回顾

- **对抗搜索和博弈**
  - 博弈论
  - 博弈中的优化决策
  - 蒙特卡罗树搜索
- **约束满足问题**
  - 定义约束满足问题
  - CSP形式体系的变体
  - CSP的回溯搜索
  - 问题的结构与问题的分解

提纲

南京大学 NANJING UNIVERSITY

- **逻辑智能体**
  - 基于知识的智能体
  - wumpus世界与吃豆人世界
  - 逻辑
  - 命题逻辑
- **一阶逻辑**
  - 一阶逻辑的语法和语义
  - 使用一阶逻辑
  - 一阶逻辑中的知识工程

提纲

南京大学 NANJING UNIVERSITY

- **逻辑智能体**
  - 基于知识的智能体
  - wumpus世界与吃豆人世界
  - 逻辑
  - 命题逻辑
- **一阶逻辑**
  - 一阶逻辑的语法和语义
  - 使用一阶逻辑
  - 一阶逻辑中的知识工程

基于知识的智能体

知识库（knowledge base，KB）：一个语句集。基于知识的智能体的核心部件。这些语句用**知识表示语言（knowledge representation language）**表达，代表了关于世界的某种断言。如果一条语句是直接给出的，而不是从其他语句推导而来的，我们就称它为**公理（axiom）**。

告知（Tell）：向知识库添加新语句。

询问（Ask）：从知识库查询已知语句。

推断（Inference）：从原有语句中推导出新语句。推断必须符合以下要求：当向知识库询问（Ask）时，答案应当遵循先前已经告知（Tell）知识库的内容而生成。

基于知识的智能体

```
function KB-AGENT( percept) returns 一个action
  persistent: KB，一个知识库
              t，一个计数器，初始为0，表示时间
  TELL(KB, MAKE-PERCEPT-SENTENCE(percept, t))
  action ←ASK(KB, MAKE-ACTION-QUERY(t))
  TELL(KB, MAKE-ACTION-SENTENCE(action, t))
  t ← t + 1
  return action
```

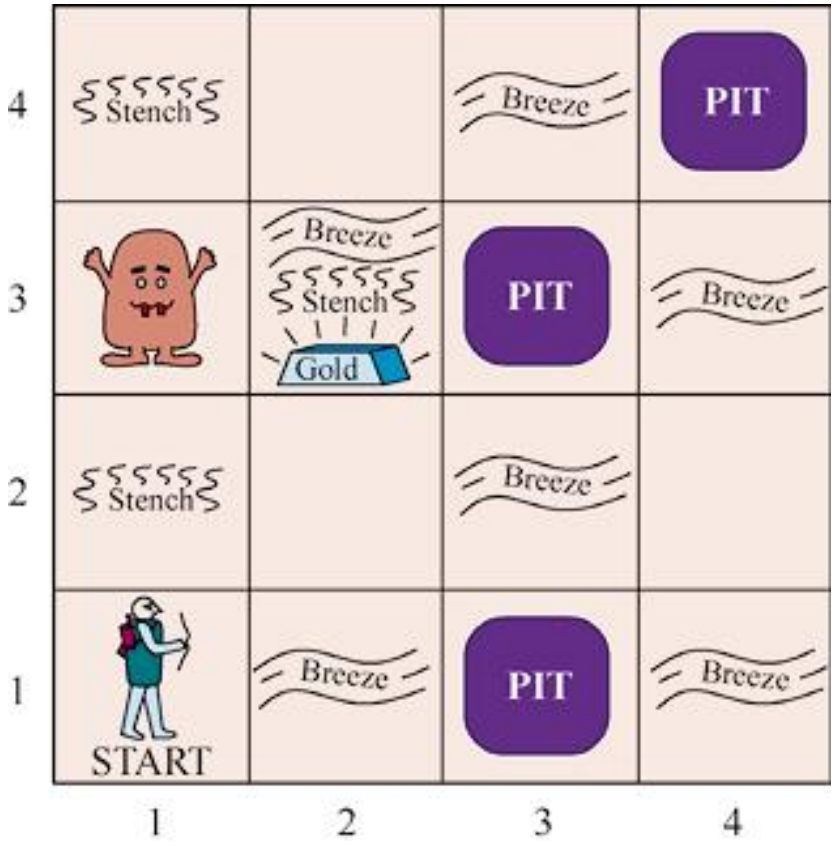
一个通用的基于知识的智能体。给定一个感知，智能体将这一感知添加进知识库，向知识库询问最优动作，并告知知识库它已经采取了这一动作。

wumpus世界

性能度量：金块 +1000, 死亡 -1000
每采取一个动作 -1, 用尽箭支 -10

环境：一个4×4的房间网格，网格四周环绕着围墙。

执行器：左转Left turn, 右转Right turn，向前Forward, 抓取Grab, 射击Shoot，攀爬Climb



一个典型的wumpus世界。智能体位于左下角，面朝东（向右）。

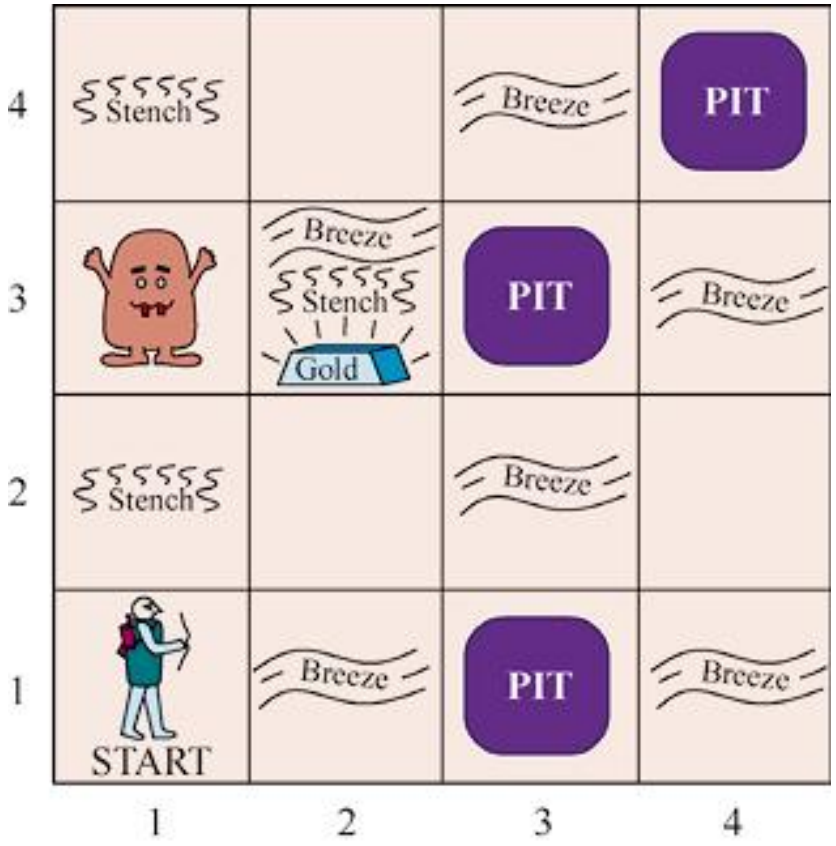
wumpus世界

性能度量：金块 +1000, 死亡 -1000
每采取一个动作 -1, 用尽箭支 -10

环境：一个4×4的房间网格，网格四周环绕着围墙。

执行器：左转Left turn, 右转Right turn，向前Forward, 抓取Grab, 射击Shoot，攀爬Climb

传感器：微风Breeze, 闪光Glitter, 臭味Stench，碰撞Bump，惨叫Scream



一个典型的wumpus世界。智能体位于左下角，面朝东（向右）。

wumpus世界

南京大学 NANJING UNIVERSITY

- **可观测的??** 否，仅是部分可观测的
- **确定性的??** 是，结果是完全可确定的
- **静态的??** 是，wumpus 和无底洞是不会移动的
- **序贯的??** 是
- **离散的??** 是
- **单智能体的??** 是，wumpus 本质上是环境的一部分

wumpus世界

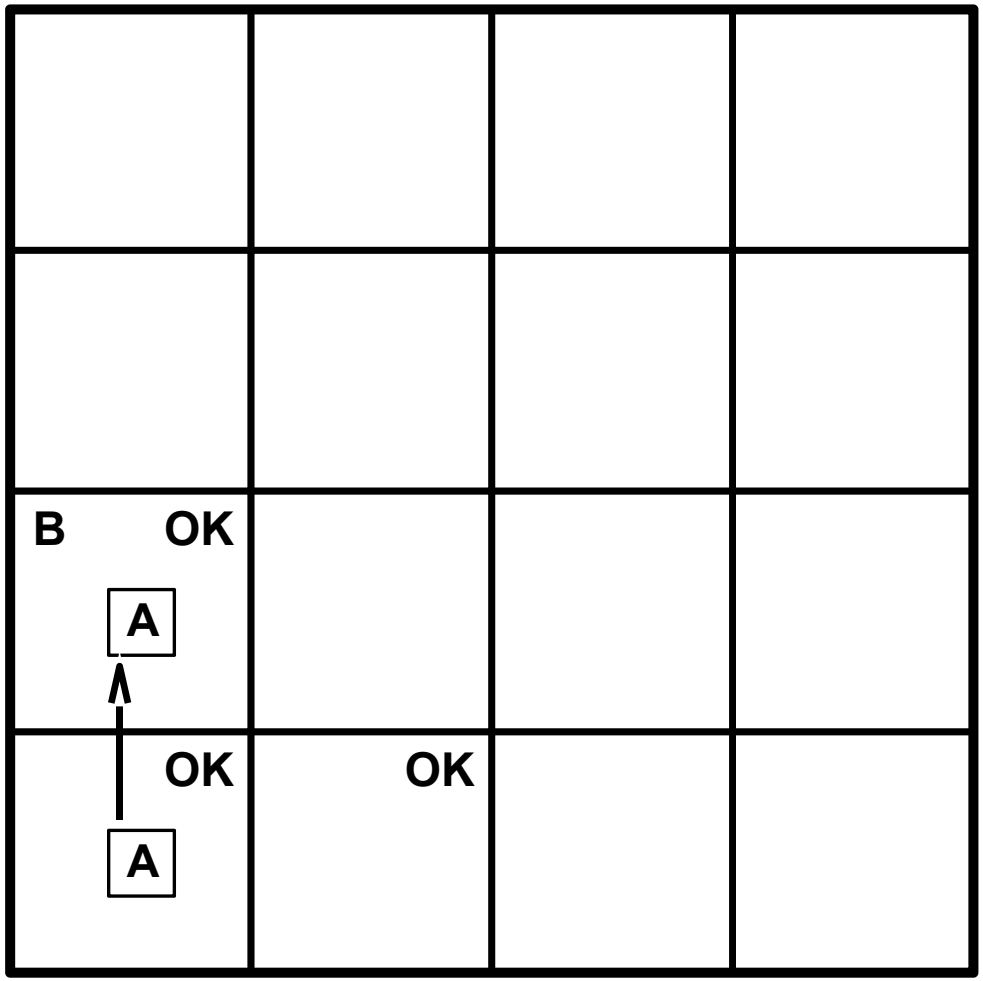
南京大学 NANJING UNIVERSITY

探索wumpus世界

OK			
OK A	OK		

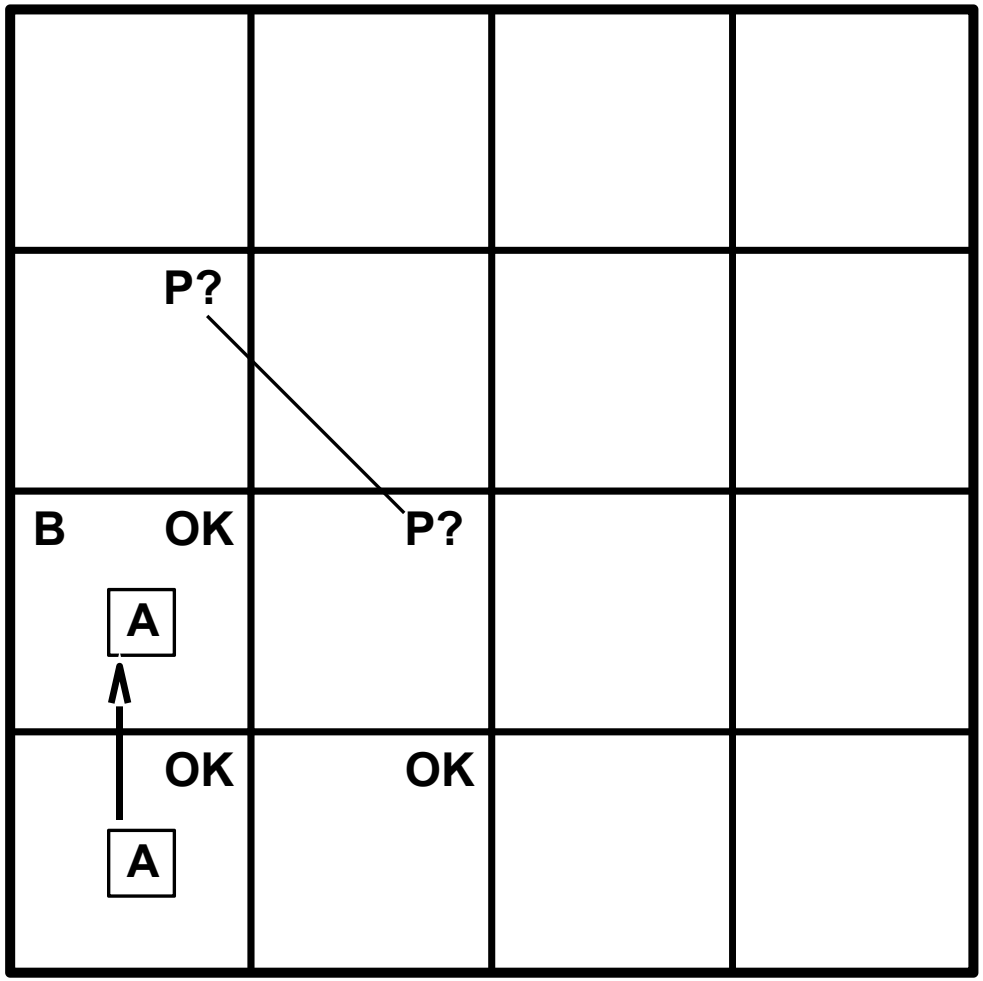
wumpus世界

探索wumpus世界



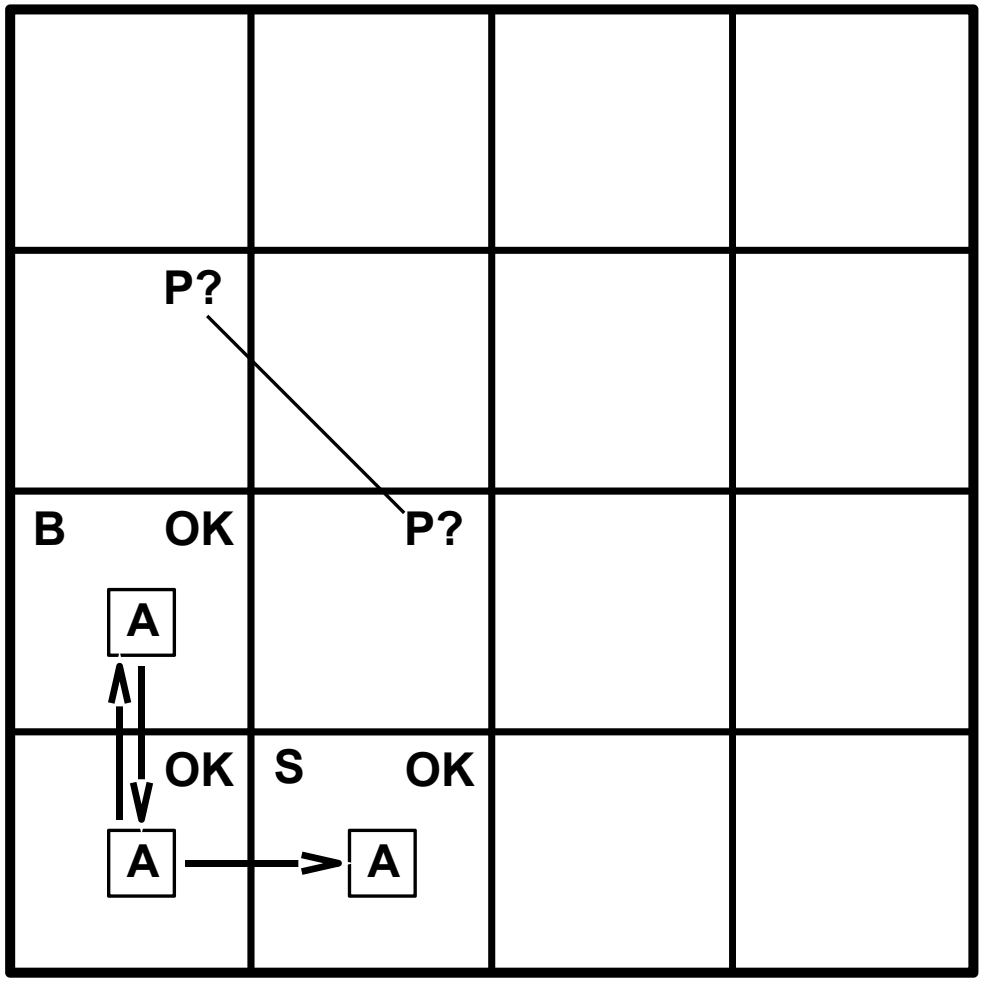
wumpus世界

探索wumpus世界



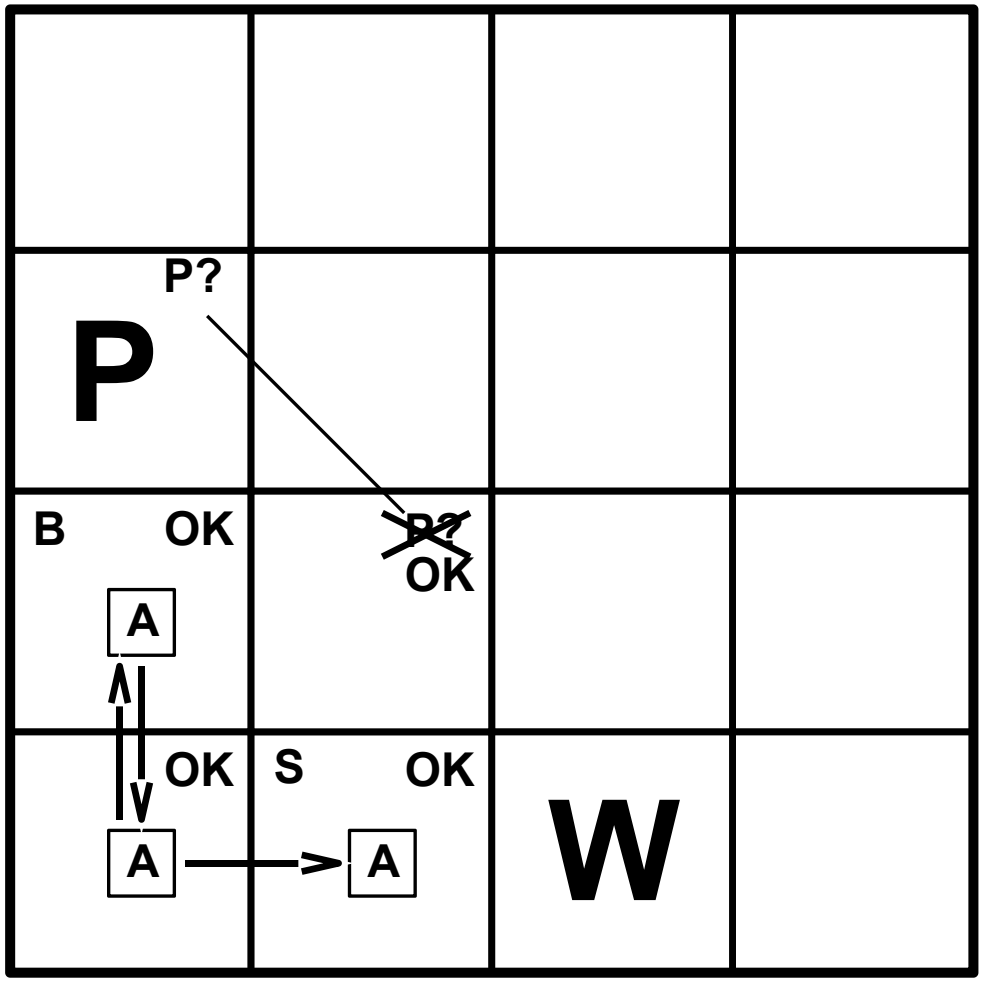
wumpus世界

探索wumpus世界



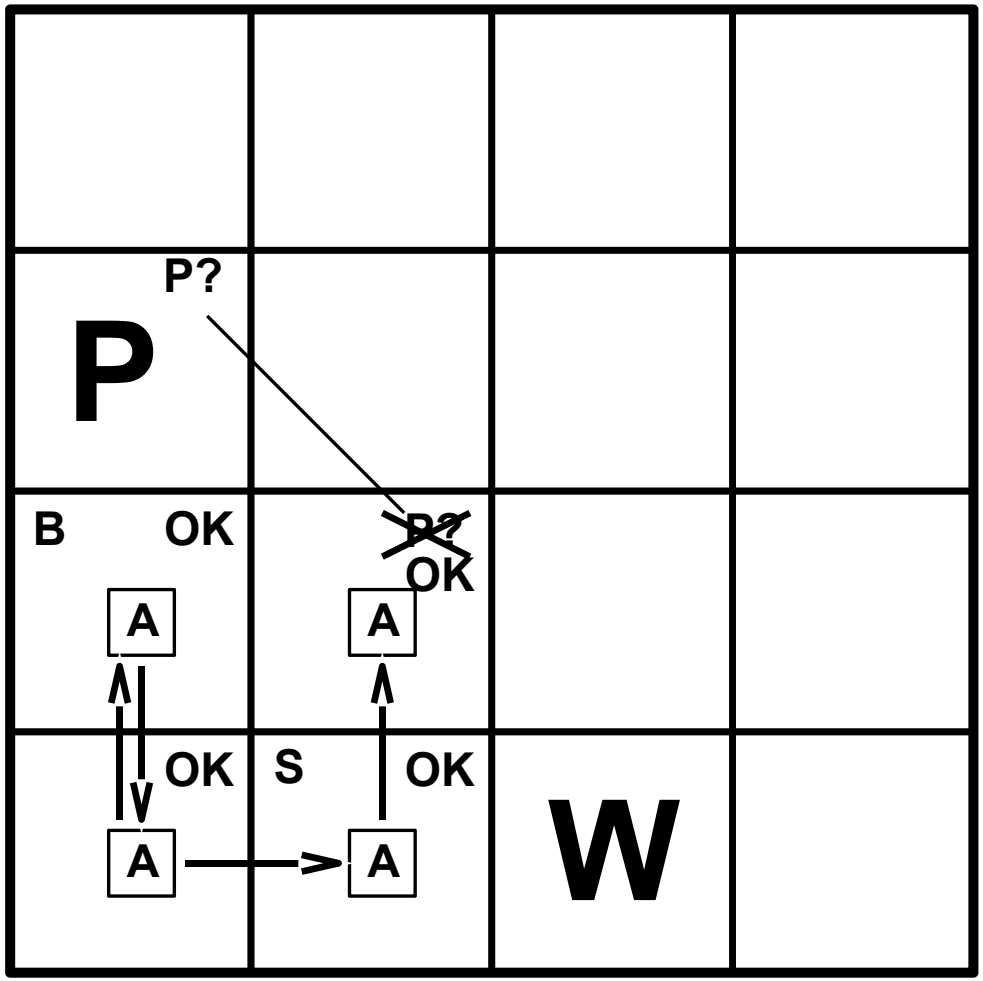
wumpus世界

探索wumpus世界



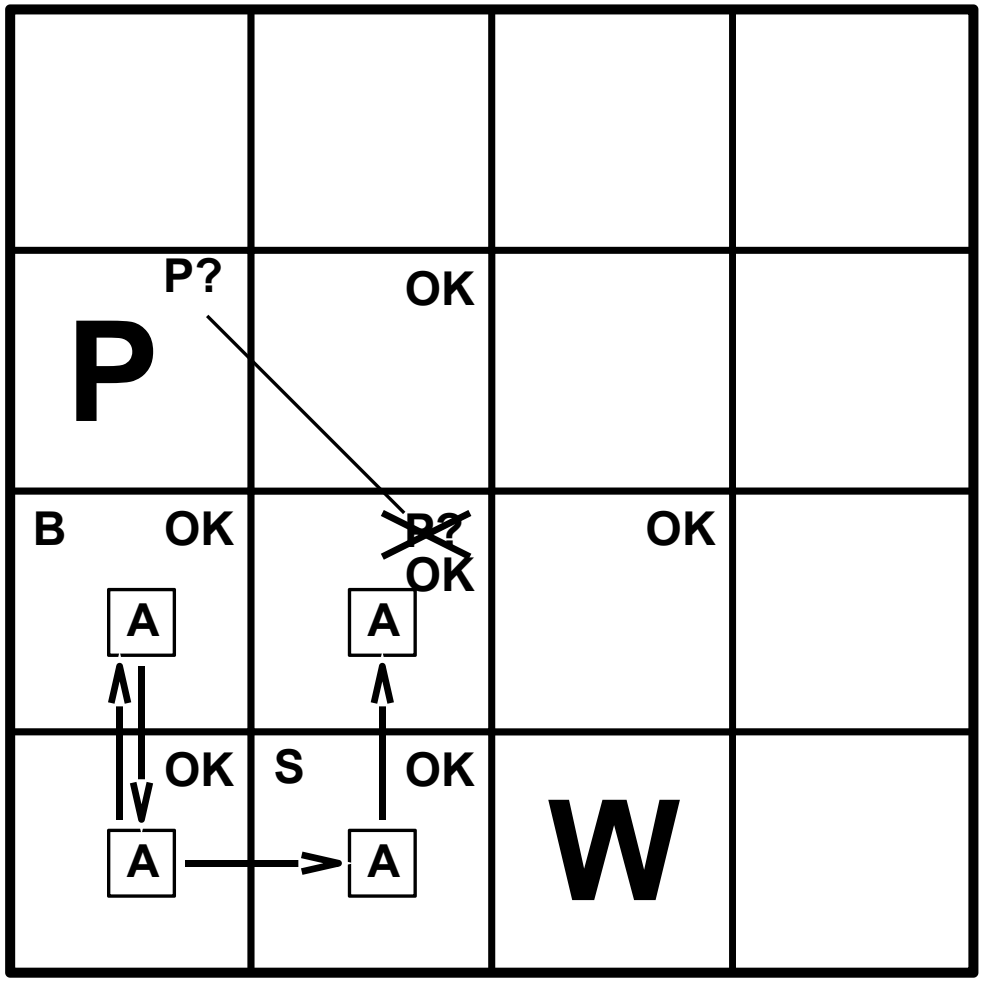
wumpus世界

探索wumpus世界



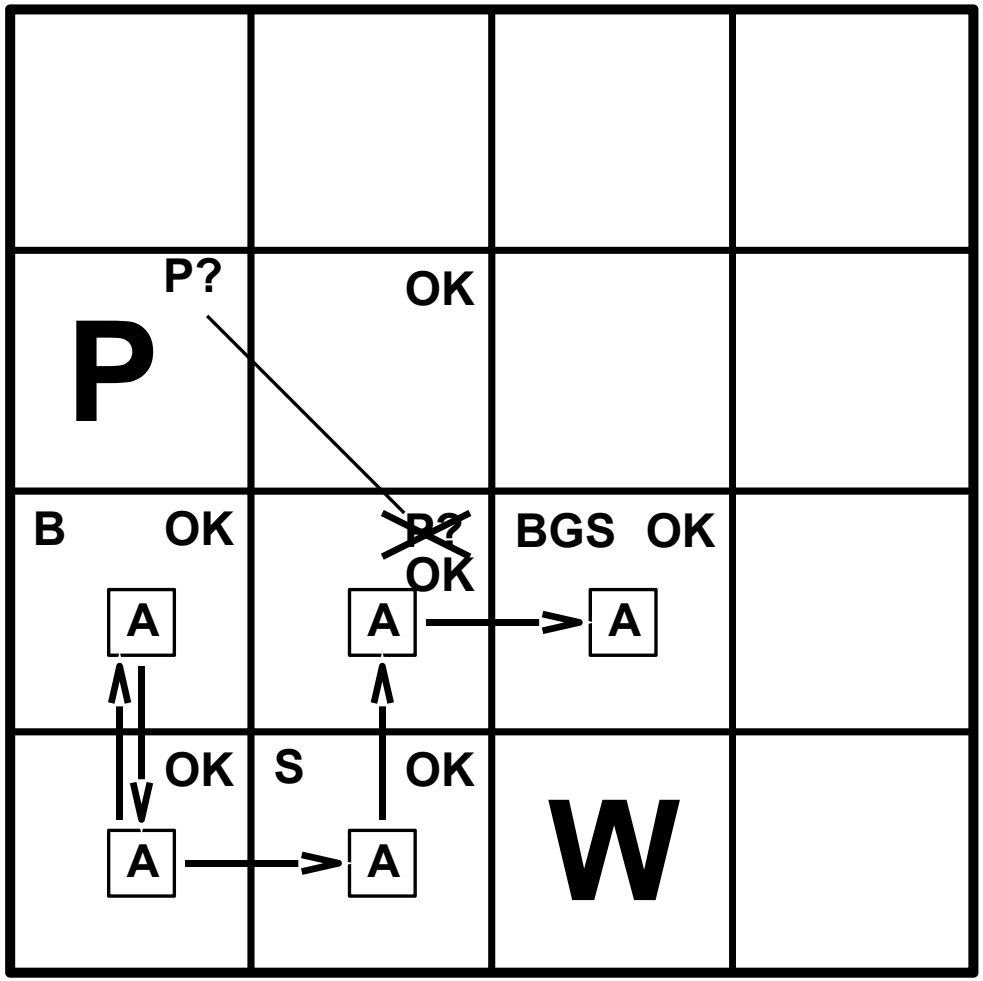
wumpus世界

探索wumpus世界



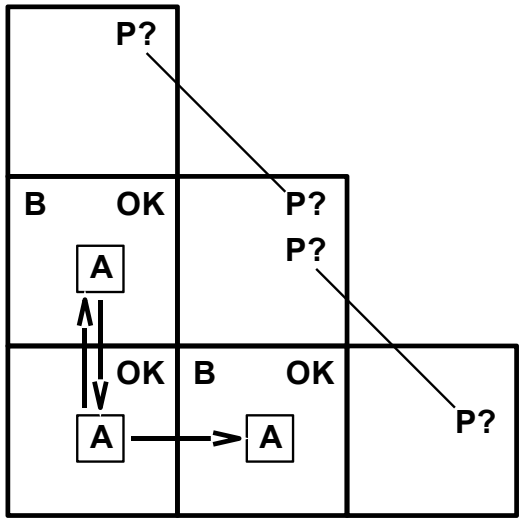
wumpus世界

探索wumpus世界



wumpus世界

探索wumpus世界：其他的可能情况

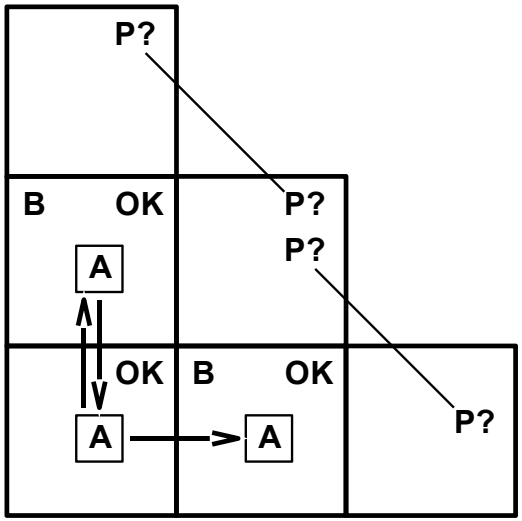


在方格(1,2) 和 (2,1)中都存在微风 ⇒ 没有绝对安全的动作

假设除了 (1,1) 外每个方格存在无底洞的概率是 0.2，(2,2) 存在无底洞的概率为 0.86, 而 (1,3) 或 (3,1) 的为 0.31。

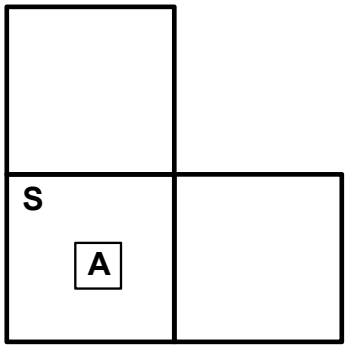
wumpus世界

探索wumpus世界：其他的可能情况



在方格(1,2) 和 (2,1)中都存在微风 ⇒ 没有绝对安全的动作

假设除了 (1,1) 外每个方格存在无底洞的概率是 0.2，(2,2) 存在无底洞的概率为 0.86, 而 (1,3) 或 (3,1) 的为 0.31。



在方格(1,1)中存在臭味 ⇒ 无法移动

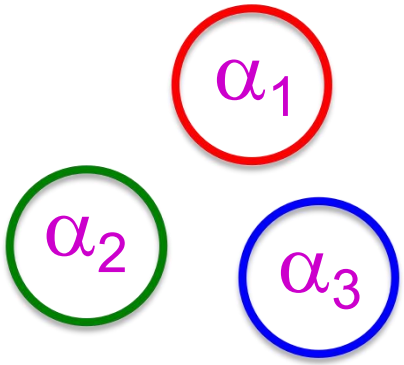
向前发射一支箭

wumpus 在前方方格 ⇒ wumpus 死亡 ⇒ 安全

wumpus 不在前方方格 ⇒ 安全

逻辑的一般概念

- **逻辑（logic）**：用于表示信息的形式化语言以便能够对世界进行推理。
- **语法（syntax）**：规定了所有的合规语句。
- **语义（semantics）**：定义每条语句在每个可能世界中的真值，即，用于判定特定模型中语句真值的规则。



语法域

语义域

逻辑的一般概念

南京大学 NANJING UNIVERSITY

- **逻辑（logic）**：用于表示信息的形式化语言以便能够对世界进行推理。
- **语法（syntax）**：规定了所有的合规语句。
- **语义（semantics）**：定义每条语句在每个可能世界中的真值，即，用于判定特定模型中语句真值的规则。

例如，在算术中

$x + 2 \geq y$ 是合规的语句; $x2 + y >$ 不是

$x + 2 \geq y$ 在一个 $x = 7,\ y = 1$ 的世界为真

$x + 2 \geq y$ 在一个 $x = 0,\ y = 6$ 的世界为假

逻辑的一般概念

- 如果语句 α 在模型 m 中为真，我们说 m 满足 α，有时也可以说 m 是 α 的一个模型。我们使用记号 $M(\alpha)$ 来代表 α 的所有模型的集合。

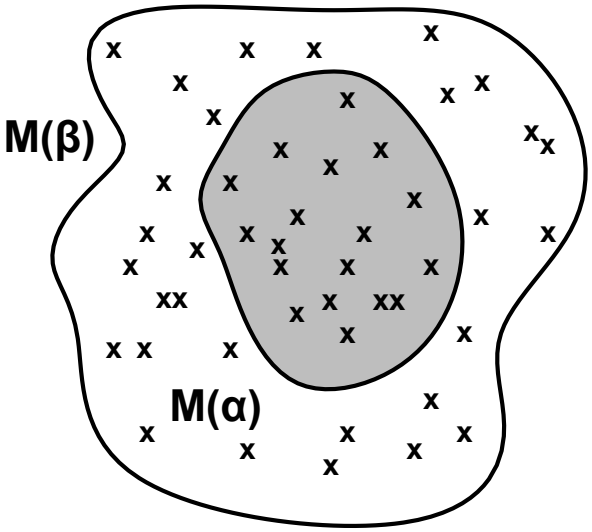
- **蕴含（entailment）** 即一个语句逻辑上引发另一个语句，其形式化定义为当且仅当在 α 为真的每个模型中 β 也为真，记为：

$$\alpha \models \beta$$

例如，$x + y = 4$ 蕴含 $4 = x + y$

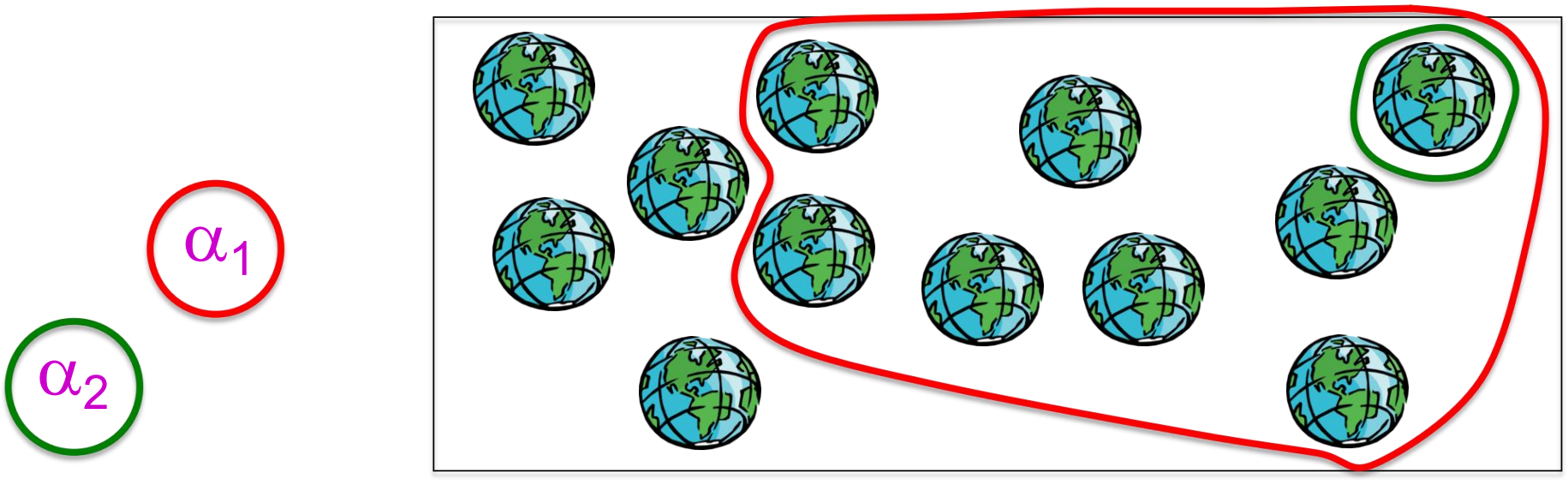
- 利用模型的记号，蕴含的定义也可以写作：

$$\alpha \models \beta \text{ 当且仅当 } M(\alpha) \subseteq M(\beta)$$



逻辑的一般概念

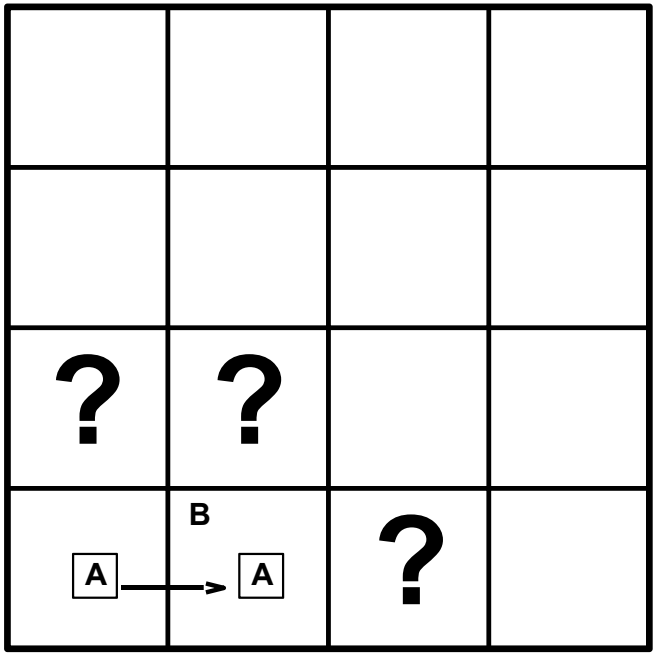
- 例如, $\alpha_2 \models \alpha_1$
- 这里 α_2 为 $\neg Q \wedge R \wedge S \wedge W$ ， α_1 为 $\neg Q$



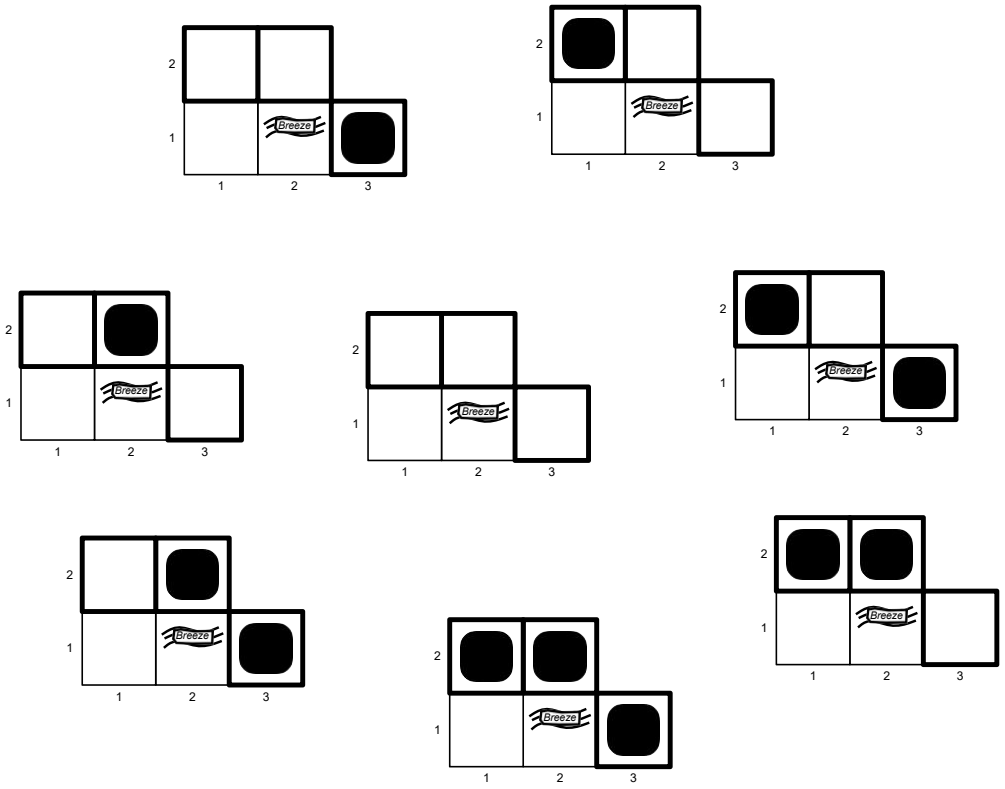
在wumpus世界中的蕴含

智能体在 [1, 1]中什么都没有探测到，在[2, 1]中探测到微风。

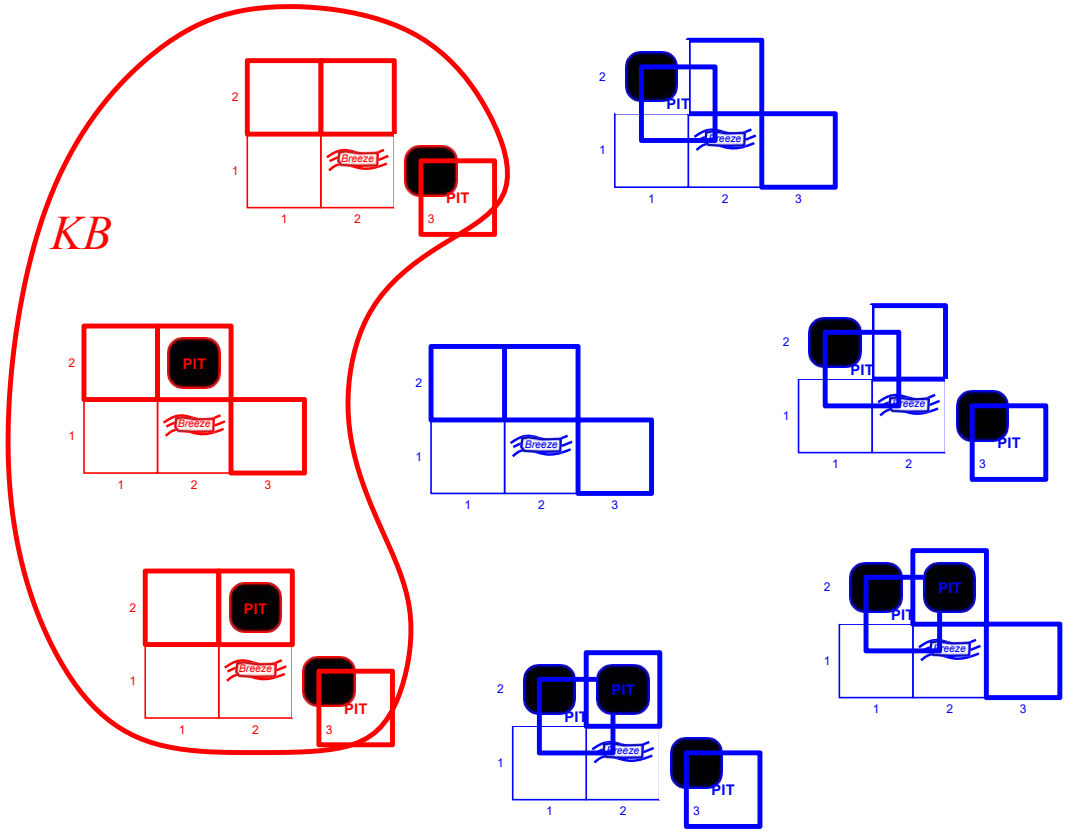
智能体所感兴趣的是相邻的方格[1, 2]、[2, 2]和[3, 1]是否有无底洞。



在wumpus世界中的蕴含

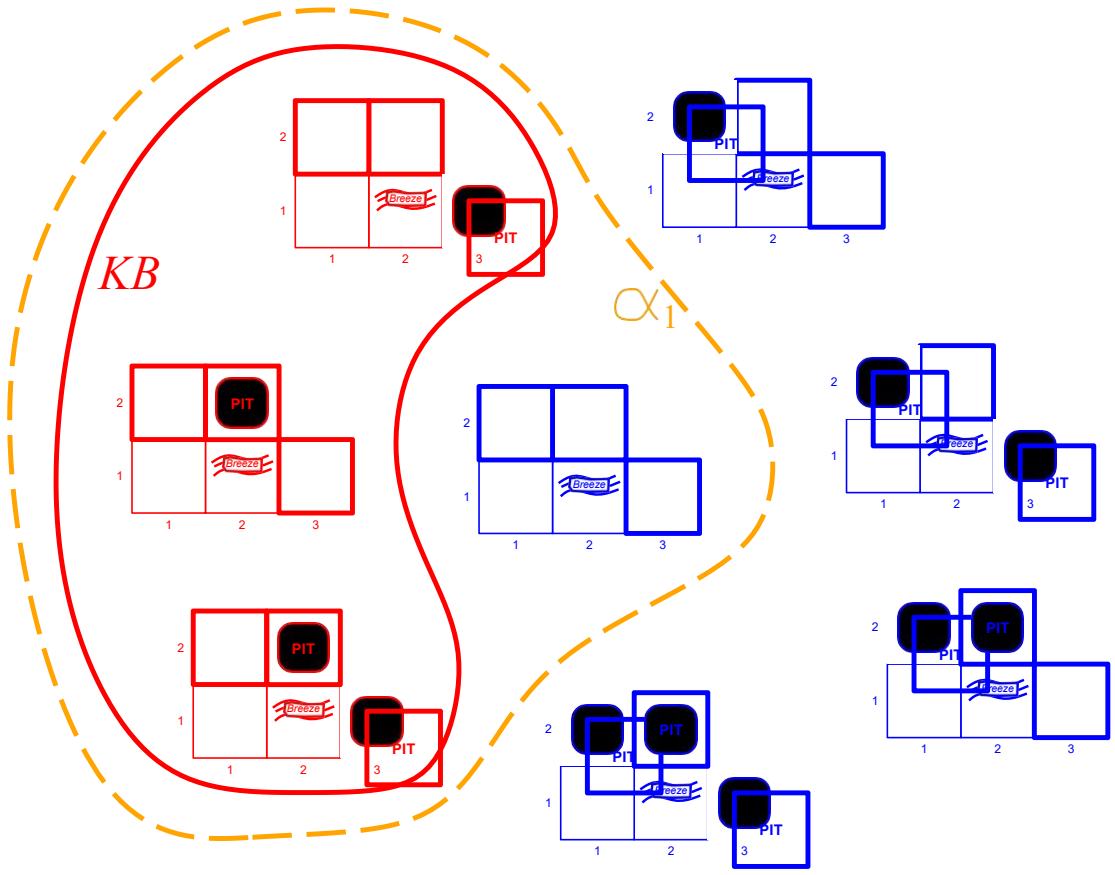


在wumpus世界中的蕴含



KB = wumpus 世界的规则 + 观测

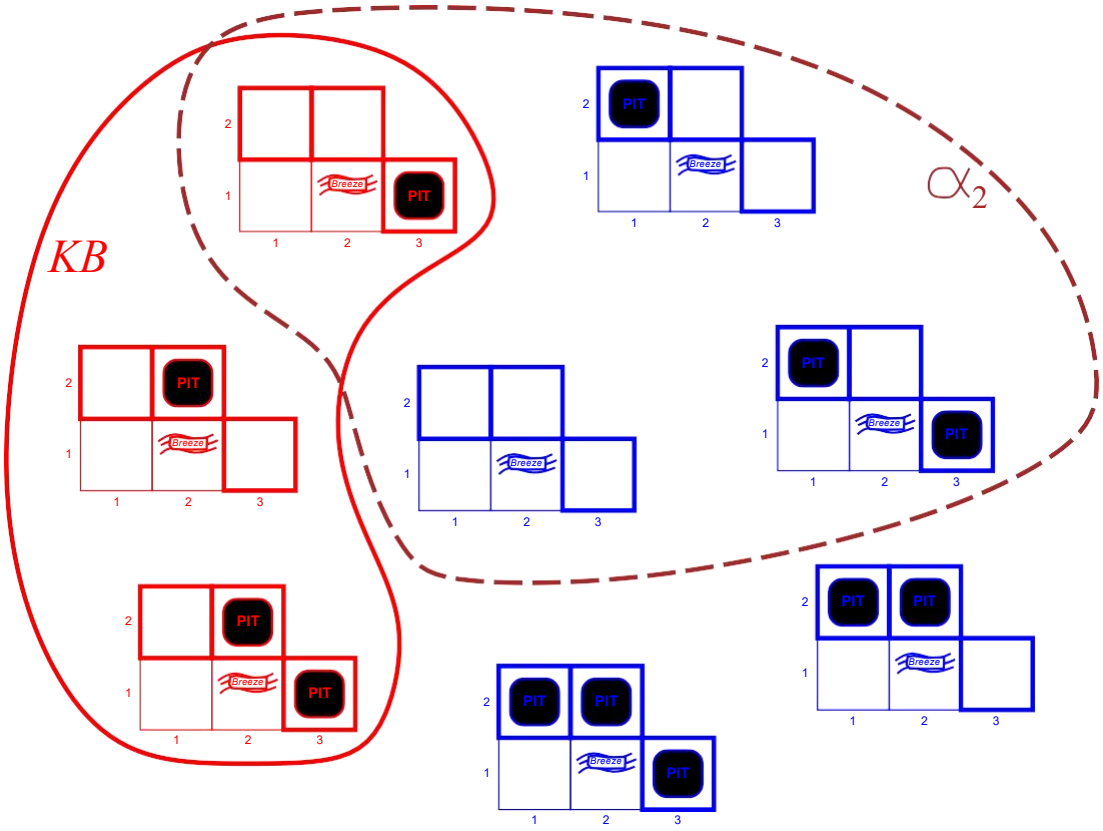
在wumpus世界中的蕴含



KB = wumpus 世界的规则 + 观测

a_1 = “[1,2] 中没有无底洞”, $KB \models a_1$, 可通过**模型检验**证明

在wumpus世界中的蕴含



KB = wumpus 世界的规则 + 观测

a_2 = “[2,2]中没有无底洞”, KB 不蕴含 a_2

逻辑推断

前述的例子不仅阐明了什么是蕴含，还展示了如何用蕴含的定义来推导出结论，即进行**逻辑推断**。如果将 KB 的所有推论的集合比作干草堆而将 α 比做一根针，那么**蕴含正如草堆中的针一样，而推断就像找到这根针的过程。**如果一个推断算法 i 可以从 KB 中推导出 α，则记为

$$KB \vdash_i \alpha$$

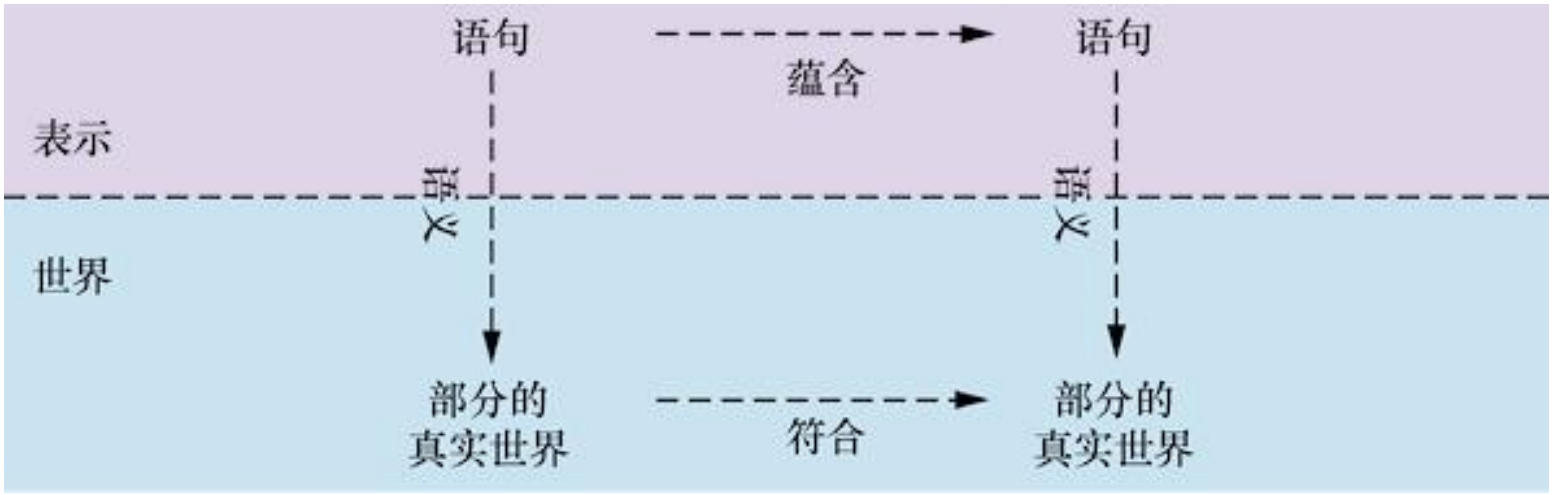
可靠性：推断算法 i 是可靠的，如果

对于任何 $KB \mid\!-_i \alpha$, 那么 $KB \models \alpha$ 也为真

完备性：推断算法 i 是完备的，如果

对于任何 $KB \models \alpha$, 那么 $KB \mid\!-_i \alpha$ 也为真

逻辑推断



语句是智能体的物理结构，而推断是从旧结构或者已有结构构建新结构的过程。逻辑推断应当确保新结构所表示的部分世界确实能够从旧结构所表示的部分世界推得。

命题逻辑：语法

命题逻辑的**语法**定义合法的语句。**原子语句（atomic sentence）**由单个**命题符号（proposition symbol）**构成。使用括号和被称作**逻辑联结词（logicalconnective）**的运算符可以将简单语句构造成**复合语句（complex sentence）**。常用的联结词有5个：

- ¬（非）。类似$\neg W_{1,3}$这样的语句称为$W1,3$的**否定**。一个**文字**要么是原子语句，即**正文字**，要么是原子语句的否定，即**负文字**。

- ∧（与）。主要联结词是∧的语句称为**合取式**，例如$W_{1,3} \wedge P_{3,1}$，其各部分称为**合取子句**。（∧看起来像是“And”中的“A”。）

- ∨（或）。主要联结词是∨的语句称为**析取式**，例如$(W_{1,3} \wedge P_{3,1}) \vee W_{2,2}$，其各部分为**析取子句**，本例中分别为$(W_{1,3} \wedge P_{3,1})$和$W2,2$。

- ⇒（蕴涵）。如$(W_{1,3} \wedge P_{3,1}) \Rightarrow W_{2,2}$这样的语句称为**蕴涵式**（implication）或条件式，其**前提**（premise）或**前件（antecedent）**是$(W_{1,3} \wedge P_{3,1})$，其**结论**（conclusion）或**后件**（consequent）是$\neg W_{2,2}$。蕴涵式也被称为**规则**（rule）或**if-then**声明。有时，蕴涵符号在一些书籍中写作⊃或→。

- ⇔（当且仅当）。语句$W_{1,3} \Leftrightarrow \neg W_{2,2}$是**双向蕴涵式**（biconditional）。

命题逻辑：语法

命题逻辑中语句的BNF文法以及从高到低排列的运算符优先级：

$$
\begin{aligned}
\text{语句} &\rightarrow \text{原子语句} \mid \text{复合语句} \\
\text{原子语句} &\rightarrow \mathit{True} \mid \mathit{False} \mid P \mid Q \mid R \mid \cdots \\
\text{复合语句} &\rightarrow (\text{语句}) \\
&\quad \mid \neg \text{语句} \\
&\quad \mid \text{语句} \wedge \text{语句} \\
&\quad \mid \text{语句} \vee \text{语句} \\
&\quad \mid \text{语句} \Rightarrow \text{语句} \\
&\quad \mid \text{语句} \Leftrightarrow \text{语句}
\end{aligned}
$$

运算符优先级 ： $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$

命题逻辑：语义

南京大学 NANJING UNIVERSITY

语义定义了用于判定特定模型中语句真值的规则。命题逻辑中，模型就是对每个命题符号设定真值，即真（true）或假（false）。命题逻辑的语义必须指定在给定模型下如何计算任一语句的真值。**这是以递归的方式实现的。**所有语句都是由原子语句和5个联结词构建的。

- $\neg P$为真，当且仅当在m中P为假。
- $P \wedge Q$为真，当且仅当在m中P和Q都为真。
- $P \vee Q$为真，当且仅当在m中P或Q中至少一个为真。
- $P \Rightarrow Q$为真，除非在m中P为真而Q为假。
- $P \Leftrightarrow Q$为真，当且仅当在m中P和Q都为真或都为假。

命题逻辑：语义

南京大学 NANJING UNIVERSITY

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
false	false	true	false	false	true	true
false	true	true	false	true	true	false
true	false	false	false	true	false	false
true	true	false	true	true	true	true

5个逻辑联结词的真值表。若要使用真值表计算在P为真、Q为假时P V Q的值，首先在左边找到P为true而Q为false的行（第3行），然后找到该行位于P V Q列处的值，得到结果true。

可以使用简单的递归过程来计算任一语句的真值, 例如,

$\neg P_{1,2} \wedge (P_{2,2} \vee P_{3,1}) = true \wedge (false \vee true) = true \wedge true = true$

命题逻辑：wumpus世界

我们已经定义了命题逻辑的语义，现在可以为wumpus世界构建一个知识库了。对于每个位置 [x, y]，需要用到下列符号：

当[x, y]有无底洞，*Px*, *y*为真。

当wumpus在[x, y]，不论其死活*Wx*, *y*都为真。

当[x, y]有微风，*Bx*, *y*为真。

当[x, y]处有臭味，*Sx*, *y*为真。

当智能体位于位置[x, y]，*Lx*, *y*为真。

命题逻辑：wumpus世界

方格[1, 2]中没有无底洞的推导过程，我们用R_i代表每个语句，以便推导。

- 方格[1, 1]中没有无底洞：$R_1 : \neg P_{1,1}$

- 一个方格有微风，当且仅当其相邻方格中有无底洞。必须对每个方格都进行这样的表示，在此我们只写出相关方格的表示：

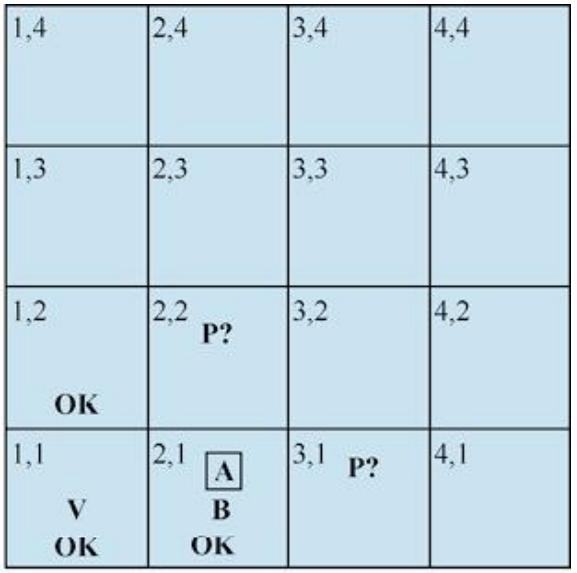
$$R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$$

$$R_3 : B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$$

- 上述语句在所有wumpus世界中都为真。我们现在为智能体在这个特定世界中已访问过的前两个方格引入微风感知，以形成右图所示的情形：

$$R_4 : \neg B_{1,1}$$

$$R_5 : B_{2,1}$$



（b）

命题逻辑：通过真值表推断

南京大学 NANJING UNIVERSITY

$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	R_1	R_2	R_3	R_4	R_5	KB
false	false	false	false	false	false	false	true	true	true	true	false	false
false	false	false	false	false	false	true	true	true	false	true	false	false
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
false	true	false	false	false	false	false	true	true	false	true	true	false
false	true	false	false	false	falsetrue	true	true	true	true	true	true	true
false	true	false	false	false	true	falsetrue	true	true	true	true	true	true
false	true	false	false	false			true	true	true	true	true	true
false	true	false	false	true	false	false	true	false	false	true	true	false
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
true	true	true	true	true	true	true	false	true	true	false	true	false

根据wumpus世界的知识库 *KB* 构建的真值表

命题逻辑：通过枚举推断

南京大学 NANJING UNIVERSITY

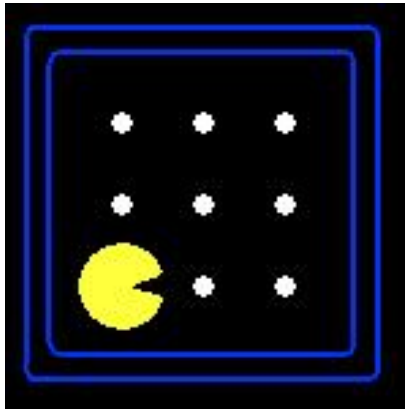
```
function TT-ENTAILS?(KB, α) returns true或false
  inputs: KB，知识库，一个命题逻辑的语句
          α，查询，一个命题逻辑的语句
  symbols ←KB和α中的命题符号列表
  return TT-CHECK-ALL(KB, α, symbols, {})

function TT-CHECK-ALL(KB, α, symbols, model) returns true或false
  if EMPTY?(symbols) then
     if PL-TRUE?(KB, model) then return PL-TRUE?(α, model)
     else return true       //当KB为false时，始终返回true
  else
     P ←FIRST(symbols)
     rest ←REST(symbols)
     return (TT-CHECK-ALL(KB, α, rest, model ∪ {P = true})
             and
             TT-CHECK-ALL(KB, α, rest, model ∪ {P = false}))
```

用于确定命题蕴含的真值表枚举算法（TT代表真值表）

命题逻辑：吃豆人的世界

南京大学 NANJING UNIVERSITY

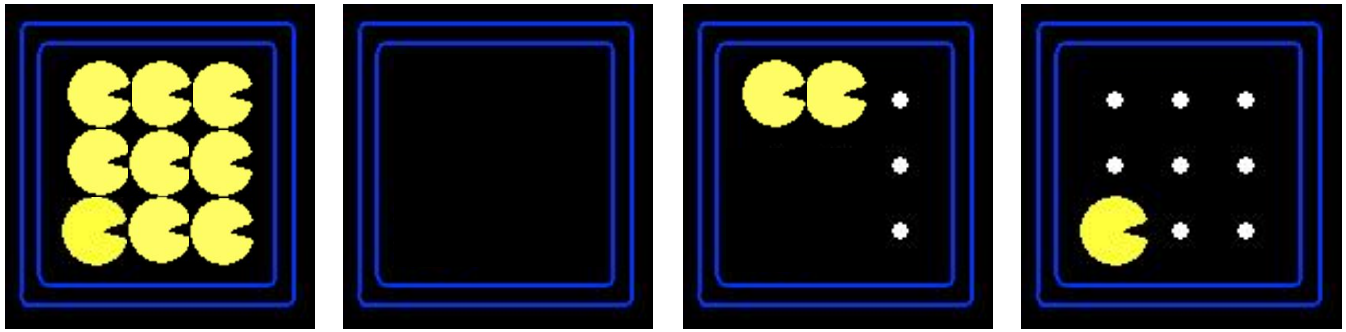
- 吃豆人知道地图，但是只感知到 NSEW 的墙壁
- 形式化：**需要哪些变量?**
  - 墙壁的位置
    - Wall_0,0 位于 [0,0] 有墙壁
    - Wall_0,1 位于 [0,1] 有墙壁，……（需要N个符号）
  - 感知
    - ~~Blocked_W (西边被墙挡住)，……~~
    - Blocked_W_0 (在第0步，西边被墙壁挡住)， ……（需要4T个符号）
  - 动作
    - W_0 (在第0步，吃豆人向西移动), E_0 ……（需要4T个符号）
  - 吃豆人的位置
    - At_0,0_0 (在第0步，吃豆人位于 [0,0]), At_0,1_0 ……（需要NT个符号）



来源：UC Berkeley CS188

命题逻辑：吃豆人的世界

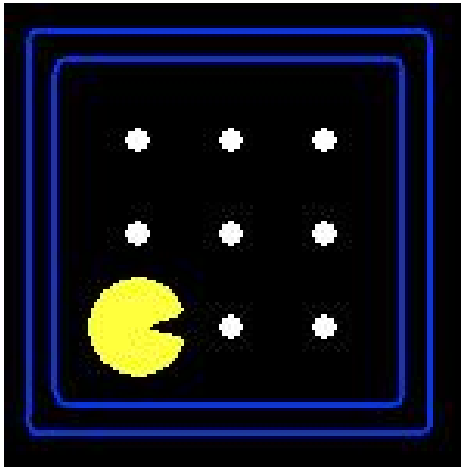
- N 个位置, T 步 => $N + 4T + 4T + NT = O(NT)$ 个变量
- $O(2^{NT})$ 个可能的世界
- N=200, T=400 => ~10^{24000} 个可能的世界
- 每一个世界上都是一段完整的“历史”
  - 但是其中大多数的世界看起来都很奇怪



命题逻辑：吃豆人的世界

南京大学 NANJING UNIVERSITY

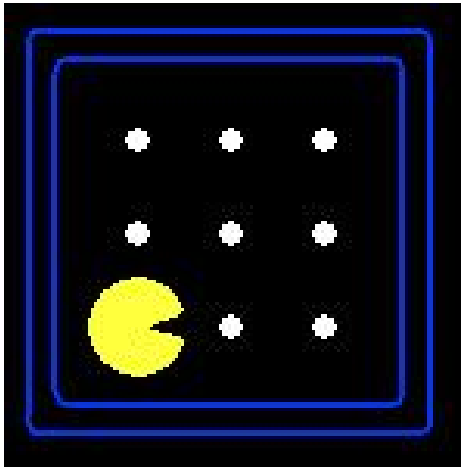
- 吃豆人知道哪里有墙壁：
  - $Wall\_0,0 \wedge Wall\_0,1 \wedge Wall\_0,2 \wedge Wall\_0,3 \wedge Wall\_0,4 \wedge Wall\_1,4 \wedge \ldots$
- 吃豆人知道哪里没有墙壁！
  - $\neg Wall\_1,1 \wedge \neg Wall\_1,2 \wedge \neg Wall\_1,3 \wedge \neg Wall\_2,1 \wedge \neg Wall\_2,2 \wedge \ldots$



命题逻辑：吃豆人的世界

南京大学 NANJING UNIVERSITY

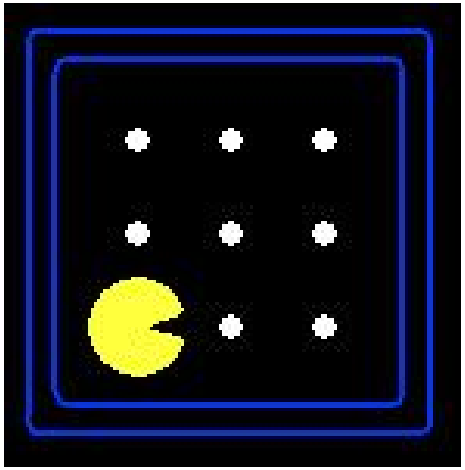
- 吃豆人不知道他自己在哪
- 但是他知道自己在下面的其中某一位置
  - $At\_1,1\_0 \vee At\_1,2\_0 \vee At\_1,3\_0 \vee At\_2,1\_0 \vee \ldots$



命题逻辑：吃豆人的世界

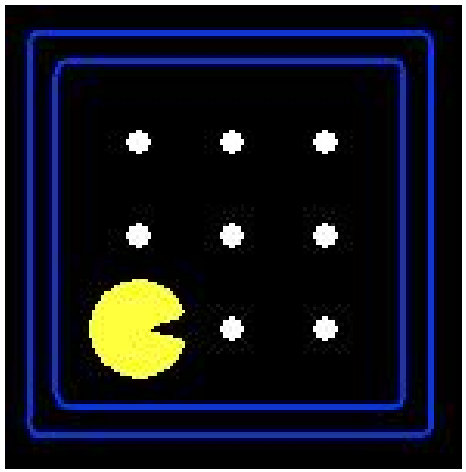
南京大学 NANJING UNIVERSITY

- 对吃豆人的感知如何产生进行描述，例如
- 在第 *t* 步吃豆人感知到西边被墙壁挡住
  当且仅当 他位于 *x,y* 并且在 *x-1,y* 存在墙壁
  - Blocked_W_0 $\Leftrightarrow$ ((At_1,1_0 $\wedge$ Wall_0,1) v
    (At_1,2_0 $\wedge$ Wall_0,2) v
    (At_1,3_0 $\wedge$ Wall_0,3) v ….)



命题逻辑：吃豆人的世界

- 每个状态变量在每步如何赋值
  - 这里我们考虑位置变量, 例如, At_3,3_17
- 一个状态变量 X 根据 后继状态公理进行赋值
  - X_t ⇔ [X_t-1 ∧ ¬(some action_t-1 made it false)] v
    [¬X_t-1 ∧ (some action_t-1 made it true)]
- 对于吃豆人的位置变量:
  - At_3,3_17 ⇔ [At_3,3_16 ∧ ¬((¬Wall_3,4 ∧ N_16) v (¬Wall_4,3 ∧ E_16) v …)]
    v [¬At_3,3_16 ∧ ((At_3,2_16 ∧ ¬Wall_3,3 ∧ N_16) v
    (At_2,3_16 ∧ ¬Wall_3,3 ∧ N_16) v …)]



命题定理证明：逻辑等价

南京大学 NANJING UNIVERSITY

逻辑等价（logical equivalence）：如果两个语句在相同的模型集合中都为真，则这两个语句逻辑等价，可以写作：$\alpha \equiv \beta$

等价的另一种定义为任意两条语句是等价的，当且仅当它们互相蕴含：

$\alpha \equiv \beta$ 当且仅当 $\alpha \models \beta$ 且 $\beta \models \alpha$

标准的逻辑等价：

$(\alpha \wedge \beta) \equiv (\beta \wedge \alpha)$ $\wedge$的交换律

$(\alpha \vee \beta) \equiv (\beta \vee \alpha)$ $\vee$的交换律

$((\alpha \wedge \beta) \wedge \gamma) \equiv (\alpha \wedge (\beta \wedge \gamma))$ $\wedge$的结合律

$((\alpha \vee \beta) \vee \gamma) \equiv (\alpha \vee (\beta \vee \gamma))$ $\vee$的结合律

$\neg(\neg\alpha) \equiv \alpha$ 双重否定律

$(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha)$ 假言易位

$(\alpha \Rightarrow \beta) \equiv (\neg\alpha \vee \beta)$ 蕴涵消去

$(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha))$ 等价消去

$\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta)$ 德摩根律

$\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta)$ 德摩根律

$(\alpha \wedge (\beta \vee \gamma)) \equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma))$ $\wedge$对$\vee$的分配律

$(\alpha \vee (\beta \wedge \gamma)) \equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma))$ $\vee$对$\wedge$的分配律

命题定理证明：有效性和可满足性

- **有效性（validity）**：如果一条语句在所有模型中都为真，则这条语句是有效的。

  例如 $True, A \vee \neg A, A \Rightarrow A, (A \wedge (A \Rightarrow B)) \Rightarrow B$

蕴含的定义可以推导出演绎定理:

$KB \models a$ 当且仅当 $(KB \Rightarrow a)$ 是有效的

命题定理证明：有效性和可满足性

南京大学 NANJING UNIVERSITY

- **有效性（validity）**：如果一条语句在所有模型中都为真，则这条语句是有效的。

  例如 $True, A \vee \neg A, A \Rightarrow A, (A \wedge (A \Rightarrow B)) \Rightarrow B$

  蕴含的定义可以推导出演绎定理:

  $KB \models a$ 当且仅当 $(KB \Rightarrow a)$ 是有效的

- **可满足性（satisfiability）**：如果一条语句在某些模型中为真或能够被满足，则这条语句是可满足的。

  如果一条语句没有模型使其为真，则这条语句是不可满足的。
  例如, $A \wedge \neg A$

  可满足可以推导出归谬法:

  $KB \models a$ 当且仅当 $(KB \wedge \neg a)$ 是不可满足的

- **单调性**：它表明蕴含的语句集只能随着信息被加入知识库而增长。

  如果 $KB \models a$，则 $KB \wedge \beta \models a$

命题定理证明：推断与证明

南京大学 NANJING UNIVERSITY

- **肯定前件**（Modus Ponens，mode that affirms的拉丁语），写作

$$\frac{\alpha \Rightarrow \beta, \alpha}{\beta}$$

- **合取消去**（and-elimination），即可以从一个合取式推导出任一合取子句：

$$\frac{\alpha \wedge \beta}{\alpha}$$

- 所有逻辑等价都可以用作推断规则。例如，等价消去可以产生两条推断规则：

$$\frac{\alpha \Leftrightarrow \beta}{(\alpha \Rightarrow \beta)(\beta \Rightarrow \alpha)} \text{ 和 } \frac{(\alpha \Rightarrow \beta)(\beta \Rightarrow \alpha)}{\alpha \Leftrightarrow \beta}$$

命题定理证明：推断与证明

南京大学 NANJING UNIVERSITY

- 推断规则和等价关系应用于wumpus世界。从含有R1到R5的知识库开始，演示如何证明[1, 2]中没有无底洞。首先给出 $R_2 : B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

（1）对R_2使用等价消去，得到

$$R_6 : (B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$$

（2）对R_6使用合取消去，得到

$$R_7 : ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$$

（3）假言易位逻辑等价关系得到

$$R_8 : (\neg B_{1,1} \Rightarrow \neg (P_{1,2} \vee P_{2,1}))$$

（4）对R_8和感知$R4$（即$\neg B_{1,1}$）使用肯定前件，得到

$$R_9 : \neg (P_{1,2} \vee P_{2,1})$$

（5）使用德摩根律，得到结论

$$R_{10} : \neg P_{1,2} \wedge \neg P_{2,1}$$

也就是，[1, 2]和[2, 1]都没有无底洞。

命题定理证明：推断与证明

应用第3章的任意搜索算法都可以找到构成这种证明的一系列步骤。只需要定义如下的证明问题：

- **初始状态（Initial State）**：最初的知识库。

- **动作（Actions）**：动作的集合，包含所有推断规则应用于所有符合上半部分推断规则的语句。

- **结果（Result）或者转移模型**：将推断规则下半部分的语句实例加入知识库。

- **目标（Goal）**：目标是含有我们想要证明的语句的状态。

命题定理证明：通过归结证明

南京大学 NANJING UNIVERSITY

单元归结（unit resolution）规则

$$\frac{\ell_1 \vee \cdots \vee \ell_k, \qquad m}{\ell_1 \vee \cdots \vee \ell_{i-1} \vee \ell_{i+1} \vee \cdots \vee \ell_k}$$

单元归结规则可以推广为**全归结规则**

$$\frac{\ell_1 \vee \cdots \vee \ell_k, \qquad m_1 \vee \cdots \vee m_n}{\ell_1 \vee \cdots \vee \ell_{i-1} \vee \ell_{i+1} \vee \cdots \vee \ell_k \vee m_1 \vee \cdots \vee m_{j-1} \vee m_{j+1} \vee \cdots \vee m_n}$$

归结使用两个子句并产生一个新的子句，该新子句包含除一对互补文字以外的原始子句的所有文字，例如

$$\frac{P_{1,1} \vee P_{3,1}, \qquad \neg P_{1,1} \vee \neg P_{2,2}}{P_{3,1} \vee \neg P_{2,2}}$$

一次只能归结一对互补文字，例如

$$\frac{P \vee \neg Q \vee R, \qquad \neg P \vee Q}{\neg Q \vee Q \vee R}$$

命题定理证明：通过归结证明

形式为子句合取式的语句被称为**合取范式（conjunctive normal form）或CNF。**把语句转换为CNF的过程：

$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

1. 消去 $\Leftrightarrow$, 替换 $\alpha \Leftrightarrow \beta$ 为 $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$.

   $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$

2. 消去$\Rightarrow$,替换$\alpha \Rightarrow \beta$ 为 $\neg\alpha \vee \beta$.

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$

3. 将 $\neg$ 内移， 通过使用德摩根律和双重否定律:

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1})$

4. 应用分配律:

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

命题定理证明：通过归结证明

function PL-RESOLUTION(KB, a) **returns** true或false
  inputs: KB，知识库，一个命题逻辑中的语句
    α，查询，一个命题逻辑中的语句

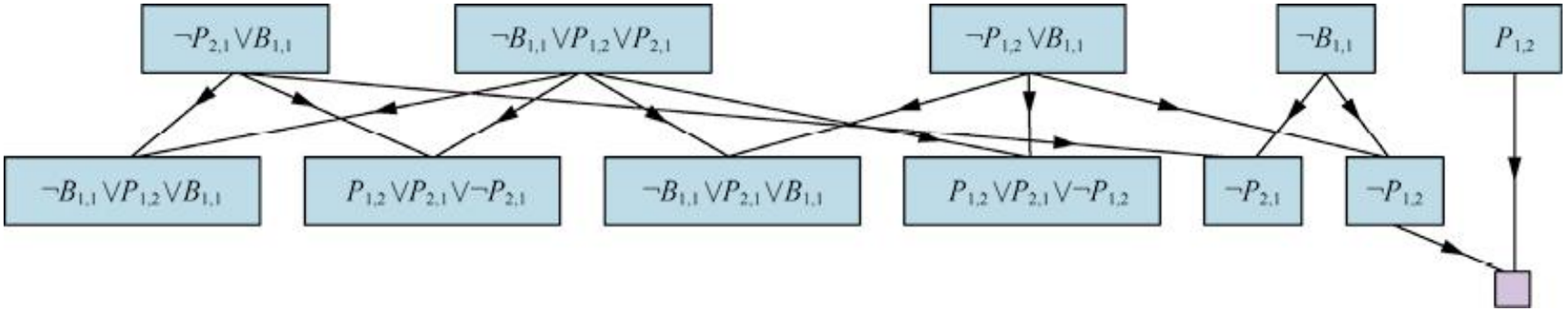
  clauses ← $KB \wedge \neg\alpha$的CNF表示的子句集合
  new ← {}
  while true **do**
    for each 子句对C_i, C_j **in** *clauses* **do**
      resolvents ← PL-RESOLVE(C_i, C_j)
      if *resolvents*包含空子句 **then return** true
      new ← *new* $\cup$ *resolvents*
    if *new* $\subseteq$ *clauses* **then return** false
    clauses ← *clauses* $\cup$ *new*

简单的命题逻辑归结算法。PL-Resolve返回对其两个输入进行归结得到的所有可能子句集合。

命题定理证明：通过归结证明

相关的知识库： $KB = R_2 \wedge R_4 = (B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})) \wedge \neg B_{1,1}$

证明：$\neg\neg P_{1,2}$



对wumpus世界的一个简单推断部分运用PL-Resolution来证明查询。

命题定理证明：霍恩子句

- **确定子句（definite clause）**，它是文字的析取式，其中只有一个为正文字。
- **霍恩子句（Horn clause）**，它是文字的析取式，其中最多只有一个为正文字。因此所有的确定子句都是霍恩子句，没有正文字的子句也是霍恩子句，也被称为目标子句（goal clause）。
- 仅含有确定子句的知识库很有意义，原因有3个：

（1）每个确定子句都可以写成一个蕴涵式，前提是正文字的合取式，结论是一个正文字。

（2）用霍恩子句进行推断可以通过前向链接（forward-chaining）算法和反向链接（backward-chaining）算法完成。这些算法的推断步骤很直观，便于人类理解。这类推断是逻辑编程（logic programming）的基础。

（3）用霍恩子句确定蕴含关系所需的时间与知识库大小呈线性关系。

命题定理证明：霍恩子句

南京大学 NANJING UNIVERSITY

合取范式、霍恩子句、确定子句、目标子句的文法：

$$
\begin{aligned}
\textit{CNF}\text{语句} &\rightarrow \text{子句}_1 \wedge \cdots \wedge \text{子句}_n \\
\text{子句} &\rightarrow \text{文字}_1 \vee \cdots \vee \text{文字}_m \\
\text{事实} &\rightarrow \text{符号} \\
\text{文字} &\rightarrow \text{符号} \mid \neg\text{符号} \\
\text{符号} &\rightarrow P \mid Q \mid R \mid \cdots \\
\text{霍恩子句范式} &\rightarrow \text{确定子句范式} \mid \text{目标子句范式} \\
\text{确定子句范式} &\rightarrow \textit{Fact} \mid (\text{符号}_1 \wedge \cdots \wedge \text{符号}_l) \Rightarrow \text{符号} \\
\text{目标子句范式} &\rightarrow (\text{符号}_1 \wedge \cdots \wedge \text{符号}_l) \Rightarrow \textit{False}
\end{aligned}
$$

命题定理证明：前向链接

南京大学 NANJING UNIVERSITY

```
function PL-FC-ENTAILS?(KB, q) returns true或false
  inputs: KB，知识库，一个命题确定子句集
          q，查询，一个命题符号
  count ← 一个表，其中count[c]为子句c的初始符号数量
  inferred ← 一个表，其中inferred[s] 初始对所有符号为false
  queue ← 一个符号队列，初始符号为KB中已知为true的符号

  while queue不空 do
      p ←POP(queue)
      if p = q then return true
      if inferred[p] = false then
          inferred[p] ← true
          for each p在c.PREMISE中的知识库中的子句c do
              count[c]减1
              if count[c] = 0 then 将c.CONCLUSION添加到queue
  return false
```

命题逻辑的前向链接算法

命题定理证明：前向链接

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

$B \wedge L \Rightarrow M$

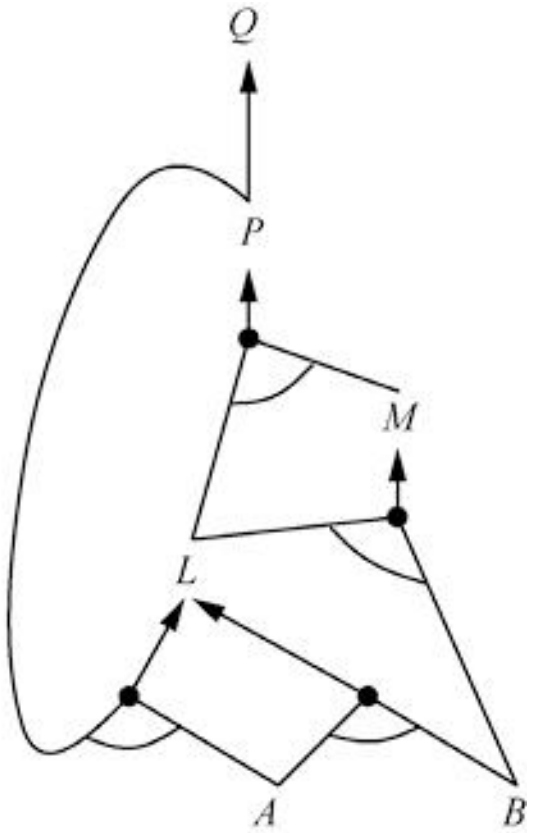
$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B

（a）



（b）

（a）一个霍恩子句集。（b）相应的与或图

命题定理证明：前向链接

前向链接算法示例

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

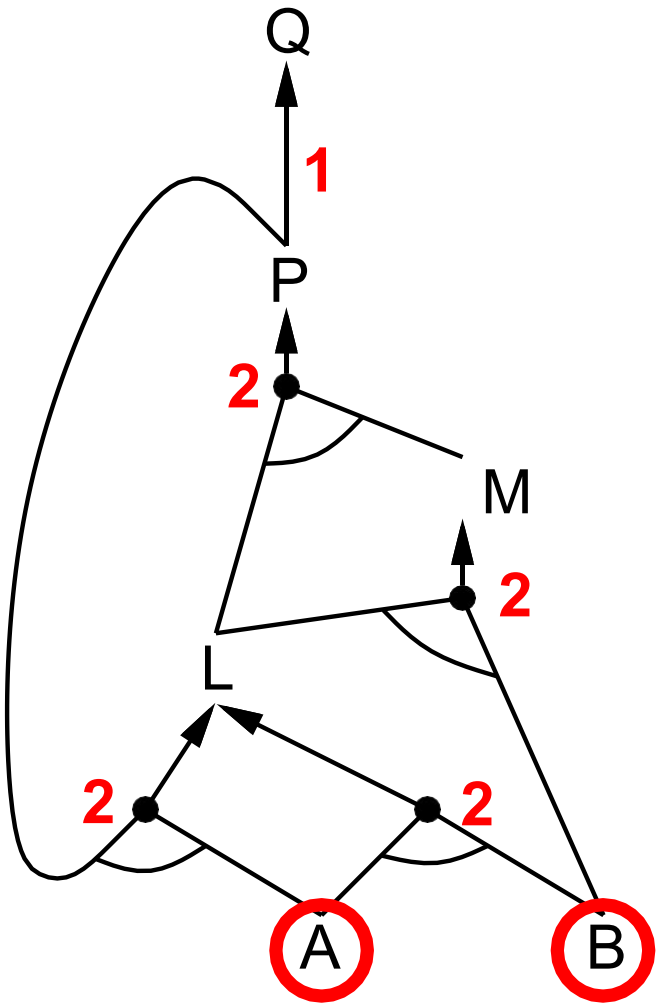
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：前向链接

前向链接算法示例

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

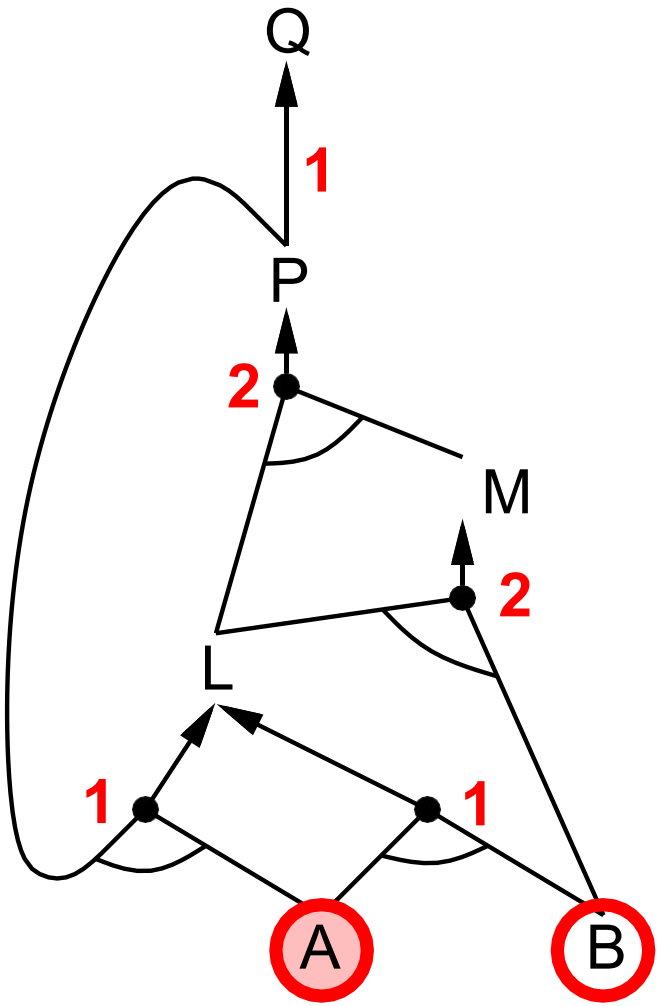
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：前向链接

前向链接算法示例

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

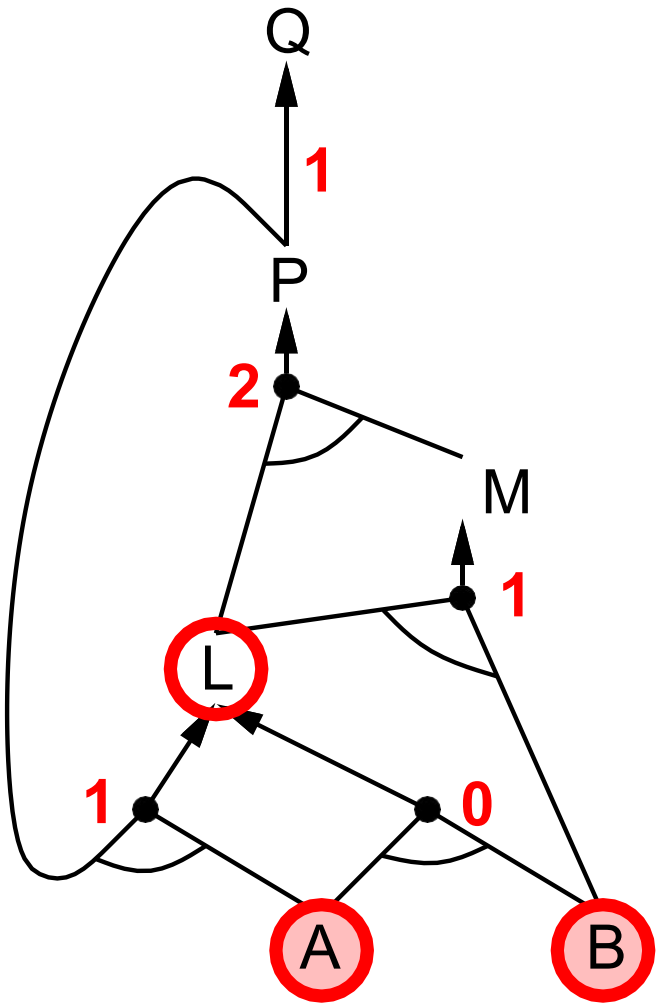
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：前向链接

南京大学 NANJING UNIVERSITY

前向链接算法示例

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

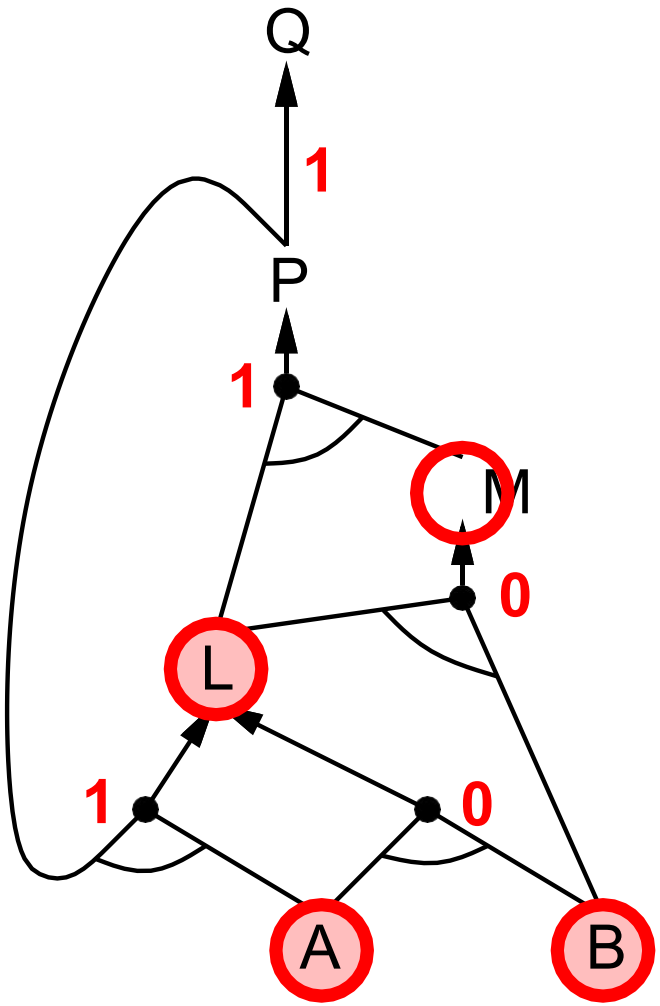
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：前向链接

前向链接算法示例

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

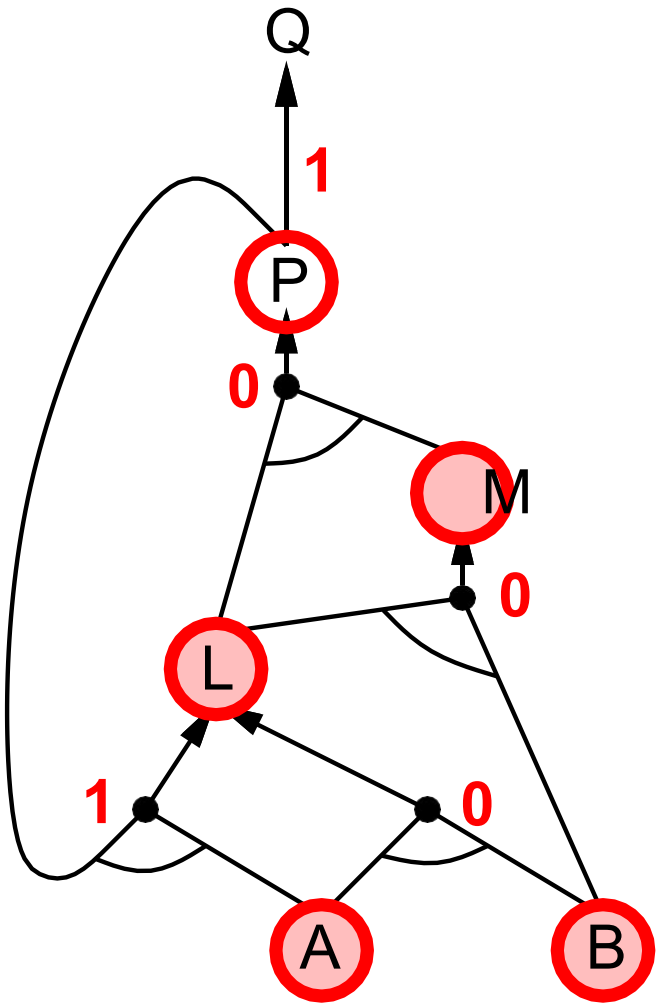
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：前向链接

南京大学 NANJING UNIVERSITY

前向链接算法示例

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

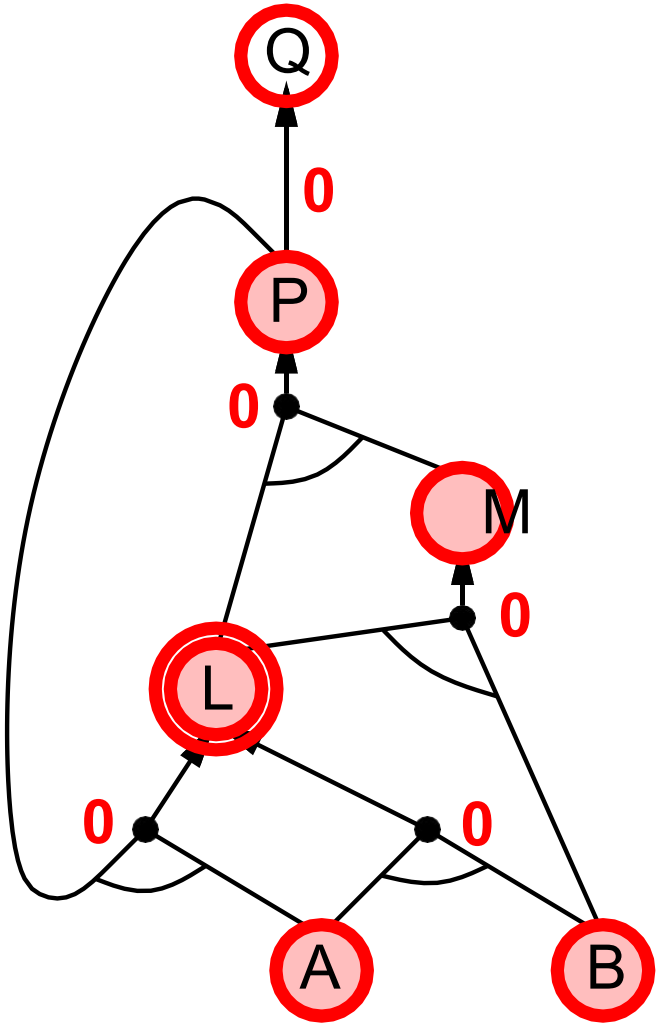
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：前向链接

南京大学 NANJING UNIVERSITY

前向链接算法示例

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

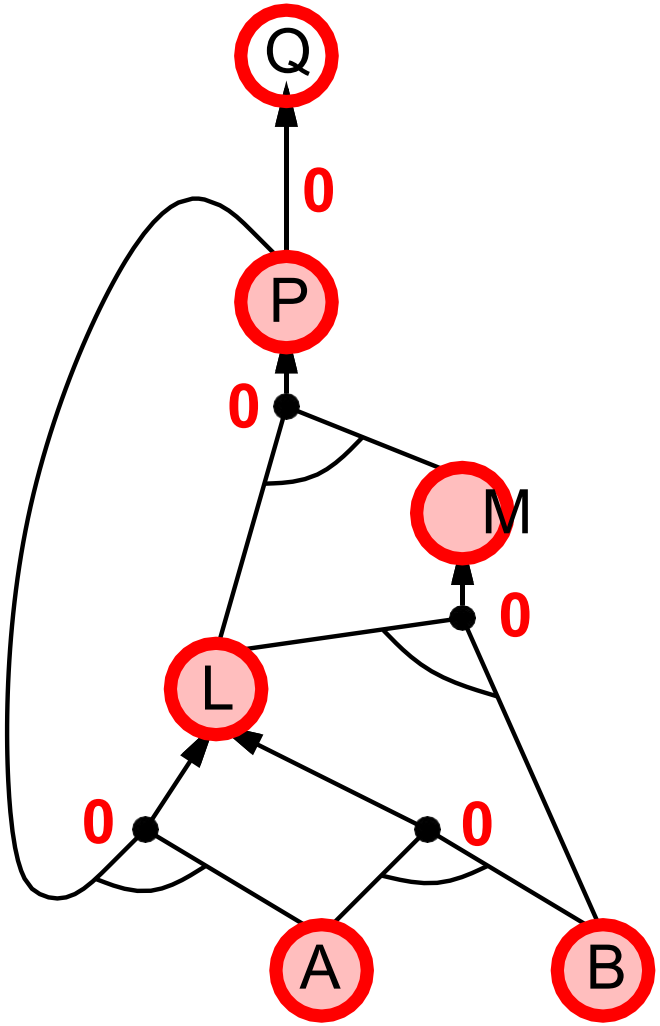
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：前向链接

南京大学 NANJING UNIVERSITY

前向链接算法示例

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

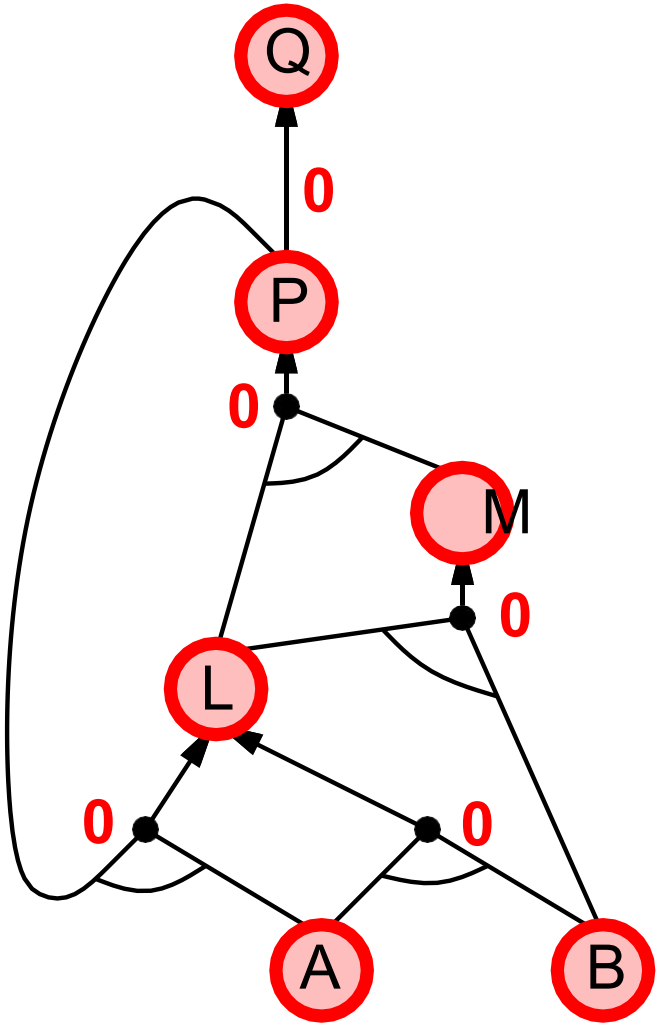
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：反向链接

- **思想:** 从查询 q 开始反向运作。
- 为了通过反向链接证明q,
  首先检查 q 是否已知为真, 否则
  使用反向链接证明知识库中结论为 q 的蕴涵式
- 避免循环：检查新的子目标是否已经在目标堆栈里
- 避免重复性工作：检查新的子目标
  1) 是否已经被证明, 或者
  2) 是否已经失败

命题定理证明：反向链接

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

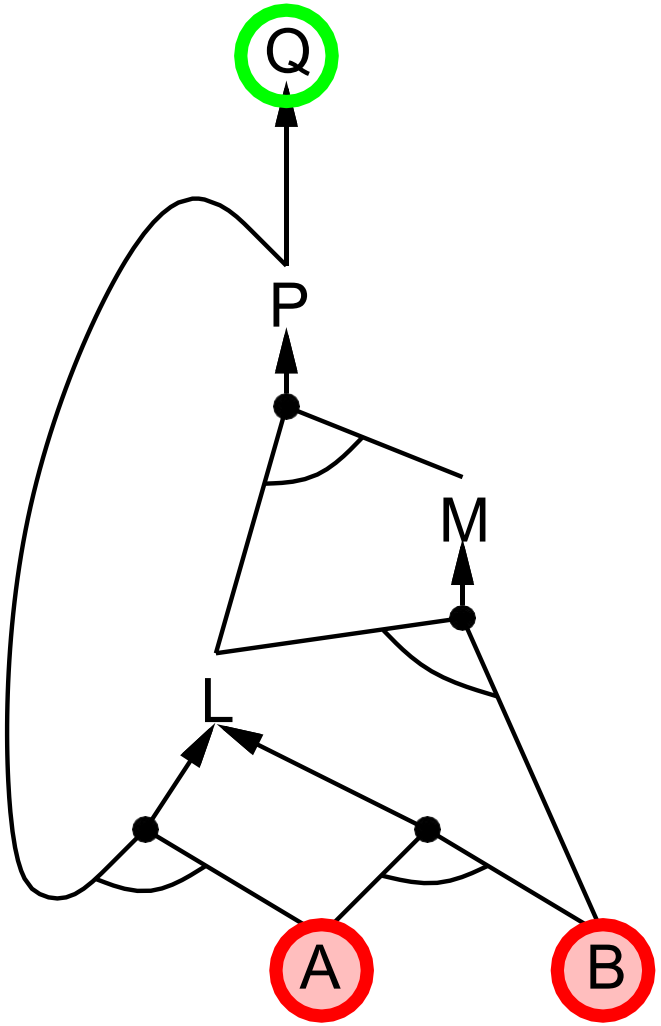
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：反向链接

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

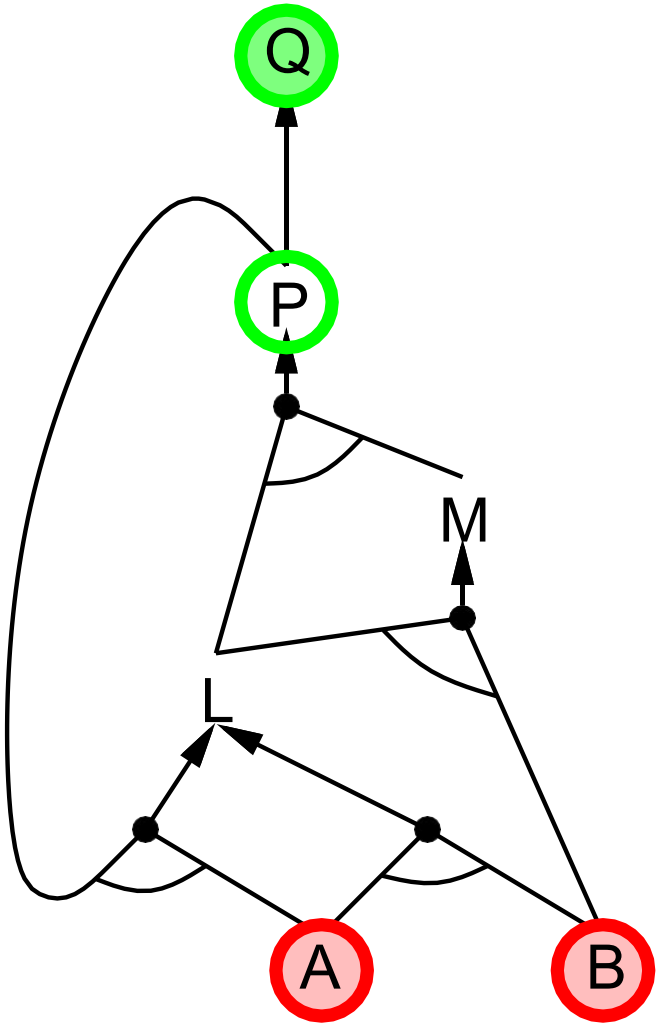
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：反向链接

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

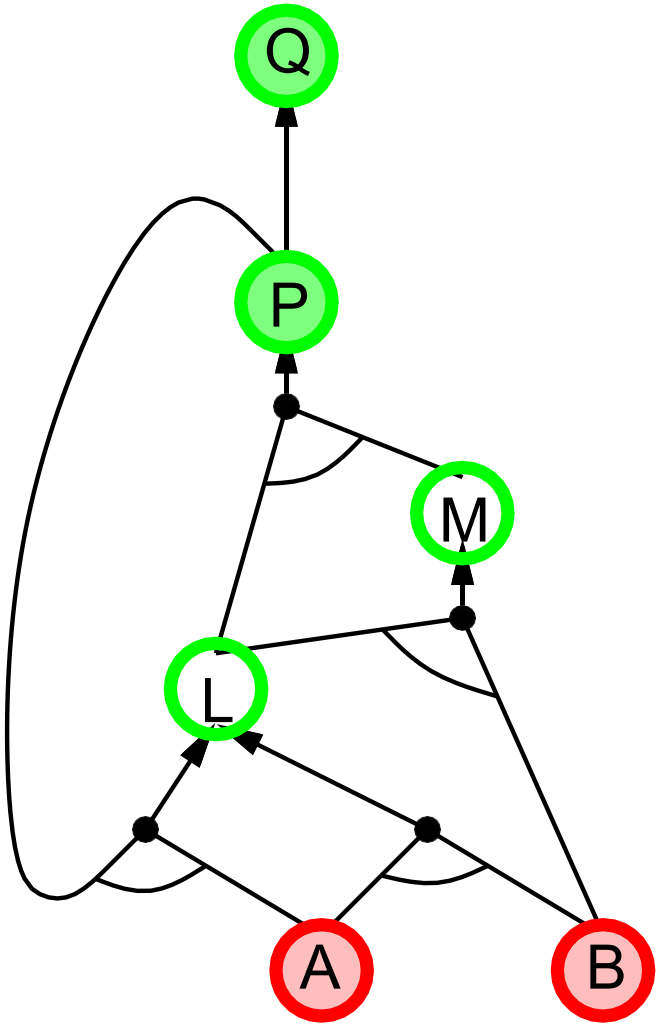
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：反向链接

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

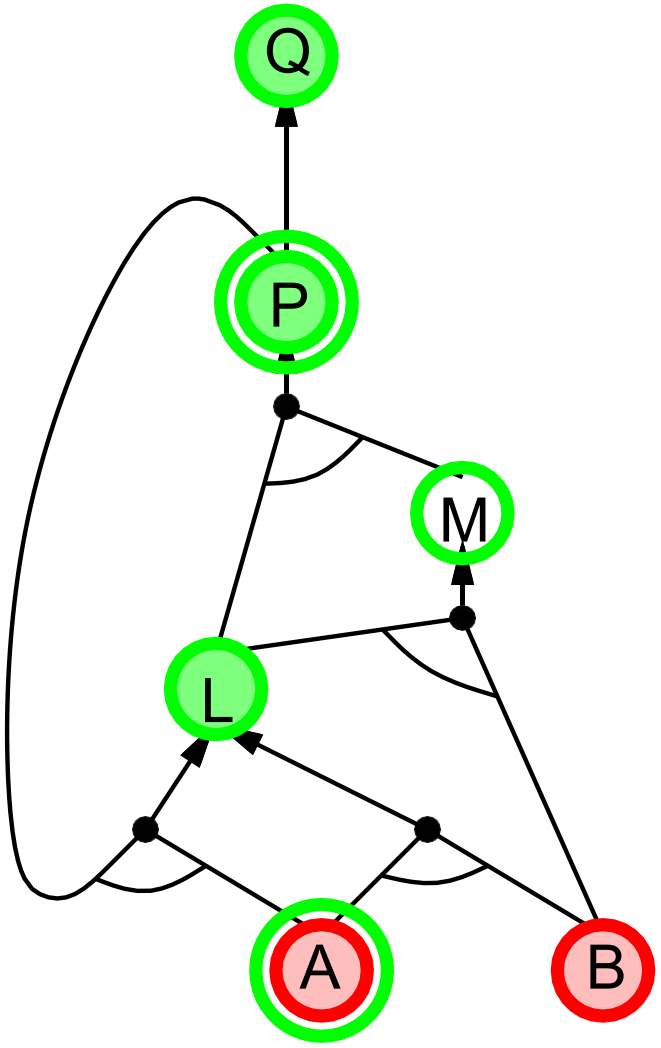
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：反向链接

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

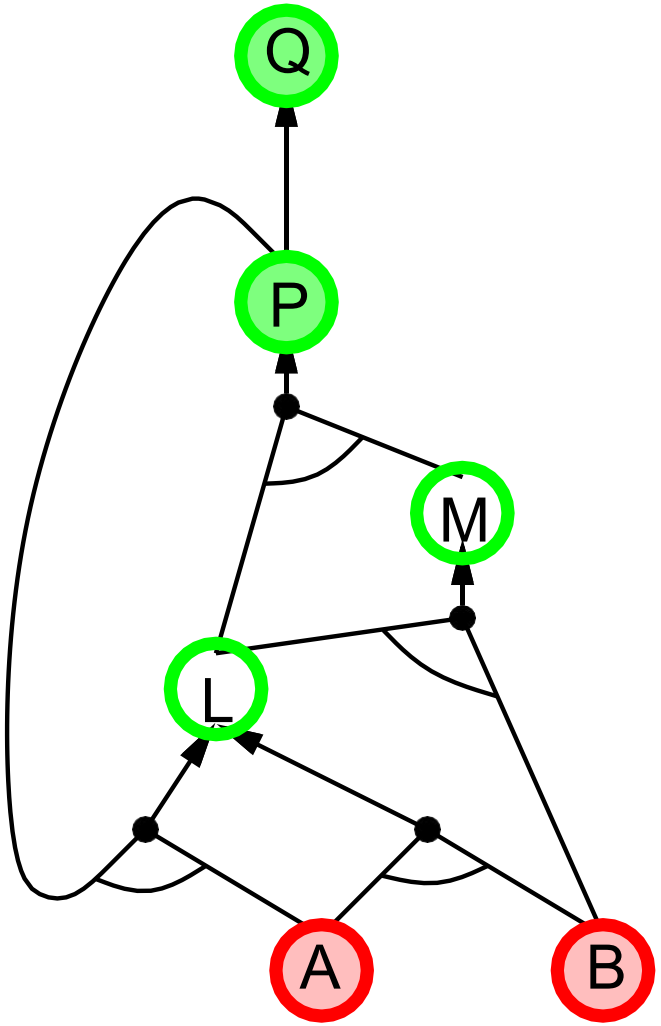
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：反向链接

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

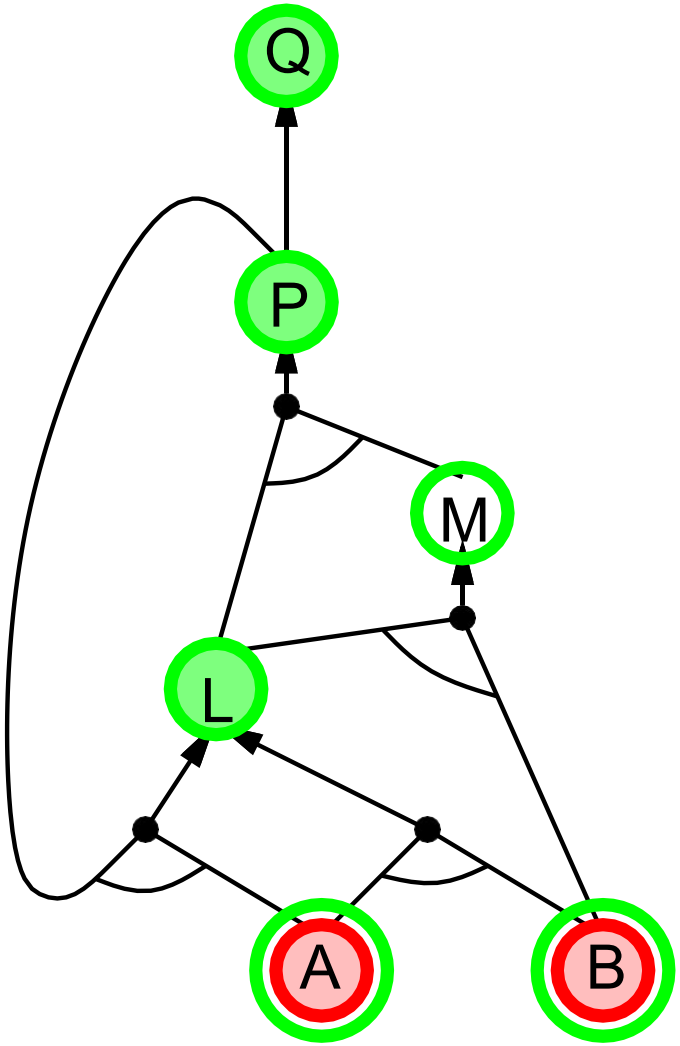
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：反向链接

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

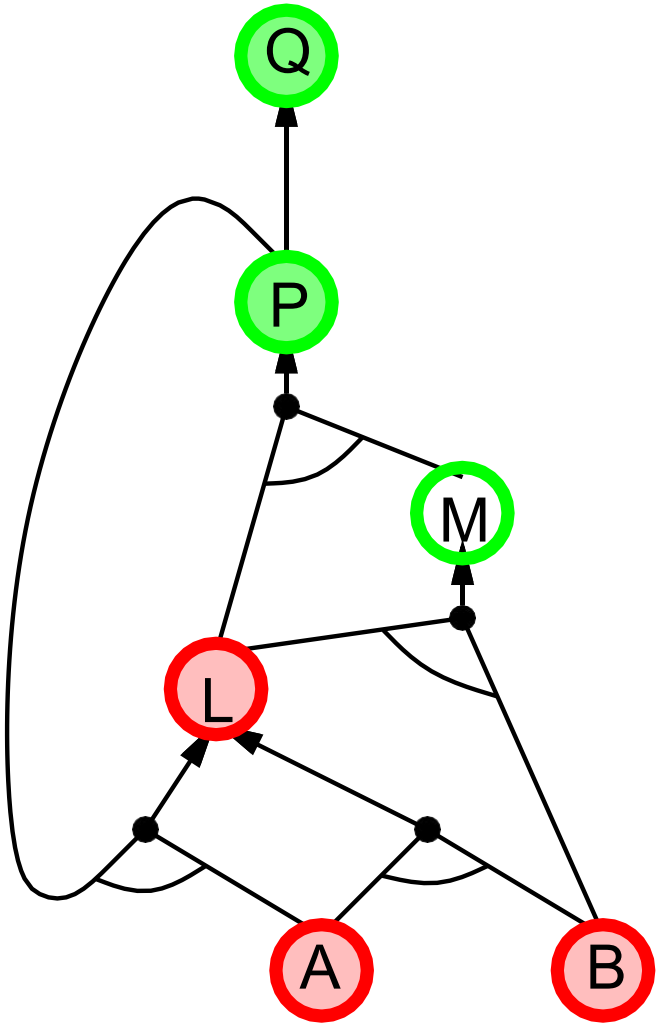
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：反向链接

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

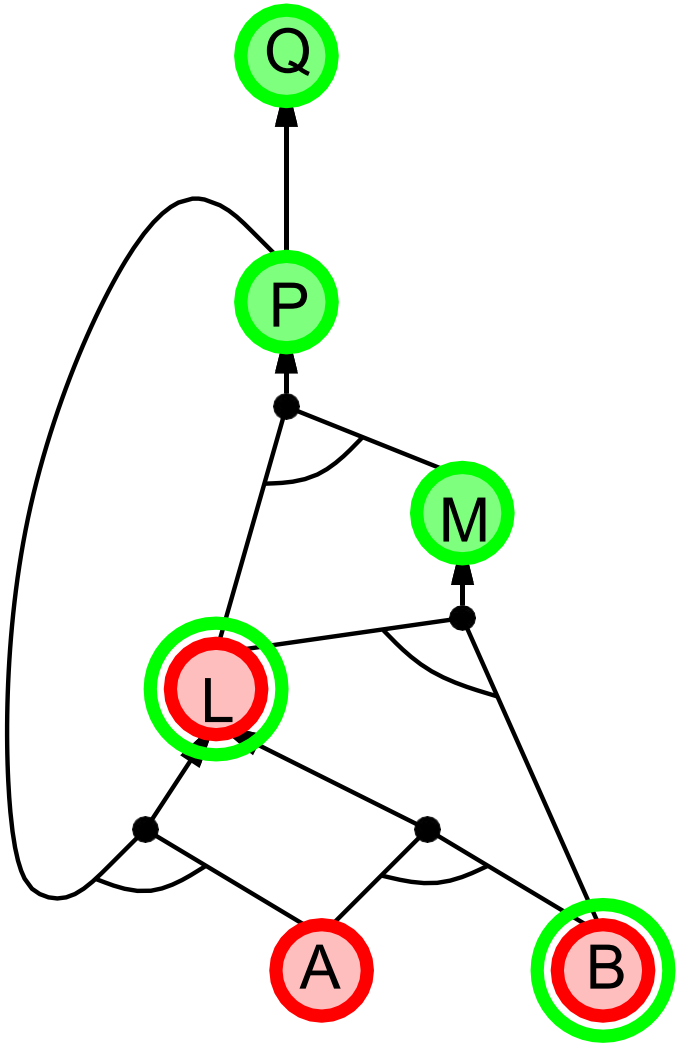
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：反向链接

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

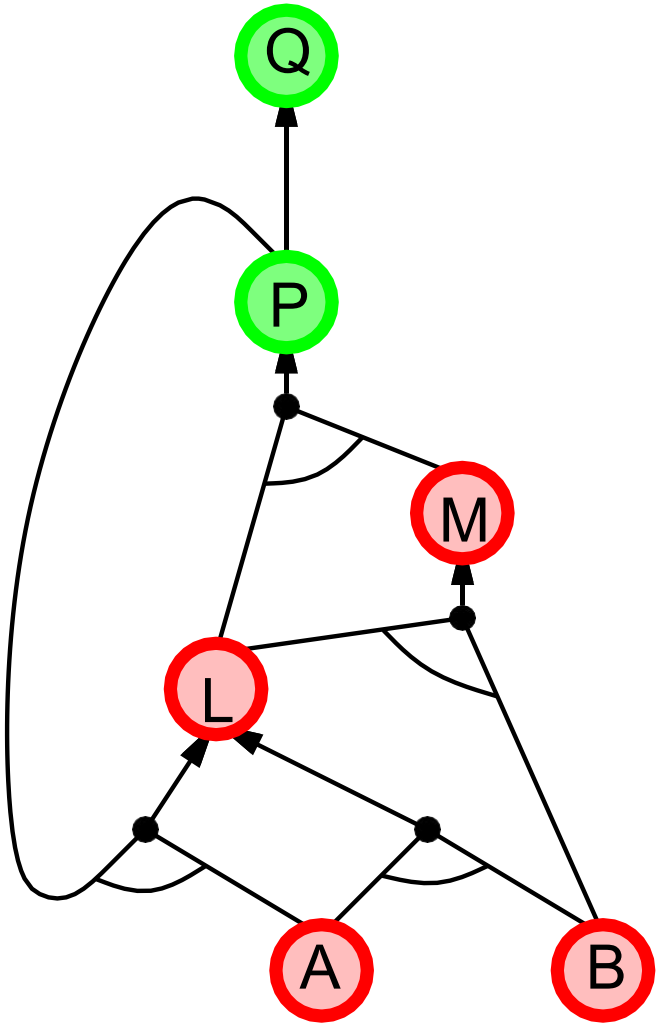
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：反向链接

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

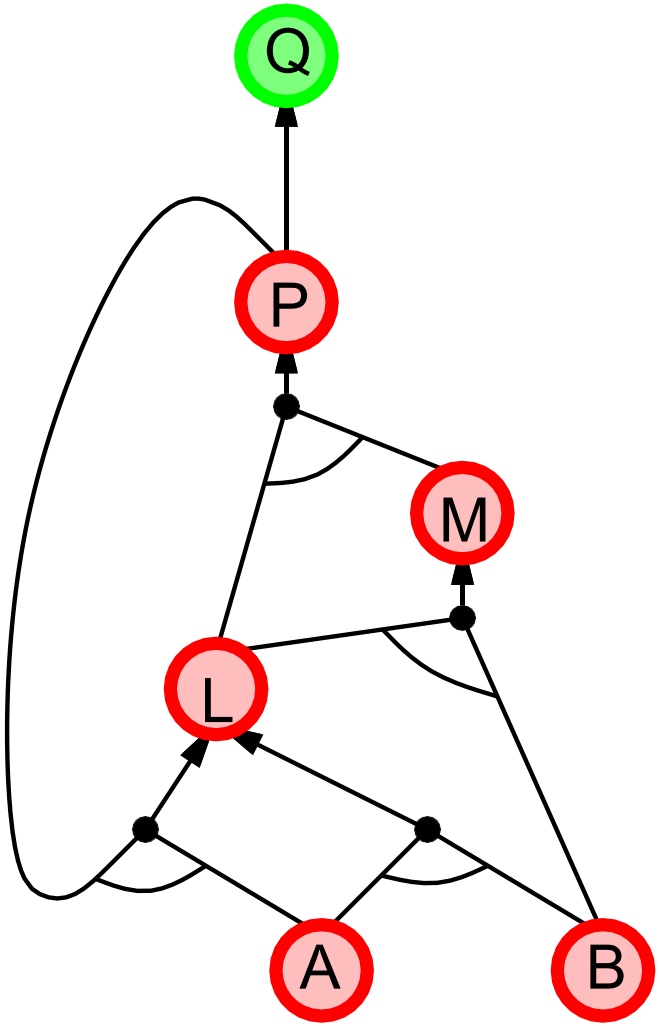
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：反向链接

$P \Rightarrow Q$

$L \wedge M \Rightarrow P$

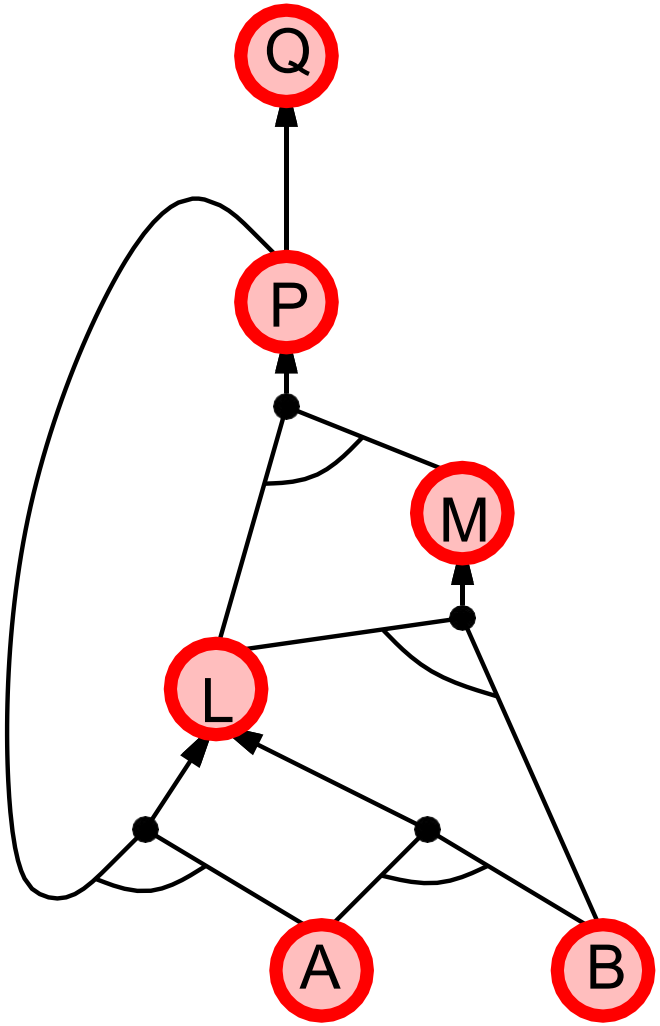
$B \wedge L \Rightarrow M$

$A \wedge P \Rightarrow L$

$A \wedge B \Rightarrow L$

A

B



命题定理证明：前向vs反向链接

- 前向链接是数据驱动（data-driven），是一种自动、无意识的处理，可能会做大量与目标无关的工作。
- 反向链接是一种目标导向推理（goal-directed reasoning）它对于回答类似“我现在该做什么？”和“我的钥匙在哪里？”这样的特定问题非常有用。通常，反向链接的代价远小于知识库规模的线性变化，因为这个过程仅涉及相关的事实。

小结

- 一个**基于知识的智能**体由一个知识库和一套推断机制组成。它的运作方式是在知识库中存储关于世界的语句，使用推断机制推断新语句，并使用这些语句来决定采取何种动作。
- 一种表示语言是用其**语法**和**语义**来定义的，语法规定了语句的结构，语义定义了每个可能世界或模型中每条语句的真值。
- 语句之间的**蕴含**关系对于我们对推理的理解非常重要。在所有α为真的世界中β也为真，则语句α蕴含语句β 。其等价定义包括语句$\alpha \Rightarrow \beta$的**有效性**和语句$\alpha \wedge \neg\beta$的不可满足性。
- 推断是从旧语句推得新语句的过程。**可靠的**推断算法只推出蕴含的语句，**完备的**算法则可以推得所有蕴含的语句。
- **命题逻辑**是由**命题符号**和**逻辑联结词**构成的简单语言。它可以处理已知为真、为假或完全未知的命题。
- **推断规则**是可靠推断的模式，它可以用于证明命题定理。**归结规则**能产生一个用于知识库的完备推断算法，以合取范式的形式表示。**前向链接**和**反向链接**是霍恩形式知识库的非常自然的推理算法。

提纲

南京大学 NANJING UNIVERSITY

- 逻辑智能体
  - 基于知识的智能体
  - wumpus世界与吃豆人世界
  - 逻辑
  - 命题逻辑
- **一阶逻辑**
  - 一阶逻辑的语法和语义
  - 使用一阶逻辑
  - 一阶逻辑中的知识工程

命题逻辑的优劣

- **陈述性（declarative）**：在命题逻辑中知识与推断是独立的，而推断完全是领域无关的。

- **合成性（compositionality）**：一条语句的含义是其各个组成部分的含义的一个函数。

- 命题逻辑作为一种因子化表示，缺乏能够简洁描述具有多个对象的环境的表达能力。

一阶逻辑

命题逻辑假设世界包含事实，
一阶逻辑（类似自然语言）假设世界包含

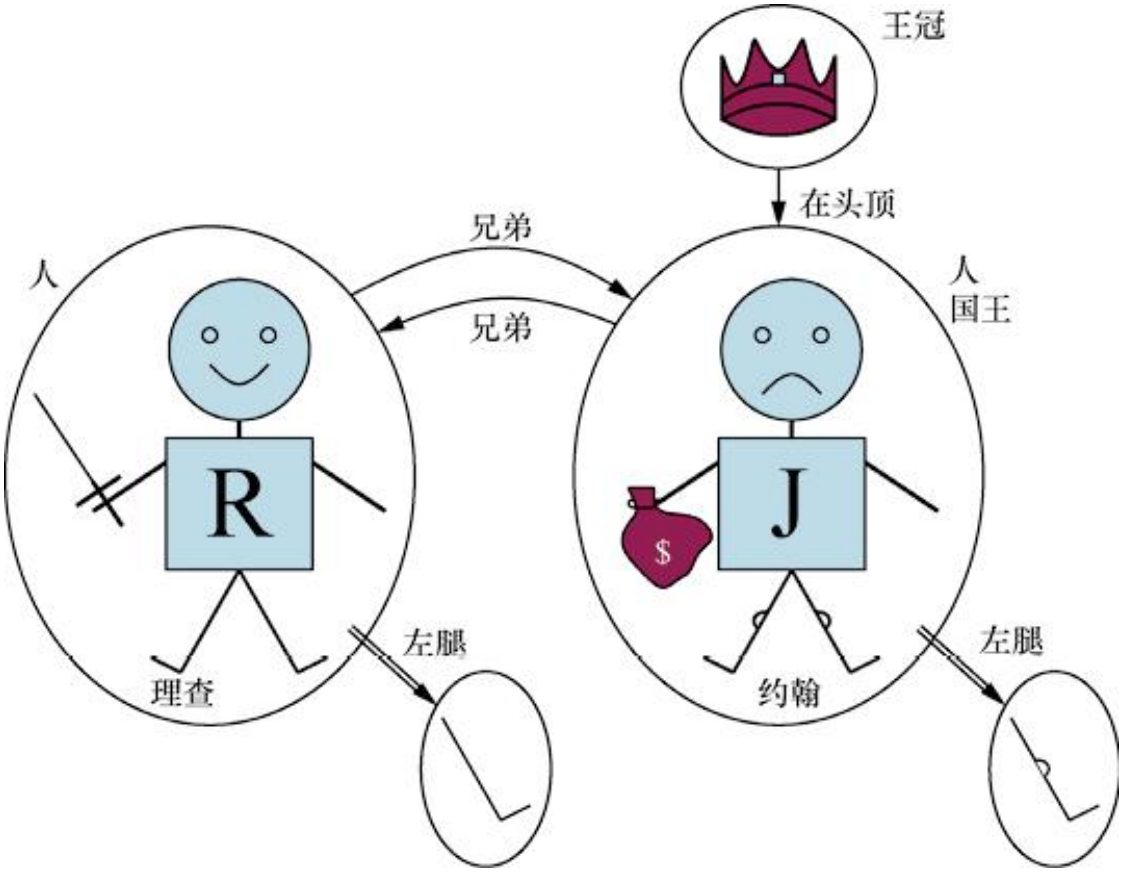
- 对象: people, houses, numbers, theories, Ronald McDonald, colors, baseball games, wars, centuries . . .
- 关系: red, round, bogus, prime, multistoried . . ., brother of, bigger than, inside, part of, has color, occurred after, owns, comes between, . . .
- 函数: father of, best friend, third inning of, one more than, end of
  . . .

一阶逻辑

语言	本体论约定 （世界上存在的东西）	认识论约定 （智能体对事实的认识）
命题逻辑	事实	真/假/未知
一阶逻辑	事实，对象，关系	真/假/未知
时态逻辑	事实，对象，关系，时间	真/假/未知
概率论	事实	信念度$\in[0,1]$
模糊逻辑	具有真实度$\in[0,1]$ 的事实	已知的区间值

形式化语言及其本体论约定和认识论约定

一阶逻辑的语法和语义



含有5个对象、2个二元关系（兄弟和在头顶）、3个一元关系（人、国王和王冠）和1个一元函数（左腿）的模型

一阶逻辑的语法和语义

南京大学 NANJING UNIVERSITY

- 一阶逻辑的基本语法元素是代表对象、关系和函数的符号。因此，符号分为3种：

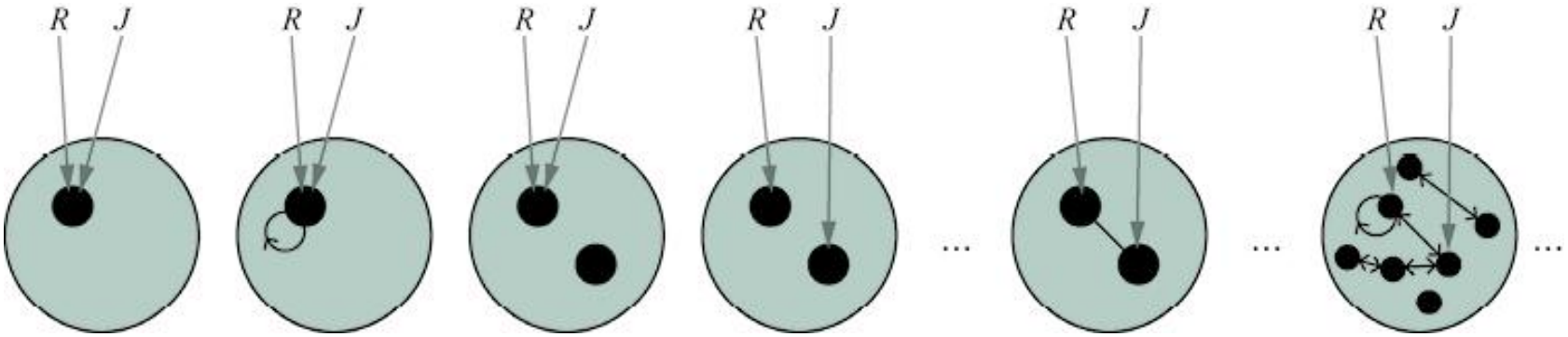
1）代表对象的**常量符号（constant symbol）**

2）代表关系的**谓词符号（predicate symbol）**

3）代表函数的**函数符号（function symbol）**

- 每个模型必须提供所需的信息来确定任意给定语句为真还是为假。因此，除了它的对象、关系和函数，每个模型还要包含一套确切指明常量、谓词和函数符号指代的是哪个对象、关系和函数的**解释（interpretation）**。

一阶逻辑的语法和语义



含有两个常量符号的语言中全部模型的集合的部分成员、R和J以及一个二元关系符号。每种常量符号的解释用灰色箭头标明。每个模型中，相关的对象用箭头连接。

一阶逻辑的语法和语义

南京大学 NANJING UNIVERSITY

- **项（term）**是指代对象的逻辑表达式。
- **复合项**的组成是一个函数符号后跟随一个括号，括号中是一系列项，作为该函数符号的参数。
- **项的形式化语义：**考虑项f(t , ..., tn)。函数符号f指代模型中的某个函数（不妨称为F），参数项指代域中的对象（称为d , ..., dn)，整个项就指代将函数F应用于d , ..., dn产生的对象，即函数的值。
- **原子语句**（或简称原子）是由谓词符号以及其后可能存在的括号中的一系列项组成的，例如：$Brother(Richard, John)$
- **复合语句：**与命题演算的语法和语义一样，可以使用**逻辑联结词**构建更为复杂的语句。

$$\neg Brother(LeftLeg(Richard), John)$$

$$Brother(Richard, John) \wedge Brother(John, Richard)$$

$$King(Richard) \vee King(John)$$

$$\neg King(Richard) \Rightarrow King(John)$$

一阶逻辑的语法和语义

南京大学 NANJING UNIVERSITY

全称量词：

$\forall x\ P$ ，这里符号 x 被称为变量， P 为任意逻辑语句，表明 P 对每个对象 x 都为真。例如，

所有国王都是人：$\forall x\, King(x) \Rightarrow Person(x)$

如果 P 在根据一个模型的给定解释构建的所有可能**扩展解释（extended interpretation）**下为真，则 $\forall x\ P$ 在该模型中为真，其中每个扩展解释给出了 x 指代的域元素。

$x \rightarrow$ 狮心理查　　狮心理查是一位国王 $\Rightarrow$ 狮心理查是一个人

$x \rightarrow$ 约翰国王　　约翰国王是一位国王 $\Rightarrow$ 约翰国王是一个人

$x \rightarrow$ 理查的左腿　　理查的左腿是一位国王 $\Rightarrow$ 理查的左腿是一个人

$x \rightarrow$ 约翰的左腿　　约翰的左腿是一位国王 $\Rightarrow$ 约翰的左腿是一个人

$x \rightarrow$ 王冠　　王冠是一位国王 $\Rightarrow$ 王冠是一个人

一阶逻辑的语法和语义

南京大学 NANJING UNIVERSITY

存在量词：

$\exists x\ P$ ，表明 P 至少对于一个对象 x 为真。例如，

约翰国王的头顶有王冠：$\exists x\ Crown(x) \wedge OnHead(x, John)$

狮心理查是王冠 $\wedge$ 狮心理查在约翰的头顶

约翰国王是王冠 $\wedge$ 约翰国王在约翰的头顶

理查的左腿是王冠 $\wedge$ 理查的左腿在约翰的头顶

约翰的左腿是王冠 $\wedge$ 约翰的左腿在约翰的头顶

王冠是王冠 $\wedge$ 王冠在约翰的头顶

量词嵌套

- 兄弟是同胞
  $\forall x\ \forall y\ \ Brother(x,y) \Rightarrow Sibling(y,x)$
- “同胞” 是对称关系
  $\forall x,y\ \ Sibling(x,y) \Leftrightarrow Sibling(y,x)$
- 每个人都喜爱一些人
  $\forall x\ \exists y\ Loves(x,y)$
- 有人被所有人喜爱
  $\exists y\ \forall x\ Loves(x,y)$
- 当两个量词与相同的变量名合用时会引起一些混淆。考虑语句
  $\forall x\ (Crown(x) \vee (\exists x\ Brother(Richard,x)))$

一阶逻辑的语法和语义

南京大学 NANJING UNIVERSITY

全称和存在量词的关系

$\forall x \ \neg Likes(x, Parsnips)$ 等价于 $\neg \exists x \ Likes(x, Parsnips)$

$\forall x \ Likes(x, IceCream)$ 等价于 $\neg \exists x \ \neg Likes(x, IceCream)$

量化语句和非量化语句的德摩根律如下:

$$\neg \exists x \ P \equiv \forall x \ \neg P \qquad \neg (P \lor Q) \equiv \neg P \land \neg Q$$

$$\neg \forall x \ P \equiv \exists x \ \neg P \qquad \neg (P \land Q) \equiv \neg P \lor \neg Q$$

$$\forall x \ P \equiv \neg \exists x \ \neg P \qquad P \land Q \equiv \neg (\neg P \lor \neg Q)$$

$$\exists x \ P \equiv \neg \forall x \ \neg P \qquad P \lor Q \equiv \neg (\neg P \land \neg Q)$$

一阶逻辑的语法和语义

南京大学 NANJING UNIVERSITY

等词符号（equality symbol）表示两个项指代相同的对象。

例如：Father(John) = Henry 表示Father(John)指代的对象与Henry 指代的对象是相同的。

它也可以与否定合用，表示两项不是相同的对象。要表示理查至少有两个兄弟，我们可以写成：

$$\exists x, y \; Brother(x, Richard) \wedge Brother(y, Richard) \wedge \neg (x = y)$$

使用一阶逻辑

南京大学 NANJING UNIVERSITY

一阶逻辑的断言与查询

- 一阶逻辑中的语句是通过Tell添加到知识库的，与在命题逻辑中完全一样。这种语句被称为**断言（assertion）**。例如，我们可以断言约翰是国王、理查是人以及所有的国王都是人：

  $\text{TELL}(KB, King(John))$

  $\text{TELL}(KB, Person(Richard))$

  $\text{TELL}(KB, \forall x\ King(x) \Rightarrow Person(x))$

- 使用Ask对知识库提问。例如：$\text{ASK}(KB, King(John))$

- 使用Ask提出的问题被称为**查询**或目标。一般来说，知识库中逻辑蕴含的所有查询都应该得到肯定的回答。

- 我们可以提出量化的问题：$\text{ASK}(KB, \exists x\ Person(x))$

- 如果我们想了解使语句为真的x的值，我们就需要另一个函数：

  $\text{ASKVARS}(KB, Person(x))$

  返回一系列答案。在这个例子中有两个答案：$\{x/John\}$ $\{x/Richard\}$

使用一阶逻辑

南京大学 NANJING UNIVERSITY

wumpus世界的知识库

"感知"

$\forall b, g, t \quad Percept([Smell, b, g], t) \Rightarrow Smelt(t)$

$\forall s, b, t \quad Percept([s, b, Glitter], t) \Rightarrow AtGold(t)$

反射: $\forall t \quad AtGold(t) \Rightarrow Action(Grab, t)$

使用内部状态的反射: 我们已经获得金条了吗?

$\forall t \quad AtGold(t) \wedge \neg Holding(Gold, t) \Rightarrow Action(Grab, t)$

$Holding(Gold, t)$ 无法被观测

$\Rightarrow$ 追踪变化很重要

知识工程的过程

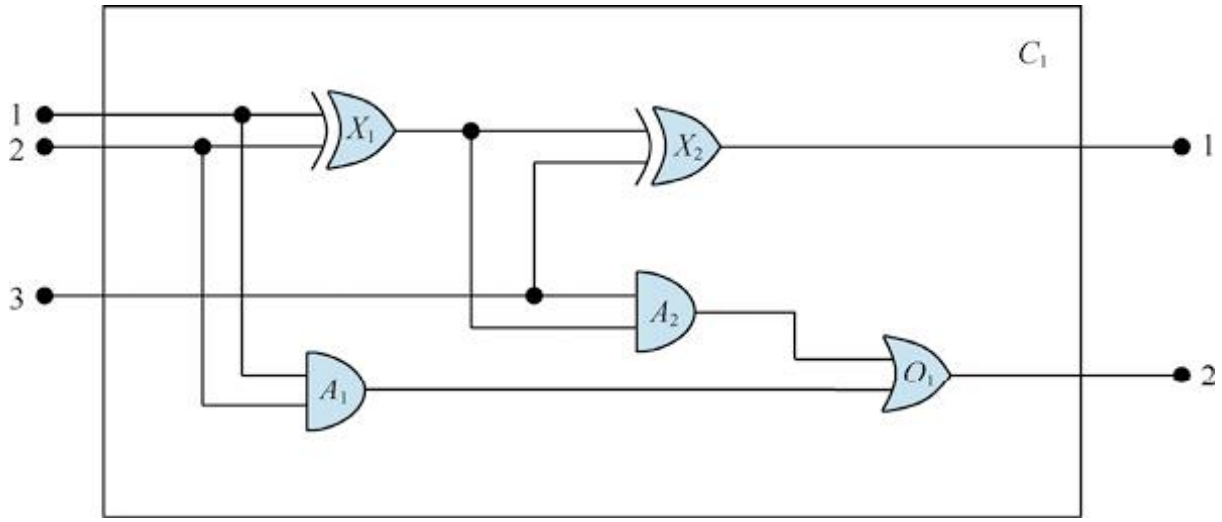
知识工程 (Knowledge engineering): 知识库构建的一般过程。

1. 确定问题。
2. 收集相关知识。
3. 确定谓词、函数和常量的词汇表。
4. 对论域的通用知识编码。
5. 对问题实例的描述编码。
6. 向推断过程提出查询并获得答案。
7. 调试并评估知识库。

一阶逻辑中的知识工程

南京大学 NANJING UNIVERSITY

知识工程的过程



一位全加器的数字电路C1。前两个输入是要相加的两位，第三个输入是进位位。第一个输出是和，第二个输出是通往下一个加法器的进位位。电路包含两个异或门、两个与门和一个或门路。

小结

- 知识表示语言应当是说明性的、合成式的、有表达能力的、上下文无关的且无歧义的。
- 逻辑之间的区别在于其**本体论约定**和**认识论约定**，命题逻辑仅约定事实的存在，一阶逻辑则约定对象和关系的存在，因而增加了表达能力，适用于像wumpus世界和电子电路这样的论域。
- 一阶逻辑的语法构建于命题逻辑之上。它增加了项来表示对象，并且有全称量词和存在量词来构建关于被量化的变量的全部或部分可能值的断言。
- 一阶逻辑的一个**可能世界**或**模型**包括一个对象集和一种将常量符号映射到对象、将谓词符号映射到对象的关系、将函数符号映射到对象上的函数的解释。
- 一条原子语句为真，仅当谓词命名的关系在项命名的对象之间成立。**扩展解释**将量词变量映射到模型中的对象，定义了量化语句的真值。