第二章 知识表示方法

主要内容：

§2.1 状态空间表示

§2.2 问题归约法

§2.3 谓词逻辑表示

§2.4 语义网络法

§2.5 框架与本体

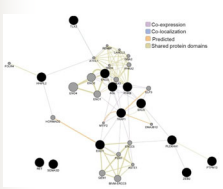
状态空间

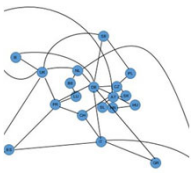
§2.1 状态空间表示

- State space method

A method for problem representation and problem-solving based on the *solution space*.

- It is based on the *State* and *Operator*.





State & Operator

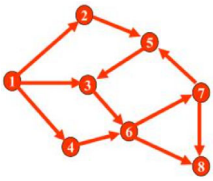
- State: Some variables, describing the difference among different things or events of some class

$$Q = [q_0, q_1, \ldots, q_n]^T$$

- Operator: A means that is used to transform a problem from one state to another.

Nodes represent states,

Arcs represent occurring events.



状态空间

- State space: a graph that represents all possible states and their relationships with the problem.

Example：3-element states $\{S, F, G\}$

S : set of all possible initial states of problem

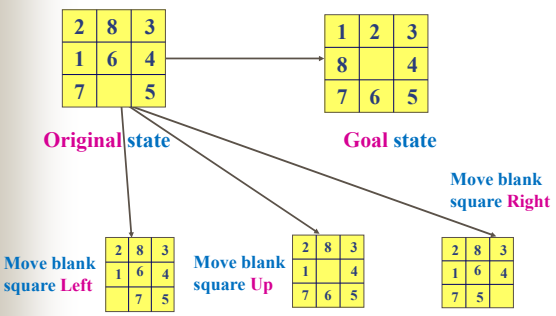
F : set of operators

G : set of goal states

State space

八数码问题

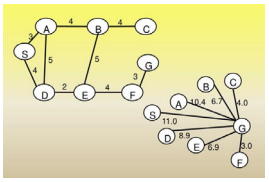
八数码问题的状态空间



基本算法

状态空间法的基本算法

- 规定一组操作(算子)，能够使状态从一个状态变为另一个状态。 $n_i \longrightarrow n_j$
- 决定一种搜索策略，使得能够从初始状态出发，沿某个路径达到目标状态。



Route Planning

Example: Route Planning

- Initial state

  Starting from ChangSha

- Operators

  Driving from a city to another city

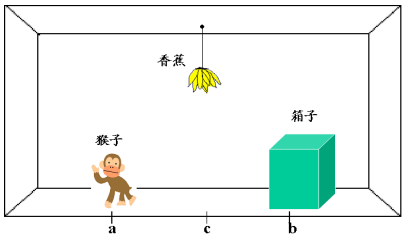
- Goal state

  Destination city - BeiJing

Route Planning

Monkey & Banana

Example: Monkey & Banana



Monkey and Banana Problem

Monkey & Banana

- Problem State Set : (W， x， Y， z)
  - W Monkey's horizontal position
  - x Monkey is above the box $x = 1$, else $x = 0$
  - Y Box's horizontal position
  - z Monkey get the banana $z = 1$, else $z = 0$
- Operator:
  - Goal (U) means monkey go to position U

    (W，0，Y，z) goto (U) (U，0，Y，z)
  - Pushbox (V) monkey pushes the box to position V:

    (W，0，W，z) pushbox (V) (V，0，V，z)

Monkey & Banana

- Climbbox: monkey climbs the top of the box

  (W，0，W，z) Climbbox (W，1，W，z)

  Attention: when used these two operators, the monkey and the box should be on the same horizontal position and the monkey is not on the top of the box.
- Grasp: monkey get the banana

  (c，1，c，0) Grasp (c，1，c，1)

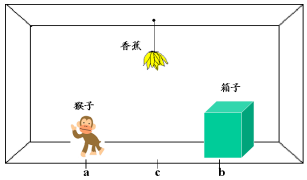
  c: the position that right below the banana

Monkey & Banana

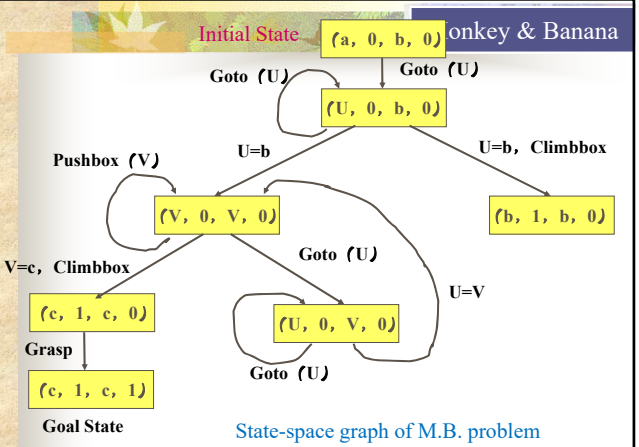
- Initial state is (a，0，b，0)
- Operator sequence from the initial state to the goal state:

  {Goto (b), Pushbox (c), Climbbox, Grasp}

  is a solution of the problem.



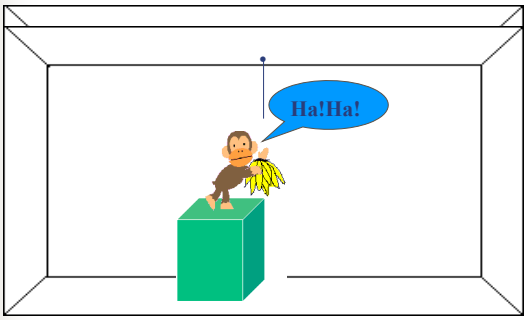
Monkey & Banana



State-space graph of M.B. problem

Monkey & Banana

Solving processes



传教士野人

举例：传教士野人问题

三个传教士M和三个野人C过河，只有一条能装下两个人的船。如果在河的一方（含船上），野人的人数大于传教士的人数，那么传教士就会有危险，提出一种安全的渡河方案！

传教士野人

状态可有多种表示方法

左岸传教士数,右岸传教士数,左岸野人数,右岸野人数,船的位置

简化表示

左岸传教士数,左岸野人数,船的位置

初始状态：(0, 0, 0)　　0：右岸

目标状态：(3, 3, 1)　　1：左岸

传教士野人

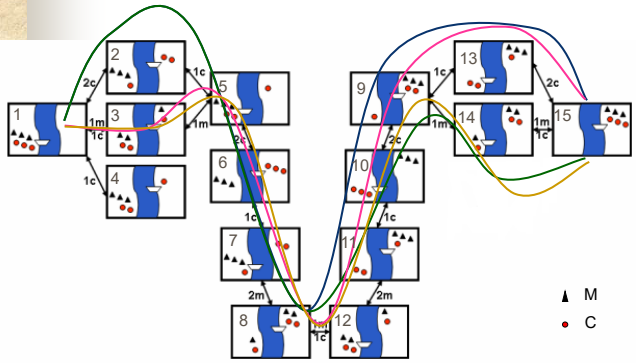
状态的转换　MC问题中的算子

- 将传教士或野人运到河对岸
- Move-1m1c-lr：将一个传教士(m)和一个野人(c)从左岸(L)运到右岸(R)
- 所有可能操作

Move-1m1c-lr　Move-1m1c-rl　Move-2c-lr
Move-2c-rl　Move-2m-lr　Move-2m-rl
Move-1c-lr　Move-1c-rl　Move-1m-lr
Move-1m-rl

传教士野人

传教士野人问题状态空间图

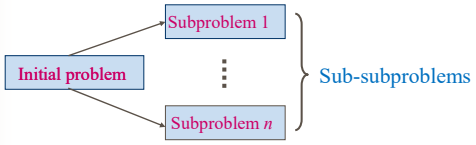


问题的规约

§2.2 问题规约法

■ **问题的规约**

将一个大的问题变换成若干子问题，子问题又可分解成更小的子问题，这样一直分解到可以直接求解为止，全部子问题的解即为大的问题的解.



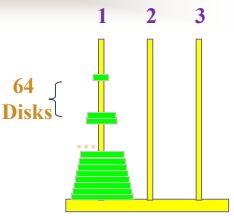
规约三要素

- **归约方法求解问题三要素**
  - 初始问题的描述
  - 一组将问题变换成子问题的变换规则
  - 一组本原问题的描述
- **问题规约的实质**
  - 从初始问题出发，建立子问题以及子问题的子问题，直至把初始问题规约为一个本原问题的集合

Hanoi Puzzle

Working Mechanism-Reduction

1. Tower of Hanoi Puzzle (THP)



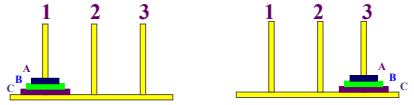
- Three pegs and sixty-four disks
- Moving times: 64 disks

$2^{64}-1 \approx 2^{64} = 10^{19.27}$

If one person moves 1 disk in one second, then to finish this problem needs more than 3000 billion years.

Hanoi Puzzle

- Three Disk Hanoi-Tower
  - Question: There are three pegs-1, 2 and 3 and three disks of different sizes A, B and C. The disks have holes in their centers so that they can be stacked on the pegs.

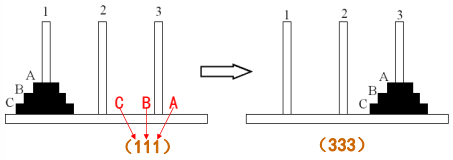


Initial configuration　　Goal configuration

Hanoi Tower Problem

Hanoi Puzzle

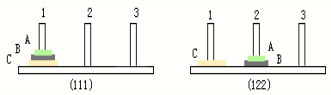
- Constrain conditions
  - Only one disk can be moved at a time;
  - Only the top disk on a peg can be moved;
  - Larger disk may never be placed on top of a smaller one.



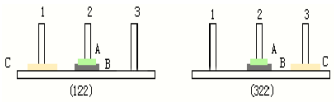
Hanoi Puzzle

原问题可以归约为3个子问题：

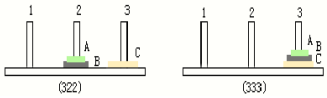
子问题1：移圆盘A和B至柱子2（借助柱子3）



子问题2：移动圆盘C至柱子3

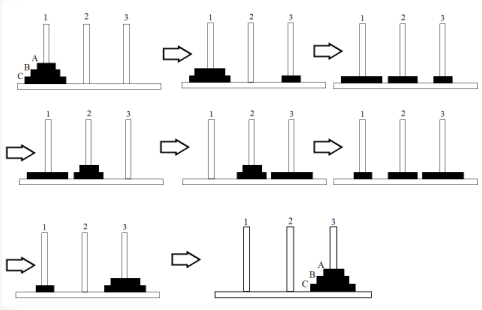


子问题3：把圆盘A和B移至柱子3（借助柱子1）



Hanoi Puzzle

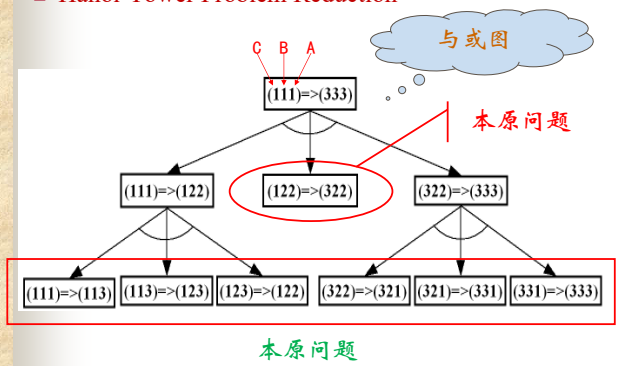
归约过程（3个圆盘）



The solving process of Hanoi tower problem

Hanoi Puzzle

- Hanoi-Tower Problem Reduction



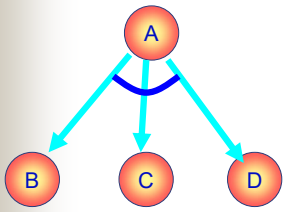
Hanoi Puzzle

Question 1: Four~Five Disk Hanoi Tower

AND/OR 的图表示

AND/OR Graph Representation

- AND graph, OR graph, AND/OR graph



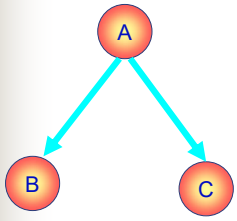
AND Graph

与图：把一个复杂问题分解为若干个较为简单的子问题，形成“与”树。

AND/OR 的图表示

AND/OR Graph Representation

- AND graph, OR graph, AND/OR graph



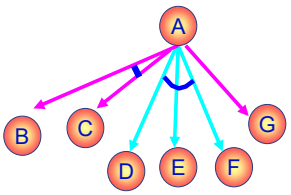
OR Graph

或图：把原问题变换为若干个较为容易求解的新问题，形成“或”树。

AND/OR 的图表示

AND/OR Graph Representation

The reduction of a problem to alternative sets of successor problems or sub-problems by graph-like structure can be conveniently represented by a diagram.

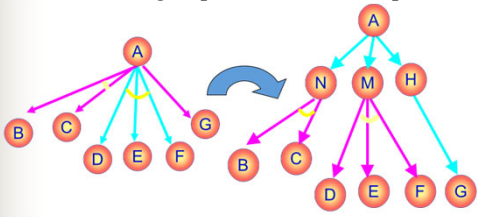


Structure for alternative sets of subproblems

AND/OR 的图表示

AND/OR Graph Representation

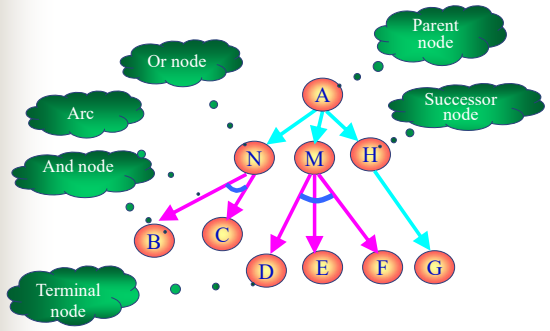
It is usual to introduce some extra nodes into the structure so that each set, containing more than one descendant problem, can be grouped below its own parent node



Structure of an AND/OR graph

AND/OR 的图表示

AND/OR Graph Representation

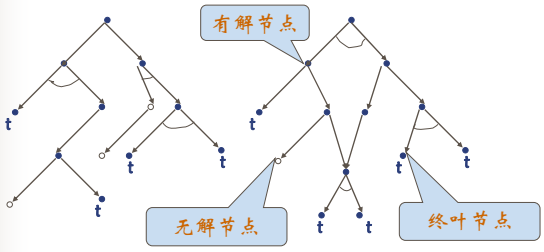


Structure of an AND/OR graph

AND/OR 的图表示

AND/OR Graph Representation

Solved node and unsolved node



The solved node and unsolved nod in an AND/OR graph

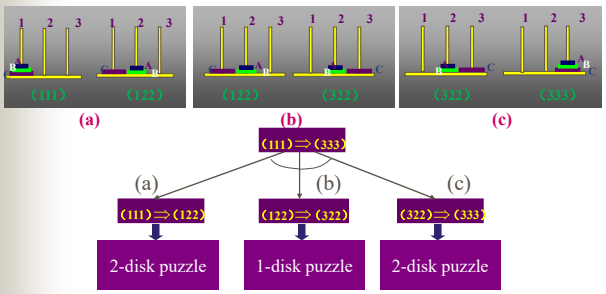
AND/OR 的图表示

AND/OR Graph Representation

- 与或图的构建规则
  - Every node represents a single problem or a set of problems.
  - Terminal nodes correspond to primitive problems.
  - Operators transform a problem into a set of sub-problems.
  - Directed links are towards every AND node of the set of sub-problems.

AND/OR 的图表示

AND/OR Graph Representation

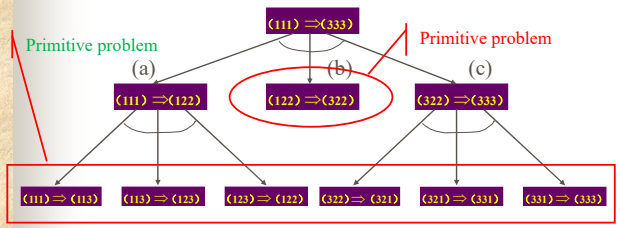


Reduction graph for the 3-disk Hanoi Tower

AND/OR 的图表示

AND/OR Graph Representation

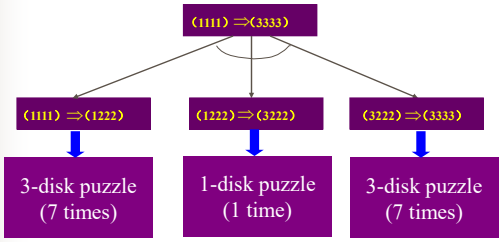
- Puzzle number (b): a primitive problem
- Puzzle number (a) and (c): 2-disk puzzle



And/Or graph for the 3-disk Hanoi Tower

AND/OR 的图表示

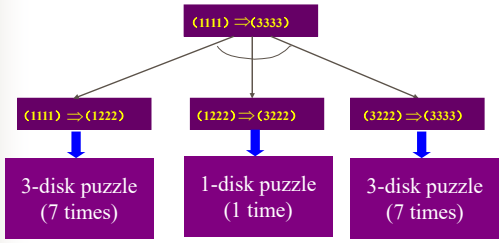
Four Disk Hanoi Tower Graph Representation



Reduction graph for the 4-disk Hanoi Tower

AND/OR 的图表示

Four Disk Hanoi Tower Graph Representation



Reduction graph for the 4-disk Hanoi Tower

Hanoi Tower

Hanoi Tower

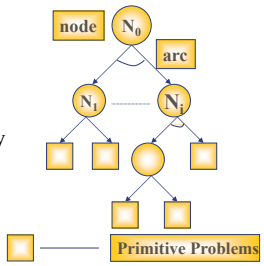
Number of Dishes	Moving times
1	1
2	2^2-1=3
3	2^3-1=7
…	…
64	2^{64}-1

AND/OR 的图表示

AND/OR Graph Representation

- Conclusions
  - A node is used to represent a problem.
  - Original problem is reduced into several subproblems and sub-subproblems.
  - The solution can be obtained by searching the AND/OR graph.
  - The solution is a solution tree.

Reduction graph



谓词逻辑法

§2.3 谓词逻辑法

一个陈述句由主语和谓语两个部分组成

在谓词逻辑中，为揭示命题内部结构及不同命题的内部结构关系，就按这两个部分进行分析

主语称为个体或客体

谓语称作谓词

个体和谓词

- 个体和谓词
  - 在原子命题中，所描述的对象，称为个体
  - 用以描述个体性质，或个体间关系的部分，称为谓词

个体

- 个体，是指可以独立存在的事物，具体的或抽象的。

  例如：李明、香蕉、电视机、精神、思想，等
  - 表示特定的个体，称为个体常元

    例如：李明、字母A、数字5
  - 表示不确定的个体，叫作个体变元

    例如：花朵、星星、男生、学校

谓词

- 谓词：用以描述个体性质，或个体间关系的部分，称为谓词
- 当谓词与一个个体相关时，刻划了该个体的性质

  例如：5是质数，A是字母　　一元谓词
- 当谓词与两个或两个以上个体相关时，刻划了个体间的关系

  例如：李明生于北京，10比5大　二元谓词

命题的谓词形式

- 命题的谓词形式

  单独的个体和谓词不能构成命题
- 命题的表示：

  将表示个体的小写字母，写在表示谓词的大写字母右侧的括号里。

谓词形式

命题的谓词形式

“李明是大学生”

其中：李明是个体，用小写a表示

“…是大学生”是谓词，用大写S表示

则该命题表示为　$S(a)$：李明是大学生

谓词形式

命题的谓词形式

“张华出生在北京”

其中：张华和北京是个体，用a表示张华，用b表示北京

“…出生在…”是谓词，用大写B表示

则该命题表示为　$B(a, b)$：张华出生在北京

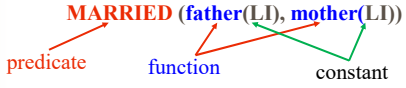
形式语言

- 一阶谓词演算：是一种形式语言，它把数学中的逻辑论证符号化。
- 形式语言：一种有别于自然语言的符号语言。

基本符号

1.谓词演算语法和语义

- 基本符号
  - 谓词符号、变量符号、函数符号、常量符号、括号和逗号



  - 原子公式 = 谓词 + 项

    Inroom (robot, r1)

2. 连词和量词

- 连词(Connectives)
  - 与及合取（AND，conjunction)

    Like (I, Music) ∧ Like (I, Sport)
  - 或及析取（OR，disjunction）

    Like (I, Music) ∨ Like (I, Sport)
  - 蕴涵（Implication） 如果→那么

    Runs (Philip, Fastest, 100m) → Wins (Philip, Champion)
  - 非（Not） ~ 或者 ¬

    ~ IN (Ma, Shanghai, Sept-2022)

- 量词
  - 全称量词

    一个原子公式$P(x)$，对于所有可能的变量x都具有真值T。这个特性可由在$P(x)$前面加上全称量词$\forall(x)$来表示。例如，句子“所有的机器人都是灰色的”可表示为：

    $\forall(x)[ROBOT(x)=>COLOR(x,GRAY)]$
  - 存在量词

    如果至少有一个x值可使$P(x)$具有真值T，那么这一特性可在$P(x)$前面加上存在量词$\exists(x)$来表示。句子“1号房间内有个物体”可表示为：$\exists(x)$ INROOM (x, R1)

- Quantifier

  Universal quantifier

  An atomic formula $P(x)$ has the value T for All variables x:

  $(\forall x)\ P(x)$

  Universal Quantifier; Bound variables; Domain of universal quantifier

  Existential quantifier

  If there Exists at least one variable of x that makes $P(x)$ has the value T : $(\exists x)\ P(x)$

  E.G. $(\exists x)$INROOM(x, r1)（1号房间内有个物体）

  Existential Quantifier; Bound variables; Domain of existential quantifier

2. 谓词公式

- 原子谓词公式

  用$P(x_1, x_2, \ldots, x_n)$表示一个n元谓词公式，其中P为n元谓词，$x_1, x_2, \ldots, x_n$为客体变量或变元。通常把$P(x_1, x_2, \ldots, x_n)$叫做谓词演算的原子公式，或原子谓词公式。
- 分子谓词公式

  用连词把原子谓词公式组成复合谓词公式，并把它叫做分子谓词公式。

- WFF (well-formed formulas)
  - Legal sentences = WFF
  - Definition
    - 原子公式是合适公式
    - If A is a WFF, then ~A is also a WFF
    - If A and B are WFF， then ($A \wedge$B)，($A \vee$B)，(A=>B)，(A←→B) are also WFF
    - If A is WFF, and x is a free variable in A, then $\forall(x)A$ and $\exists(x)A$ are also WFF
    - Only the formula by above 4 rules，would be WFF

合适公式的真值表

P	Q	P∨Q	P∧Q	P⇒Q	~P
T	T	T	T	T	F
F	T	T	F	T	T
T	F	T	F	F	F
F	F	F	F	T	T

Equivalence

■ Equivalence

Two WWFs are Equivalence if they have the same truth value under every interpretation.

(1) Double negative elimination $\sim(\sim P) \equiv P$

(2) $(P \vee Q) \equiv \sim P \rightarrow Q$

(3) De Morgan's law

$\sim(P \wedge Q) \equiv \sim P \vee \sim Q \qquad \sim(P \vee Q) \equiv \sim P \wedge \sim Q$

(4) Distributive law

$P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$

$P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$

Equivalence

(5) Commutative laws

$P \wedge Q \equiv Q \wedge P, \qquad P \vee Q \equiv Q \vee P$

(6) Associative law

$(P \wedge Q) \wedge R \equiv P \wedge (Q \wedge R)$

$(P \vee Q) \vee R \equiv P \vee (Q \vee R)$

(7) Contra-negative law

$P \rightarrow Q \equiv \sim Q \rightarrow \sim P$

Equivalence

合适公式的性质

■ $(P \wedge \sim Q) \rightarrow \sim R$

P: Jon likes cats， Q: Jon likes dogs ， R: Jon likes all kinds of pets.

■ An expression is a sentence ←→It can be formed of legal symbols

Equivalence

合适公式的性质

■ $((P \wedge \sim Q) \rightarrow \sim R) \equiv \sim P \vee Q \vee R$

- $(P \wedge \sim Q) \rightarrow \sim R$ symbols conjunction implication
- $\sim P \vee Q \vee R$ symbols disjunction
- $((P \wedge \sim Q) \rightarrow \sim R) \equiv \sim P \vee Q \vee R$ equivalence

置换与合一

3. 置换与合一

■ 置换

- 假元推理

$$\left.\begin{array}{l} W_1 \Rightarrow W_2 \\ W_1 \end{array}\right\} \rightarrow \text{Generate } W_2$$

- 全称化推理

$$(\forall x) W(x) \rightarrow W(A)$$

Bound var. Free const

- 综合推理

$$\left.\begin{array}{l} (\forall x)[W_1(x) \rightarrow W_2(x)] \\ W_1(A) \end{array}\right\} \rightarrow \text{Generate } W_2(A)$$

置换与合一

■ 合一

- 寻找项对变量的置换，以使两表达式一致。
- 如果一个置换s作用于表达式集$\{E_i\}$的每个元素，则我们用$\{E_i\}s$来表示置换例的集。我们称表达式集$\{E_i\}$是可合一的。

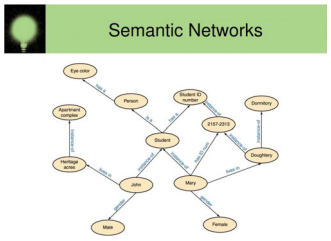
If $E_{1s} = E_{2s} = E_{3s} = \ldots$

Expression set $\{E_i\}$ is unifiable, s is unifier.

定义 & 组成

§2.4 语义网络法

- **定义 & 组成**
  - 定义：语义网络是知识的图形表示，由节点和链接组成.
  - **组成**
    - 词法
    - 结构
    - 过程
    - 语义



二元语义网络

- A collection of predicate calculus expression can be represented by a graph structure-Semantic Network.

1. Representation of Two-Element Semantic Network

（二元语义网络的表示）

-Representation of simple facts

举例：小燕是一只燕子，燕子是鸟；巢-1是小燕的巢，巢-1是巢中的一个

ISA (is-a) Relation

ISA (is-a) Relation

- The ISA (is-a) relation is often used to link instances to classes, classes to super-classes
- Some links (e.g. isPart) are inherited along ISA paths
- The semantics of a SN can be relatively very formal or informal.
  - Often defined at the implementation level

Example

- All swallow are birds

- Xiaoyan is a swallow, swallow is birds; Nest-1 is xiaoyan's nest, Nest-1 is one of the nests.

语义基元

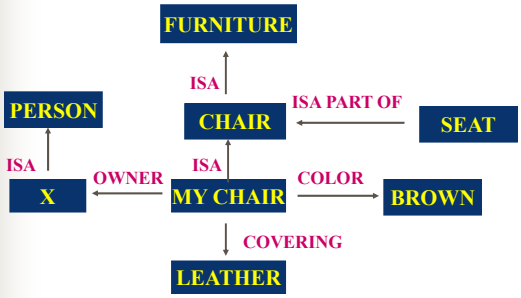
- **选择语义基元**
  - Looking for basic concepts and basic arcs

试图用一组基元来表示知识，以便简化表示，可用简单的知识来表示更复杂的知识。

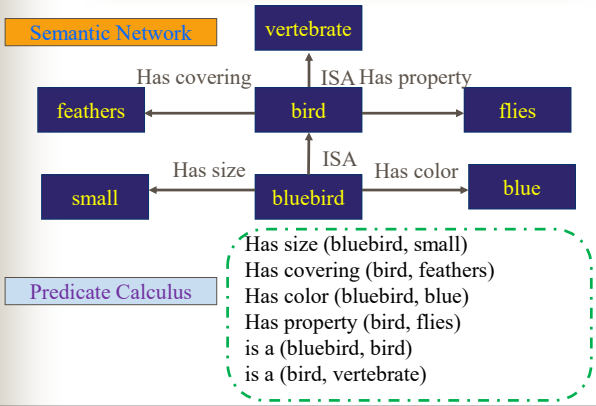
Example:

The color of my chair is brown; the covering of the chair is leather; chair is one kind of furniture; chair is part of seat; chair's owner is *X*; *X* is a person

Example



Example



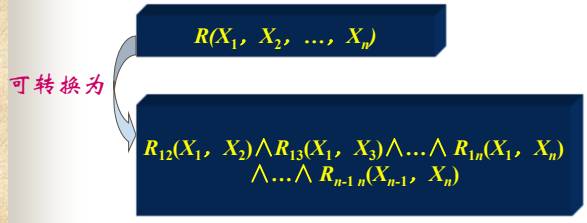
Example

- Predicate Logic = Semantic Network

多元语义网络

- 多元语义网络表示的实质

把多元关系转化为一组二元关系的组合，或二元关系的合取



Example

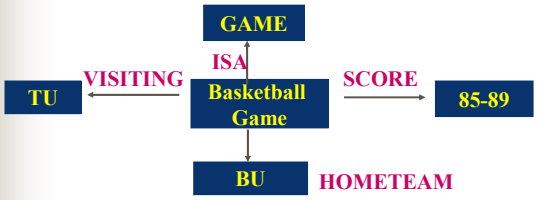
Example

There is a basketball game between two teams: Boston University (home team，BU) and Tsinghua University (visitor team，TU). The game is hold in Boston, 2012 and the score is 85:89.

- Predicate Logic

SCORE（BU，TU，（85-89））

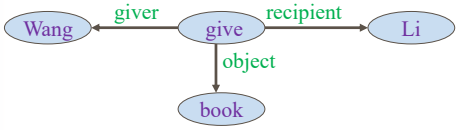
Example



Example

Example

Wang give Li a book.



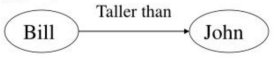
Add more nodes to transform the multi-elements relationship into several two-element relationships.

Example

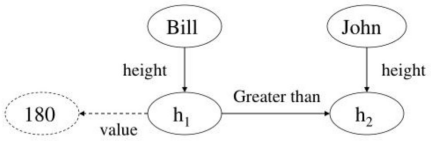
Example

"Bill is taller than John ."

- Non appropriate scheme :



- Appropriate scheme :

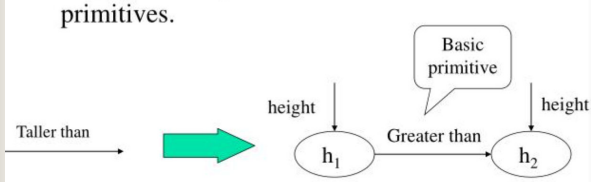


Example

Example

For an appropriate scheme:

- Draw relations on the basic of primitives.
- Represent complicated relations with this primitives.



Inheritance

- Inheritance (继承)

把事物的描述从*概念节点*传递到*实例节点*

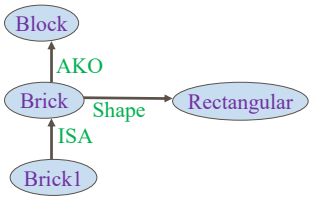
- Pass（遗留） — the descendants get the properties from the ancestors
- Add（添加） — the descendants expand the properties of the ancestors
- Exclude（排除） — stop inheriting the properties

Inheritance

Three methods for inheritance

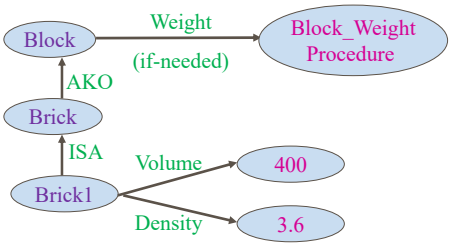
- 值继承 is-a, AKO (a kind of)

What is the shape of Brick1?



Inheritance

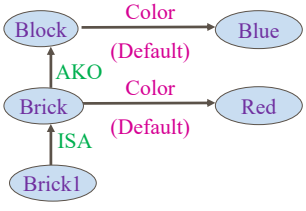
- 如果-需要“继承” If-needed (it can not inherit from the ancestors, we get it from other programs)



Inheritance

- 缺省“继承” Default—it is mostly the truth

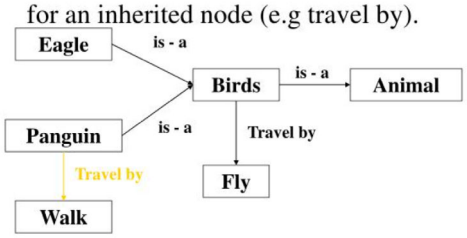
Example: the birds can fly；



Example

Example

Some times we had to override a relation for an inherited node (e.g travel by).



Example

Example

Use semantic networks to represent the below:

- Nellie is an elephant,
- He likes apples.
- Elephants are a kind of mammals,
- They live in Africa,
- And they are big animals.
- Mammals and reptiles are both animals,
- All animals have head.

Example