总复习

2024/7/2

1.1人工智能的概念

从学科角度看，人工智能(Artificial Intelligence,简记为AI)是当前科学技术迅速发展及新思想、新理论、新技术不断涌现的形势下产生的一门新学科(从学科的时间尺度上)，也是一门涉及数学、计算机科学、哲学、认知心理学、信息论、控制论等学科的交叉学科(从学科性质上看)。

1.1.3人工智能的主要学派

由于人们对“智能”本质的不同理解和认识，形成了人工智能研究的不同途径。逐步形成了符号主义、连接主义和行为主义三大学派。

符号主义

又称为逻辑主义、心理学派或计算机学派是基于物理符号系统的人工智能学派

谓词逻辑

联接主义

又称为仿生学派或生理学派，是基于神经网络与学习算法的人工智能学派

深度神经网络

行为主义

又称为进化主义或控制论学派，是基于控制论和“动作-感知”控制系统的人工智能学派

强化学习

第2章知识表示

2.1 概述

2.2谓词逻辑表示法

2.3 产生式表示法

2.4框架表示

2.5 语义网络

2.6 面向对象表示等等。

2.2.1命题

命题变元

不是 指派

原子命题

命题 特定命题

原子变元

- **命题：** 具有真假意义的陈述句。

  原子命题：不能分解成更简单的命题。

  复合命题：由连接词、标点符号和原子命题等复合构成的命题。

- **命题逻辑：** 研究命题和命题之间关系的符号逻辑系统。
- 命题常量(命题常元)：一个命题标识符表示确定的命题。
- **命题变元(命题变量)**：命题标识符只表示任意命题的位置标志。

2.2.1命题

- 命题逻辑的符号包括以下几种：

(1)命题常元：　True(T) 和False(F);

(2)命题符号：　P、Q、R、T 等；

（3）联 结 词 ：　①－ ；②A:③√:④→;　　⑤ ——。

(4)括号：()。

命题逻辑主要使用这5个联结词来表示逻辑关系，通过这些联结词，可以由简单的命题构成复杂的复合命题。

2.2.2谓词

- 谓词逻辑是命题逻辑的扩充与发展，它将一个原子命题分解成谓词与个体两部分。谓 词描述个体的属性，以及个体之间**的关系**。谓词对应于语言中的谓语，或谓语+宾语。
- 谓词逻辑继续拆分命题，把命题拆为“π”、“...是无理数”、“..是实数”这些结构，可以得出命题 R“π 是实数”这种命题。其中“...是无理数”、“...是实数”称为谓 词。上例所说的π就是**个体词**。
- 个体常项常用单词或 a,b,c 等小写字母表记，个体变项常用x,y,z等小写字母表记。

2.2.4谓词逻辑表示

口谓词公式表示知识的步骤：

01 定义谓词及个体，确定每个谓词及个体的确切含义；

02 根据所要表达的事物或概念，为每个谓词中的变元赋以特定的值；

03 根据所要表达的知识的语义，用适当的连接符将各个谓词连接起来形成谓词公式。

2.2.4谓词逻辑表示

- 例2.2用谓词逻辑表示下列知识：

武汉是一个美丽的城市，但它不是一个沿海城市。
如果马亮是男孩，张红是女孩，则马亮比张红长的高。

第一步：定义谓词如下。

BCity(x): x是一个美丽的城市
HCity(x):x是一个沿海城市
Boy(x):x 是男孩
Girl(x): x是女孩
High(x,y):x 比y长得高

2.2.4谓词逻辑表示

口例2.2 用谓词逻辑表示下列知识：

武汉是一个美丽的城市，但它不是一个沿海城市。
如果马亮是男孩，张红是女孩，则马亮比张红长的高。

第二步：将这些个体代入谓词中。

BCity(wuhan),HCity(wuhan),Boy(mal),
Girl(zhangh),High(mal,zhangh)

第三步：根据语义，用谓词连接符将他们连接起来，得到表示上述知识的谓词公式。

BCity(wuhan)A-HCity(wuhan)
(Boy(mal)AGirl(zhangh))→High(mal,zhangh)

2.2.4谓词逻辑表示

□ **例2.3用谓词逻辑表示下列知识：**

所有学生都穿彩色制服。

任何整数或者为正数或者为负数。

自然数都是大于零的整数。

解： **第一步：定义谓词如下**

Student(x):x 是学生　Uniform(x,y):x 穿y

I(x): 是整数　P(x):x 是正数　Q(x):x 是负数

N(x):x 是自然数　　L(x):x 大于零

找不清谓词和个体的话：如果能翻译成如果.....那么.....可以考虑使用蕴含结构。全称量词更适合使用蕴含

注意一元谓词和二元谓词

按照第二步和第三步，上述知识可以用谓词公式分别表示为：

(Vx)(Student(x)→Uniform(x,color))

(Vx)(I(x)→P(x)VQ(x))

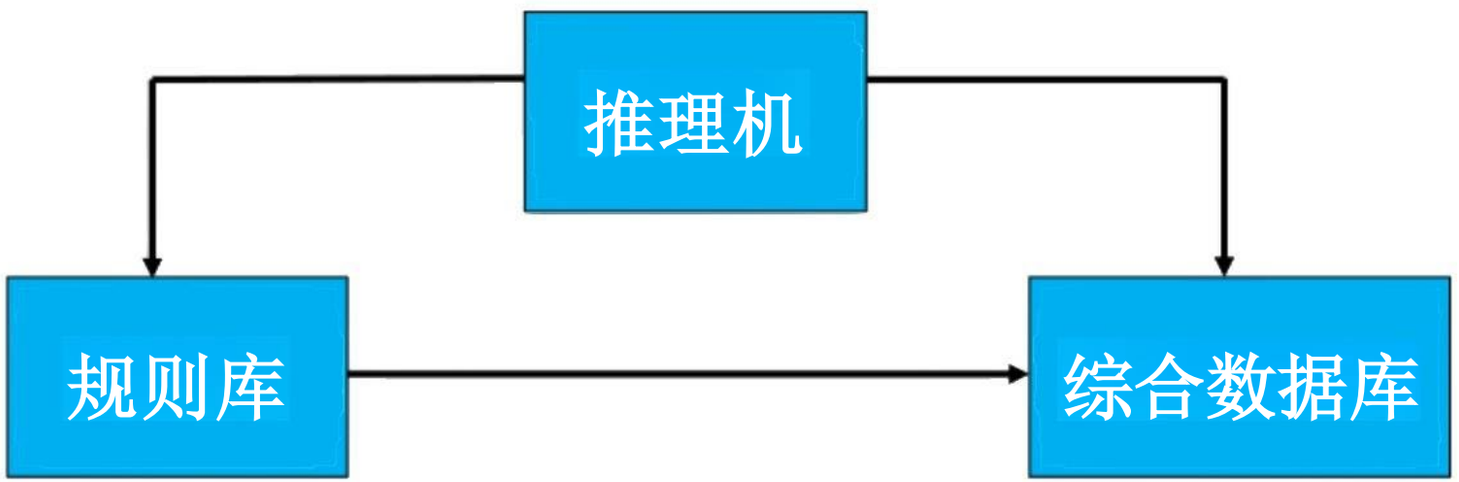
(Vx)(N(x)→L(x)AI(x))

2.3.2产生式系统的基本结构

产生式系统的组成

把一组产生式规则放在一起，让他们互相配合，协同作用，一个产生式规则生成的结论可以供另一个产生式规则作为已知事实使用，以求得问题的解决，这样的系统称为产生式系统。

一般说来， 一个产生式系统由以下三个基本部分组成：



2.3.2产生式系统的基本结构

1. 规则库

- **规则库：**用于描述相应领域内知识的产生式集合，是某领域知识(规则)的存储器，其中的规则是以产生式形式表示的。规则库中包含着将问题从初始状态转换成目标状态(或解状态)的那些变换规则。

- **规则库是专家系统的核心**，也是一般产生式系统赖以进行问题求解的基础。其中知识的完整性和一致性、知识表达的准确性和灵活性以及知识组织的合理性，都将对产生式系统的性能和运行效率产生直接的影响。

2.3.2产生式系统的基本结构

2. 综合数据库

- **综合数据库**：用于存放输入的事实、从外部数据库输入的事实以及中间结果(事实)和最后结果的工作区。

- 当规则库中的某条产生式的前提可与综合数据库中的某些已知事实匹配时，该产生式就被激活，并把用它推出的结论放入综合数据库中，作为后面推理的已知事实。

综合数据库的内容是在不断变化的，是动态的。

3. 推理机

推理机：用来控制和协调规则库与综合数据库的运行，包含了控制策略和推理方式。

2.4语义网络表示法

从结构上来看，语义网络一般由一些最基本的语义单元组成。这些最基本的语义单元被称为语义基元(基本语义联系)，可用如下三元组来表示为　(节点1，弧，节点2)

可用如图2.3所示的有向图来表示。其中A 和B 分别代表节点，而R则表示A和B之间的某种语义联系(语义关系)。

当把多个语义基元用相应的语义联系关联在一起的时候，就形成了一个语义网络。如图2.4所示。

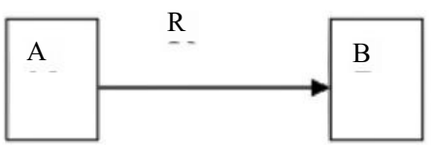


图2.3语义基元结构

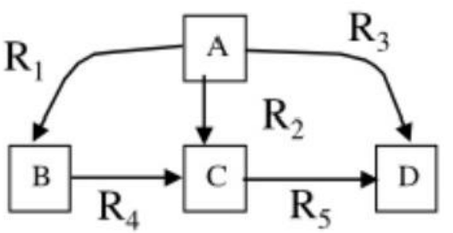


图2.4语义网络结构

2.4语义网络表示法

2.4.2语义网络的基本语义联系

1. 类属关系

类属关系是指具体有共同属性的不同事物间的分类关系、成员关系或实例关系，它体现的是“具体与抽象”、“个体与集体”的层次分类。其直观意义是“是一个”，“是一种”，“是一只”.....。在类属关系中，其一个最主要特征是属性的继承性，处在具体层的结点可以继承抽象层结点的所有属性。

2.4语义网络表示法

2.4.2 语义网络的基本语义联系

AKO(A-Kind-of) ：表示一个事物是另一个事物的一种类型。

AMO(A-Member-of) ：表示一个事物是另一个事物的成员。

ISA(Is-a) ：表示一个事物是另一个事物的实例。

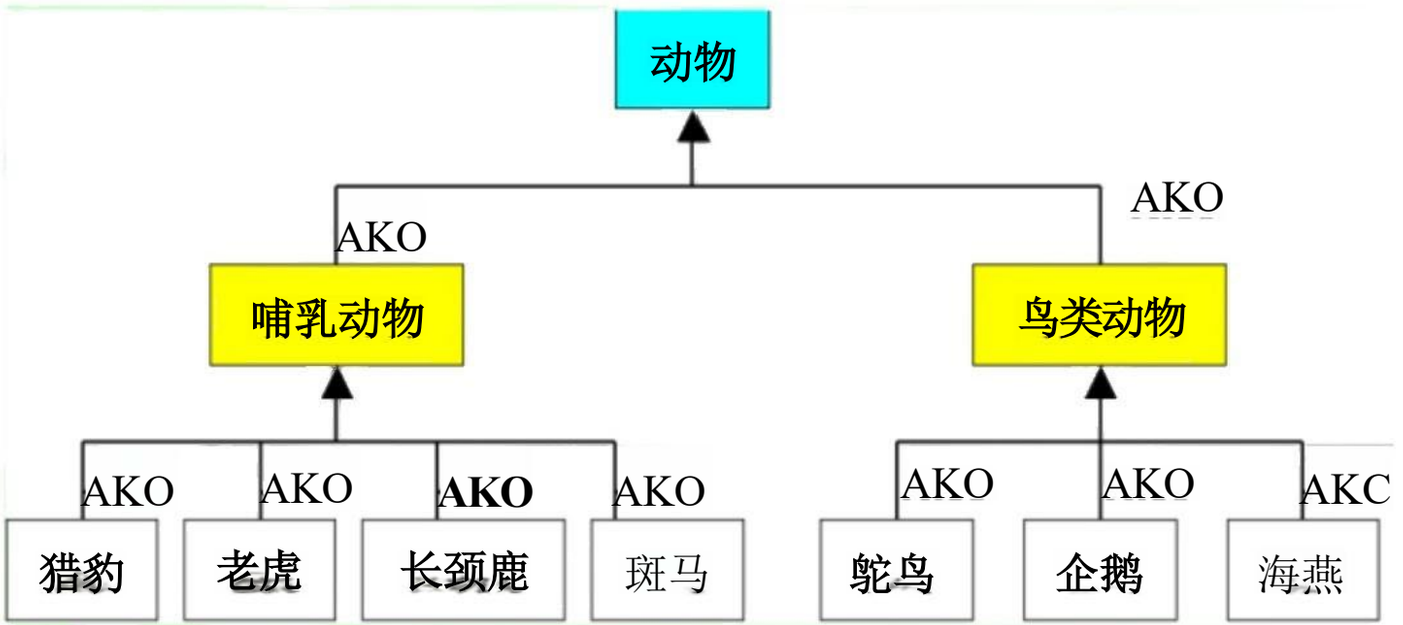


图 AKO 关系实例

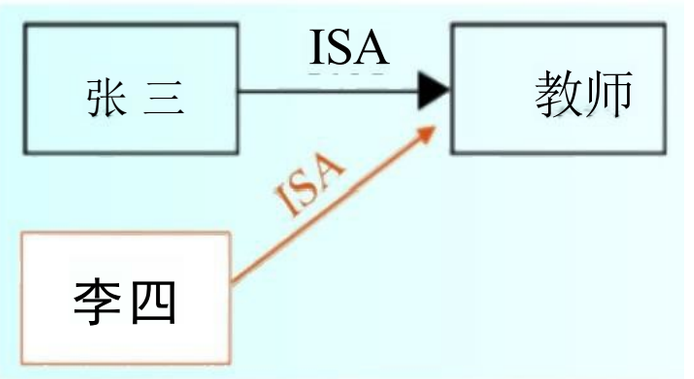


图 ISA 关系实例

2.4语义网络表示法

2.4.2语义网络的基本语义联系

2. 包含关系

包含关系也称为聚集关系，是指具有组织或结构特征的“部分与整体”之间的关系，它和类属关系的最主要的区别就是包含关系一般不具备属性的继承性。常用的包含关系的有：Part-of,Member-of,　　表示一个事物是另一个事物的一部分。Part-of联系不具备属性的继承性。例如，“轮胎是汽车的一部分”其语义网络表示如图2.7所示。

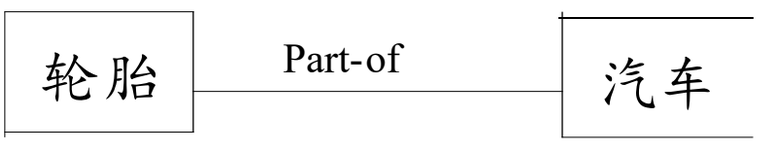


图2.7包含关系实例

2.4语义网络表示法

2.4.2语义网络的基本语义联系

3.属性关系

属性关系是指事物和其属性之间的关系。常用的属性关系有:

Have: 表示一个结点具有另一个结点所描述的属性。

Can: 表示一个结点能做另一个结点的事情。

例如,“鸟有翅膀”,“电视机可以放电视节目”。其对应的语义网络表示如图2.8所示。

图2.8属性关系实例

2.4语义网络表示法

2.4.3语义网络表示知识的方法

1. 事实性知识的表示

对于一些简单的事实，例如“鸟有翅膀”，“轮胎是汽车的一部分”，这里要描述这些事实需要两个节点，用前面给出的基本语义联系或自定义的基本语义联系就可以表示了。

对于稍微复杂一点的事实，比如在一个事实中涉及到多个事物时，就需要更多的语义网络。

通常把有关一个事物或一组相关事物的知识用一个语义网络来表示。

2.4语义网络表示法

例如：用一个语义网络来表示事实“苹果树是一种果树，果树又是树的一种，树有根、有叶”。

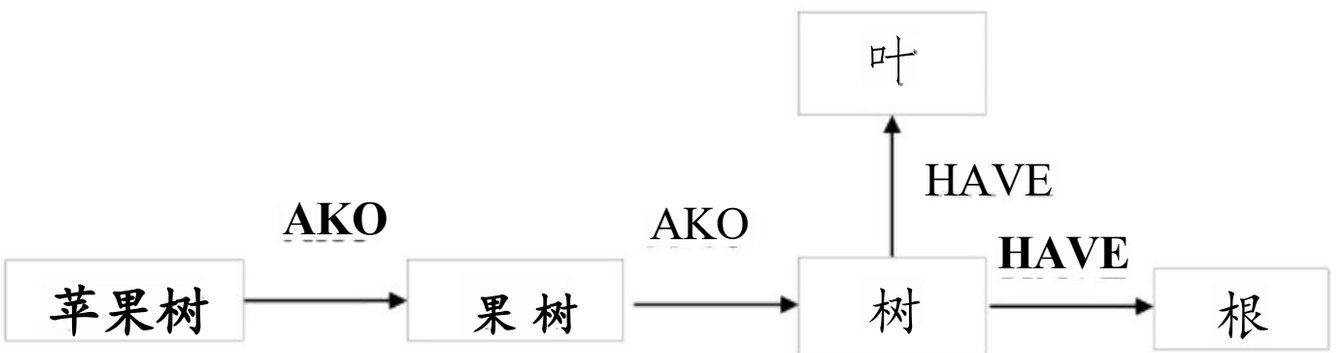


图2.14 有关苹果树的语义网络

2.4语义网络表示法

2. 情况、动作和事件的表示

为了描述那些复杂的知识，在语义网络的知识表示法中，通常采用引进附加节点的方法来解决。西蒙（Simon） 在提出的表示方法中增加了情况节点、动作节点和事件节点，允许用一个节点来表示情况、动作和事件。

(1) 情况的表示

在用语义网络表达那些由不及物动词表示的语句或没有间接宾语的及物动词表示的语句时(祈使句、动作只关联单一个体)，如果该语句的动作表示了一些其它情况，如动作作用的时间等，则需要增加一个情况节点用于指出各种不同的情况。

2.4语义网络表示法

例如： 用语义网络表示知识“请在2006年6月前归还图书”。

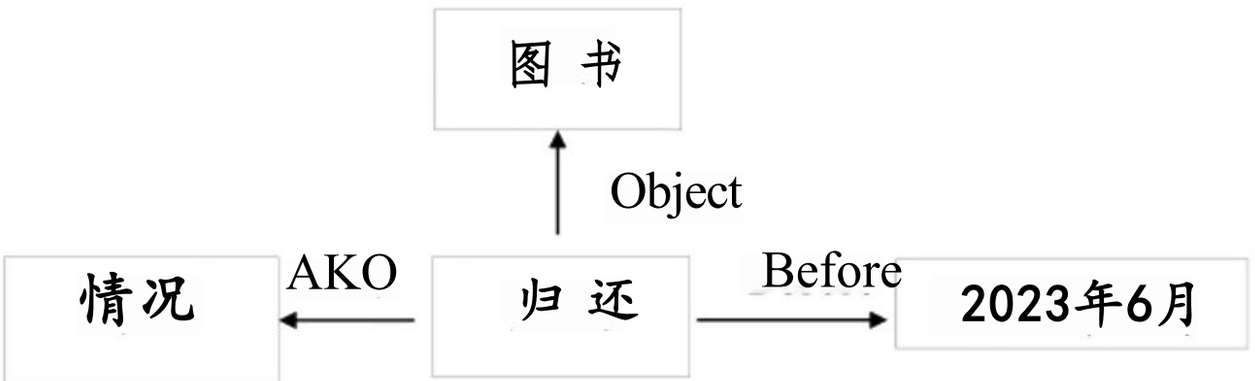


图2.15带有情况节点的语义网络

2.4语义网络表示法

有些表示知识的语句既有发出动作的主体，又有接受动作的客体。在用语义网络表示这样的知识时，可以增加一个动作节点用于指出动作的主体和客体。

例如：用语义网络表示知识“校长送给李老师一本书”。这条知识涉及到3个对象就是“书”、“校长”、“李老师”，为了表示这个事实，增加一个“送给”节点。其语义网络表示如图2.16所示。

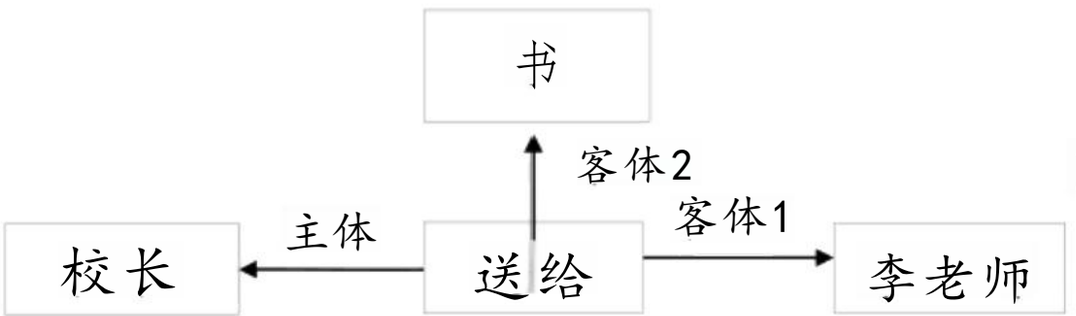


图2.16带有动作节点的语义网络

2.4语义网络表示法

(3) 事件的表示

如果要表示的知识可以看成是发生的一个事件，那么可以增加一个事件节点来描述这条知识。

例如：用语义网络表示知识“中国队与日本队两国的国家女子足球队在中国进行一场比赛，结局的比分是3 :2”。其语义网络表示如图2.17所示。

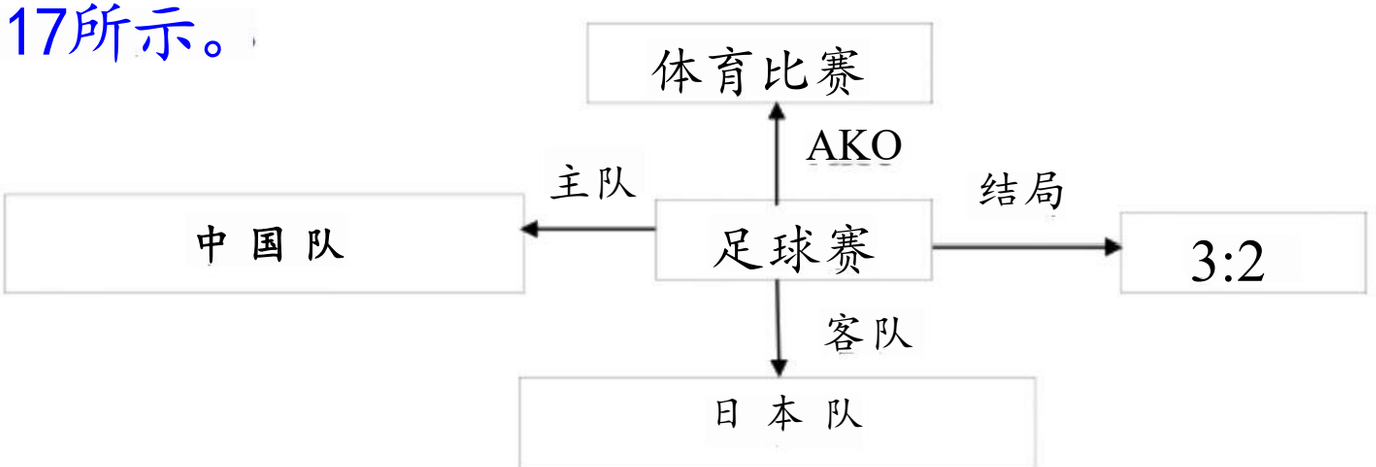


图2.17带有事件节点的语义网络

2.4语义网络表示法

3. 连词和量词的表示(逻辑关系的表示)

在稍微复杂一点的知识中，经常用到象：

“并且”

“或者”

“所有的”

“有一些”

等这样的联结词或量词，在谓词逻辑表示法中，很容易就可以表示这类知识。

而谓词逻辑中的连词和量词可以用语义网络来表示。因此，语义网络也能表示这类知识。

2.4语义网络表示法

3. 连词和量词的表示

(2)存在量词与全称量词的表示

对存在量词可以直接用“AKO”、“ISA”等语义关系来表示。

对全称量词可以采用亨德里克(G.G.Hendrix)提出的语义网络分区技术来表示，也称为分块语义网络(Partitioned Semantic Net)，以解决量词的表示问题。

该技术的基本思想是：把一个复杂的命题划分成若干个子命题，每一个子命题用一个简单的语义网络来表示，称为一个子空间，多个子空间构成一个大空间。每个子空间看作是大空间中的一个节点，称为超节点。空间可以逐层嵌套，子空间之间用弧相互连接。

2.4语义网络表示法

存在和全称量词的表示(1/4)

- 例2-19 用语义网络表示如下事实：

“每个学生都学习了一门程序设计语言”

为表示语义网络，定义如下节点：

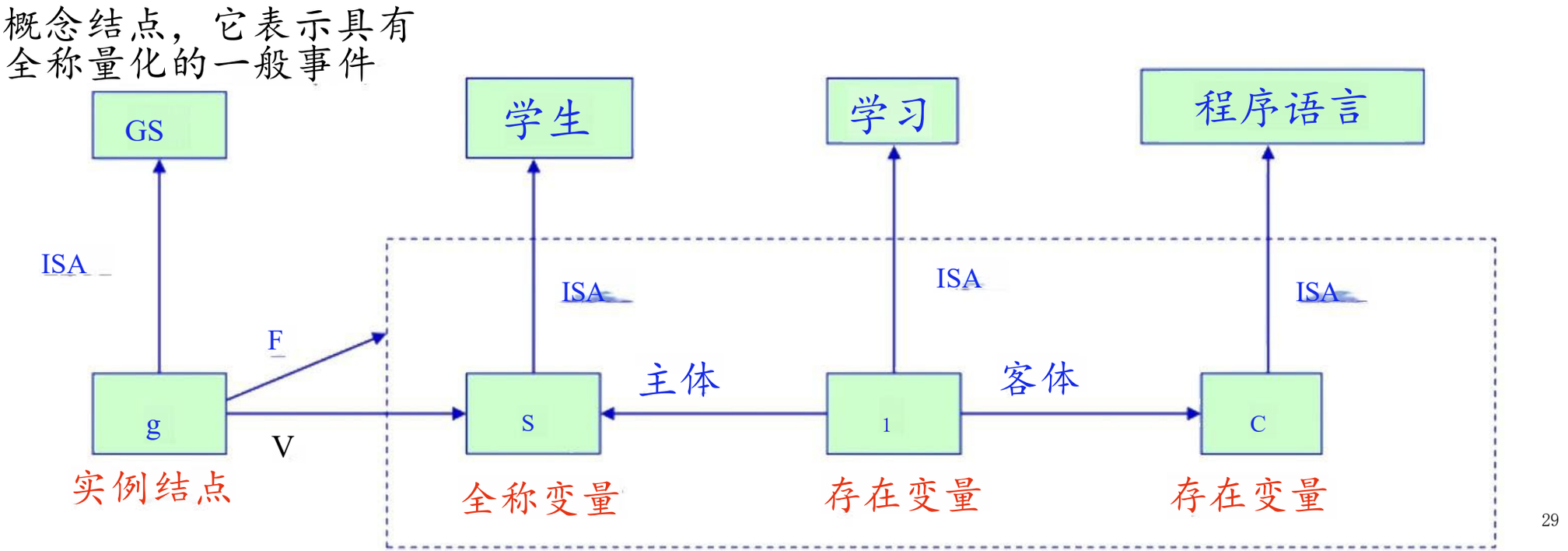
- GS是一个概念结点，它表示具有全称量化的一般事件。
- g是一个实例结点，代表GS 中的一个具体例子，如上所提到的事实。
- s是一个全称变量，表示任意一个学生。
- 1是一个存在变量，表示某一次学习。
- c是一个存在变量，表示某一门程序设计语言。

这样，s、1、c之间的语义联系就构成一个子空间，它表示对每一个学生s，都存在一个学习事件1和一门程序设计语言c。

2.4语义网络表示法

存在和全称量词的表示(2/4)

- 在从结点g引出的三条弧中，弧“ISA”说明结点g是GS 中一个实例；弧“F”说明它所代表的子空间及其具体形式；弧“v” 说明它所代表的全称量词。

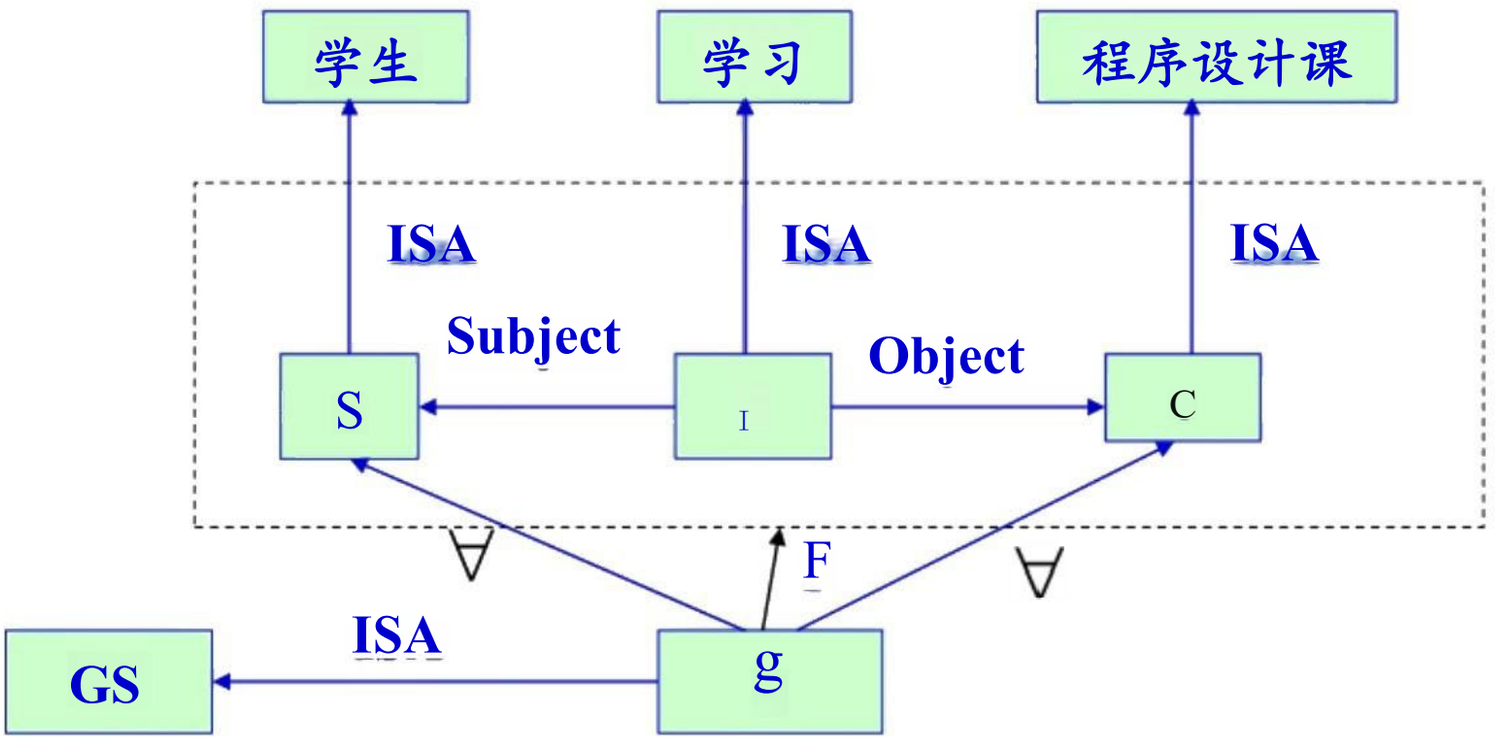


每一个全称量词都需要一条弧，子空间中有多少个全称量词，就需要有多少条弧。

2.4语义网络表示法

存在和全称量词的表示(3/4)

- 每一个全称量词都需要一条这样的弧，子空间中有多少个全称量词，就需要有多少条这样的弧。
- 例2-19用语义网络表示事实：
- “每个学生都学习了所有的程序设计课程”
- 其语义网络如下图所示。其中，结点g有两条指向全称变量的弧。



第3章确定性推理

本章主要内容：

3.1 推理的基本概念

3.2推理的逻辑基础

3.3自然演绎推理

3.4归结演绎推理

3.2.2谓词公式的永真性和可满足性

附1：谓词演算的基本等价式

- 双重否定律(double negation law):

  7(P(x))=P(x)

- 德·摩根定律(Mogen law):

  7(P(x)VQ(x))=┐P(x)A-Q(x)

  7(P(x))AQ(x))=7 P(x)V7Q(x)

- 逆否律(inverse-negation law):

  P(x)→Q(x)=7Q(x)→┐P(x)

- 分配律(assignment law):

  P(x)A(Q(x)VR(x))=(P(x)AQ(x)V(P(x)△R(x))

  P(x)V(Q(x)△R(x))=(P(x)VQ(x))A(P(x)VR(x))

3.2.2谓词公式的永真性和可满足性

附1:谓词演算的基本等价式

- 结合律(association law):

  (P(x)AQ(x))△R(x)=P(x)A(Q(x)△R(x))

  (P(x)VQ(x))VR(x)=P(x)V(Q(x)VR(x))

- 蕴含等价式(implication law):

  P(x)→Q(x)三P(x)VQ(x)

- 易名规则(rename law):

  VxP(x)VVxQ(x)=VxP(x)VVyQ(y)

- 量词转换律(quantifier transform law):

  7 VxP(x)=3x-P(x)

  73xP(x)=Vx-P(x)

3.2.2谓词公式的永真性和可满足性

附1:谓词演算的基本等价式

- 量词分配律(quantifier assignment law):

  3x(P(x)VQ(x))=3xP(x)V3xQ(x)

  Vx(P(x)AQ(x))=VxP(x)AVxQ(x)

  Vx(P→Q(x))=P→VxQ(x)

  3x(P→Q(x))=P→3xQ(x)

- 量词交换律(quantifier commutative law):

  VxVy(P(x,y))=VyVx(P(x,y))

  3x3y(P(x,y))=3y3x(P(x,y))

  3xVy(P(x,y))→Vy3x(P(x,y))

  Vx3y(P(x,y))→3y3x(P(x,y))

3.4.1 子句型

1. 子句与子句集

- 定义3.12 原子谓词公式及其否定统称为文字。
- 例如，P(x)、Q(x)、-P(x)、7Q(x) 等都是文字。
- 定义3.13 任何文字的析取式称为子句。
- 例如，P(x)VQ(x),P(x,f(x))VQ(x,g(x)) 都是子句。
- 定义3.14 不含任何文字的子句称为空子句。
- 由于空子句不含有任何文字，也就不能被任何解释所满足，因此空子句是永假的，不可满足的。
- 空子句一般被记为□或NIL。
- 定义3.15 由子句或空子句所构成的集合称为子句集。

3.4.1 子句型

- 在谓词逻辑中，任何一个谓词公式都可以通过应用等价关系及推理规则化成相应的子句集。其化简步骤如下：

- (1) 消去连接词“→”和“→”

反复使用如下等价公式：

$P \rightarrow Q \Leftrightarrow \neg P \vee Q$

$P \rightarrow Q \Leftrightarrow (P \wedge Q) \vee (\neg P \wedge \neg Q)$

即可消去谓词公式中的连接词“→”和“→”。

- 例如公式

$(\forall x)\{(\forall y)P(x,y) \rightarrow \neg(\forall y)[Q(x,y) \rightarrow R(x,y)]\}$

经等价变化后为

$(\forall x)\{\neg(\forall y)P(x,y) \vee \neg(\forall y)[\neg Q(x,y) \vee R(x,y)]\}$

3.4.1子句型

2. 子句集的化简(2/6)

(2) 减少否定符号的辖域

将每个否定符号“—”移到紧靠谓词的位置，使得每个否定符号最多只作用于一个谓词上。

反复使用双重否定率：7(-P)⇔P

德摩根定律：7(PAQ)⇔¬ PV-Q;7(PVQ)⇔¬ PA-Q

量词转换率

7(Vx)P(x)→(3x)7P(x)

7(3x)P(x)⇔(Vx)7P(x)

例如，(Vx){7(Vy)P(x,y)V-(Vy)[-Q(x,y)VR(x,y)]} 经等价变换后为：

(Vx){(3y)¬ P(x,y)V(3y)[Q(x,y)A¬ R(x,y)]}

3.4.1子句型

2. 子句集的化简(3/6)

- (3) 对变元标准化

在一个量词的辖域内，把谓词公式中受该量词约束的变元全部用另外一个没有出现过的任意变元代替，使不同量词约束的变元有不同的名字。

例如，上式经变换后为 (Vx){(3y)-P(x,y)V(3y) [Q(x,y)A-R(x,y)]}

(Vx){(3y)┐P(x,y)V(3z) [Q(x,z)A┐R(x,z)]}

(4) 化为前束范式

化为前束范式的方法：把所有量词都移到公式的左边，并且在移动时不能改变其相对顺序。由于第(3)步已对变元进行了标准化，每个量词都有自己的变元，这就消除了任何由变元引起冲突的可能，因此这种移动是可行的。

例如，上式化为前束范式后为

(Vx)(3y)(3z) {┐P(x,y)V [Q(x,z)A┐R(x,z)]}

3.4.1子句型

(5) 消去存在量词

消去存在量词时，需要区分以下两种情况：

若存在量词不出现在全称量词的辖域内(即它的左边没有全称量词)，只要用一个新的个体常量替换受该存在量词约束的变元，就可消去该存在量词。

若存在量词位于一个或多个全称量词的辖域内，例如

$(Vx_1)..(Vx_0)(3y)P(x_1,X_2.....,Xny)$

则需要用Skolem函数$f(x_1,X_2 \ldots, xn)$替换受该存在量词约束的变元y，然后再消去该存在量词。

例如，上步所得公式中存在量词(3y)和(3z)都位于(Vx)的辖域内，因此都需要用Skolem函数来替换。设替换y和z的Skolem函数分别是f(x)和g(x)，则替换后的式子为 (Vx)(3y)(3z){7 P(x,y)V[Q(x,z)A **-R(x,z)**]}

2. 子句集的化简(5/6)

(Vx) {7 P(x,f(x)V[Q(x,g(x)△-R(x,g(x))]}

3.4.1 子句型

(6) 化为Skolem 标准型

Skolem标准型的一般形式为

$$(Vx_1)...(Vxn)M(x_1,X_2.....,Xn)$$

其 中 ， M(x_1,X_2,...,Xn) 是Skolem标准形的母式，它由子句的合取所构成。把谓词公式化为Skolem 标准形需要使用以下等价关系

$$PV(Q△R)→(PVQ)A(PVR)$$

例 如 ， 前面的公式化为Skolem 标准形后为

(Vx){[-P(x,f(x))VQ(x,g(x)]△[P-(x,f(x))V -R(x,g(x))]}

(7) 省掉全称量词

由于母式中的全部变元均受全称量词的约束，并且全称量词的次序已无关紧要，因此可以省掉全称量词。但剩下的母式，仍假设其变元是被全称量词量化的。 例如，上式消去全称量词后为

[-P(x,f(x))VQ(x,g(x)]A[-P(x,f(x))V -R(x,g(x))]

3.4.1 子句型

(8) 根据合取词拆分公式

在母式中消去所有合取词，把母式用子句集的形式表示出来。其中，子句集中的每一个元素都是一个子句。

例如，上式的子句集中包含以下两个子句

P(x,f(x)VQ(x,g(x)),

P(x,f(x)V-R(x,g(x)).

(9) 更换变量名称

对子句集中的某些变量重新命名，使任意两个子句中不出现相同的变量名。由于每一个子句都对应着母式中的一个合取元，并且所有变元都是由全称量词量化的，因此任意两个不同子句的变量之间实际上不存在任何关系。这样，更换变量名是不会影响公式的真值的。

例如，对前面的公式，可把第二个子句集中的变元名x更换为y，得到如下子句集：7P(x,f(x)VQ(x,g(x),┐P(y,f(y))V┐R(y,g(y))

化简谓词公式

- ①消去连接词“→”和“→”；
- ②减少否定的辖域(内移否定符号)；
- ③变量标准化(变量换名)；
- ④化为前束范式；
- ⑤消去存在量词；
- ⑥化为Skolem标准型；
- ⑦省掉全称量词
- ⑧根据合取词拆分公式；
- ⑨更换变量名称

3.4.2鲁滨逊归结原理

- 鲁滨逊归结原理基本思想

- **首 先**把欲证明问题的结论否定，并加入子句集，得到一个扩充的子句集S'。然 后设法检验子句集S'是否含有空子句，若含有空子句，则表明S'是不可满足的；若不含有空子句，则继续使用归结法，在子句集中选择合适的子句进行归结，直至导出空子句或**不能**继续归结为止。

- 鲁滨逊归结原理包括：(1)命题逻辑归结原理；

  (2)谓词逻辑归结原理

定理3.5子句集S是不可满足的，当且仅当存在一个从S到空子句的归结过程。

3.4.2鲁滨逊归结原理

谓词逻辑的归结

- 在谓词逻辑中，由于子句集中的谓词一般都含有变元，因此不能像命题逻辑那样直接消去互补文字。而需要先用一个最一般合一 (MGU) 对变元进行代换，然后才能进行归结。

- 例3.9 设C_1=P(a)VR(x), C_2=—P(y)VQ(b), 求 C_12

- 解：取L_1=P(a), L_2=—P(y), 则L_1和L_2不考虑互补性时的最一般合一是σ={a/y}。根据定义4.17, 可得

$$C12=(\{C_1\sigma\}-\{L_1\sigma\})U(\{C_2\sigma\}-\{L_2\sigma\})$$

$$=(\{P(a),R(x)\}-\{P(a)\})U(\{—P(a),Q(b)\}-\{—P(a)\})$$

$$=(\{R(x)\})U(\{Q(b)\})=\{R(x),Q(b)\}$$

$$=R(x)VQ(b)$$

- 例3.18“激动人心的生活”问题
- 假设：所有不贫穷并且聪明的人都是快乐的，那些看书的人是聪明的。李明能看书且不贫穷，快乐的人过着激动人心的生活。
- 求证：李明过着激动人心的生活。

解： 先定义谓词：

Poor(x)x　是贫穷的

Smart(x)x　是聪明的

Happy(x)x　是快乐的

Read(x)x　能看书

Exciting(x)x　过着激动人心的生活

- 再将问题用谓词表示如下：
- “所有不贫穷并且聪明的人都是快乐的”
- (Vx)((—Poor(x)△Smart(x))→Happy(x))
- “那些看书的人是聪明的”
  (Vy)(Read(y)→Smart(y))
- “李明能看书且不贫穷”
- Read(Liming)A—Poor(Liming)
- “快乐的人过着激动人心的生活”
- (Vz)(Happy(z)→Exciting(z))
- 目标“李明过着激动人心的生活”的否定
  —Exciting(Liming)

- 将上述谓词公式转化为子句集如下：

· (1)Poor(x)V—Smart(x)VHappy(x)

· (2)—Read(y)VSmart(y)

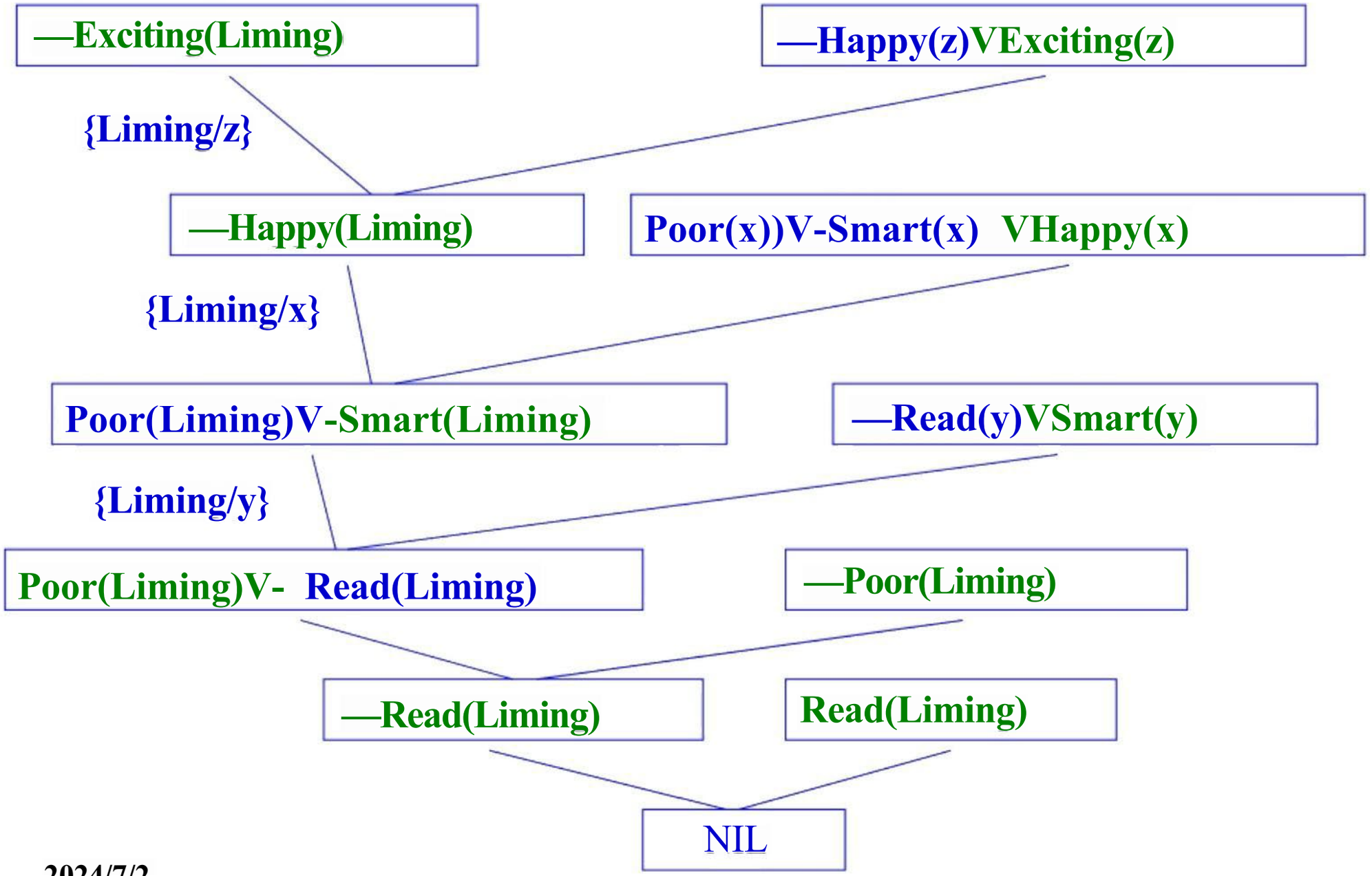
· (3)Read(Liming)

· (4)—Poor(Liming)

· (5)—Happy(z)VExciting(z)

- (6)—Exciting(Liming)(结论的否定)

2. 谓词逻辑的归结


—Exciting(Liming)
—Happy(z)VExciting(z)
{Liming/z}
—Happy(Liming)
Poor(x))V-Smart(x) VHappy(x)
{Liming/x}
Poor(Liming)V-Smart(Liming)
—Read(y)VSmart(y)
{Liming/y}
Poor(Liming)V- Read(Liming)
—Poor(Liming)
—Read(Liming)
Read(Liming)
NIL

第4章搜索策略

本章主要内容：

4.1 引言

4.2状态空间搜索

4.3盲目搜索

4.4启发式搜索

4.5问题规约

4.6博弈

2024/7/2

4.1引言

搜索可以根据是否使用启发式信息分为

(1)盲目搜索(也叫非启发式搜索)

盲目搜索是指在问题的求解过程中，不运用启发式信息，只按照一般的逻辑法则或控制性知识，在预定的控制策略下进行搜索，在搜索过程中获得的中间信息不用来改进控制策略。

典型的盲目搜索：深度优先搜索、宽度优先搜索。

(2)启发式搜索

启发式搜索是指在问题的求解过程中，为了提高搜索效率，运用与问题有关的启发式知识，即解决问题的策略、技巧、窍门等实践经验与知识，来指导搜索朝着最有希望的方向前进，加速问题求解并找到最优解。

典型的启发式搜索： A 算法、 A*算法。

4.3盲目搜索

4.3.1宽度优先搜索

- 宽度优先搜索(Breadth First Search,BFS)又称为广度优先搜索。

·OPEN 表中节点简单的排序方式：

- 宽度优先——扩展当前节点后生成的子节点总是置于OPEN表的后端，即OPEN表作为队列，先进先出，使搜索优先向横向方向发展。

4.3.1宽度优先搜索

采用队列结构，宽度优先算法可以表示如下：

(1) 把初始节点So, 放入OPEN表。

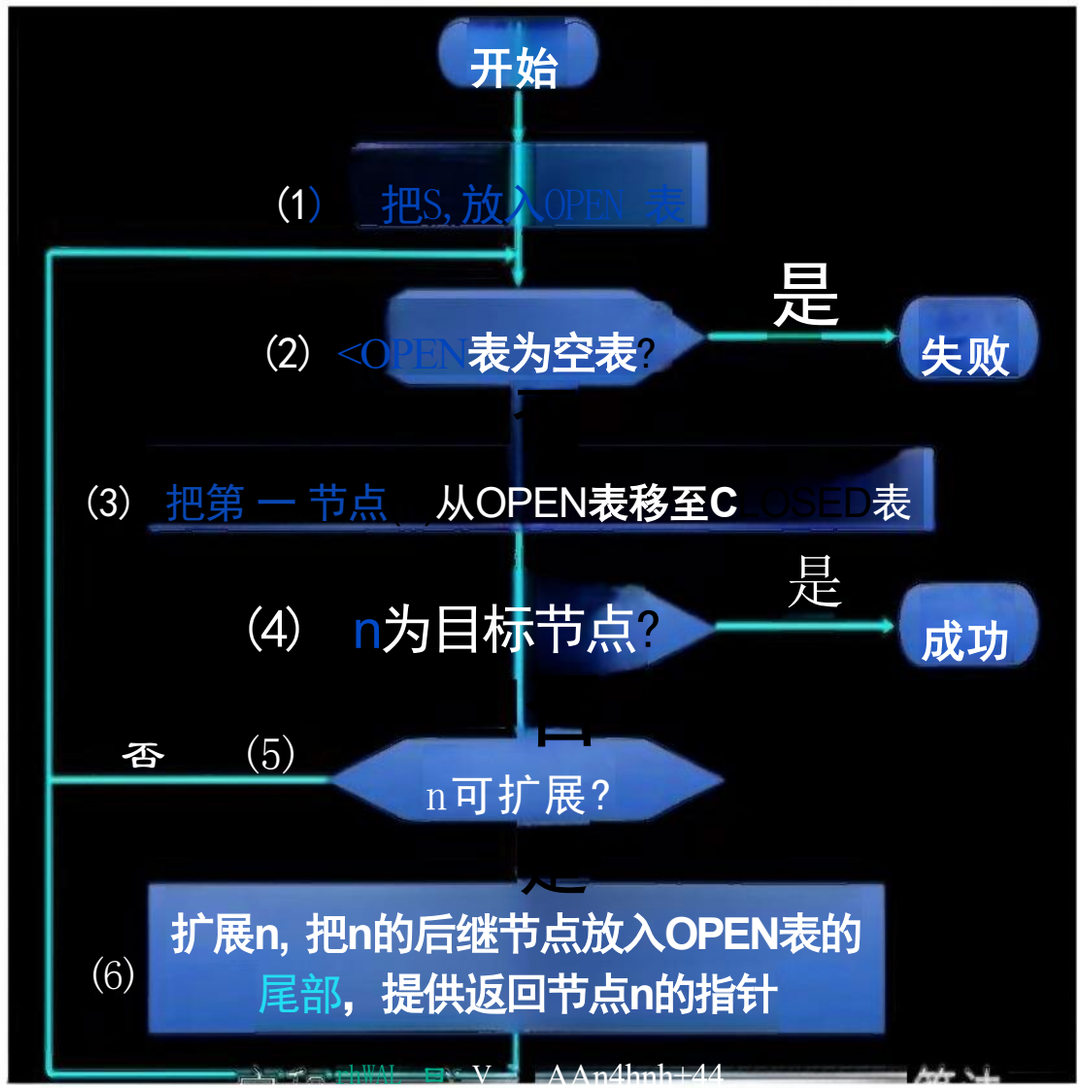
(2) 如果OPEN表为空。则问题无解，失败并退出。

(3) 把OPEN表中的第一个节点取出放入CLOSE表中，并按顺序冠以编号n;

(4) 考察节点n是否为目标节点。若是，则求得了问题的解，成功并退出。

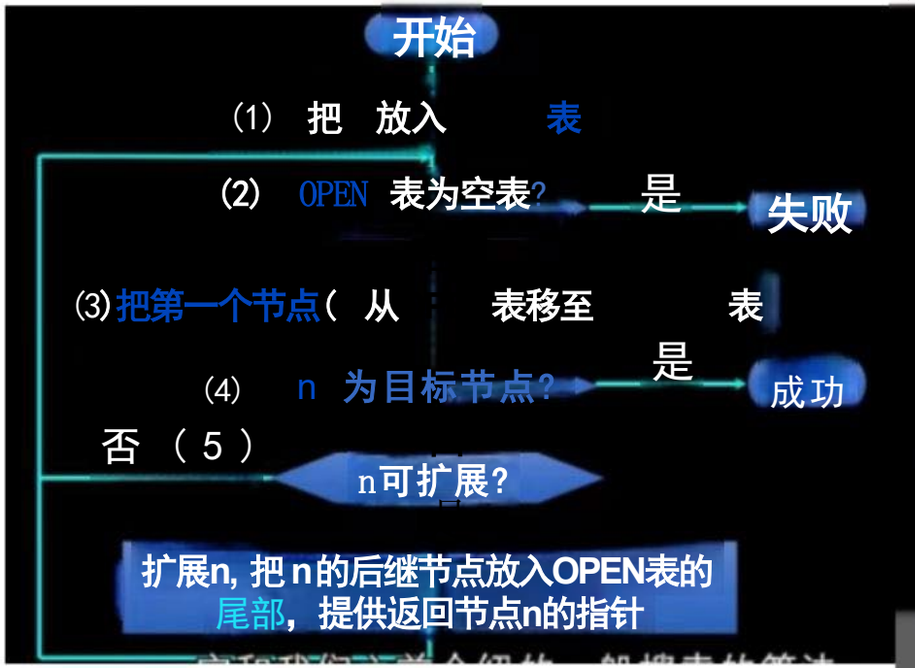
(5) 若节点n不可扩展，则转第(2)步；

(6) 否则扩展节点n，将其子节点放到OPEN表的尾部，并为每一个子节点都配置指向父节点的指针，然后转第(2)步。

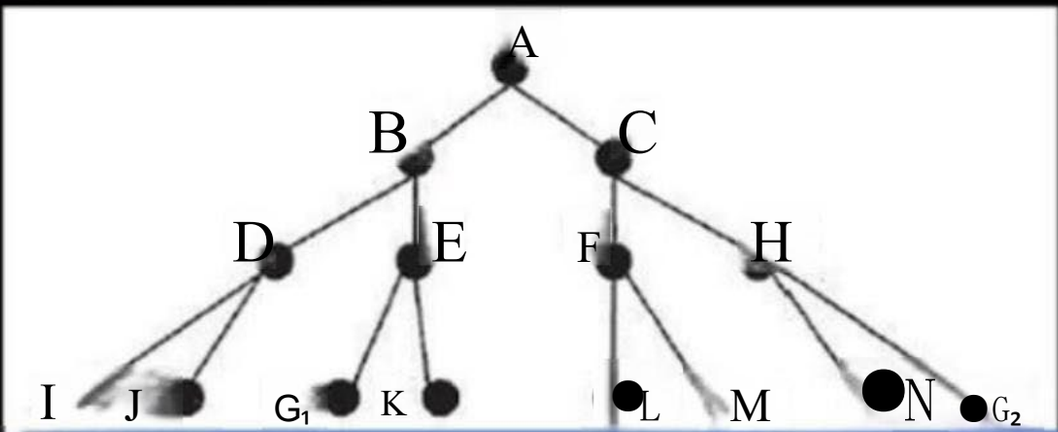


4.3.1宽度优先搜索

采用队列结构，宽度优先算法可以表示如下：



广度



OPEN=[A]; CLOSED=[]

OPEN =[B,C]; CLOSED =[A]

OPEN =[C,D,E]; CLOSED =[B,A]

OPEN =[D,E.F.H]; CLOSED=[C,B,A]

OPEN =[E,F,H,J]; CLOSED =[D,C,B,A]

OPEN =[F.H,IJ,G_1K]; CLOSED =[E,D,C,B.A]

OPEN =[H, JG $_1$, KLM 直到我们这里头的G状态

……直到G_1出现在OPEN 表的最左端。

4.3.2深度优先搜索

- 深度优先搜索(Depth First Search,DFS)，是一种一直向下的搜索策略。

·OP EN表中节点的排序方式：

- 深度优先——扩展当前节点后生成的子节点总是置于OPEN表的前端，即OPEN表作为栈，后进先出，使搜索优先向纵深方向发展。

DFS的基本思想

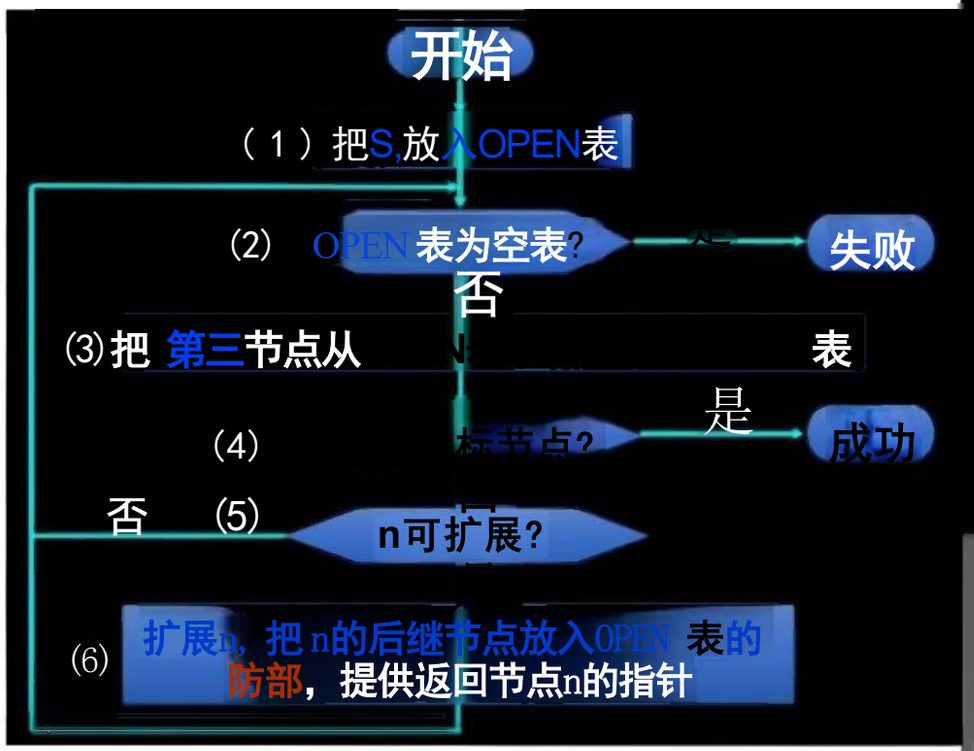
- 深度优先搜索生成节点并与目标节点进行比较是沿着树的最大深度方向进行的，只有当上次访问的节点不是目标节点，而且没有其他节点可以生成的时候，才转到上次访问节点的父节点。
- 转移到父节点后，该算法会搜索父节点的其他的子节点。

DFS的基本思想

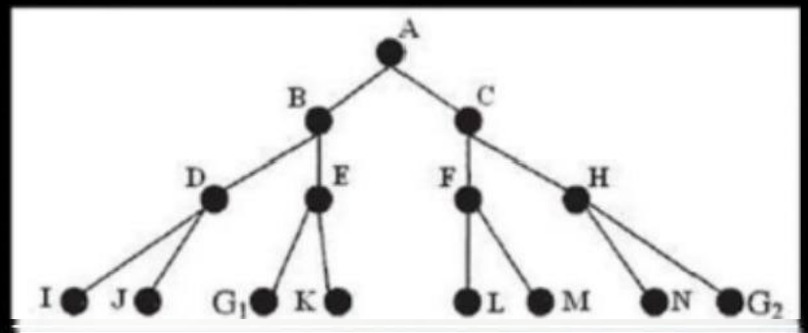
- 深度优先搜索也称为回溯搜索，它总是首先扩展树的最深层次上的某个节点，只是当搜索遇到一个死亡节点(非目标节点而且不可扩展)，搜索方法才会返回并扩展浅层次的节点。
- 上述原理对树中的每一节点是递归实现的，实现该递归过程的比较简单的一种方法是采用栈。

基于栈实现的DFS流程

基于栈实现的深度优先搜索算法：



深度优先搜索的示例



OPEN=[A];	CLOSED=[]
OPEN=[.C];	CLOSED=[A]
OPEN =[D,E,C];	CLOSED=[B.A]
OPEN =[J,E,C];	CLOSED=[D,B.A]
OPEN =[J,E,C]	CLOSED=[I,D,B.A]
OPEN =[E,C];	CLOSED=[.I.D,B.A]
OPEN =[G_1K,C]	CL数据结构，D,B.A]

G_1出现在OPEN表的最左端，将G_1移至CLOSED表，找到解。

4.3.3有界深度搜索和迭代加深搜索

有限深度优先搜索的搜索过程

(1) 把初始节点S_0放入OPEN 表中，置S0的深度d(S0)=0。

(2) 如果OPEN表为空，则问题无解，失败并退出。

(3) 把OPEN 表中的第一个节点取出放入CLOSE 表。按顺序编号n。

(4) 考察节点n是否为目标节点。若是，则求得了问题的解，成功并退出。

(5) 如果节点n的深度d(n)=dm, 则转第(2)步。

(6) 如果节点n不可扩展，则转第(2)步。

(7) 扩展节点n。将其子节点放入OPEN表的首部，并为其配置指向父节点的指针。然后转第(2)步。

4.4启发式搜索

- 前面讨论的各种搜索方法都是按事先规定的路线进行搜索，没有用到问题本身的特征信息，具有较大的盲目性，产生的无用节点较多，搜索空间较大，效率不高。
- 如果能够利用问题自身的一些特征信息来指导搜索过程，则可以缩小搜索范围，提高搜索效率。
- 像这样利用问题自身特征信息来引导搜索过程的方法称为启发式方法。

启发式搜索

其主要功能：用来评估节点重要性。

评估函数

- 评估函数f(x)定义为从初始节点S_0出发，约束地经过节点x到达目标节点Sg的所有路径中最小路径代价的估计值。

- 评估函数的一般形式为：

$$f(x)=g(x)+h(x)$$

其中：g(x)——从初始节点S_0到节点x的实际代价；

h(x)——从x到目标节点Sg的最优路径的评估代价，它体现了问题的启发式信息，其形式要根据问题的特性确定，h(x)称为启发式函数。

4.4.3实现启发式搜索的关键因素和A*算法

- 实现启发式搜索应考虑的关 键 因 素：

  (1)搜索算法的可采纳性(Admissibility)；

  (2)启发式函数h(n)的强弱及其影响。

1. 搜索算法的可采纳性(Admissibility)

- 在存在从初始状态节点到目标状态节点解答路径的情况下，若一个搜索算法总能找到最短(代价最小)的解答路径，

  则称该状态空间中的搜索算法具有可采纳性，也叫最优性。

- 如宽度优先的搜索算法是可采纳的，只是其搜索效率不高。

2. 启发式函数h(n)的强弱及其影响。

- 定义f*(n)=g*(n)+h*(n)
- 评价函数f与f*的比较

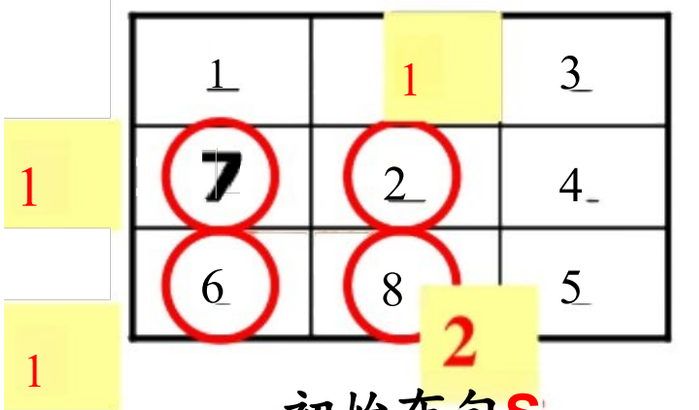
·f(n)、g(n)、h(n)分别是f*(n)、g*(n)、h*(n)的近似值(估计值)

- 可以用h(n)接近h*(n)的程度去衡量启发式函数的强弱。

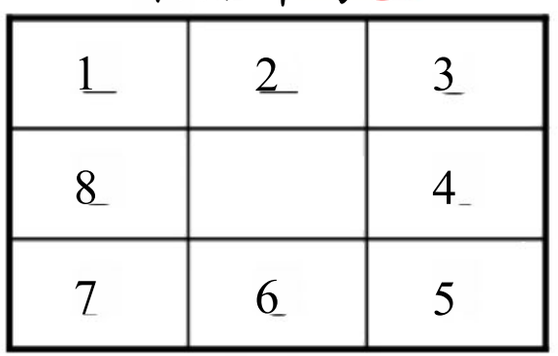
·h(n) 尽可能靠近h*(n)——A*算法的关键。

A* 算法实例——八数码游戏

·f(s) 计算实例



初始布句S



目标布局ng

h*=5

f(s) = d(s)：当前节点深度 + w(S)：错位的棋牌个数

= 0 + 4

= 4

f(s) = d(s)：当前节点深度 + p(S)：各错位棋牌移动距离

= 0 + 5

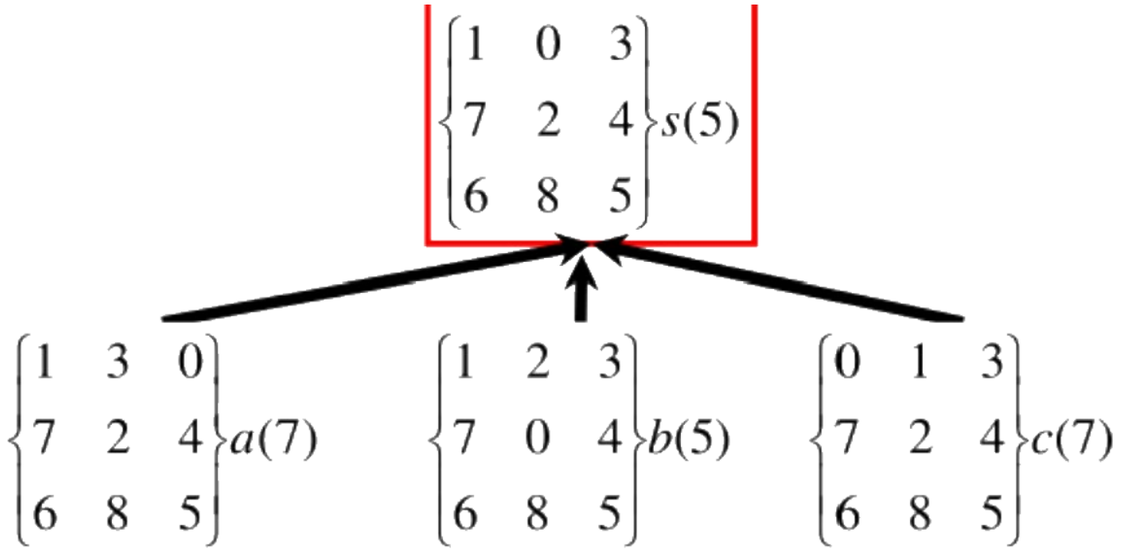
= 5

$$\begin{Bmatrix} 1 & 0 & 3 \\ 7 & 2 & 4 \\ 6 & 8 & 5 \end{Bmatrix} s(5)$$

初始化

OPEN:={s5}

CLOSE:={}



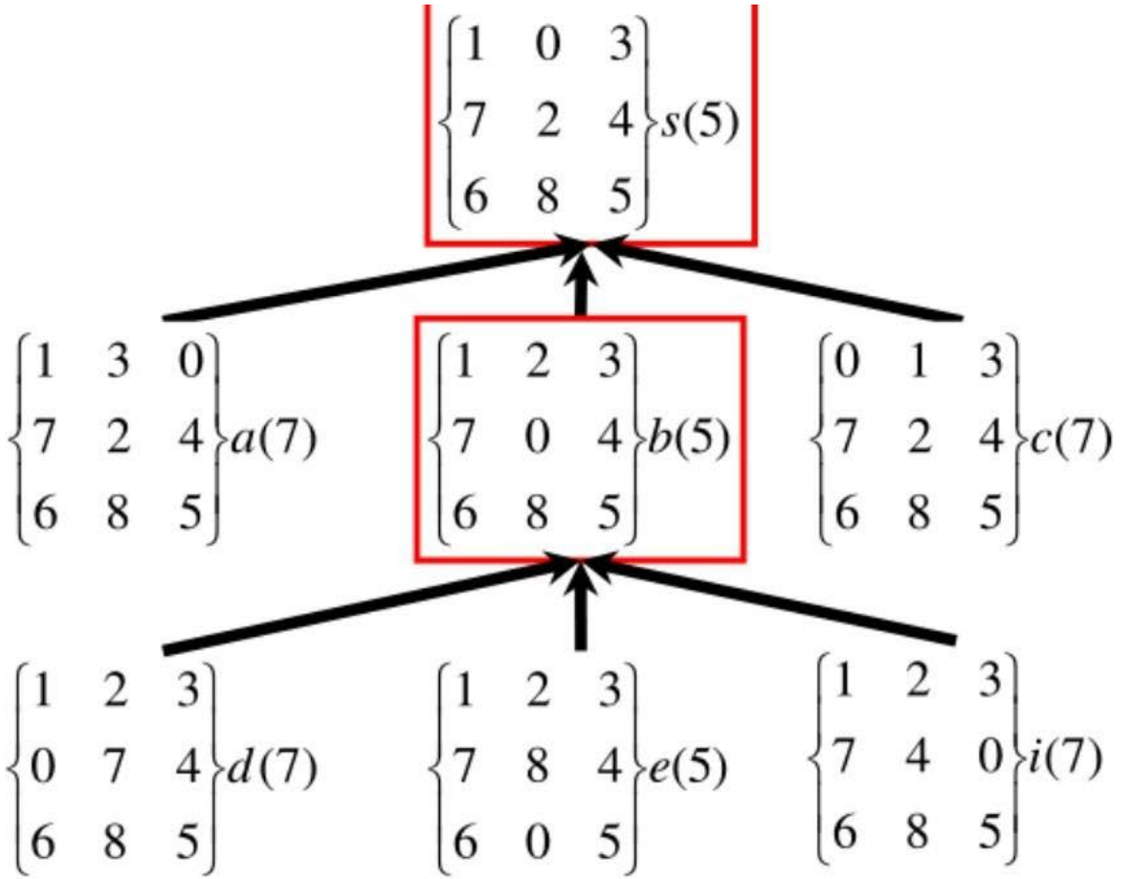
循环1

OPEN:={b5

a7

c7}

CLOSE:={s5}



循 环 2

OPEN:={e5

a7

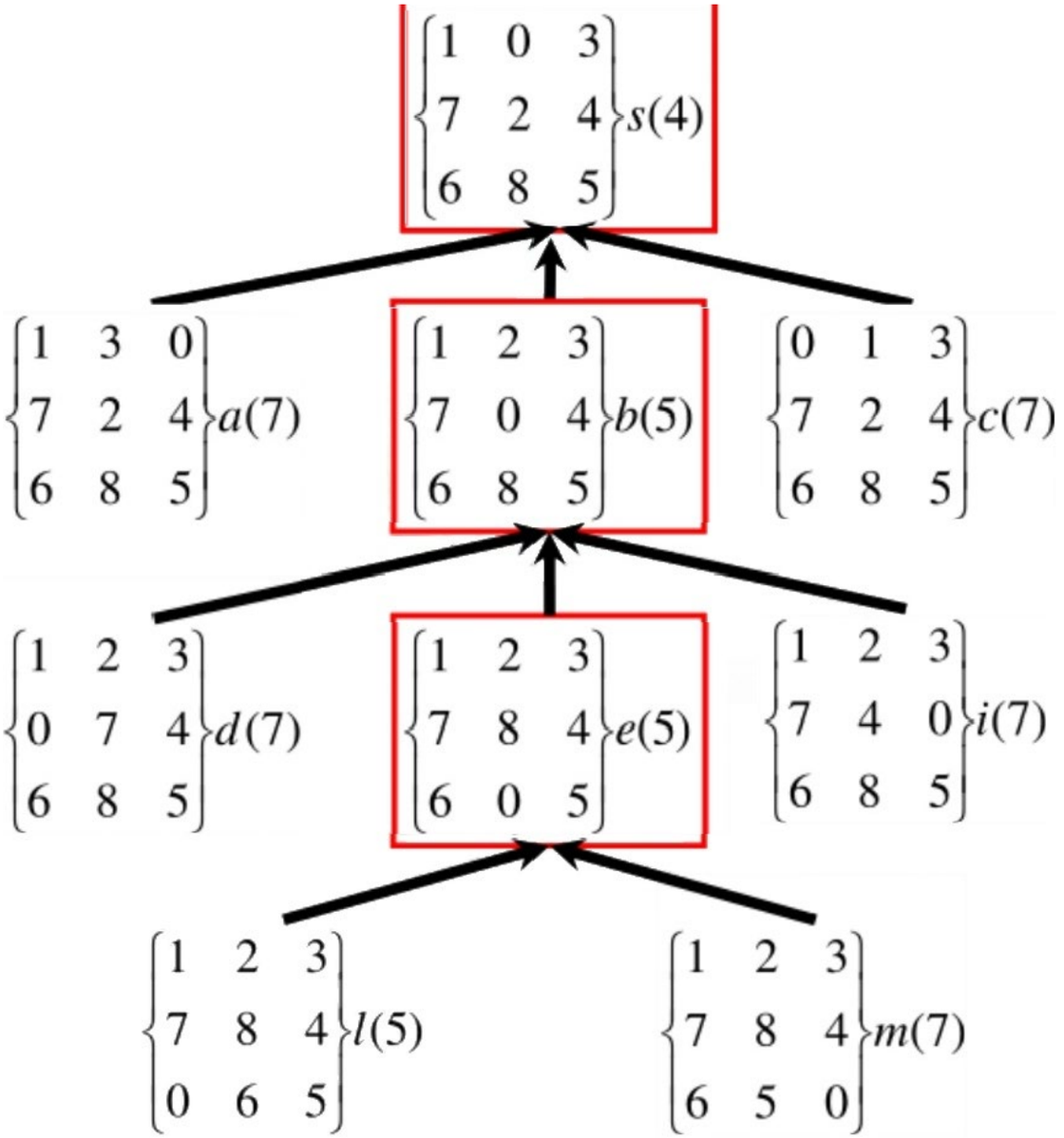
c7

d7

i7 }

CLOSE:={s5

b5 }



循 环 3

OPEN:={15

a7

c7

d7

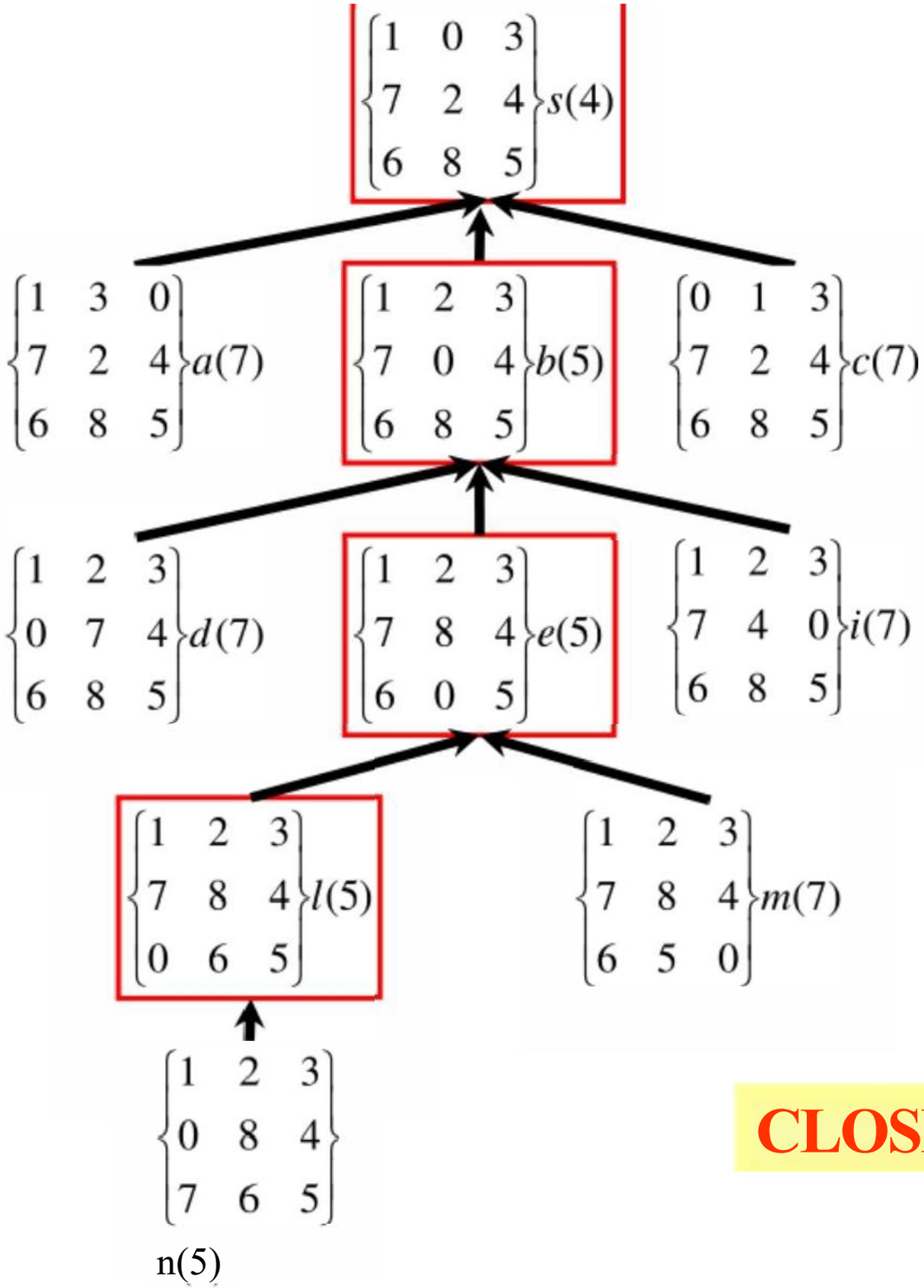
i7

m7}

CLOSE:={s5

b5

e5 }



循环4

OPEN:={n5

a7

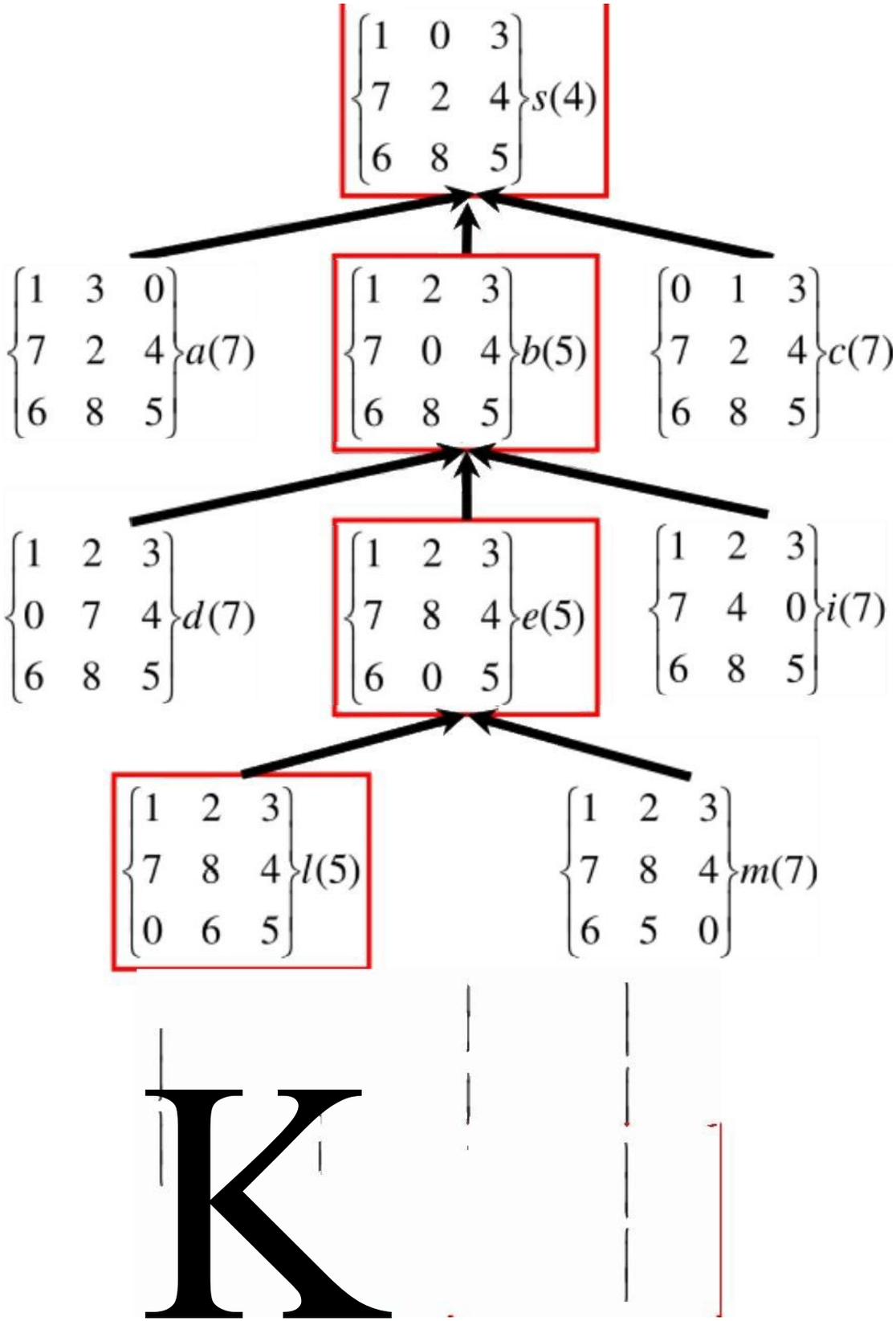
c7

d7

i7

m7}

CLOSE:={s5,b5,e5,15}



循 环 5

OPEN:={g5

a7

c7

d7

订

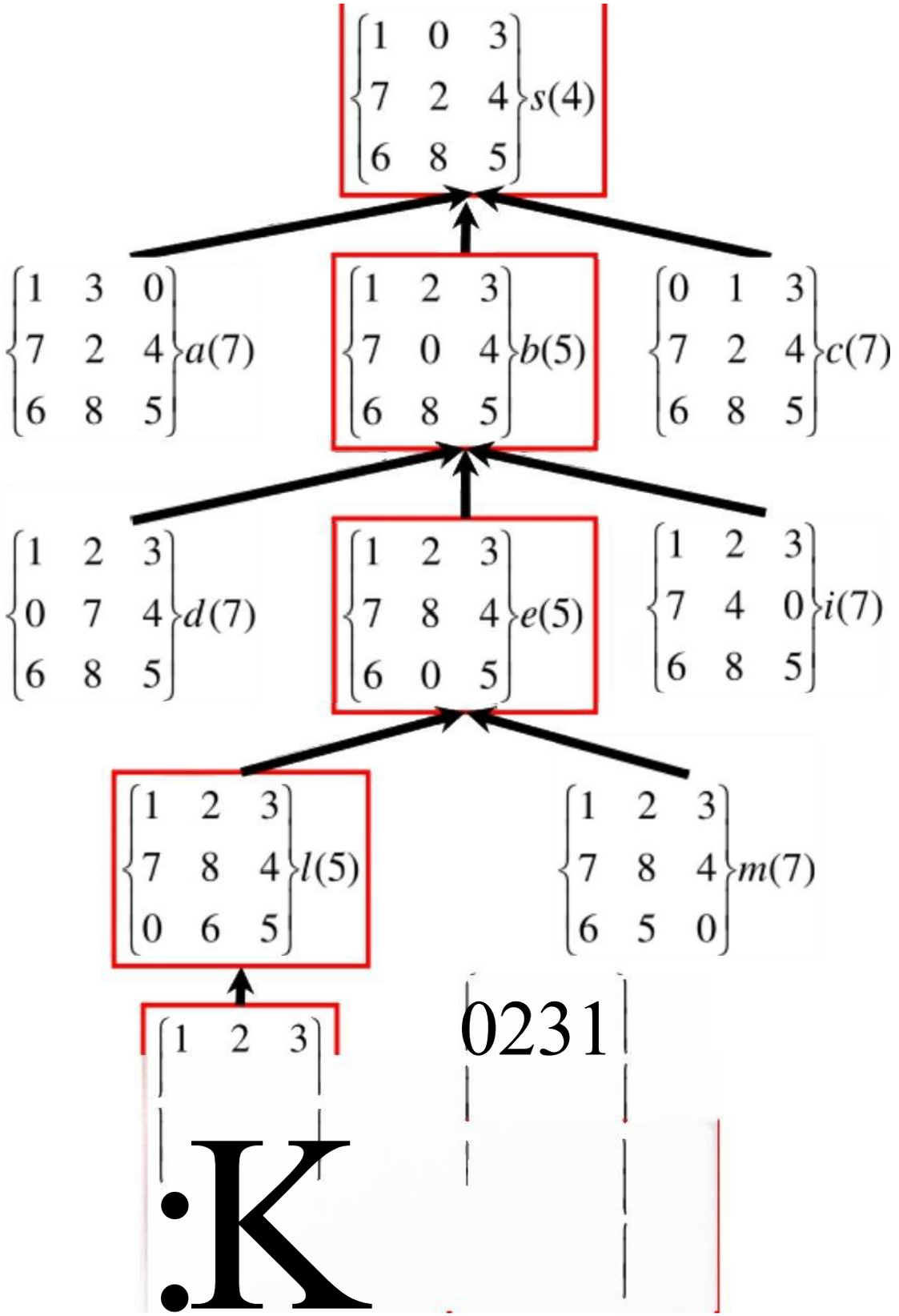
m7

07}

CLOSE:={s5,b5,

e5,15,

n5 }



循 环 6

成功结束

最理想搜索图G

4.6博弈

这里讲的博弈是二人博弈，二人零和、全信息、非偶然博弈，博弈双方的利益是完全对立的。

其对立性主要表现在下列几个方面：

(1) 对垒的双方MAX和MIN轮流采取行动，博弈的结果只能有3种情况：MAX胜、MIN败 ；MAX败 ，MIN胜；平局。

(2) 在对垒过程中，任何一方都了解当前的格局和过去的历史。

(3) 任何一方在采取行动前都要根据当前的实际情况，进行得失分析，选择对自己最为有利而对对方最不利的对策，在不存在“碰运气”的偶然因素，即双方都很理智地决定自己的行动。

什么是博弈树

博 弈 树就是把双人博弈过程用图的形式表示出来，**这样就可以得到一棵AND-OR**树，这种AND-OR树称为博弈树。在 博弈树中，那些下一步该MAX 走的节点称为MAX 节点，而下一步该MIN 走的节点称为MIN 节点。

求解策略

在人工智能中可以采用问题规约搜索法来求解博弈问题。

下面就来讨论博弈中两种最基本的搜索方法：

(1) 极大极小过程 (MINMAX 过程)；

(2) α－β 过程。

与节点倒推取最小，或节点倒推取最大

- 下图所示是向前看两步，共四层的博弈树，用□表示MAX，用○表示MIN，端节点上的数字表示它对应的估价函数的值。在MIN处用圆弧连接，用0表示其子节点取估值最小的格局。

图中节点处的数字，在端节点是估价函数的值，称它为静态值，在MIN处取最小值，在　　处取最大值。

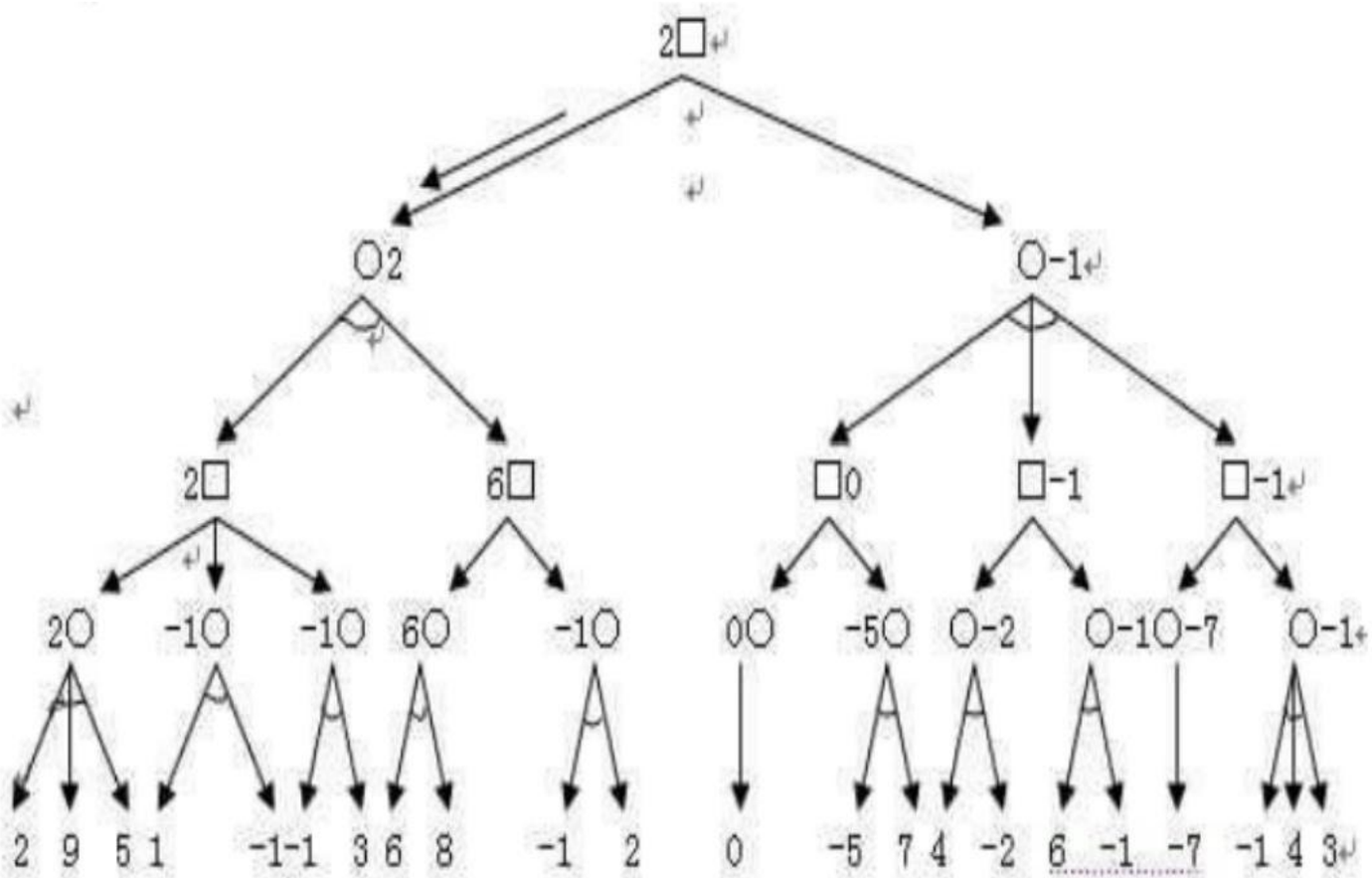


图3-18四层博弈树

第5章不确定性推理

5.1概述

5.2概率方法

5.3主观Bayes 方法

5.4可信度方法

5.5证据理论

5.3主观Bayes方法

1. 规则不确定性

在主观Bayes方法中，规则的不确定性知识可表示为：

IF E THEN (LS,LN)H

其中(LS,LN)用来表示该知识的强度。LS（充分性度量）和LN（必要性度量）。

为讨论方便，引入几率函数如下：

$$O(X)=\frac{P(X)}{1-P(X)} \quad \text{或} \quad O(X)=\frac{P(X)}{P(\neg X)}$$

5.3主观Bayes方法

1. 规则不确定性

修改的Bayes公式

$$O(HIE)=LS\times O(H) \tag{5.18}$$

$$O(HI\text{-}E)=LN\times 0(H) \tag{5.19}$$

公式(5.18)与公式(5.19)就是修改的Bayes公式。从这两个公式可以看出：

当E为真时，可以通过LS将H的先验几率O(H)更新为其后验几率O(HIE)；

E为假时，可以通过LN将H的先验几率O(H)更新为其后验几率O(H|E)；

5.3主观Bayes方法

- 2. 证据不确定性
- (1) 证据肯定为真

当证据E肯定为真，即全证据一定出现时，此时P(E)=P(E IS)=1。

将H的先验几率更新为后验几率的公式为：

$$O(H|E)=LS\times O(H)$$

- (2) 证据肯定为假

当证据E 肯定为假，即证据不出现时，P(E)=P(EIS)=0, P(qE)=1。将H的先验几率更新为后验几率的公式为：

$$O(H\eta\ E)=LN\times O(H)$$

例题分析

例 设 有 规 则

IF E THEN (200,0.1) H

已知证据E必然发生，并且P(H)=0.03, 求 H的后验概率。

解：因为P(H)=0.03, 则

$$O(H)=\frac{P(H)}{1-P(H)}$$ =0.03/(1-0.03)=0.0309

O(HIE)=LS×O(H)=200×0.0309=6.18

$$P(H\mid E)=\frac{O(H\mid E)}{1+O(H\mid E)}$$

=6.18/(1+6.18)=0.8607

2. 可信度的计算

■设有如下规则：

If E_1 Then $H(CF(H,E_1))$

If E_2 Then $H(CF(H,E_2)$

■则结论H 的综合可信度可分以下两步计算：

第一步：分别对每条规则求出其CF(H)。即

$CF1(H)=CF(H,E_1)\times\max(0,CF(E_1))$

$CF2(H)=CF(H,E_2)\times\max(0,CF(E_2))$

2. 可信度的计算

第二步：用如下公式求E_1与E_2对H 的综合可信度：

$$CF(H)=\begin{cases} CF_1(H)+CF_2(H)-CF_1(H)\times CF_2(H) & \text{若CR()20CR(D)0} \\ CF_1(H)+CF_2(H)+CF_1(H)\times CF_2(H) & \\ CF_1(H)+CF_2(H) & \text{若}CF_1(H)\text{与}CF_2(H)\text{异号} \end{cases}$$

- 在后来基于MYCIN 基础上形成的EMYCIN 中，对上式做了如下的修改：

- 如果CF1(H) 和CF2(H)异号，则：

2. 可信度的计算

$$CF(H)=\frac{CF_1(H)+CF_2(H)}{1-\min\{|CF_1(H)|,|CF_2(H)|\}}$$

其他情况不变。

如果可由多条知识推出同一个结论，并且这些规则的前提相互独立，结论的可信度又不相同，则可以将上述合成过程推广应用到多条规则支持同一条结论，且规则前提可以包含多个证据的情况。这时合成过程是先把第一条与第二条合成，然后再用该合成后的结论与第三条合成，依次进行下去，直到全部合成完为止。

例子分析：

- 例5.4 设有如下一组规则：

R1:IF E_1THNN H(0.9)

R2:IF E_2THEN H(0.6)

R3:IF E_3THEN H(-0.5)

R4:IF E_4AND(E_5OR E_6)THEN E_1(0.8)

已知：CF(E_2)=0.8, CF(E_3)=0.6, CF(E_4)=0.5,

CF(E_5)=0.6,CF(E_6)=0.8

求H 的综合可信度CF(H)。

R1:IF E_1THNN H(0.9)

R2:IF E_2THEN H(0.6)

R3:IF E_3THEN H(-0.5)

R4:IF E_4AND(E_5OR E_6)THEN E_1(0.8)

已知：$CF(E_2)=0.8, CF(E_3)=0.6, CF(E_4)=0.5, CF(E_5)=0.6, CF(E_6)=0.8$

- 解：由R_4得到：

$$CF(E1)=0.8\times\max\{0,CF(E_4\text{AND}\ (E_5\text{OR}\ E_6))\}$$
$$=0.8\times\max\{0,\min\{CF(E_4),CF(E_5\text{OR}\ E_6)\}\}$$
$$=0.8\times\max\{0,\min\{CF(E_4),\max\{CF(E_5),CF(E_6)\}\}\}$$
$$=0.8\times\max\{0,\min\{0.5,0.8\}\}$$
$$=0.8\times\max\{0,0.5\}$$
$$=0.4$$

- 由R1得到：

$$CF_1(H)=CF(H,E_1)\times\max\{0,CF(E_1)\}$$
$$=0.9\times\max\{0,0.4\}=0.36$$

- 由R_2得到：

$$CF_2(H)=CF(H,E_2)\times\max\{0,CF(E_2)\}$$
$$=0.6\times\max\{0,0.8\}=0.48$$

- 由R_3得到：

$$CF_3(H)=CF(H,E_3)\times\max\{0,CF(E_3)\}$$
$$=-0.5\times\max\{0,0.6\}=-0.3$$

- 根据结论不确定性的合成算法得到：

$$CF_1,2(H)=CF_1(H)+CF_2(H)-CF_1(H)\times CF_2(H)$$
$$=0.36+0.48-0.36\times0.48$$

$$=0.84-0.17=0.67$$

$$CF_{1,2,3}(H)=\frac{CF_{1,2}(H)+CF_3(H)}{1-\min\{|CF_{1,2}(H)|,|CF_3(H)|\}}$$

$$=\frac{0.67-0.3}{1-\min\{0.67,0.3\}}=\frac{0.37}{0.7}$$

=0.53

- 这就是所求出的综合可信度，即CF(H)=0.53。

5.5.1证据理论的形式化

在D-S 理论中，有四个函数非常重要，分别是：

概率分配函数

信任函数

似然函数

类概率函数

5.5.1证据理论的形式化

定义5.3设函数m:$2^{\Theta}\rightarrow[0,1]$，且满足

$$m(\varphi)=0$$

$$\sum_{A\subseteq\Omega} m(A)=1$$

则称m是2^{Θ}上的概率分配函数，m(A)称为A的基本概率。

语义：m(A)表示依据当前的环境对假设集A的信任程度。

5.5.1证据理论的形式化

2. 信任函数

定义5.4 信任函数(Belief Function)

$$\mathrm{Bel}:2^{2}\rightarrow[0,1]$$

对任意的AC Ω 有

$$Bel(A)=\sum_{B\subseteq A}m(B)$$

Bel(A)表示当前环境下，对假设集A的信任程度，其值为A的所有子集的基本概率之和，表示对A的总的信任度。

5.5.1证据理论的形式化

3、似然函数

定义5.5 似然函数(Plausibility Function)

$$Pl:2^{2}\rightarrow[0,1]$$

对任意的 ACΩ,有：Pl(A)=1-Bel(-A),其 中 ， A=Q-A。 从Ω去掉A 而不是从幂集去掉A

似然函数又称为不可驳斥函数或上限函数。由于Bel(A)表示对A为真的信任度，Bel(q A)表示对7A 的信任度，即A 为假的信任度，因此，Pl(A)表示对A 为非假的信任度。

5.5.1证据理论的形式化

推论1:设有信任函数m, 似然函数Pl, 则有

$$Pl(A)=\sum_{A\cap B\neq\phi} m(B)$$

因为：

$$\begin{aligned}
Pl(A)-\sum_{A\cap B\neq\phi} m(B) &= 1-Bel(\neg A)-\sum_{A\cap B\neq\phi} m(B)\\
\vdots \qquad &= 1-(Bel(\neg A)+\sum_{A\cap B\neq\phi} m(B))\\
&= 1-(\sum_{C\subseteq\neg A} m(C)+\sum_{A\cap B\neq\phi} m(B))\\
&= 1-\sum_{D\subseteq\Omega} m(D)\\
&= 0
\end{aligned}$$

5.5.1证据理论的形式化

- 例题：有限集Q={ 红， 黄， 蓝}，若定义2^2上的一个基本函数m:m(φ,{ 红}，{黄}，{蓝}，{红， 黄}，{红， 蓝}，{黄，蓝}，{红， 黄， 蓝})={0，0.3，0，0.1，0.2，0.2，0.1，0.1}，
- 计算Bel({红， 黄})，Pl({红})的值。

5.5.1证据理论的形式化

·Bel({ 红, 黄})=m({ 红})+m({ 黄})+m({ 红 , 黄})=0.3+0+0.2=0.5。

·Pl({ 红})=1−Bel(-{ 红})=1−Bel({黄, 蓝})=1−(m({黄})+m({蓝})+m({黄, 蓝}))=1−(0+0.1+0.1)=0.8

5.5.2证据理论推理模型

2. 类概率函数

利用信任函数Bel(A)和似然函数Pl(A)，可以定义A 的类概率函数，并把它作为A的不确定性度量。

$$f(A)=Bel(A)+\frac{|A|}{|\Omega|}(Pl(A)-Bel(A))$$

其中IAI、 |Ω|分别表示A 和Ω中包含元素的个数。

类概率函数f(A)也可以用来度量证据A 的不确定性。

5.5.2证据理论推理模型

- 例题：设学生考试成绩的论域为{A,B,C,D,E},

  小王成绩得A、 得B、 得A 或B 的基本概率分别分配到0.2、0.1、0.3, Bel({C,D,E}) 为0.2;

- 给出Bel({A,B})、Pl({A,B}) 和f({A,B})。

5.5.2证据理论推理模型

·Bel({A,B})=m({A})+m({B})+m({A,B})=0.2+

0.1+0.3=0.6

·Pl({A,B})=1-Bel({C,D,E})=1-0.2=0.8

·f({A,B})=Bel({A,B})+|{A,B}l/IUI·[Pl({A,B})

-Bel({A,B})]=0.6+2/5·(0.8-0.6)=0.6+0.08=

0.68

第六章机器学习

6.1机器学习概述

6.2决策树学习

6.3其它机器学习方法

6.1.2学习系统

2、学习系统的基本要求

通常一个学习系统应该满足如下的基本要求：

1) 具有适当的学习环境

学习系统中环境并非指通常的物理条件，而是指学习系统进行学习时所必需的信息来源。

2) 具有一定的学习能力

学习系统应通过与环境反复多次相互作用，逐步学到有关知识，并且要使系统在学习过程中通过实践验证、评价所学知识的正确性。

6.1.2学习系统

3) 能用所学的知识解决问题

学习系统能把学到的信息用于对未来的估计、分类、决策和控制。

4) 能提高系统的性能

提高系统的性能是学习系统最终目标。通过学习，系统随之增长知识，提高解决问题的能力，使之能完成原来不能完成的任务，或者比原来做得更好。

6.1.2学习系统

3、学习系统的基本模型

由此看来：学习系统至少应有：环境、知识库、学习环节和执行环节四个基本部分。一种典型的机器学习系统(迪特里奇 (Dietterich) 学习模型)如图6-1所示。：

6.1.2学习系统

环境向系统的学习部件提供某些信息，学习环节利用这些信息修改知识库，增进执行部件的效能；执行环节根据知识库完成任务，同时把获得的信息反馈给学习环节。

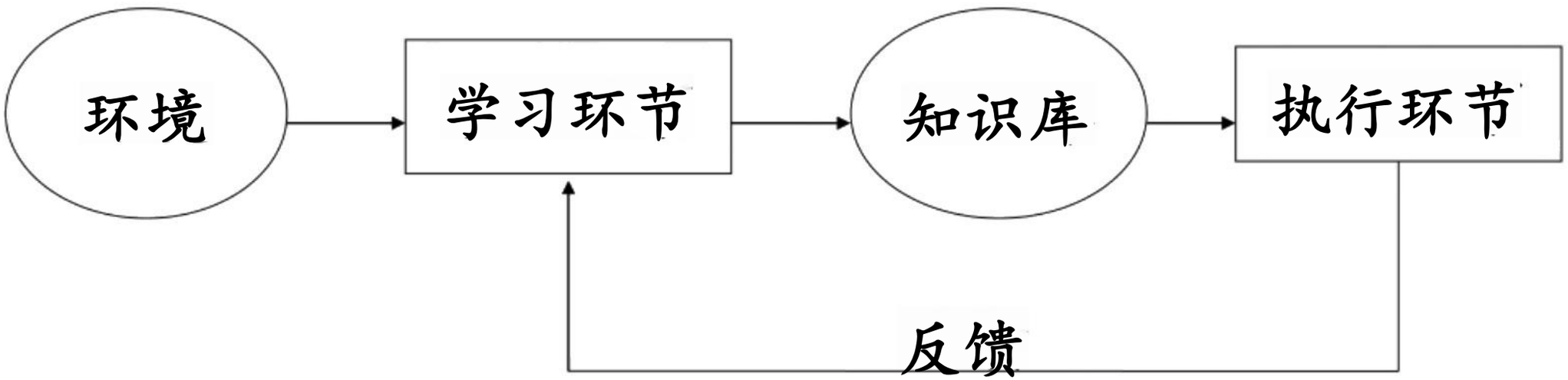


图6.1简单的学习模型

6.3.3基本的决策树算法ID3

口昆兰(Quinlan)于1983年对CLS 进行了发展，研制了 ID3 算法。该算法不仅能方便地表示概念属性-值信息的结构，而且能从大量实例数据中有效地生成相应的决策树模型。

- 大多数决策树学习算法是一种核心算法的变体
- 采用自顶向下的贪婪搜索遍历可能的决策树空间
- ID3是这种算法的代表

6.3.3基本的决策树算法ID3

- **ID3的思想**
  - 自顶向下构造决策树
  - 从“哪一个属性将在树的根节点被测试”开始
  - 使用统计测试来确定每一个实例属性单独分类训练样例的能力
- **ID3的过程**
  - 分类能力最好的属性被选作树的根节点
  - 根节点的每个可能值产生一个分支
  - 训练样例排列到适当的分支
  - 重复上面的过程

6.3.3基本的决策树算法ID3

□ 基本的决策树学习算法ID3：

ID3(Examples,Target _atribute,Attributes)

即训练样例集

是这棵树要预测的目标属性

是除目标属性外决策树测试的属性列表

6.3.3基本的决策树算法ID3

- ID3算法是一种自顶向下增长树的贪婪算法，在每个节点选取能最好地分类样例的属性。继续这个过程直到这棵树能完美分类训练样例，或所有的属性都已被使用过。

- 那么，在决策树生成过程当中，应该以什么样的顺序来选取实例集中实例的属性进行扩展呢?即如何选择具有最高信息增益的属性为最好的属性?

ID3算法-最佳分类属性

■设给定正负实例的集合为S，构成训练窗口。ID3算法视S为一个离散信息系统，并用信息熵表示该系统的信息量。当决策有k个不同的输出时，S的熵为：

$$Entropy(S) \equiv H(S) = -\sum_{i=1}^{k} P_i \log_2 P_i$$

■其中P_i表示第i类输出所占训练窗口中总的输出数量的比例。

天气	温度	湿　度	风速	活动
晴	炎热	高	弱	No
晴	炎热	高	强	No
阴	炎热	高	弱	Yes
雨	适中	高	弱	Yes
雨	寒冷	正常	弱	Yes
雨	寒冷	正常	强	No
阴	寒冷	正常	强	Yes

$$Entropy(S) = Entropy[4+,3-]$$
$$= -\frac{4}{7}log_2\frac{4}{7} - \frac{3}{7}log_2\frac{3}{7}$$

ID3算 法 -最佳分类属性

- 为了检测属性重要性，通过属性的信息增益Gain来评估其重要性。对于属性A，假设其值域为(v1, V2, …, Vn)，则训练实例S中属性A的信息增益Gain可以定义为：

$$Gain(S,A) = Entropy(S) - \sum_{i=1}^{n} \frac{|S_i|}{|S|} Entropy(S_i)$$

$$=Entropy(S)-Entropy(SIA)$$

$$=H(S)-H(SIA)$$

- S;表示S中属性A 的值为v;的子集；　|S;|表示集合的势。

ID3算法-最佳分类属性

口信息增益

- 昆兰建议选取获得信息量最大的属性作为扩展属性。
- 最小熵原理：因为获得信息量最大，即信息增益Gain最大，等价于条件熵H(SIA;)最小，即不确定性最小。
- 因此，也可以以条件熵H(SIA)最小作为选择属性重要性标准。H(SIA) 越小，说明A;引入的信息最多，系统熵下降的越快。
- ID3 算法是一种贪婪搜索(Greedy Search)算法，即选择信息量最大的属性进行决策树分裂，计算中表现为使训练例子集的熵下降最快。

ID3算 法 -最佳分类属性—例子分析

Instances	No. of Wings	Broken Wings	Living status	area/weight	Fly
1	2	0	alive	2.5	T
2	2	1	alive	2.5	F
3	2	2	alive	2.6	F
4	2	0	alive	3.0	
5	2	0	dead	3.2	F
6	0	0	alive	0	F
7	1	0	alive	0	F
8		0	alive	3.4	T
9	2	0	alive	2.0	F

■对于给出的例子，选取整个训练集为训练窗口，有3个正实例，6个负实例，采用记号[3+, 6–]表示总的样本数据S。

则**S的熵为：**

$$Entropy(S) \equiv H(S) = -\sum_{i=1}^{k} P_i \log_2 P_i$$

$$Entropy[3+,6-] = -\frac{3}{9}\log_2(3/9) - \frac{6}{9}\log_2(6/9) = 0.9179bit$$

ID3算法-最佳分类属性一例子分析

■计算属性Living Status的信息增益，该属性值域为(alive,dead),则

Salive=[3+,5-],Sdead=[0+,1-]

■先计算Entropy(Salive),Entropy(Sdead),如下：

$$Entropy(Salive)=Entropy[3+,5-]$$

$$=-\frac{3}{8}\log_2(3/8)-\frac{5}{8}\log_2(5/8)=0.9544bit$$

$$Entropy(Sdead)=Entropy[0+,1-]$$

$$=-\frac{0}{1}\log_2(0/1)-\frac{1}{1}\log_2(1/1)=0bit$$

ID3 算 法 -最佳分类属性-例子分析

■所以，living status的信息增益为

$$Gain(S, a) = Entropy(S) - Values(a) \quad Entropy(S_{alive}) - \frac{|S_{dead}|}{|S|} Entropy(S_{dead})$$

$$= Entropy(S) - \frac{|S_{alive}|}{|S|} Entropy(S_{alive}) - \frac{|S_{dead}|}{|S|} Entropy(S_{dead})$$

同样可计算其他属性的信息增益。

ID3算法—最佳分类属性—例子分析

■计算属性No. of Wings的信息增益，该属性值域为(0, 1, 2)，则

S_0=[0+,1-],S_1=[0+,1-],S_2=[3+,4-]

■先计算Entropy(S_0)，Entropy(S_1)，Entropy(S_2)，如下：

$$Entropy(S_0)=Entropy[0+,1-]=0bit$$

$$Entropy(S_1)=Entropy[0+,1-]=0bit$$

$$Entropy(S_2)=Entropy[3+,4-]$$

$$=-\frac{3}{7}\log_2(3/7)-\frac{4}{7}\log_2(4/7)=0.9852bit$$

ID3 算 法 -最佳分类属性一例子分析

■ No.of Wings的信息增益计算公式如下：

$$= Entropy(S) - \frac{|S_0|}{|S|} Entropy(S_0) - \frac{|S_1|}{|S|} Entropy(S_1) - \frac{|S_2|}{|S|} Entropy(S_2)$$

$$= 0.9179 - \frac{7}{9} \times 0.9852 = 0.1516 bit$$

ID3算 法 -最佳分类属性一例子分析

■计算属性Broken Wings的信息增益，该属性值域为(0,1,2),则

$S_0=[3+,4-],S_1=[0+,1-],S_2=[0+,1-]$

■ 先 计 算Entropy(S_0),Entropy(S_1),Entropy(S_2), 如下：

$$Entropy(S_0)=Entropy[3+,4-]$$

$$=-\frac{3}{7}\log_2(3/7)-\frac{4}{7}\log_2(4/7)=0.9852bit$$

$$Entropy(S_1)=Entropy[0+,1-]=0bit$$

$$Entropy(S_2)=Entropy[0+,1-]=0bit$$

ID3 算 法 -最佳分类属性—例子分析

■ Broken Wings的信息增益计算公式如下：

$$= Entropy(S) - \frac{|S_0|}{|S|} Entropy(S_0) - \frac{|S_1|}{|S|} Entropy(S_1) - \frac{|S_2|}{|S|} Entropy(S_2)$$

$$= 0.9179 - \frac{7}{9} \times 0.9852 = 0.1516 bit$$

ID3 算 法 -最佳分类属性-例子分析

■计算Area/Weight的信息增益，该属性值域为(>=0的实数)，可分为{>=2.5}和{<2.5}两个集合，则

S>=2.5=[3+,3-],S<2.5=[0+,3-]

■先计算Entropy(S>=2.5)，Entropy(S<2.5)，如下：

$$Entropy(S\_25)=Entropy[3+,3-]$$

$$=-\frac{3}{6}\log_2(3/6)-\frac{3}{6}\log_2(3/6)=1bit$$

$$Entropy(S\_5)=Entropy[0+,3-]$$

$$=-\frac{0}{3}\log_2(0/3)-\frac{3}{3}\log_2(3/3)=0bit$$

ID3 算 法 -最佳分类属性一例子分析

■ Area/Weight的信息增益计算公式如下：

$$= Entropy(S) - \frac{|S_{>=2.5}|}{|S|} Entropy(S_{>=2.5}) - \frac{|S_{<2.5}|}{|S|} Entropy(S_{<2.5})$$

$$= 0.9179 - \frac{6}{9} \times 1 = 0.2512 bit$$