?	3,4	4,4
1,3 W!	2,3 A S G OK B	3,3 P?	4,3
1,2 S V OK	2,2 V OK	3,2 OK	4,2
1,1 V OK	2,1 B V OK	3,1 P!	4,1

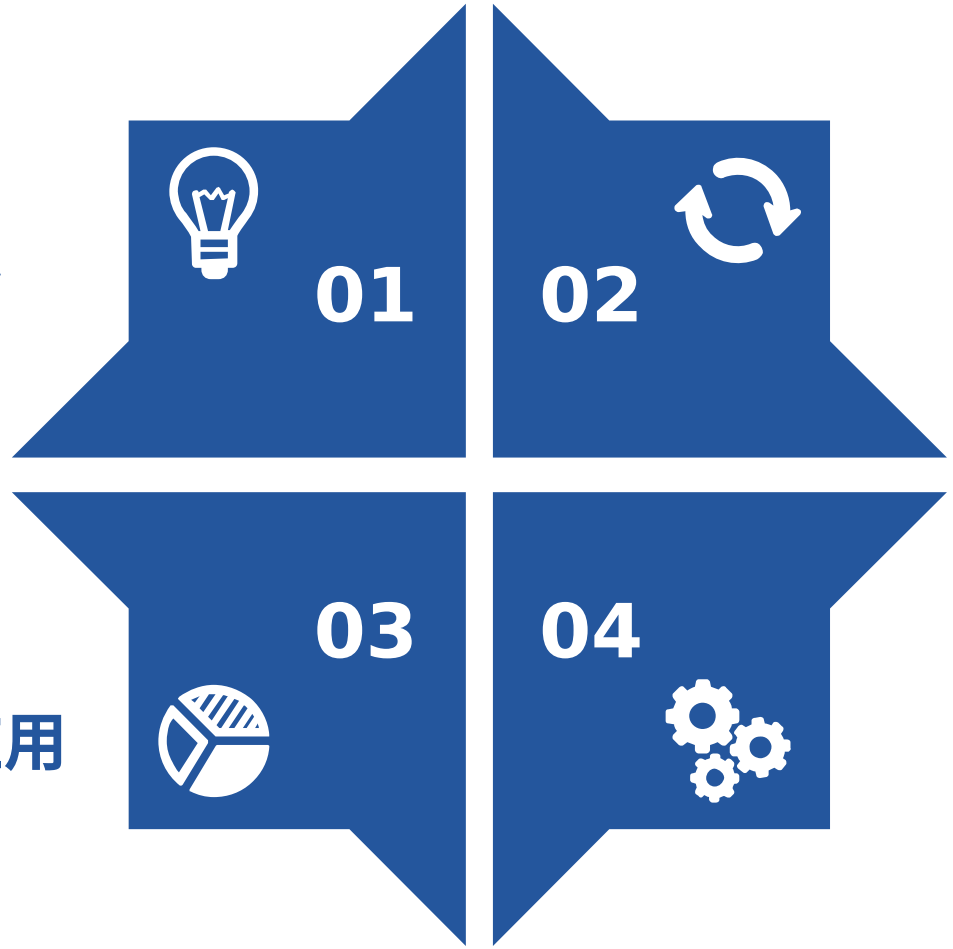
percept [Stench,Breeze,Glitter,None,None] In [2,3]



the agent detects a glitter, so it should grab the gold and then return home.

同济大学
TONGJI UNIVERSITY

知识表示方法



命题逻辑与谓词逻辑

归结原理及应用

逻辑推理案例

命题逻辑

同济大学 TONGJI UNIVERSITY

定义 命题：确定为真或为假的陈述句

命题通常用小写符号来表示，如p或者 q。命题总是具有一个“真值”，它只有两种可能性：真或假，分别用符号T（True）和F（False）表示。只有具有确定真值的陈述句才是命题，无法判断真或假的描述性句子不能作为命题

- 北京是中国的首都　（✓，真命题）
- 请出去　（×，祈使句）
- 您去开会吗?　（ ×，疑问句）
- 13能被6整除　（ ✓，　假命题）
- 这座山真高啊！（ ×，感叹句）
- 我正在说谎　（ ×，悖论）

原子命题：不包含其他命题作为其组成部分的命题，又称简单命题。

定义 复合命题：包含其他命题作为其组成部分的命题

命题联结词	表示形式	意义
与 (and)	$p \wedge q$	命题合取（conjunction），即“p且q”
或 (or)	$p \vee q$	命题析取（disjunction），即“p或q”
非 (not)	$\neg p$	命题否定（negation），即“非p”
条件 (conditional)	$p \rightarrow q$	命题蕴含（implication），p称为前件，q称为后件，即“如果p，则q”
双向条件 (bi-conditional)	$p \leftrightarrow q$	命题双向蕴含（bi-implication），即“p当且仅当q”

五种主要的命题联结词

主要联结词构成的复合命题的真值表及其涵义说明

- 条件命题联结词中前件为假时，无论后件取值如何，复合命题均为真。

- “如果p，则q（$p \rightarrow q$）”定义的是一种蕴含关系，也就是命题q包含命题p（p是q的子集）。p不成立相当于p是一个空集，而空集是任何集合的子集。因此，当p不成立时，“如果p，则q（$p \rightarrow q$）”恒为真。

F	F	T	F	F	T	T
F	T	T	F	T	T	F
T	F	F	F	T	F	F
T	T	F	T	T	T	T

五种主要联结词构成的复合命题的真值表

命题逻辑

定义逻辑等价：具有相同的真假结果，一般用≡来表示

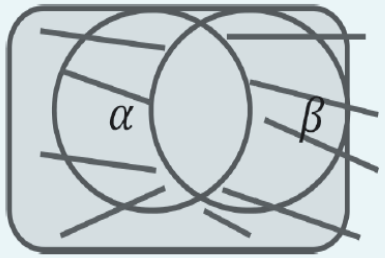
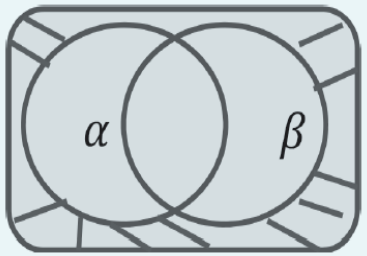
- 给定命题p和命题q，如果p和q在所有情况下都具有相同的真假结果，那么p和q在逻辑上等价，一般用≡来表示，即$p \equiv q$。
- 逻辑等价为命题进行形式转换带来了可能。基于命题转换，我们不再需要逐一列出 p和q的真值表来判断两者是否在逻辑上等价，而是可直接根据已有逻辑等价公式来判断p和q在逻辑上是否等价。

$\alpha \wedge \beta \equiv \beta \wedge \alpha$ （∧的交换律）	$(\alpha \rightarrow \beta) \equiv \neg\alpha \vee \beta$ （蕴涵消除）
$\alpha \vee \beta \equiv \beta \vee \alpha$ （∨的交换律）	$(\alpha \leftrightarrow \beta) \equiv (\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$ （双向消除）
$(\alpha \wedge \beta) \wedge \gamma \equiv \alpha \wedge (\beta \wedge \gamma)$ （∧的结合律）	$\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta)$ （De Morgan定律）
$(\alpha \vee \beta) \vee \gamma \equiv \alpha \vee (\beta \vee \gamma)$ （∨的结合律）	$\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta)$ （De Morgan定律）
$\neg(\neg\alpha) \equiv \alpha$ （双重否定）	$(\alpha \wedge (\beta \vee \gamma)) \equiv (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$ （∧对∨的分配律）
$(\alpha \rightarrow \beta) \equiv \neg\beta \rightarrow \neg\alpha$ （逆否命题）	$(\alpha \vee (\beta \wedge \gamma)) \equiv (\alpha \vee \beta) \wedge (\alpha \vee \gamma)$ （∨对∧的分配律）

一些逻辑等价的例子

命题逻辑

表 2.4　若干逻辑等价的解释

$(\alpha \rightarrow \beta) \equiv \neg\beta \rightarrow \neg\alpha$（逆否命题）	秋天天气变凉→大雁南飞越冬≡大雁没有南飞越冬→秋天天气没有变凉
$(\alpha \rightarrow \beta) \equiv \neg\alpha \vee \beta$（蕴涵消除）	α 为假，则命题恒为真；α 为真，命题与 β 真值相同
$\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta)$（德摩根定律）	
$\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta)$（德摩根定律）	

若干逻辑等价的解释

为什么要从命题逻辑到谓词逻辑

在命题逻辑中，原子命题是最基本的单位。由原子命题出发，通过使用命题联结词，构成复合命题。命题逻辑只能把复合命题分解为简单命题（即原子命题），无法对原子命题所包含的丰富语义（如个体、部分或全体等）进行刻画。因此，命题逻辑无法表达局部与整体、一般与个别的关系

α：所有的人都是要死的

β：苏格拉底是人

γ：所以苏格拉底是要死的

苏格拉底三段论

虽然苏格拉底三段论是正确的，但无法通过命题逻辑来进行推理判断（$\alpha \wedge \beta \rightarrow \gamma$）。其中的原因是上述的原子命题蕴涵了个体（苏格拉底）、群体（所有的人）和关系（都是要死的）等内在丰富语义，命题逻辑无法表现这种内在丰富语义。因此，需要引入更加强大的逻辑表示方法，分离其主语（个体或群体）和谓语（关系），这就是谓词逻辑。

无法用命题逻辑来表示苏格拉底三段论

谓词逻辑

同济大学 TONGJI UNIVERSITY

定义 谓词逻辑：刻画主体（个体和群体）之间逻辑关系的方法

在谓词逻辑中，将原子命题进一步细化，分解出个体、谓词和量词，来表达个体与总体的内在联系和数量关系，这就是谓词逻辑的研究内容。

- **个体：个体是指所研究领域中可以独立存在的具体或抽象的概念。**
- **谓词：谓词是用来刻画个体属性或者描述个体之间关系存在性的元素，其值为真或为假**

- 包含一个参数的谓词称为一元谓词，表示一元关系，通常用于刻画个体是否包含特定的属性，如$P(x)$：x是质数，表示某个数是否是质数；包含多个参数的谓词称为多元谓词，用于表示个体之间的多元关系，通常用于描述个体之间是否存在特定的关联，如$Father(x, y)$表示x是y的父亲。
- 函数是从定义域到值域的映射；谓词是从定义域到$\{True, False\}$的映射。

从自然语言描述到谓词描述

① Richard 是国王

② Lucy 和 Lily 是姐妹

③ 北京是中国的首都

① King(Richard)。其中，Richard是一个个体常量，King是一个描述“国王”这个一元关系的谓词。

② Sister(Lucy, Lily)。其中，Lucy和Lily是两个个体常量，Sister是一个描述“姐妹”这个二元关系的谓词。

③ Capital(北京, 中国)。其中，北京和中国是两个个体常量，Capital是一个描述“首都”这个二元关系的谓词。

定义 全称量词和存在量词

- 全称量词：全称量词表示一切的、所有的、凡是、每一个等，用符号∀表示
- 存在量词：存在量词表示存在、有一个、某些等，用符号∃表示

$\forall x$表示定义域中的所有个体，$(\forall x)P(x)$表示定义域中的所有个体具有性质P；$\exists x$表示定义域中存在一个或若干个个体，$(\exists x)P(x)$表示定义域中存在一个个体或若干个体具有性质P。

① $(\forall x)(A(x) \vee B(x)) \equiv (\forall x)A(x) \vee (\forall x)B(x)$不成立

② $(\forall x)(A(x) \wedge B(x)) \equiv (\forall x)A(x) \wedge (\forall x)B(x)$成立

③ $(\exists x)(A(x) \vee B(x)) \equiv (\exists x)A(x) \vee (\exists x)B(x)$成立

④ $(\exists x)(A(x) \wedge B(x)) \equiv (\exists x)A(x) \wedge (\exists x)B(x)$不成立

从自然语言描述到谓词描述

(1)所有的国王都是人

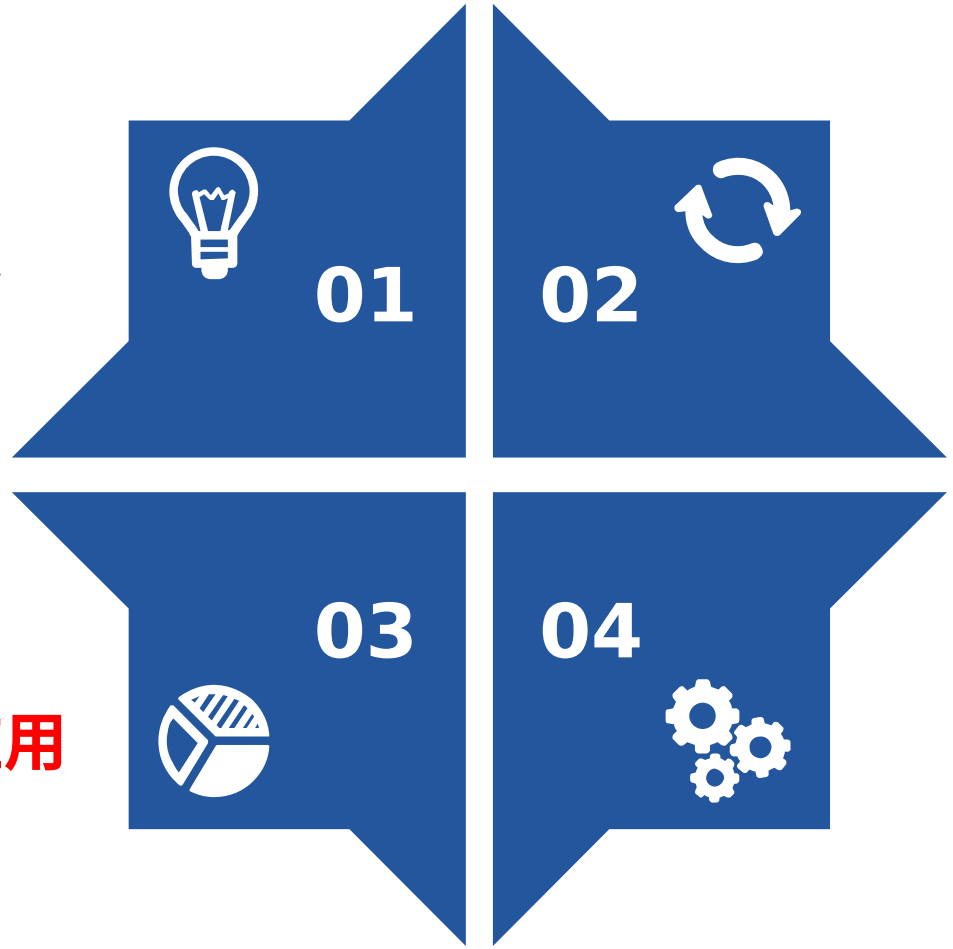
(2)所有的国王都头戴皇冠

(1) “所有的国王都是人”表示的含义为“对于所有的x，如果x是国王，那么x是人”，其符号化表示为$(\forall x)(King(x) \rightarrow Person(x))$。其中$x$是变量符号，由于$x$受到全称量词的约束，因此$x$是约束变元；$King(x)$是一个一元谓词，表示$x$是国王，$Person(x)$是一个一元谓词，表示$x$是人。

(2) “所有的国王都头戴皇冠”表示的含义为“对于所有的x，如果x是国王，那么x头戴皇冠”，符号化表示为$(\forall x)(King(x) \rightarrow Head_On(\text{Crown}, x))$。其中$x$是变量符号，由于$x$受到全称量词的约束，因此$x$是约束变元；Crown是一个常量符号，表示皇冠； $King(x)$是一个一元谓词，表示x是国王，$Head_On(\text{Crown}, x)$是一个二元谓词，表示x头戴皇冠。

同济大学
TONGJI UNIVERSITY

知识表示方法



命题逻辑与谓词逻辑

归结原理及应用

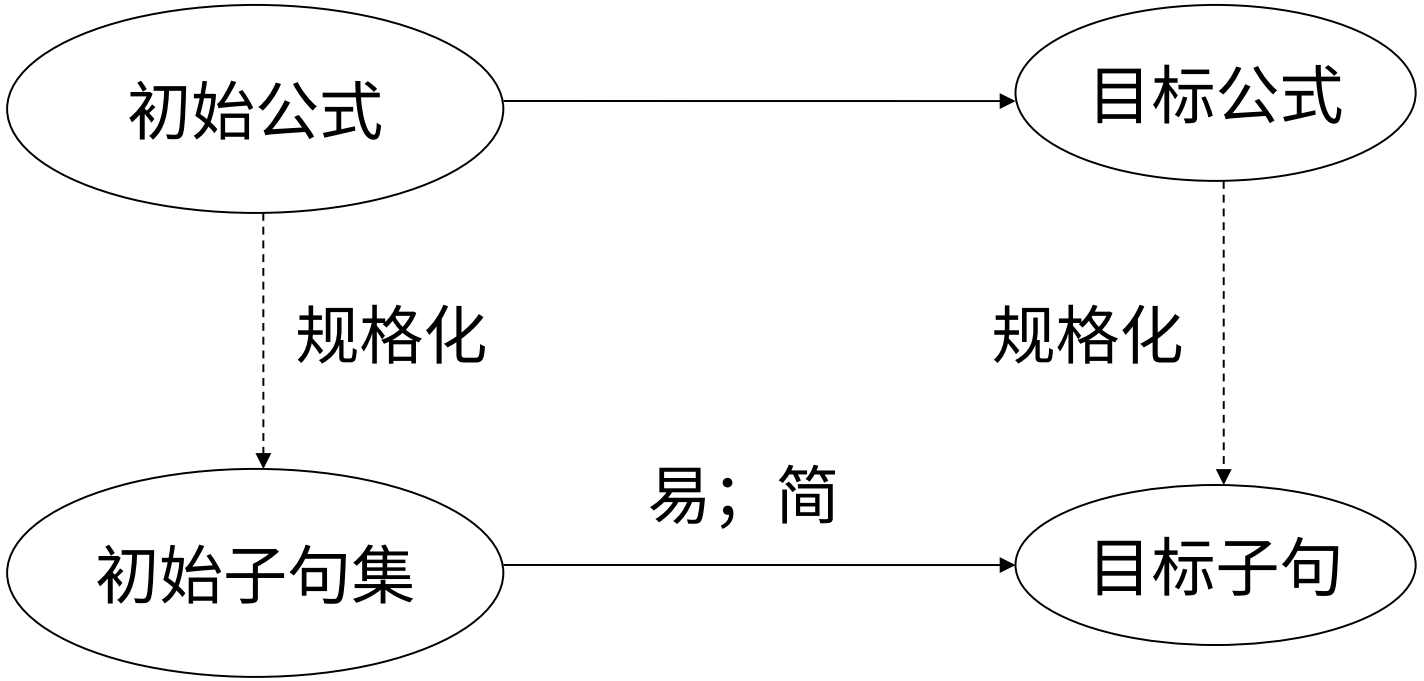
逻辑推理案例

归结原理及应用 - 一阶逻辑化为子句

同济大学 TONGJI UNIVERSITY

1. 为什么要化为子句

$(\forall x)\{[\neg P(x)\vee Q(x)]\rightarrow(\exists y)[S(x,y)\wedge Q(x)]\}\wedge(\forall x)[P(x)\vee B(x)]$



归结原理及应用 - 一阶逻辑化为子句

- **子句概念**

谓词逻辑中，把原子公式及原子公式的否定统称为文字

【定义 4.1 】任何组成合取范式的析取式称为子句。

例如 $P \vee Q$ 、$\boxtimes P$ （x， $f(x)$， y）$\vee Q(y) \vee R(f(x))$ *都是子句*

【定义 4.2 】不包含任何文字的子句称为空子句，表示为 NIL

由于空子句不包含有文字，它不能被任何解释满足，所以空子句是永假的，不可满足的

由子句构成的集合称为子句集

一阶逻辑中，任何一个一阶逻辑公式都可以化成一个子句集

归结原理及应用 - 一阶逻辑化为子句

同济大学 TONGJI UNIVERSITY

将一阶逻辑公式化为子句集的步骤:

(1) **消蕴含和等价**：利用 $P \to Q = \neg P \vee Q$；$P \leftrightarrow Q = (P \wedge Q) \vee (\neg P \wedge \neg Q)$ 等价关系消去蕴含符“→”和双条件符“↔”

(2) **否定内移**：利用 $\neg\neg P = P$；$\neg(P \vee Q) = \neg P \wedge \neg Q$；$\neg(P \wedge Q) = \neg P \vee \neg Q$；$\neg(\exists x)P = (\forall x)(\neg P)$；$\neg(\forall x)P = (\exists x)(\neg P)$ 等价关系把否定符号“¬”移到紧靠谓词位置上

(3) **变量标准化**：利用 $(\forall x)P(x) = (\forall y)P(y)$；$(\exists x)P(x) = (\exists y)P(y)$ 等价关系将变量标准化，即使每个量词采用不同的变量

归结原理及应用 - 一阶逻辑化为子句

将一阶逻辑公式化为子句集的步骤:

(4) **消去存在量词∃:**

如果存在量词不在任何一个全称量词的辖域内，则该存在量词不依赖于任何其它的变量，因此可用一个新的个体常量代替 如将 (∃x)P（x）化为 P(A)

如果存在量词是在全称量词的辖域内（如在公式 (∀y)（(∃x)P（x，y））中，变量 x 的取值依赖于变量 y 的取值）

由 **Skolem 函数（f(y)）**表示依赖关系 注意，函数名应是原合式公式中没有的。

归结原理及应用 - 一阶逻辑化为子句

将一阶逻辑公式化为子句集的步骤：

(5) 将公式化为前束形：把所有全称量词移到公式的左边，并使每个量词的辖域包含这个量词后面的整个部分，所得的公式称为前束形

(6) 化为合取范式：利用 P∨（Q∧R）＝（P∨Q）∧（P∨R）；

（P∧Q）∨（P∧R）＝P∧（Q∨R）等价关系将母式化为合取范式（**子句的合取式**）

(7) 略去全称量词：母式中的变量都是全称量词量化的变量

(8) 消去合取符号∧，把母式用**子句集**表示

如：P∧Q 可表示为成子句集： P Q

(9) 子句变量标准化：重新命名变量，使每个子句中的变量符号不同

归结原理及应用 - 一阶逻辑化为子句

讲下列一阶逻辑表示化为子句：

【例】将 $(\forall x)\{[\neg P(x)\vee\neg Q(x)]\rightarrow(\exists y)[S(x,y)\wedge Q(x)]\}\wedge(\forall x)[P(x)\vee B(x)]$ 化成子句集

转换过程遵照上述 9 个步骤：

(1) $(\forall x)\{\neg[\neg P(x)\vee Q(x)]\vee(\exists y)[S(x,y)\wedge Q(x)]\}\wedge(\forall x)[P(x)\vee B(x)]$

(2) $(\forall x)\{[P(x)\wedge Q(x)]\vee(\exists y)[S(x,y)\wedge Q(x)]\}\wedge(\forall x)[P(x)\vee B(x)]$

(3) $(\forall x)\{[P(x)\wedge Q(x)]\vee(\exists y)[S(x,y)\wedge Q(x)]\}\wedge(\forall w)[P(w)\vee B(w)]$

(4) $(\forall x)\{[P(x)\wedge Q(x)]\vee[S(x,f(x))\wedge Q(x)]\}\wedge(\forall w)[P(w)\vee B(w)]$

归结原理及应用 - 一阶逻辑化为子句

(5) $(\forall x)(\forall w)$

$\{[P(x) \wedge Q(x)] \vee [S(x, f(x)) \wedge Q(x)]\} \wedge [P(w) \vee B(w)]$

(6) $(\forall x)(\forall w)\{[P(x) \vee S(x, f(x))] \wedge Q(x) \wedge [P(w) \vee B(w)]\}$

(7) $[P(x) \vee S(x, f(x))] \wedge Q(x) \wedge [P(w) \vee B(w)]$

(8) 子句集为: $P(x) \vee S(x, f(x))$; $Q(x)$; $P(w) \vee B(w)$

(9) 子句变量标准化后，最终的子句集为:

$P(x) \vee S(x, f(x))$; $Q(y)$; $P(w) \vee B(w)$

归结原理

- **置换和合一**

为了使用推理规则，如假言推理、假言三段论等，一个推理系统必须决定两个表达式是否相同或匹配：

两个表达式匹配当且仅当其语法是等价的

一个表达式的项可以是常量、变量或函数，**合一**就是寻找项对变量的**置换**而使表达式一致的过程，合一是人工智能中很重要的过程。

如，为了使公式 P(x,f(y),B) 与 P(x,f(B),B ）匹配，可以用常量 B 代替变量 y，从而使两个公式一致。 称为通个置换 {B/y} 就可使上述公式集合一。

归结原理

- **置换和合一**

置换可用有序对的集合 $s=\{t_1/v_1, t_2/v_2, \cdots, t_n/v_n\}$ 表示，其中 t_i/v_i 表示将表达式中所有的变量 v_i 都用项 t_i 代替，t_i 可以是变量、常量或函数。

注意，一个变量不能用含有同一变量的项来代替，被置换的一定是变量

一般可用 Es 表示一个表达式 E 用一个置换 s 所得到的表达式的置换。

如：$P(z, f(w), A) = (P(x, f(y), A))_s$

归结原理

- **置换和合一**

例如：有表达式P（x，f（y），A），通过不同的置换，可分别得到：

P（z，f（w），A）

相应的置换为 s1 = { z/x ， w/y }

P（x，f（B），A） 相应的置换为 s2 = { B/ y }

P（g(z)，f（B），A）

相应的置换为 s3 = { g(z)/x ， B /y }

P（C，f（B），A）

相应的置换为 s4 = { C/x ， B/y }

归结原理

- **归结原理**

归结原理又称为消解原理，它是定理证明基础。

由谓词公式转化为子句集的过程中可以看出，在子句集中子句之间是合取关系，其中只要有一个子句不可满足，则子句集就不可满足。若一个子句集中包含空子句，则这个子句集一定是不可满足的。

归结原理就是基于这一认识提出来的。

【定义 5.1 】若 P 是原子谓词公式，则称 P 与☒ P 为互补文字 。

归结原理

- **一阶逻辑归结**

在一阶逻辑中，子句中含有变量

为将归结原理应用于含有变量的子句，应找出一个置换，作用于给定的两个子句，使它们包括互补的文字，然后才能进行归结。

例 1: 子句集 S={P(x)∨Q(x)，☒ P(A)∨R(y)}，子句集中的两个子句不能直接归结，但若对子句集先进行置换 s={A/x}，则两个子句分别为 P(A)∨Q(A) 和 ☒ P(A)∨R(y)，这时再进行归结，归结结果为 Q(A)∨R(y) 。

归结原理

【定义 5.2】设 C_1 和 C_2 是两个没有相同变量的子句，并分别表示成两个文字集合 $\{L_i\}$ 和 $\{M_i\}$，$\{l_i\}$ 是 $\{L_i\}$ 的一个子集，$\{m_i\}$ 是 $\{M_i\}$ 的一个子集，若 s 是集合 $\{l_i\}$ 和 $\{\neg m_i\}$ 的并集的最简合一者，则称

$$C_{12} = \{\{L_i\}-\{l_i\}\}_s \vee \{\{M_i\}-\{m_i\}\}_s$$

为 C_1 和 C_2 的归结式。

当两个子句作归结时，子集 $\{l_i\}$ 和 $\{m_i\}$ 的选取可能有多种形式，所以得到的归结式不是唯一的。

归结原理

例 2： 设有两个子句 $P(A)\vee\neg Q(x)\vee R(x)$ 和 $P(y)\vee Q(B)$，则由如下两种归结方法：

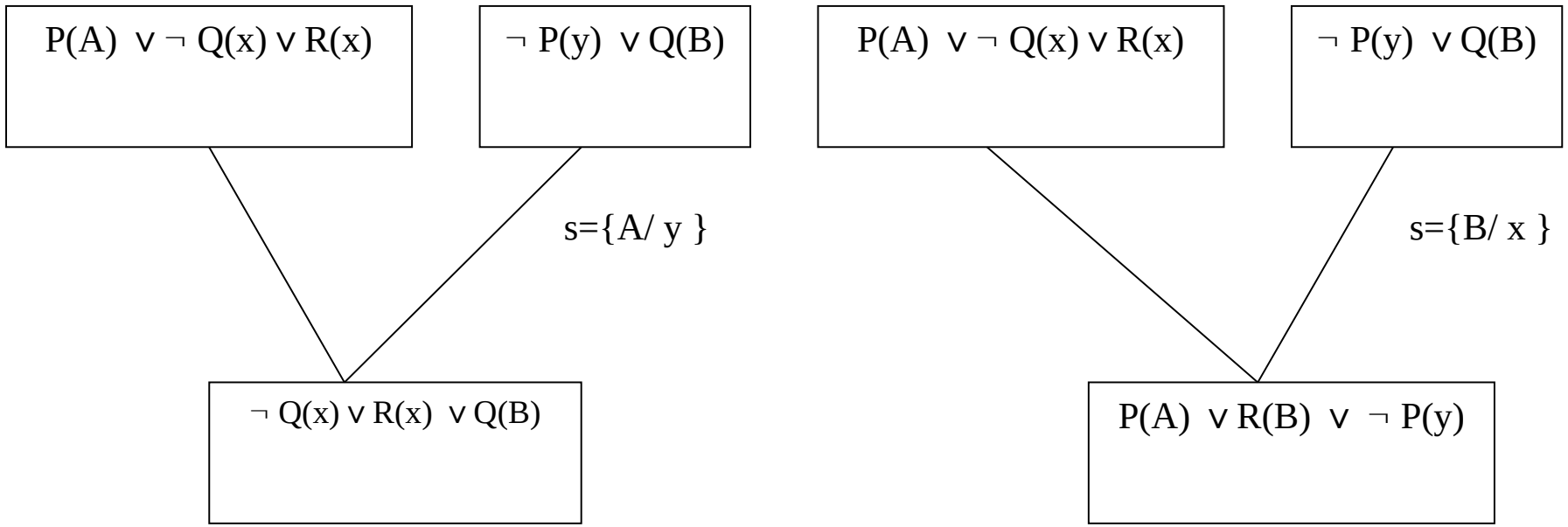
① 取 $\{l_i\}=\{P(A)\}$，$\{m_i\}=\{\neg P(y)\}$，$\{l_i\}$ 和 $\{\neg m_i\}$ 的最简合一者为 $s=\{A/y\}$，此时归结结果为 $Q(x)\vee R(x)\vee Q(B)$

② 取 $\{l_i\}=\{\neg Q(x)\}$，$\{m_i\}=\{Q(B)\}$，$\{l_i\}$ 和 $\{\neg m_i\}$ 的最简合一者为 $s=\{B/x\}$，此时归结结果为 $P(A)\vee R(B)\vee\neg P(y)$

- **注意：** 在求归结式时，**不能同时消去两个互补文字**对，消去两个互补文字对所得的结果不是两个亲本子句的逻辑推论。

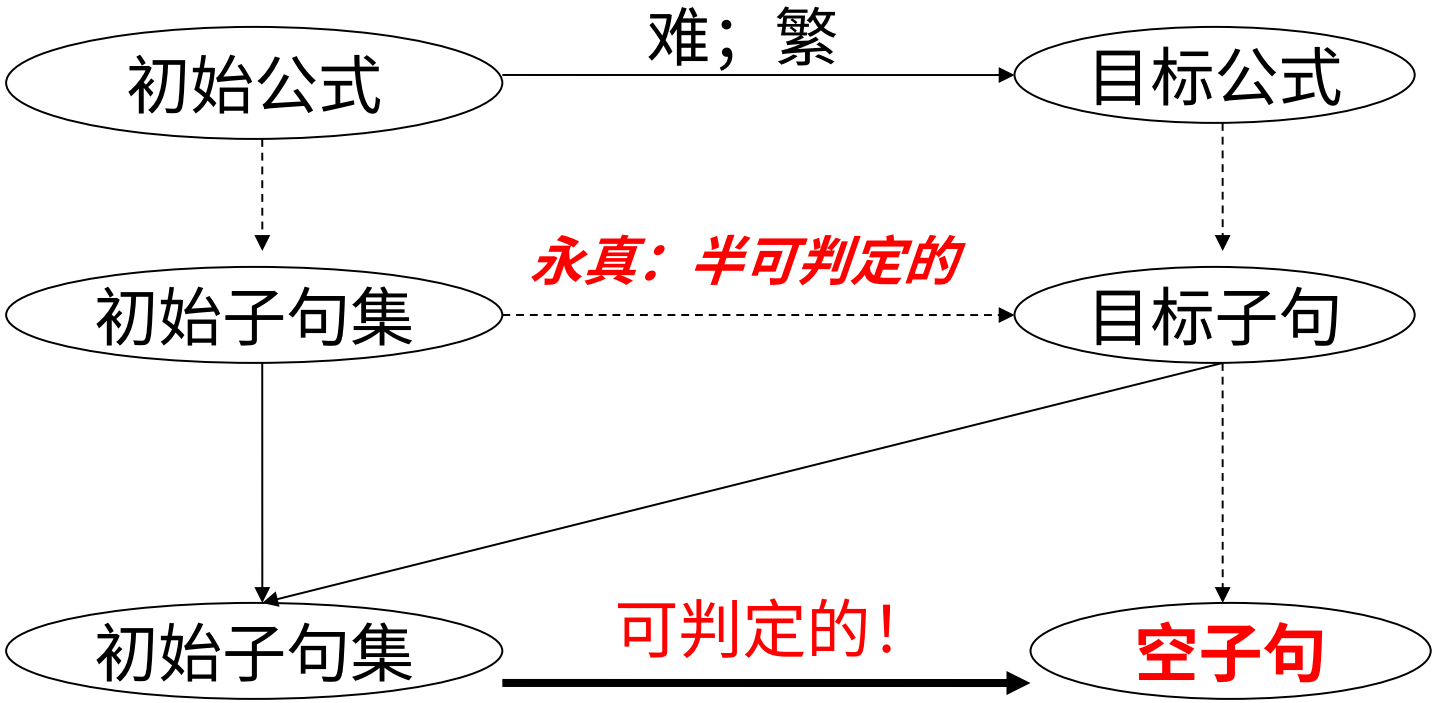
归结原理

例 2 的归结过程：



归结原理

- **归结反演**



归结原理

- **归结反演**

归结反演就是利用归结和反演实现定理的证明

具体过程:

(1) 将定理证明的前提谓词公式转化为子句集 F

(2) 将求证的目标表示成合适的谓词公式 G （目标公式）

(3) 将目标公式的否定式¬ G 转化成子句的形式，并加入到子句集 F 中，得到子句集 S

(4) 应用归结原理对子句集中的子句进行归结，并把每次归结得到的归结式都并入 S 中。如此反复进行，若归结得到一个空子句 NIL，则停止归结¬ 证明了 G 为真

归结原理的应用

- 应用归结原理进行定理证明

步骤：

设要被证明的定理可用谓词公式表示为如下的形式：

$$A_1 \wedge A_2 \wedge \ldots \wedge A_n \rightarrow B$$

(1) 首先否定结论 B，并将否定后的公式～ B 与前提公式集组成如下形式的谓词公式：

$$G = A_1 \wedge A_2 \wedge \ldots \wedge A_n \wedge \sim B$$

(2) 求谓词公式 G 的子句集 S 。

(3) 应用归结原理，证明子句集 S 的不可满足性。

归结原理的应用

例 6.1】

- 已知前提为 F:(∀x){[P(x,y)∧Q(y)]→(∃y)[R(y)∧S(x,y)]}
- 求证结论 G:¬(∃x)R(x)→(∀x)(∀y)[P(x,y)→¬Q(y)] 成立
- 证明：先按前面所讲的方法将前提和结论化为子句集：

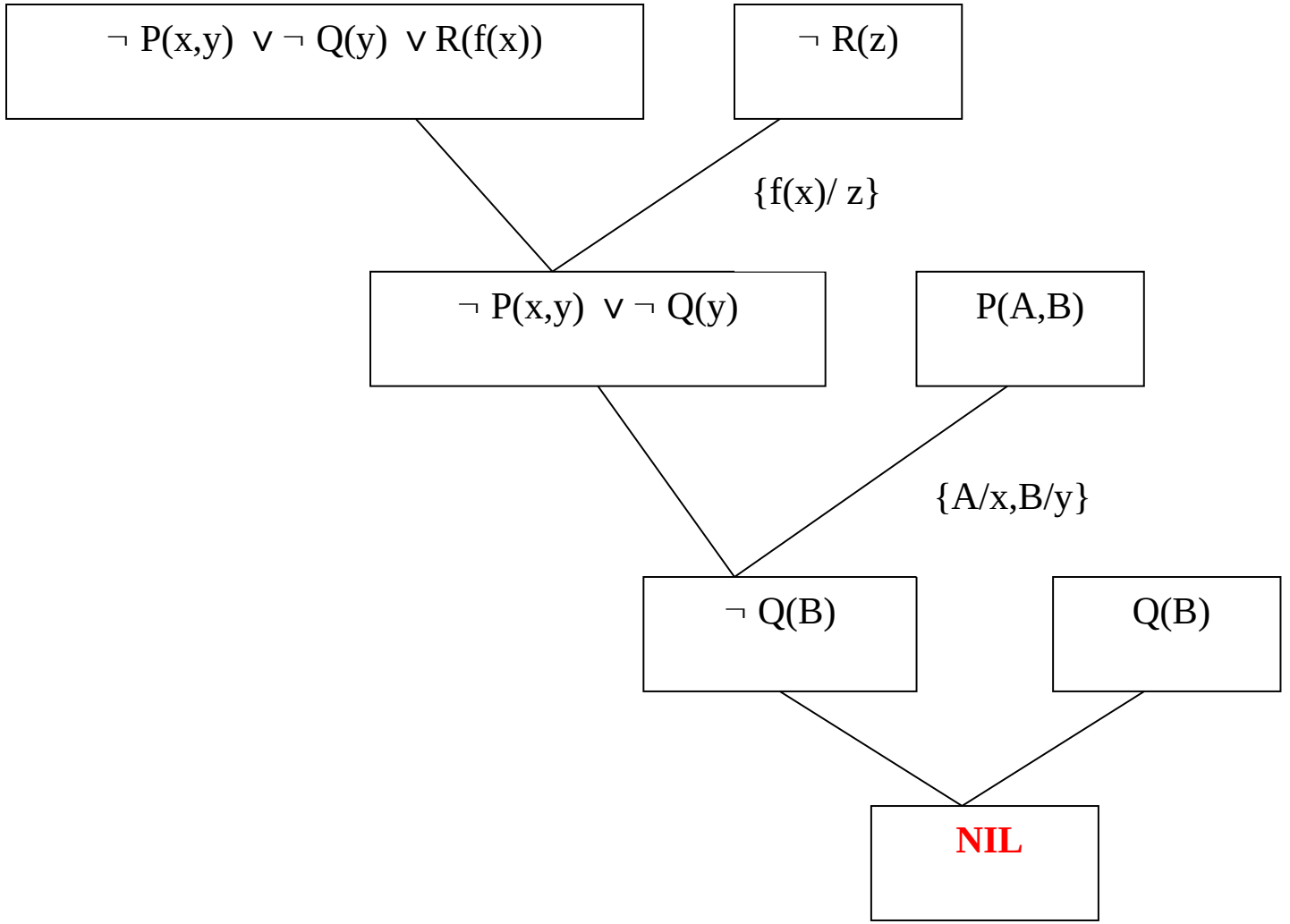
前提 F 所对应的子句集为：☒ P(x,y)∨¬Q(y)∨R(f(x))

¬ P(x,y)∨¬Q(y)∨S(x, f(x))

结论 G 否定对应子句集为：☒ R(z)；　P(A,B)；　Q(B)

归结原理的应用

- 归结过程如下：



归结原理的应用

例 6.2. 已知：某些病人喜欢所有的医生，

没有一个病人喜欢任意一个骗子。

证明：任意一个医生都不是骗子。

证明： 知识表示：令

P(x)： x 是病人　　D(x)： x 是医生

Q(x)： x 是骗子　　L(x, y)： x 喜欢 y

A1: $\exists x\,(P(x) \wedge \forall y(D(y) \rightarrow L(x, y)))$

A2: $\forall x(P(x) \rightarrow \forall y(Q(y) \rightarrow \sim L(x, y)))$

B: $\forall x(D(x) \rightarrow \sim Q(x))$

我们要证明 B 是 A1 和 A2 的逻辑结果，即公式 $A1 \wedge A2 \wedge \sim B$ 是不可满足的。

归结原理的应用

同济大学 TONGJI UNIVERSITY

$A1=\exists x\,(P(x) \wedge \forall y(\sim D(y) \vee L(x, y)))$

$=\exists x\ \forall y\,(P(x) \wedge (\sim D(y) \vee L(x, y)))$

$\dashrightarrow \forall y\,(P(a) \wedge (\sim D(y) \vee L(a, y)))$

$A2=\forall x(P(x) \rightarrow \forall y(\sim Q(y) \vee \sim L(x, y)))$

$=\forall x(\sim P(x) \vee \forall y(\sim Q(y) \vee \sim L(x, y)))$

$=\forall x \forall y\,(\sim P(x) \vee \sim Q(y) \vee \sim L(x, y))$

$\sim B=\sim(\forall x(D(x) \rightarrow \sim Q(x)))$

$=\exists x\,(D(x) \wedge Q(x))$

$\dashrightarrow D(b) \wedge Q(b)$

因此，公式 $A1 \wedge A2 \wedge \sim B$ 的子句集为

$S = \{P(a), \sim D(y) \vee L(a,y), \sim P(x) \vee \sim Q(y) \vee \sim L(x, y), D(b), Q(b)\}$

归结原理的应用

S 不可满足的归结演绎序列为:

(1) P(a)

(2) ～D(y) ∨ L(a, y)

(3) ～P(x) ∨ ～Q(y) ∨ ～L(x, y)

(4) D(b)

(5) Q(b)

(6) L(a, b) 由 (2)、(4) mgu:{b/y}

(7) ～Q(y) ∨ ～L(a, y) 由 (1)、(3) mgu:{a/x}

(8) ～L(a, b) 由 (5)、(7) mgu:{b/y}

(9) □ 由 (6)、(8)

归结原理的应用

- 利用归结原理求取问题答案

步骤：

（1）把已知前提条件用谓词公式表示出来，并化成相应的子句集，设该子句集的名字为 S_1。

（2）把待求解的问题也用谓词公式表示出来，然后将其否定，并与一谓词 ANSWER 构成析取式。谓词 ANSWER 是一个专为求解问题而设置的谓词，其变量必须与问题公式的变量完全一致。

（3）把（2）中的析取式化为子句集，并把该子句集与 S_1 合并构成子句集 S。

（4）对子句集 S 应用归结原理进行归结，在归结的过程中，通过合一，改变 ANSWER 中的变元。

（5）如果得到归结式 ANSWER，则问题的答案即在 ANSWER 谓词中。

归结原理的应用

例 6.3. 任何兄弟都有同一个父亲，

John 和 Peter 是兄弟，且 John 的父亲是 David ，

问 :Peter 的父亲是谁?

解 第一步：将已知条件用谓词公式表示出来，并化成子句集，那么要先定义谓词。

（1） 定义谓词:

设 Father(x,y) 表示 x 是 y 的父亲。

Brother(x,y) 表示 x 和 y 是兄弟。

归结原理的应用

（2） 将已知事实用谓词公式表示出来。

F1 ： 任何兄弟都有同一个父亲。

$\forall x \forall y \forall z\ (Brother(x,y) \wedge Father(z,x) \rightarrow Father(z,y))$

F2： John 和 Peter 是兄弟。

Brother(John,Peter)

F3： John 的父亲是 David 。

Father(David, John)

（3） 将它们化成子句集得：

$S1=\{ \sim Brother(x,y) \vee \sim Father(z,x) \vee Father(z,y),$

Brother(John,Peter), Father(David,John)$\}$

归结原理的应用

第二步：把问题用谓词公式表示出来，

并将其否定与谓词 ANSWER 作析取。

设 Peter 的父亲是 u ，则有： Father(u,Peter) 。

将其否定与 ANSWER 作析取，得：

G： ～ Father(u,Peter)∨ANSWER(u)

第三步：将上述公式 G 化为子句集 S2, 并将 S1 和 S2 合并到 S 。

S2 ={ ～ Father(u,Peter)∨ANSWER(u)}

S= S1∪S2

归结原理的应用

第四步：应用归结原理进行归结

（5）～Brother(John,y)∨Father(David,y)

（1）与（3）归结 σ={David/z,John/x}

（6）～Brother(John,Peter)∨ANSWER(David)

（4）与（5）归结 σ={David/u,Peter/y}

（7） ANSWER(David) （2）与（6）归结

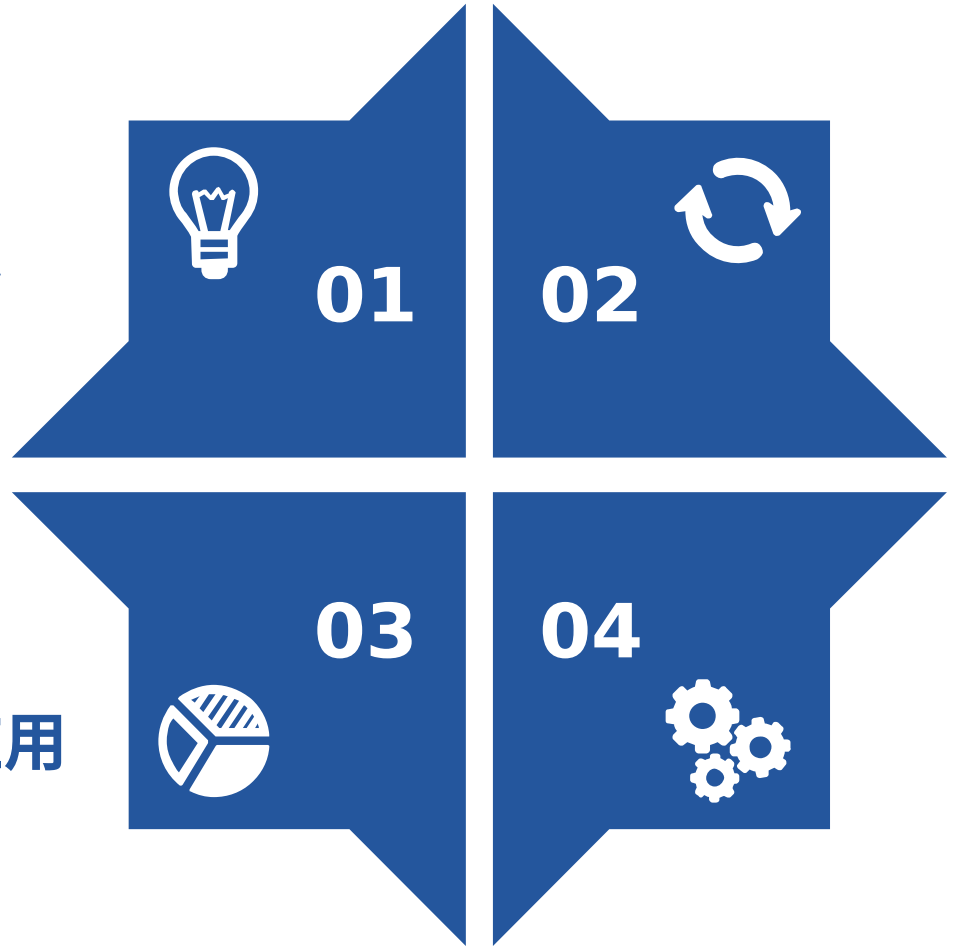
第五步：

得到了归结式 ANSWER(David)，答案即在其中，所以 u=David 。

即 Peter 的父亲是 David 。

同济大学
TONGJI UNIVERSITY

知识表示方法



命题逻辑与谓词逻辑

归结原理及应用

逻辑推理案例

案例 1： 破案推理

同济大学 TONGJI UNIVERSITY

案例背景：

在某刑事案件中，一名侦探正在调查一桩**盗窃案**。已知：

- 盗窃发生在**晚上**。
- 犯罪嫌疑人要么是**张三**（ZhangSan），要么是**李四**（LiSi），但不可能是两人一起作案。
- 目击者证实：**罪犯在案发时戴着黑色手套**。
- 调查发现：张三家中有**黑色手套**，但李四没有。
- 另有证据表明：**罪犯作案时必须会撬锁**，但张三不会撬锁，而李四是锁匠。
- 由于犯罪现场没有强行进入的痕迹，说明**罪犯是用撬锁进入的**。

目标：使用**归结原理**进行逻辑推理，确定罪犯是谁。

案例 1： 破案推理

同济大学 TONGJI UNIVERSITY

（1）定义谓词

- Suspect(X)： X 是嫌疑人
- BlackGloves(X)： X 有黑色手套
- Criminal(X)： X 是罪犯
- Lock(X)： X 会撬锁
- WearsGloves(X)： X 戴黑色手套
- CrimeEntry(Lock)：犯罪通过撬锁进入

（2）谓词公式

在某刑事案件中，一名侦探正在调查一桩**盗窃案**。已知：

F1: 罪犯只能是张三或李四：

$Suspect(X) \rightarrow (X = ZhangSan \vee X = LiSi)$

F2: 罪犯戴黑色手套：

$WearsGloves(X) \rightarrow BlackGloves(X)$

F3: 张三有黑色手套，李四没有：

$BlackGloves(ZhangSan)$， $\neg BlackGloves(LiSi)$

F4: 罪犯必须会撬锁：

$Criminal(X) \rightarrow Lock(X)$

F5: 张三不会撬锁，李四是锁匠：

$\neg Lock(ZhangSan)$， $Lock(LiSi)$

F6: 罪犯作案时是通过撬锁进入的： CrimeEntry (Lock)

案例 1： 破案推理

（3）子句集

步骤 1： 转化为子句集

S1 = {¬Suspect(X)∨(X = ZhangSan∨ X = LiSi),
¬WearsGloves(X)∨BlackGloves(X),BlackGloves(ZhangSan), ¬BlackGloves(LiSi), ¬Criminal(X)∨Lock(X),
¬Lock(ZhangSan),Lock(LiSi),CrimeEntry(Lock)}

步骤 2： 把待求解的问题用谓词公式表示，并构成析取式

谁是罪犯? Criminal(Y) 然后，我们取它的否定:

¬Criminal(Y)

并与谓词 ANSWER(Y) 结合形成析取式:

¬Criminal(Y)∨ANSWER(Y)

步骤 3： 把析取式化为子句集，并合并到 S1 ，形成 S

将上述问题的析取式转换为子句集 S2 ：

S2 = {¬Criminal(Y)∨ANSWER(Y)}

合并 S1 和 S2 ：

S = S1∪S2

案例 1： 破案推理

（4）归结原理归结

步骤 1: 假设张三是罪犯 Criminal(ZhangSan)，跟 F4 的子句归结，罪犯必须会撬锁：Lock(ZhangSan)

步骤 2: 已知张三不会撬锁（F5）： ¬Lock(ZhangSan)，

与步骤 1 归结，矛盾（Lock (ZhangSan) 和 ¬Lock(ZhangSan)），

得出结论：张三不可能是罪犯，¬Criminal(ZhangSan)。

步骤 3: 由于 F1 指出罪犯只能是张三或李四： Criminal(ZhangSan)∨Criminal(LiSi)

归结得到： Criminal(LiSi)

步骤 4: 归结到 ANSWER

¬Criminal(Y)∨ANSWER(Y)

与步骤 3 归结： ANSWER(LiSi)

步骤 5: 李四是罪犯!

案例 1： 破案推理

同济大学 TONGJI UNIVERSITY

Python 实现：

```
# 定义规则和已知事实
def rules_and_facts():
    # 规则 1：罪犯只能是张三或李四
    def rule1(x):
        return x == "ZhangSan" or x == "LiSi"
    # 规则 2：罪犯必须戴黑色手套
    def rule2(x):
        return f"WearsGloves({x}) → BlackGloves({x})"
    # 规则 3：张三有黑色手套，李四没有
    def rule3():
        return ["BlackGloves(ZhangSan)", "¬BlackGloves(LiSi)"]
    # 规则 4：罪犯必须会撬锁
    def rule4(x):
        return f"Criminal({x}) → Lock({x})"
    # 规则 5：张三不会撬锁，李四是锁匠
    def rule5():
        return ["¬Lock(ZhangSan)", "Lock(LiSi)"]
    # 规则 6：罪犯作案时是通过撬锁进入的
    def rule6():
        return "CrimeEntry(Lock)"
    return rule1, rule2, rule3, rule4, rule5, rule6

# 归结推理过程
def reasoning():
    rule1, rule2, rule3, rule4, rule5, rule6 = rules_and_facts()
    # 步骤 1：假设张三是罪犯
    suspect = "ZhangSan"
    print(f" 假设 {suspect} 是罪犯 ")
    # 步骤 2：根据规则 4 推导出张三必须会撬锁
    print(f" 根据规则 4，罪犯必须会撬锁： Lock({suspect})")
    # 步骤 3：根据事实，张三不会撬锁（规则 5）
    print(f" 已知张三不会撬锁： ¬Lock({suspect})")
    # 步骤 4：得出矛盾（Lock(ZhangSan) 和 ¬Lock(ZhangSan)）
    if "Lock(ZhangSan)" != "¬Lock(ZhangSan)":
        print(f" 发生矛盾，推导出： ¬Criminal(ZhangSan)")
    # 步骤 5：通过排除法，得出罪犯只能是李四
    print(" 由于张三被排除为罪犯，根据规则 1，得出： Criminal(LiSi)")
    print(" 因此，李四是罪犯： Criminal(LiSi)")
# 执行推理过程
reasoning()
```

案例 2： 复杂自动驾驶场景推理与决策

同济大学 TONGJI UNIVERSITY

案例背景：

在复杂的城市交叉路口，自动驾驶系统需要根据实时感知信息做出决策：

- 交通信号灯：红灯。
- 前方车辆：前方有一辆汽车正在变道，且其速度较快。
- 行人：行人在斑马线位置，距离车辆较近。
- 紧急情况：后方有急救车正在逼近，且正在发出警报声。
- 路况信息：前方有坑洼路段，限速为 30km/h 。

目标：通过归结原理推理出如何安全通过交叉口，避免发生碰撞，同时考虑紧急车辆的优先通行。

案例 2： 复杂自动驾驶场景推理与决策

同济大学 TONGJI UNIVERSITY

（1）定义谓词

- TrafficSignal(S, C)： 信号灯 S 的状态为 C （红、绿、黄）
- Stop(V)： 车辆 V 停止
- Proceed(V)： 车辆 V 继续行驶
- SlowDown(V)： 车辆 V 减速
- Near(V, O)： 车辆 V 靠近对象 O （行人、前方车辆）
- CrossRoadSignal(S, C)： 表示交叉车道信号 S 的状态为 C
- PedestrianNear(P)： 行人 P 靠近斑马线
- PedestrianFar(P)： 行人 P 远离车辆。
- VehicleChangLane(F)： 前方车辆 F 变道
- VehicleSpeed(F, S)： 前方车辆 F 的速度为 S （快 / 慢）
- EmergencyVehicle(EV)： 急救车 EV 逼近。
- SirenActive(EV)： 急救车 EV 发出警报。
- Yield(EV)： 车辆需要让行急救车 EV
- RoadCondition(R, C)： 道路 R 的状态为 C （坑洼、限速等）
- RoadSpeedLimit(R, S)： 道路 R 的限速为 S
- SpeedLimit(R, S)： 道路 R 的限速为 S km/h 。
- Decision(V, A)： 车辆 V 的决策为 A （停车、行驶、减速、让行）

案例 2： 复杂自动驾驶场景推理与决策

同济大学 TONGJI UNIVERSITY

（2）谓词公式

F1：交通信号规则

如果信号灯是红灯，车辆应停车。

TrafficSignal(TrafficLight,Red)→Stop(Vehicle)

如果信号灯是绿灯，且没有行人在斑马线上，车辆可以行驶。

TrafficSignal(TrafficLight,Green)∧¬PedestrianNear(Pedestrian)→Proceed(Vehicle)

如果信号灯是黄灯，且距离交叉口较近，车辆应减速并准备停车。

TrafficSignal(TrafficLight,Yellow)∧Near(Vehicle,Intersection)→SlowDown(Vehicle)

如果交叉道口有交叉车道信号，且信号允许通行，车辆可以继续行驶。

CrossRoadSignal(CrosswalkSignal,Green)→Proceed(Vehicle)

案例 2： 复杂自动驾驶场景推理与决策

（2）谓词公式

F2：行人规则

如果行人处于斑马线上，且与车辆较近，车辆应停车。

PedestrianNear(Pedestrian) → Stop(Vehicle)

如果行人离车辆较远，车辆可以通过。

PedestrianFar(Pedestrian) → Proceed(Vehicle)

F3：前方车辆规则

如果前方车辆变道且接近，车辆应减速。

VehicleChangLane(FrontVehicle) ∧ Near(Vehicle, FrontVehicle) → SlowDown(Vehicle)

如果前方车辆速度过快，且车辆接近，车辆应减速。

VehicleSpeed(FrontVehicle, Fast) ∧ Near(Vehicle, FrontVehicle) → SlowDown(Vehicle)

案例 2： 复杂自动驾驶场景推理与决策

（2）谓词公式

F4：紧急车辆规则

如果后方有急救车且正在发出警报，其他车辆必须给急救车让路。

EmergencyVehicle(EV) ∧ SirenActive(EV)→ Yield(EV)

F5：路况信息规则

如果前方道路上有坑洼且限速为 30km/h ，车辆应减速。

RoadCondition(Pothole) ∧ RoadSpeedLimit(Road, 30) → SlowDown(Vehicle)

F6：车辆速度规则

如果车辆的速度超过限速，车辆应减速。

VehicleSpeed(Vehicle, Speed) ∧ Speed > SpeedLimit(Road ， S) → SlowDown(Vehicle)

案例 2： 复杂自动驾驶场景推理与决策

同济大学 TONGJI UNIVERSITY

（3）子句集

步骤1：转化为子句集

S1 = {¬TrafficSignal(TrafficLight,Red)∨Stop(Vehicle),
¬TrafficSignal(TrafficLight,Green)∨PedestrianNear(Pedestrian)∨Proceed(Vehicle),

¬TrafficSignal(TrafficLight,Yellow)∨¬Near(Vehicle,Intersection)∨SlowDown(Vehicle),

¬CrossRoadSignal(CrosswalkSignal,Green)∨Proceed(Vehicle),

¬PedestrianNear(Pedestrian)∨Stop(Vehicle), ¬PedestrianFar(Pedestrian)∨Proceed(Vehicle),

¬VehicleChangLane(FrontVehicle)∨¬Near(Vehicle, FrontVehicle)∨SlowDown(Vehicle),

¬VehicleSpeed(FrontVehicle,Fast)∨¬Near(Vehicle, FrontVehicle)∨SlowDown(Vehicle),

¬EmergencyVehicle(EV)∨¬SirenActive(EV)∨Yield(EV),

¬RoadCondition(Pothole)∨¬RoadSpeedLimit(Road,30)∨SlowDown(Vehicle),

¬VehicleSpeed(Vehicle, Speed)∨Speed ≤ SpeedLimit(RoadS)∨SlowDown(Vehicle)}

案例 2： 复杂自动驾驶场景推理与决策

同济大学 TONGJI UNIVERSITY

（3）子句集

步骤 2：把待求解的问题用谓词公式表示，并构成析取式

在当前环境下，自动驾驶车辆的最优决策是什么?

用谓词表示： Decision(V,A) 取其否定：

¬Decision(V,A)

并与谓词 ANSWER(V,A) 结合形成析取式：

¬Decision(V,A)∨ANSWER(V,A)

步骤 3：把析取式化为子句集，并合并到 S1，形成 S

将上述问题的析取式转换为子句集 S2 ：

S2 = {¬Decision(V,A)∨ANSWER(V,A)}

合并 S1 和 S2 ：

S = S1∪S2

案例 2： 复杂自动驾驶场景推理与决策

同济大学 TONGJI UNIVERSITY

（4）归结原理归结

步骤 1: 假设 TrafficSignal(TrafficLight,Red)，跟 F1 的子句归结，车辆应停车： Stop(Vehicle)

步骤 2: 假设行人靠近车辆 PedestrianNear(Pedestrian)，跟 F2 的子句归结，车辆应停车： Stop(Vehicle)

步骤 3: 假设前方车辆变道且车速过快 VehicleChangingLane(FrontVehicle)∧VehicleSpeed(FrontVehicle,Fast)，

跟 F3 的子句归结： SlowDown(Vehicle)，但由于车辆已经停下，减速推理不会影响当前行为。

步骤 4: 假设 EmergencyVehicle(EV)∧SirenActive(EV)，跟 F4 的子句归结： Yield(EV)

步骤 5: 假设 RoadCondition(Pothole)∧RoadSpeedLimit(Road, 30)，跟 F5 的子句归结：SlowDown(Vehicle)

由于步骤 1 和步骤 2 已经归结出 Stop(Vehicle)，此规则不会影响最终决策。

步骤 6: 假设 VehicleSpeed(Vehicle,Speed)∧Speed>SpeedLimit(Road,S)，跟 F6 的子句归结：SlowDown(Vehicle)，

但车辆已经停车，减速无影响。

决策： Decision(Vehicle,Stop)∧Decision(Vehicle,Yield(EV))，即车辆停车且在急救车到达时给其让行。

案例 2： 复杂自动驾驶场景推理与决策

同济大学 TONGJI UNIVERSITY

Python 实现：

```
# 定义规则和已知事实
def rules_and_facts():
    # 规则 1： 交通信号规则
    def rule1():
        return [
            "TrafficSignal(Red) -> Stop(Vehicle)",
            "TrafficSignal(Green) ∧ ¬PedestrianNear(Pedestrian) -> Proceed(Vehicle)",
            "TrafficSignal(Yellow) ∧ Near(Vehicle, Intersection) -> SlowDown(Vehicle)",
            "CrossRoadSignal(Green) -> Proceed(Vehicle)"
        ]
    # 规则 2： 行人规则
    def rule2():
        return [
            "PedestrianNear(Pedestrian) -> Stop(Vehicle)",
            "PedestrianFar(Pedestrian) -> Proceed(Vehicle)"
        ]
    # 规则 3： 前方车辆规则
    def rule3():
        return [
            "VehicleChangingLane(FrontVehicle) ∧ Near(Vehicle, FrontVehicle) -> SlowDown(Vehicle)",
            "VehicleSpeed(FrontVehicle, Fast) ∧ Near(Vehicle, FrontVehicle) -> SlowDown(Vehicle)"
        ]
    # 规则 4： 紧急车辆规则
    def rule4():
        return [
            "EmergencyVehicle(EV) ∧ SirenActive(EV) -> Yield(EV)"
        ]
    # 规则 5： 道路状况规则
    def rule5():
        return [
            "RoadCondition(Pothole) ∧ RoadSpeedLimit(Road, 30) -> SlowDown(Vehicle)"
        ]
    # 规则 6： 车辆速度规则
    def rule6():
        return [
            "VehicleSpeed(Vehicle, Speed) ∧ Speed > SpeedLimit(Road, S) -> SlowDown(Vehicle)"
        ]
    return rule1(), rule2(), rule3(), rule4(), rule5(), rule6()

# 归结推理过程
def reasoning():
    rule1, rule2, rule3, rule4, rule5, rule6 = rules_and_facts()
    # 假设已知事实
    facts = {
        “TrafficSignal(Red)”: True,
        “PedestrianNear(Person1)”: True,
        “VehicleChangLane(FrontVehicle)”: True,
        “VehicleSpeed(FrontVehicle, Fast)”: True,
        “EmergencyVehicle(EV)”: True,
        “SirenActive(EV)”: True,
        “RoadCondition(Pothole)”: True,
        “RoadSpeedLimit(Road, 30)”: True
    }
    # 归结步骤 1： 处理交通信号
    if facts[“TrafficSignal(Red)”]:
        print(“ 步骤 1： 交通信号是红灯，车辆应停车。” )
        decision = “Stop(Vehicle)”
    # 归结步骤 2： 行人规则
    if facts["PedestrianNear(Person1)"]:
        print(" 步骤 2： 行人在斑马线上，车辆应停车。 ")
        decision = "Stop(Vehicle)"
    # 归结步骤 3： 前方车辆规则
    if facts["VehicleChangLane(FrontVehicle)"] and facts["VehicleSpeed(FrontVehicle, Fast)"]:
        print(" 步骤 3： 前方车辆变道且速度快，车辆应减速。 ")
        if decision != "Stop(Vehicle)":
            decision = "SlowDown(Vehicle)"
    # 归结步骤 4： 紧急情况
    if facts["EmergencyVehicle(EV)"] and facts["SirenActive(EV)"]:
        print(" 步骤 4： 紧急车辆正在接近，需让行。 ")
        if decision == "Stop(Vehicle)":
            print(" 车辆已停车，等待急救车通过后继续行驶。 ")
        else:
            decision = "Yield(EV)"
    # 归结步骤 5： 路况信息
    if facts["RoadCondition(Pothole)"] and facts["RoadSpeedLimit(Road, 30)"]:
        print(" 步骤 5： 前方有坑洼路段，车辆应减速。 ")
        if decision != "Stop(Vehicle)":
            decision = "SlowDown(Vehicle)"
        print(f" 最终决策： {decision}")
reasoning()
```