

## 第3章 搜索推理技术

Ch.3 Searching & Reasoning Tech.

# Main Contents

## 第3章 搜索推理技术

- 3.1 图搜索策略
- 3.2 盲目搜索
- 3.3 启发式搜索
- 3.4 消解原理
- 3.5 规则演绎系统
- 3.6 不确定性推理
- 3.7 概率推理
- 3.8 主观贝叶斯方法
- 3.9 小结

## ❖ 问题求解 (Problem-solving)

### ❖ 表示 (Problem formulation)

● 状态空间表示 (State Space Representation)

### ❖ 搜索 (Searching)

● 无信息搜索 (Uninformed Search)

● 有信息搜索 (Informed Search)

# 1. 问题求解——什么是问题求解

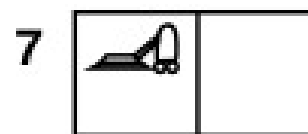
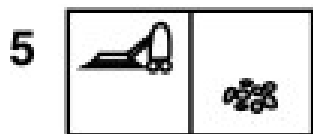
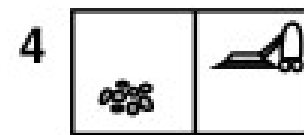
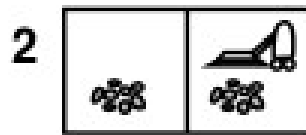
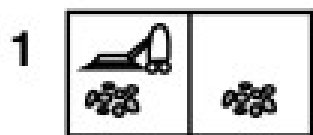
- ❊ 问题求解是人工智能的核心问题之一
- ❊ 问题求解的目的
  - ❖ 机器自动找出某问题的正确解决策略
  - ❖ 更进一步，能够举一反三，具有解决同类问题的能力
- ❊ 是从人工智能初期的智力难题、棋类游戏、简单数学定理证明等问题的研究中开始形成和发展起来的一大类技术
- ❊ 求解的手段多种多样
- ❊ 其中搜索技术是问题求解的主要手段之一
  - ❖ 问题表示
  - ❖ 解的搜索

## 2.问题表示

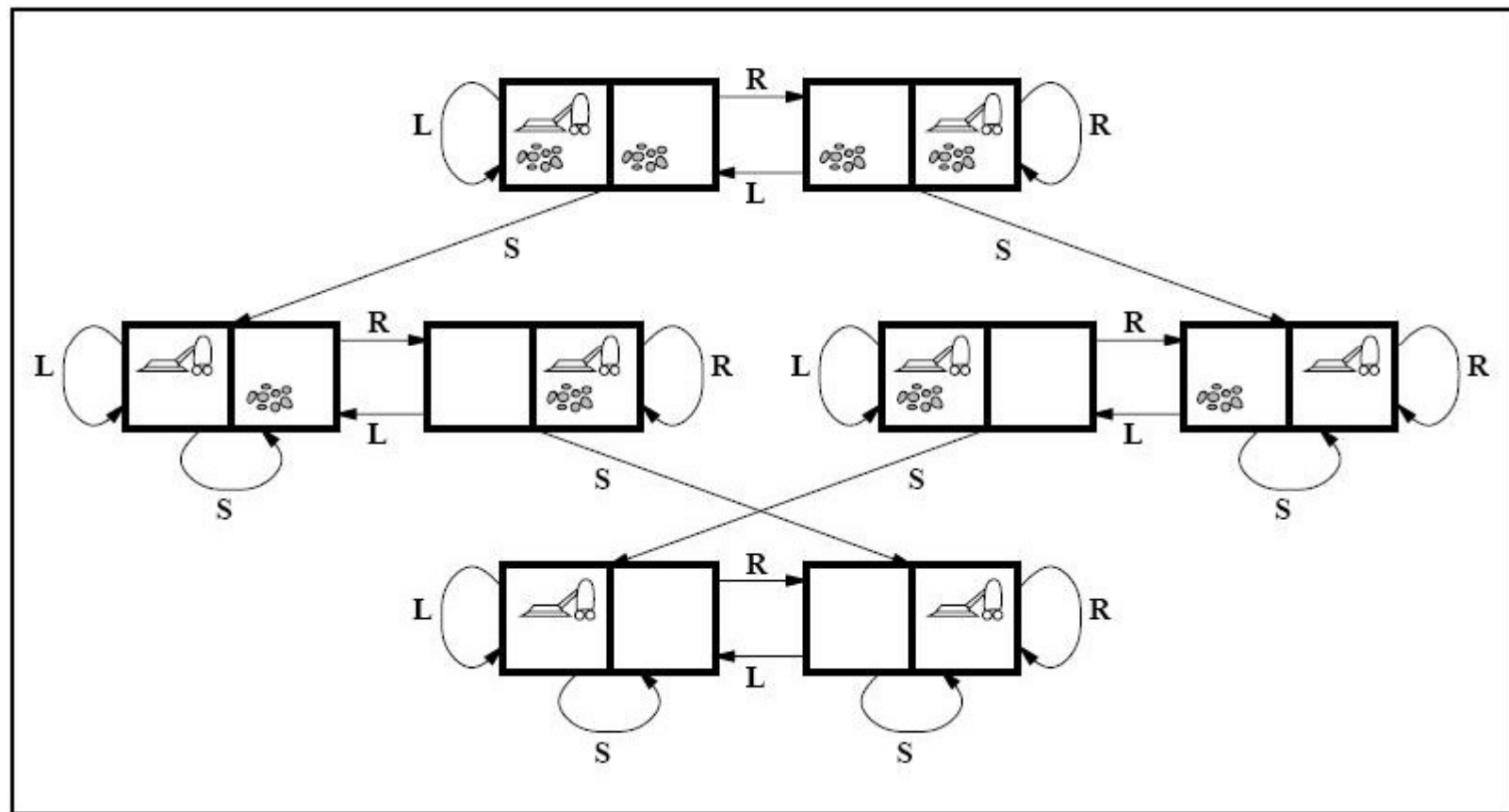
### 例：真空吸尘器的世界

- 假设：吸尘器的世界只有两块地毯大小，地毯或者是脏的，或者是干净的
- 吸尘器能做的动作只有三个  
{向左(**L**eft), 向右(**R**ight), 吸尘(**S**uck)}
- 一共有多少种可能的情况？

状态



# 状态空间图



## 3.1 图搜索策略 (Graph Search Strategy)

### ❁ 图搜索控制策略(Control Strategy of Graph Search)

一种在图中寻找路径的方法。

图中每个节点对应一个状态，每条连线对应一个操作符。这些节点和连线(即状态与操作符)又分别由产生式系统的数据库和规则来标记。求得把一个数据库变换为另一数据库的规则序列问题就等价于求得图中的一条路径问题。

## 📍 图的搜索

### 📍 无信息搜索（盲目搜索）

📍 宽度优先搜索

📍 深度优先搜索

### 📍 有信息搜索（启发式搜索）

📍 A算法

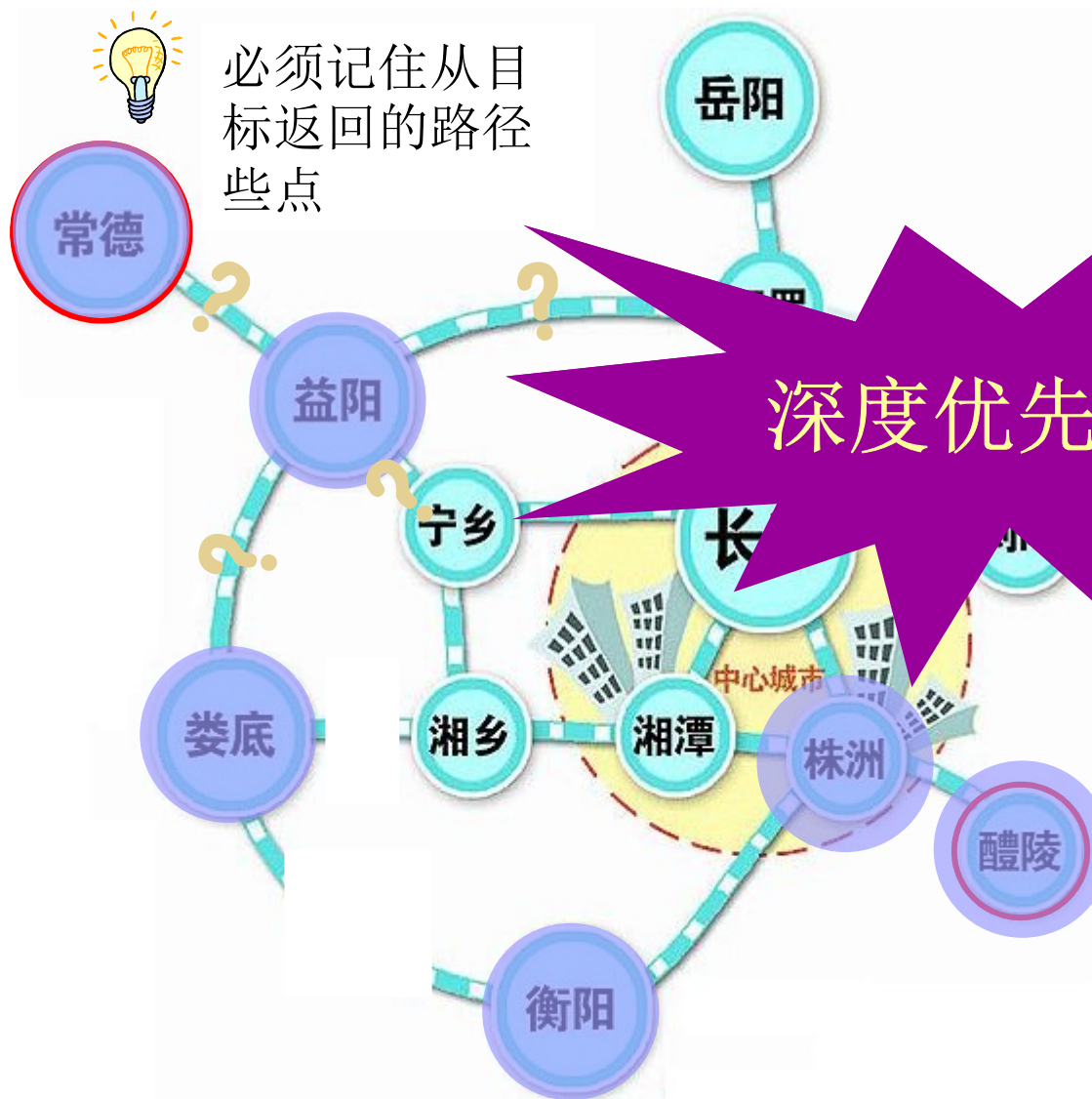
📍 A\*算法

图搜索的一般策略

怎么样在图中搜索路径呢？



# 图的搜索过程



状态: (城市名)

例子: 常德→益阳

益阳→常德

益阳→汨罗

益阳↔宁乡

益阳↔娄底

...

# 图的搜索过程



必须记住下一步还可以走哪些点

OPEN表(记录还没有扩展的点)



必须记住哪些点走过了

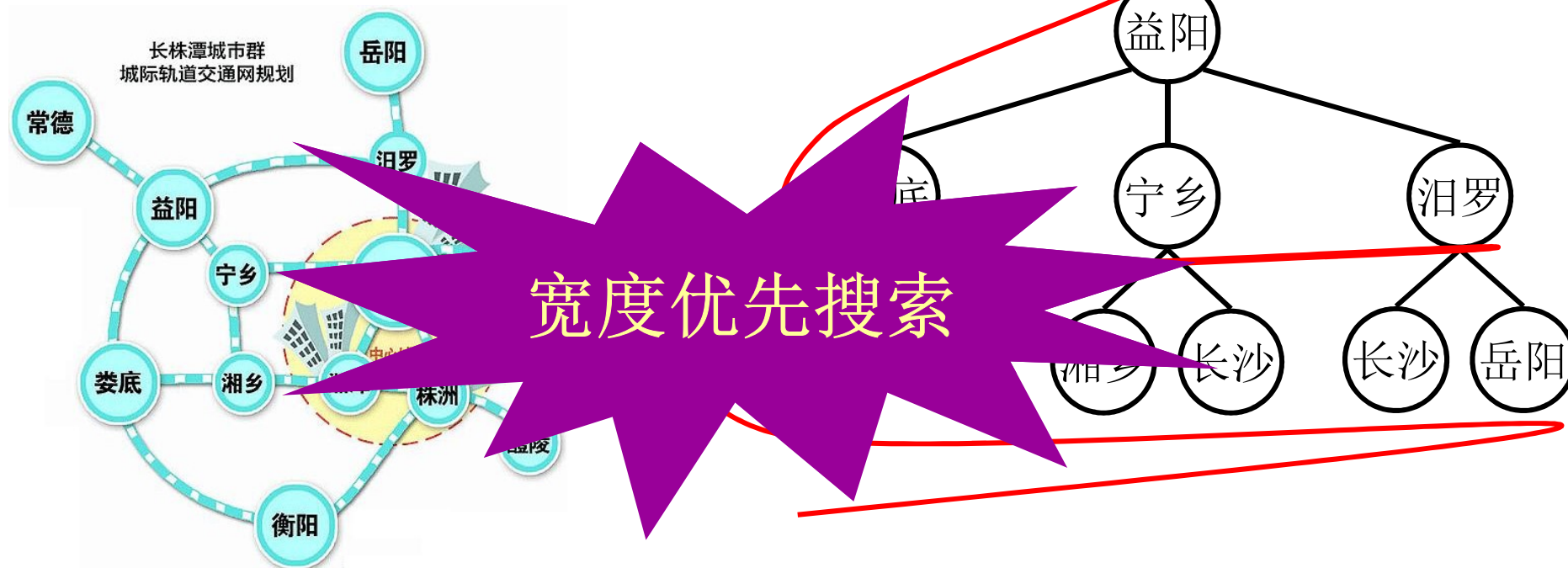
CLOSED表(记录已经扩展的点)



必须记住从目标返回的路径

每个表示状态的节点结构中必须有指向父节点的指针

# 图的搜索过程

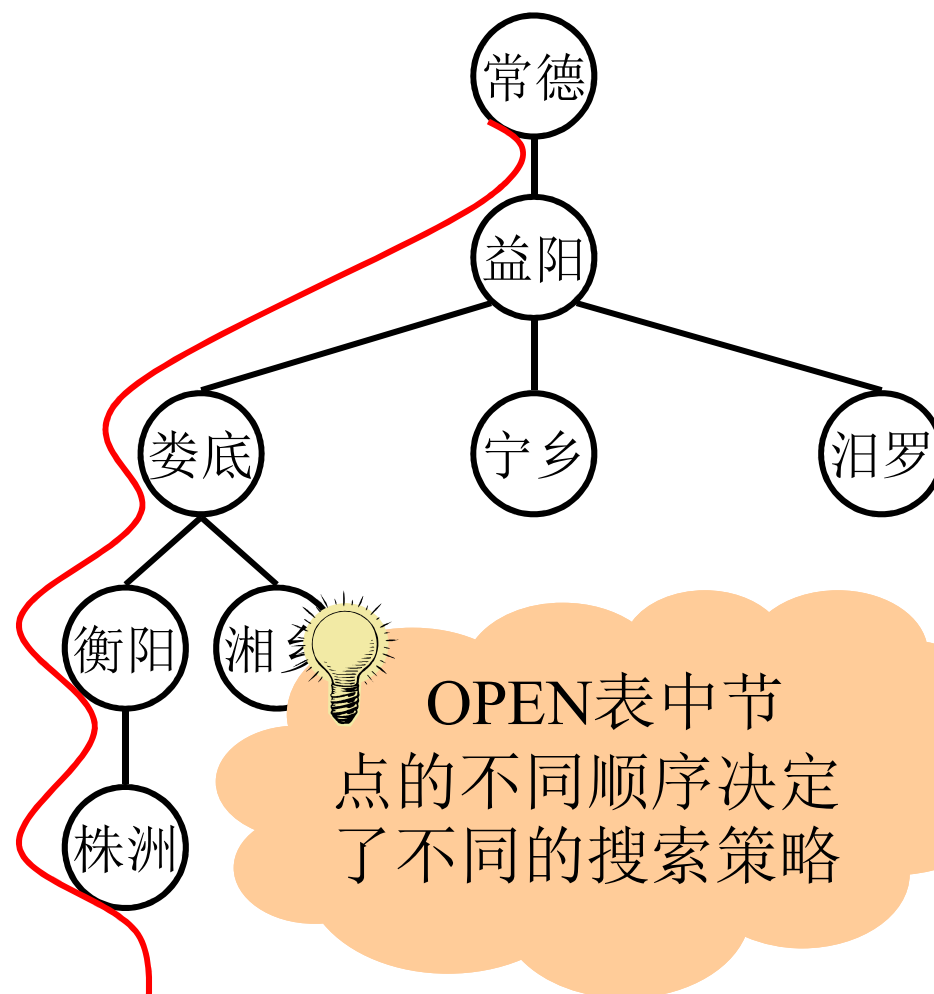
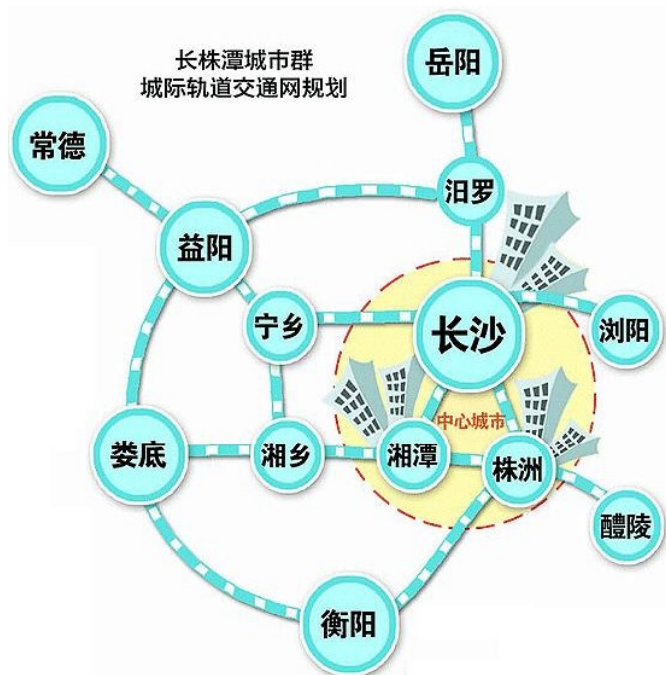


OPEN: 常德 益阳 娄底 宁乡 汨罗 衡阳 湘乡 长沙 岳阳

CLOSED:

**OPEN表中节点先进先出(FIFO)——队列(Queue)**

# 图的搜索过程



OPEN: 常德 益阳 娄底 衡阳 株洲

CLOSED: 宁乡 汨罗 湘乡

**OPEN表中节点后进先出(LIFO)——栈(Stack)**

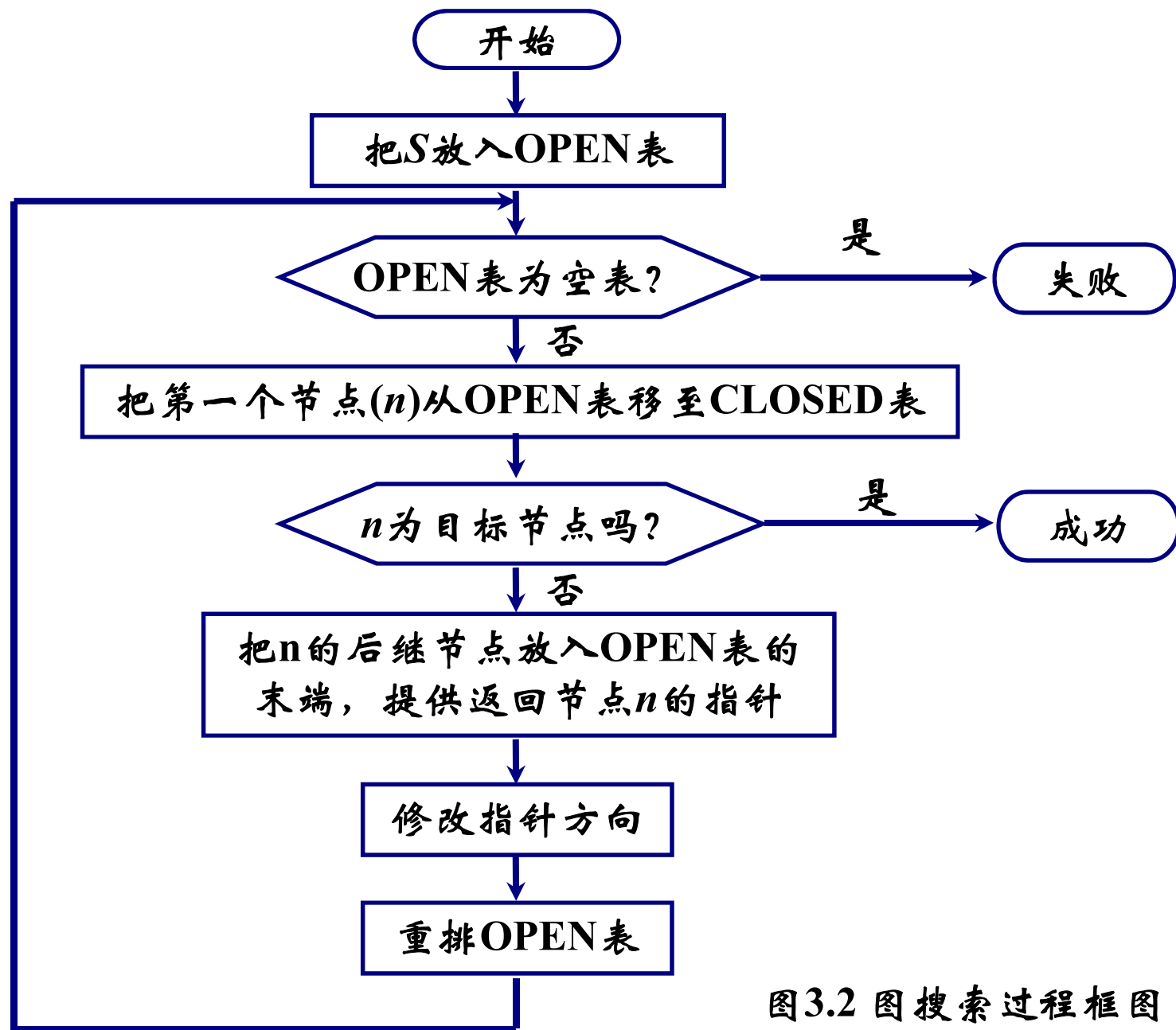


图3.2 图搜索过程框图

## 3.2 盲目搜索 (Blind Search)

- ⊕ 特点：没有先验知识，不需重排OPEN表
- ⊕ 种类：宽度优先、深度优先、等代价搜索等

### 3.2.1 宽度优先搜索(Breadth-first search)

#### ❖ 定义

以接近起始节点的程度逐层扩展节点的搜索方法。

#### ❖ 特点：

一种高代价搜索，但若有解存在，则必能找到它。

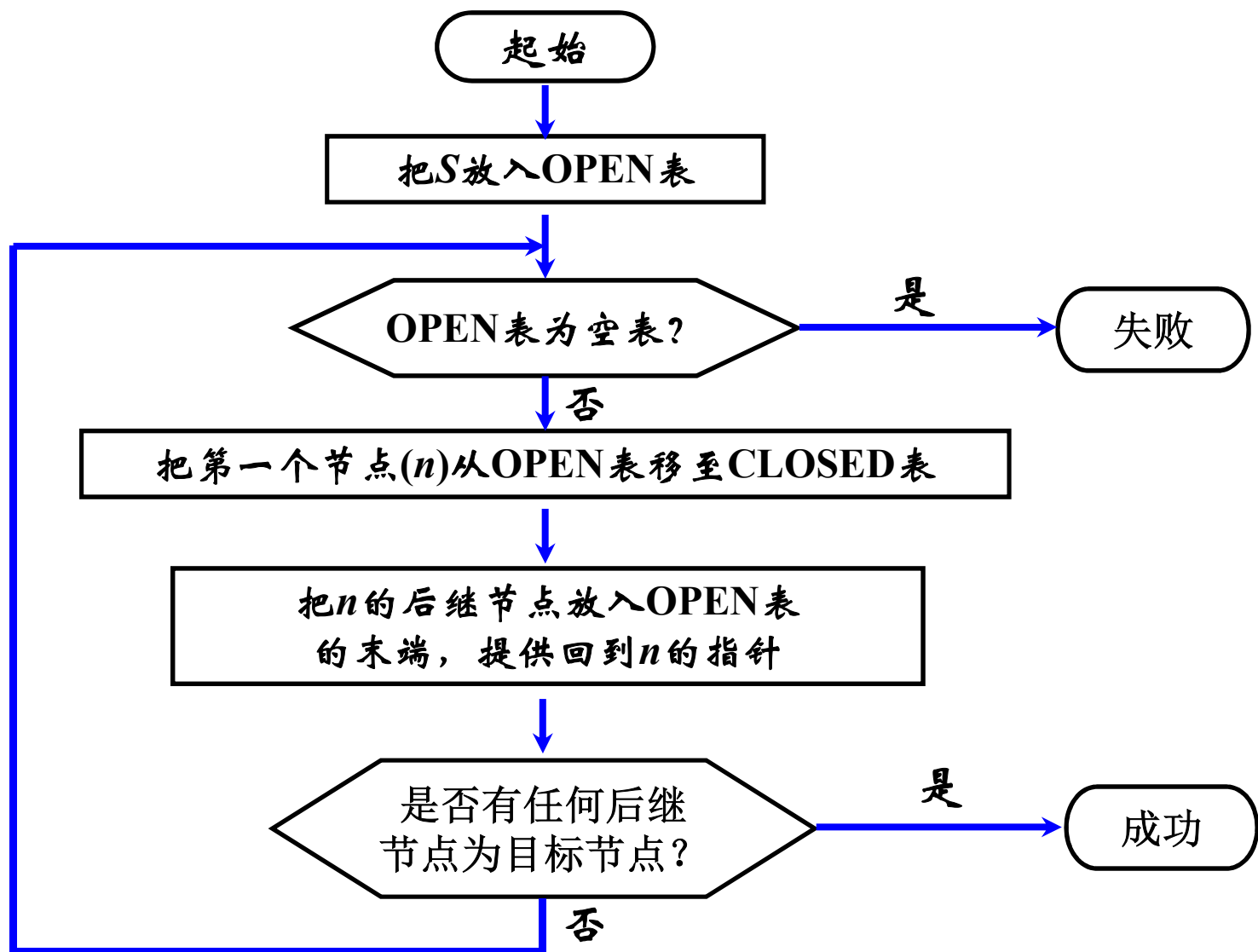


图3.4 宽度优先算法框图



例

八数码难题 (8-puzzle problem)

2	8	3
1		4
7	6	5

(start state)



1	2	3
8		4
7	6	5

(goal state)

规定：将牌移入空格的顺序为：从空格左边开始  
顺时针旋转。不许斜向移动，也不返回先辈节点。  
从下图可见，要扩展26个节点，共生成46个节点  
之后才求得解（目标节点）。





### 3.2.2 深度优先搜索（Depth-first Search）

- ❖ 定义

首先扩展最新产生的(即最深的)节点。

- ❖ 算法

防止搜索过程沿着无益的路径扩展下去，往往给出一个节点扩展的最大深度——深度界限。

与宽度优先搜索算法最根本的不同在于：将扩展的后继节点放在OPEN表的前端。（算法框图见教材）

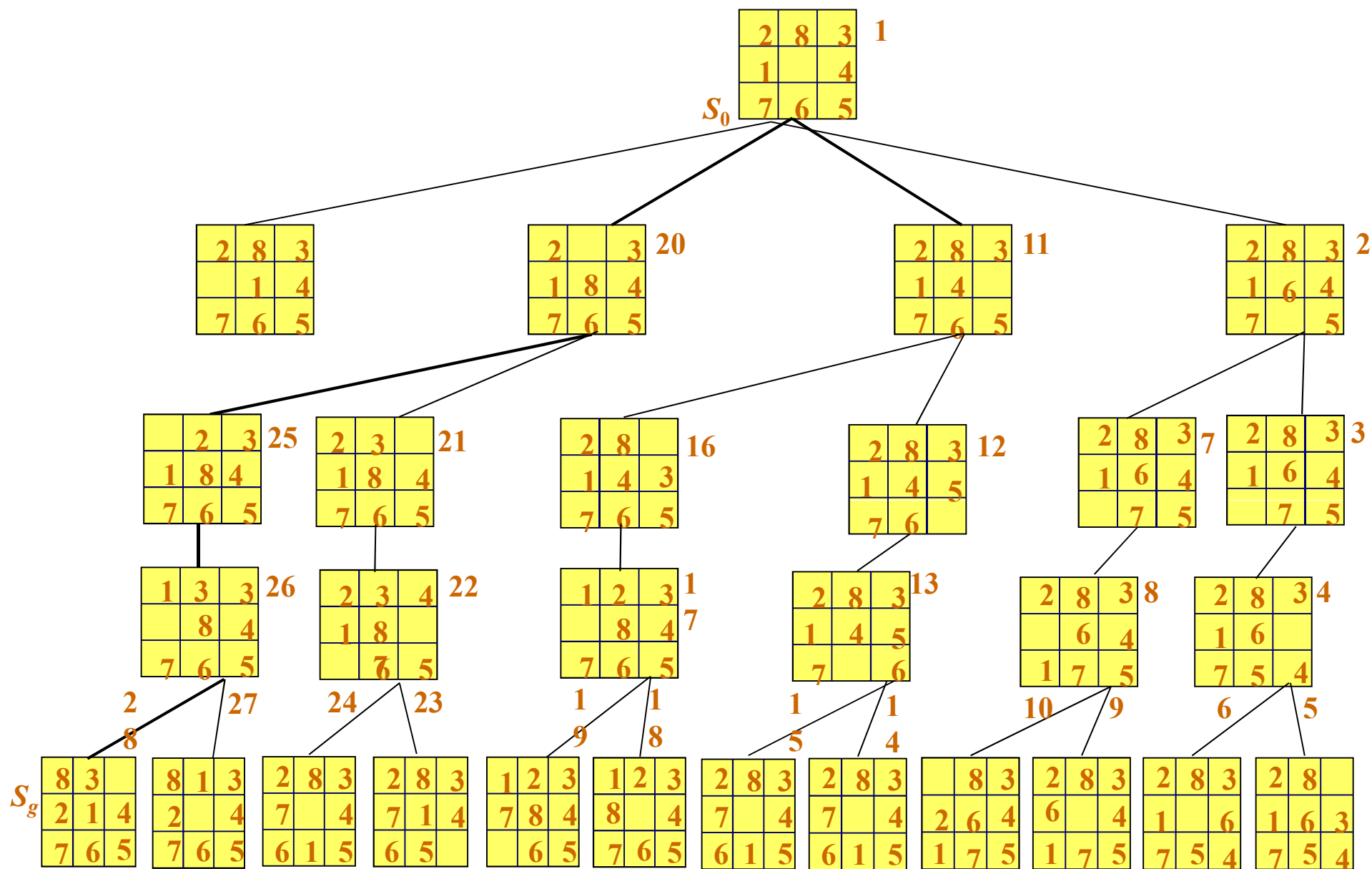
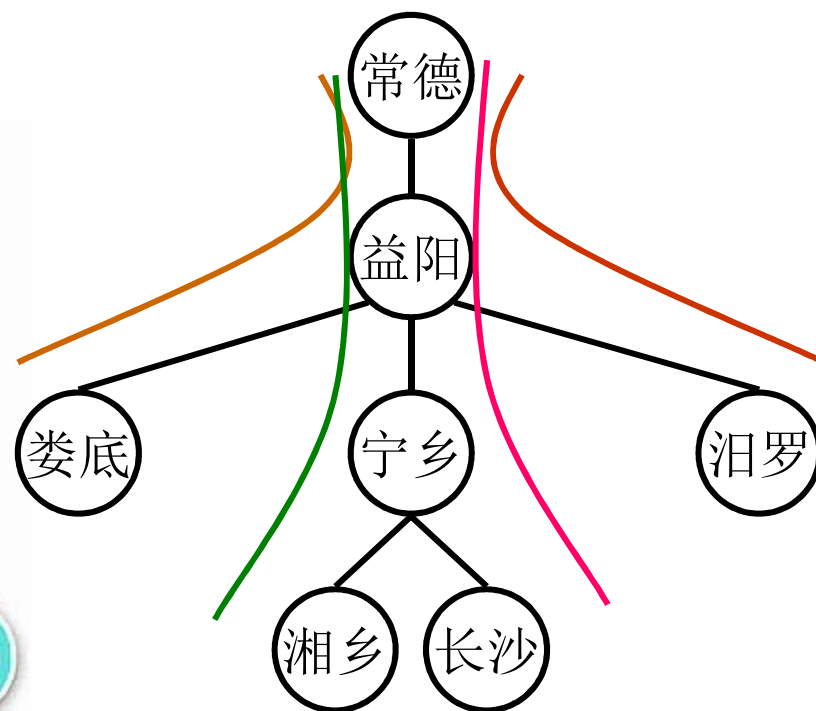


图3.8 八数码难题的有界深度优先搜索树

- ❖ 八数码难题以棋子的移动次数作为代价，每次状态的转换，代价相同，宽度优先可找到最优解
- ❖ 很多时候转换到不同状态的代价可能不同
  - ❖ 通常希望找到最优解（最小代价）



1. 在常-益-娄，常-益-宁，常-益-汨三条线路中选择最短的
2. 在接下来的四条线路中再选择最短的

### 3.2.3 等代价搜索 (Uniform cost search)

#### ❖ 定义

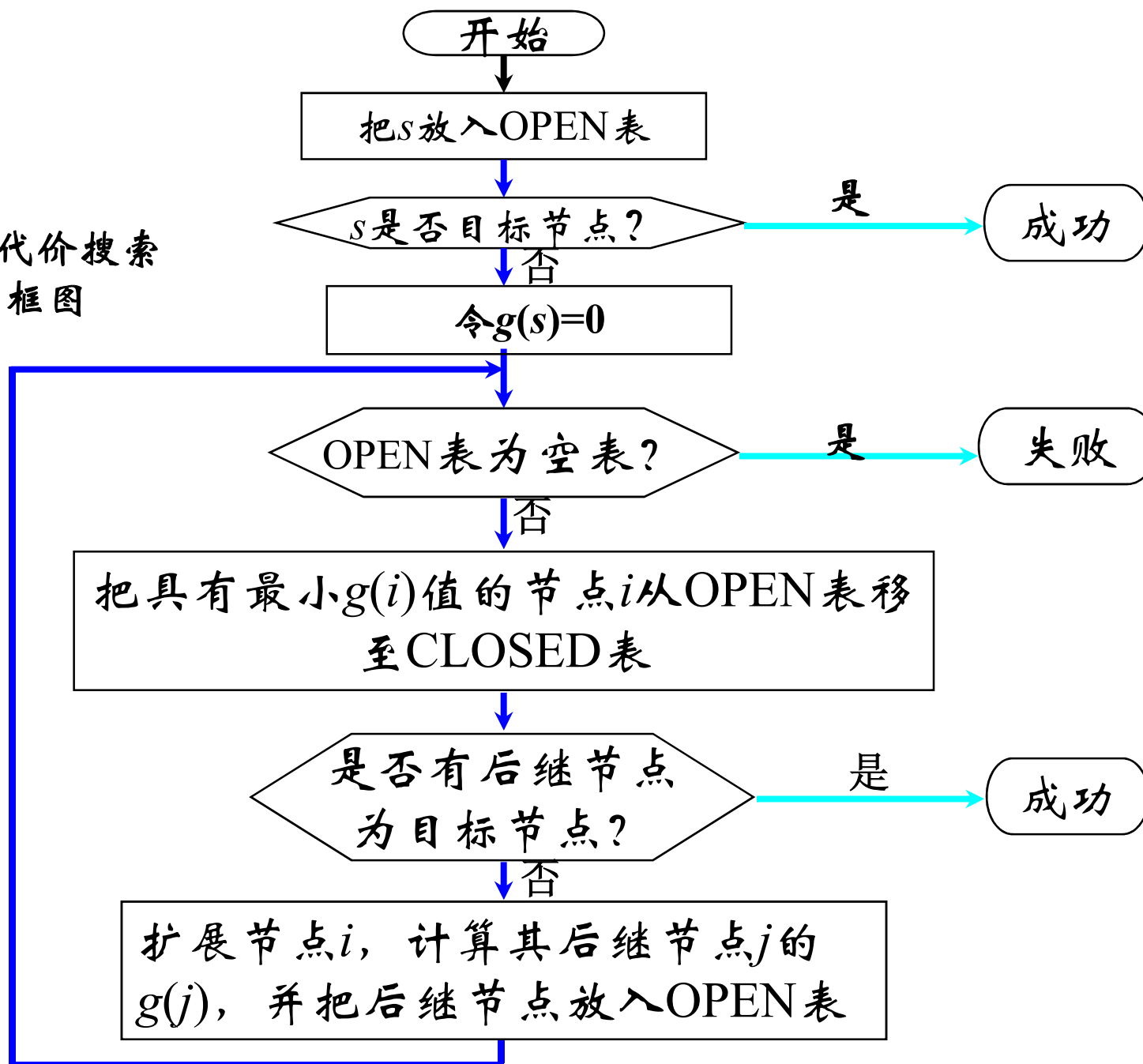
是宽度优先搜索的一种推广，不是沿着等长度路径断层进行扩展，而是沿着等代价路径断层进行扩展。

搜索树中每条连接弧线上的有关代价,表示时间、距离等花费。

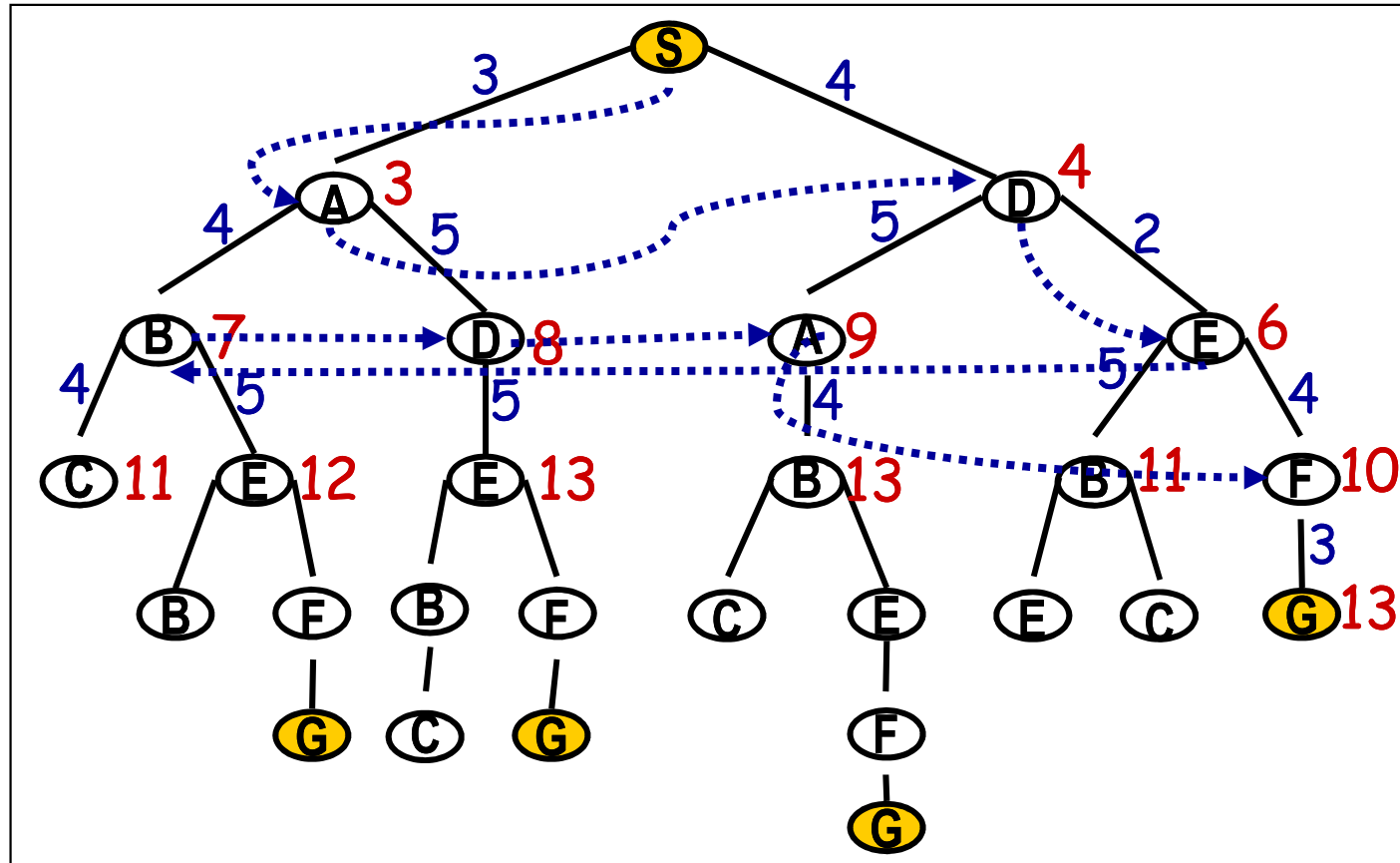
#### ❖ 算法

若所有连接弧线具有相等代价，则简化为宽度优先搜索算法。

图3.9 等代价搜索  
算法框图



# 等代价搜索





# 是什么影响了搜索的效率？

## ✿ 八数码难题

2	8	3
1		4
7	6	5

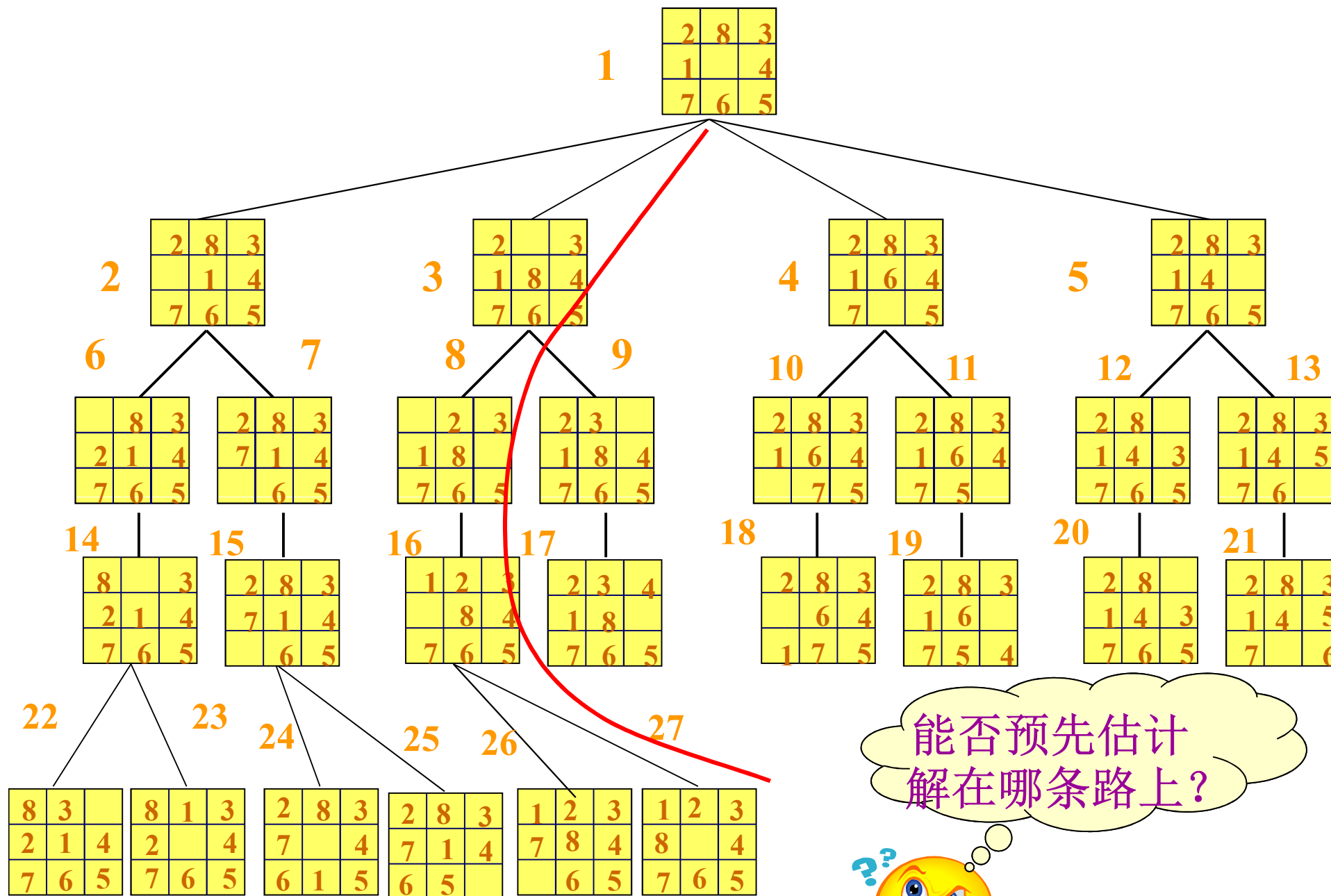
(初始状态)



1	2	3
8		4
7	6	5

(目标状态)

操作：空格上移，空格下移，空格左移，空格右移



能否预先估计  
解在哪条路上？



## 3.3 启发式搜索 (Heuristic Search)

- ✿ 特点：重排OPEN表，选择最有希望的节点加以扩展
- ✿ 种类：有序搜索、A\*算法等

### 3.3.1 启发式搜索策略和估价函数

- ✿ 盲目搜索可能带来组合爆炸
- ✿ 启发式信息 (Heuristic information)  
用来加速搜索过程的有关问题领域的特征信息。

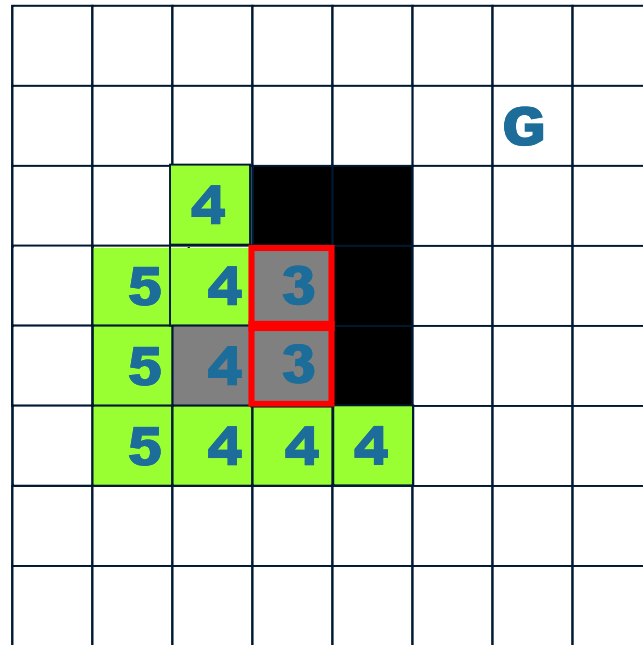
- ❖ 引入估价函数(*evaluation function*)来估计节点位于解路径上的“希望”，函数值越小“希望”越大
- ❖ 搜索过程中按照估价函数的大小对OPEN表排序, 每次选择估价函数值最小的节点作为下一步考察的节点
- ❖ At the heart of such algorithms there is the concept of a *heuristic function*(启发函数)
- ❖ 与前面将搜索策略统一到“图一般搜索策略”的框架下类似，将一般策略稍作修改，可建立有信息搜索的一般搜索策略——“最佳优先搜索(Best First Search)”，也称有序搜索

## Eg. Cost as True Distance

				2	1	0	
			3	2	1	1	
	5	4	3	2	2	2	
	5	4	3	3	3		
	5	4	4	4			

优先选择距离目标最近的点

## Eg. Cost as True Distance (little change)



# 估价函数

- ✚ 是启发式搜索中最重要的因素
- ✚ 启发式搜索和盲目搜索的不同就体现在对OPEN表按估价函数的大小排序
- ✚ 不同的估价函数所体现出来的搜索效率不同，甚至天差地远
- ✚ 不同的估价函数也决定了不同的启发式搜索算法

# 估价函数

- ✚ 不同的估价函数定义对应于不同的算法

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

- ✚  $h(x)=0$ : UCS, 非启发式算法
- ✚  $g(x)=0$ : 贪婪搜索, 无法保证找到解
- ✚ 即使采用同样的形式  $f(x) = g(x) + h(x)$ , 不同的定义和不同的值也会影响搜索过程



## 估价函数对算法的影响

### 例：八数码问题

❏  $f(x) = g(x) + h(x)$

❏  $g(x)$ : 从初始状态到 $x$ 需要进行的移动操作的次数

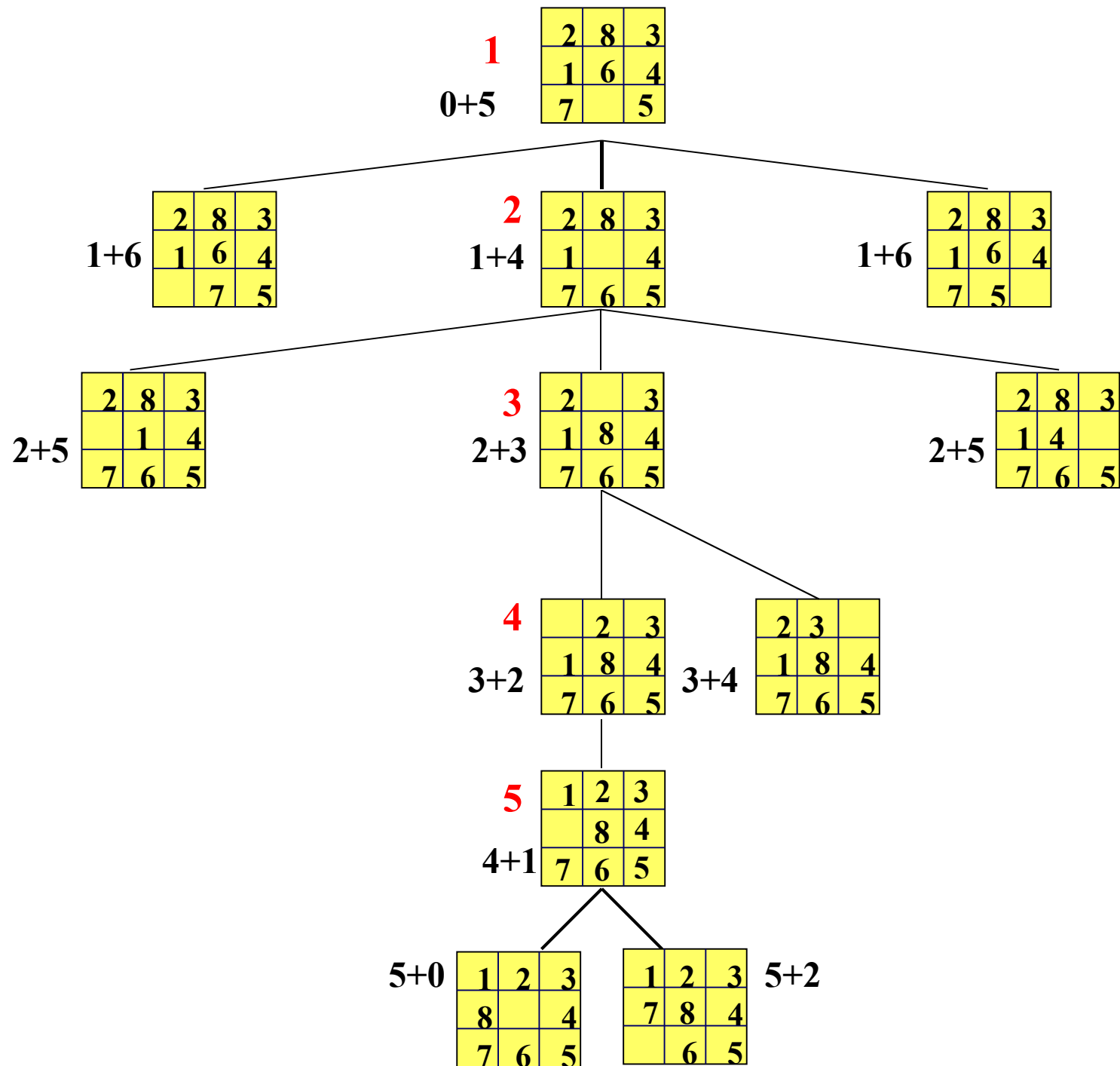
❏  $h(x)$ : 所有棋子与目标位置的曼哈顿距离之和

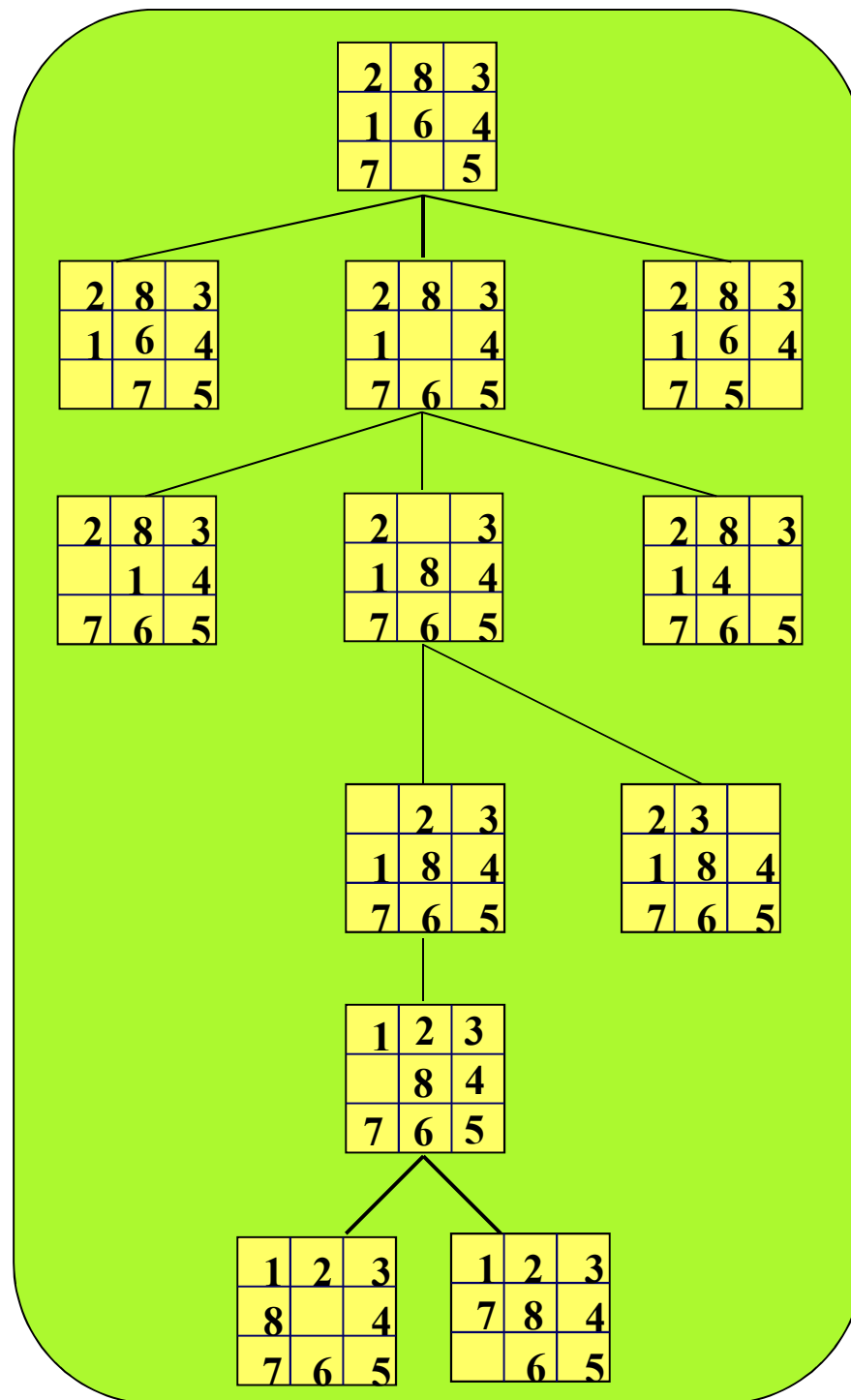
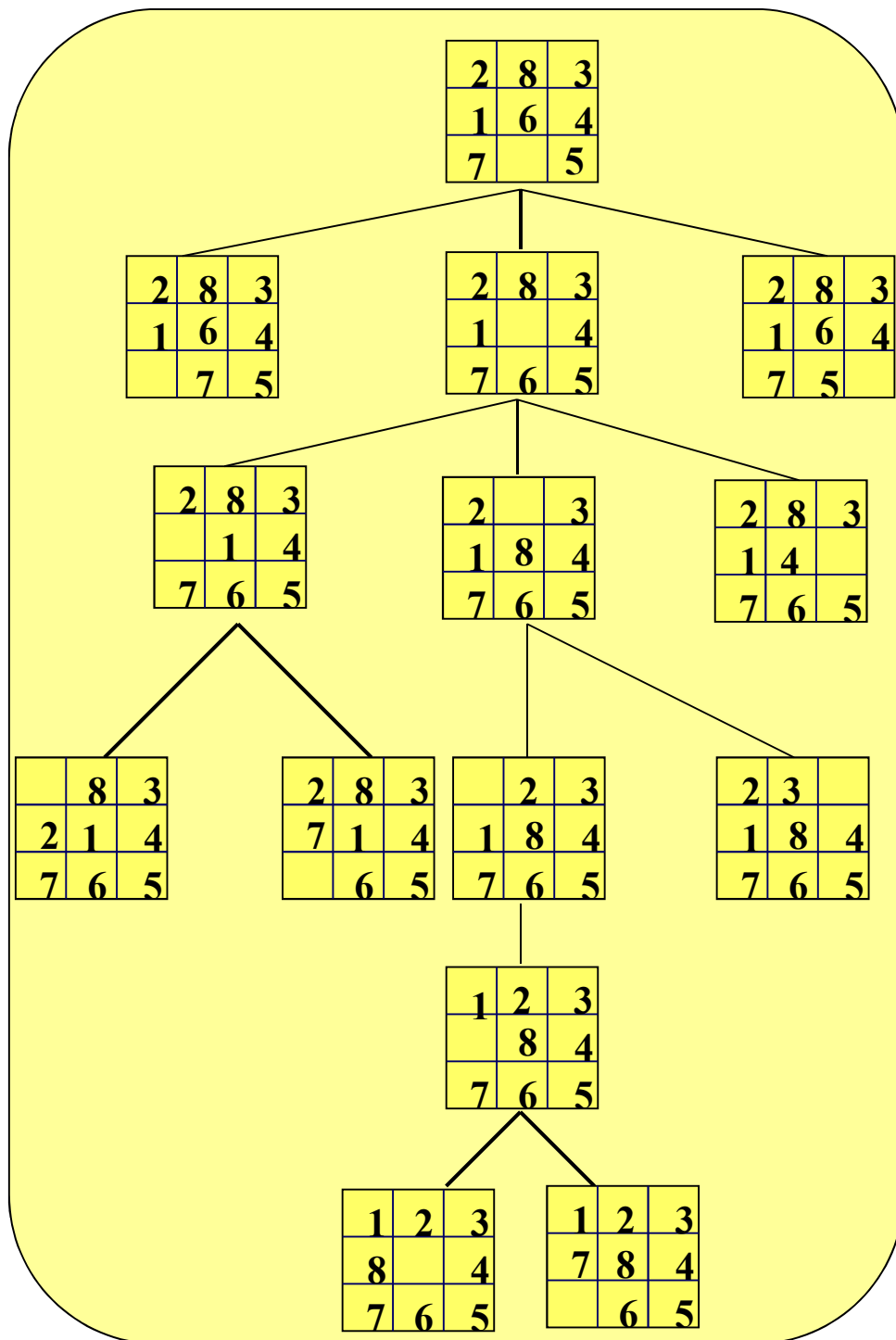
🟢 曼哈顿距离: 两点之间水平距离和垂直距离之和

🟢 仍满足估价函数的限制条件

2	8	3
7	1	4
	6	5

$$h(x) = 2 + 1 + 1 + 2 = 6$$





### 3.3.2 有序搜索 (Ordered Search)

#### ❖ 实质

选择OPEN表上具有最小 $f$ 值的节点作为下一个要扩展的节点。

❖ 算法

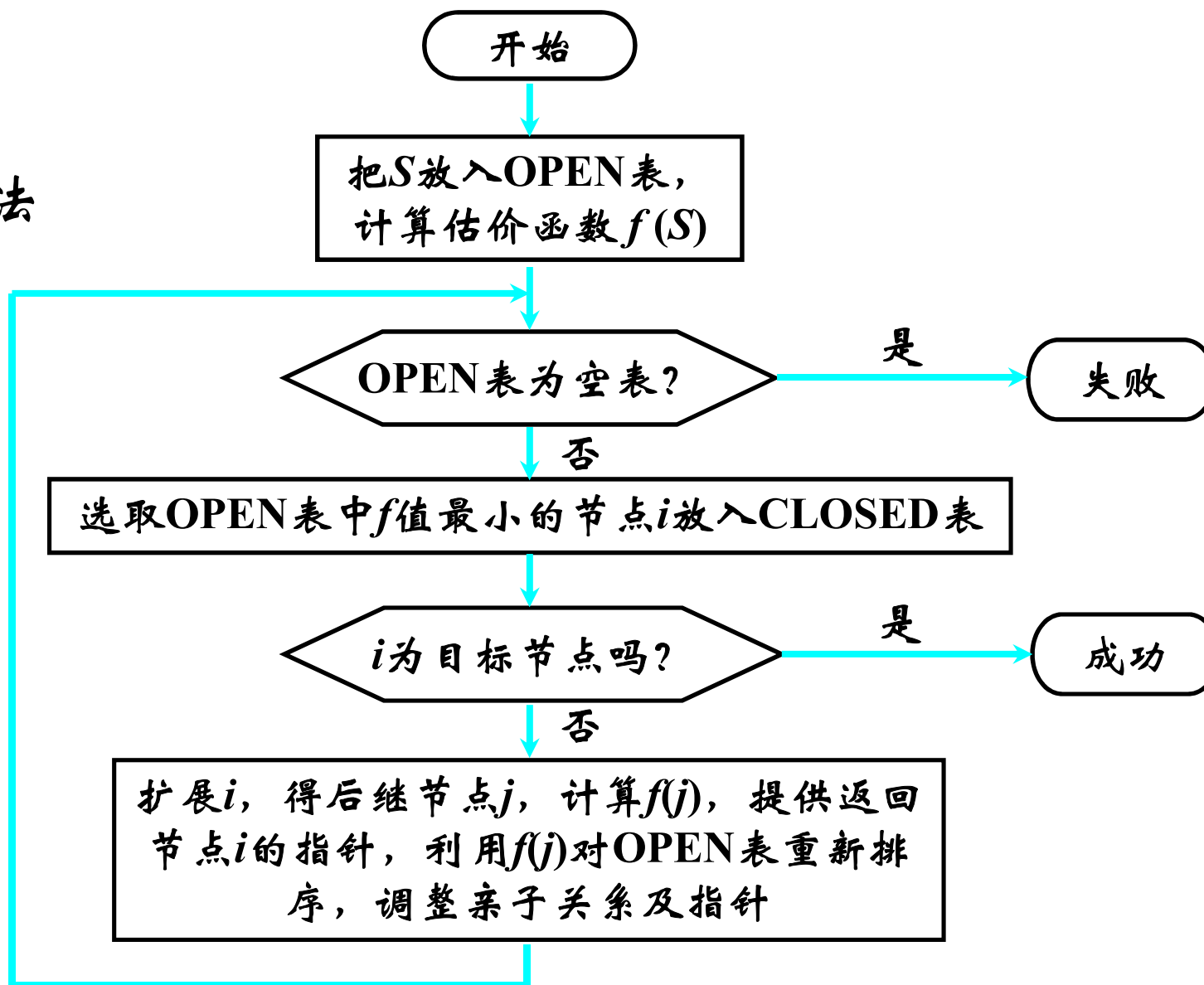


图3.10 有序搜索算法框图

## Example

八数码难题 (8-puzzle problem)

2	8	3
1	6	4
7		5



1	2	3
8		4
7	6	5

(初始状态)

(目标状态)

八数码难题的有序搜索树见下图:



### 3.3.3 A\*算法 (Algorithm A\*)

#### ✚ 估价函数的定义

对节点 $n$ 定义 $f^*(n)=g^*(n)+h^*(n)$ ,表示从 $S$ 开始约束通过节点 $n$ 的一条最佳路径的代价。

希望估价函数 $f$  定义为:  $f(n)=g(n)+h(n)$

——  $g$ 是 $g^*$ 的估计,  $h$ 是 $h^*$ 的估计

#### ✚ A\*算法的定义

定义1 在GRAPHSEARCH过程中, 如果第8步的重排OPEN表是依据 $f(x)=g(x)+h(x)$ 进行的, 则称该过程为A算法。

定义2 在A算法中, 如果对所有的 $x$ 存在 $h(x) \leq h^*(x)$ , 则称 $h(x)$ 为 $h^*(x)$ 的下界, 它表示某种偏于保守的估计。

定义3 采用 $h^*(x)$ 的下界 $h(x)$ 为启发函数的A算法, 称为A\*算法。当 $h=0$ 时, A\*算法就变为有序搜索算法。





Nils Nilsson  
尼尔逊



Bertram Raphael  
拉斐尔

- ✚ 1964年，尼尔逊提出一种算法以提高最短路径搜索的效率，被称为A1算法
- ✚ 1967年，拉斐尔改进了A1算法，称为A2算法

## ✿ 特征:

### ▣ 估价函数

$$f(x) = \underline{g(x)} + \underline{h(x)}$$

从起始状态到当前状态 $x$ 的代价

从当前状态 $x$ 到目标状态的估计代价（启发函数）

$$f(x) = g(x) \text{ —— UCS}$$

$$f(x) = h(x) \text{ —— 贪婪算法}$$

虽提高了算法效率，但不能保证找到最优解

### 3.3.3 A\*算法



Peter Hart  
彼得.哈特

- ✚ 1968年，彼得.哈特对A算法进行了很小的修改，并证明了当估价函数满足一定的限制条件时，算法一定可以找到最优解
- ✚ 估价函数满足一定限制条件的算法称为A\*算法

A\*算法的限制条件  $f(x) = \underbrace{g(x)}_{\text{大于0}} + \underbrace{h(x)}_{\text{不大于}x\text{到目标的实际代价}}$

# A\*算法示例

## ❖ 八数码问题

## ❖ 估价函数的定义

❖  $f(x) = g(x) + h(x)$

❖  $g(x)$ : 从初始状态到 $x$ 需要进行的移动操作的次数

❖  $h(x)$  =  $x$ 状态下错放的棋子数

满足限制条件

错放的棋子越  
少越好!



2	8	3
7	1	4
	6	5

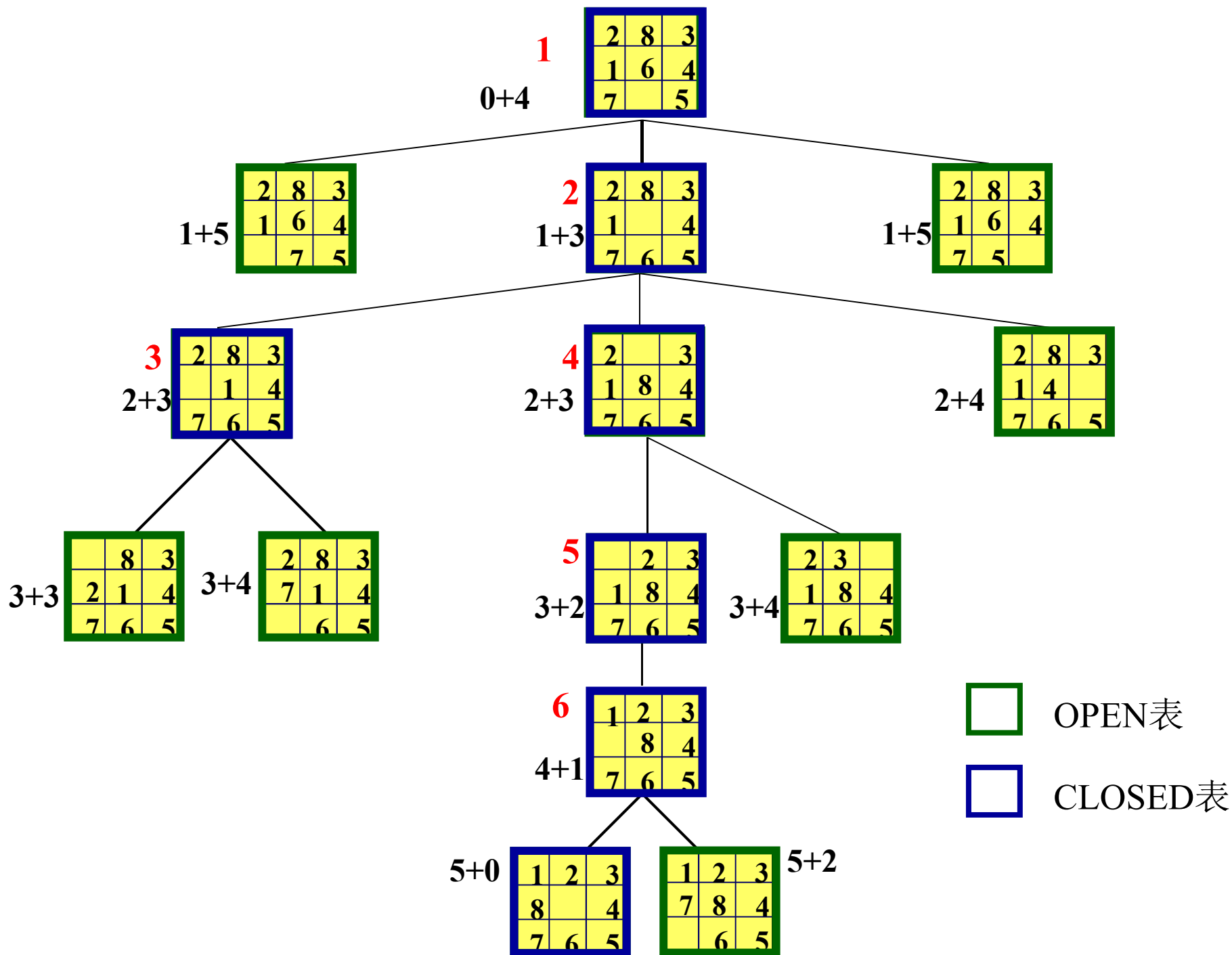
$$h(x)=4$$

1	2	3
7	8	4
	6	5

$$h(x)=2$$

1	2	3
	8	4
7	6	5

$$h(x)=1$$



## 3.4 消解原理 (Resolution Principle)

回顾:

原子公式 (atomic formulas)

文字—一个原子公式及其否定。

子句—由文字的析取组成的合适公式。

消解—对谓词演算公式进行分解和化简, 消去一些符号, 以求得导出子句, 又称归结。

# 消解原理（归结原理）

## Resolution Principle

- ❖ 美国数学家鲁滨逊提出消解原理（1965年）
- ❖ 基本的出发点：要证明一个命题为真都可以通过证明其否命题为假来得到
- ❖ 将多样的推理规则简化为一个——  
消解



鲁滨逊

# 什么叫消解

## 例1:

小王说他下午或者去图书馆或者在家休息

小王没去图书馆

R——小王下午去图书馆

S——小王下午在家休息

$$\left. \begin{array}{l} R \vee S \\ \sim R \end{array} \right\} \Rightarrow S$$

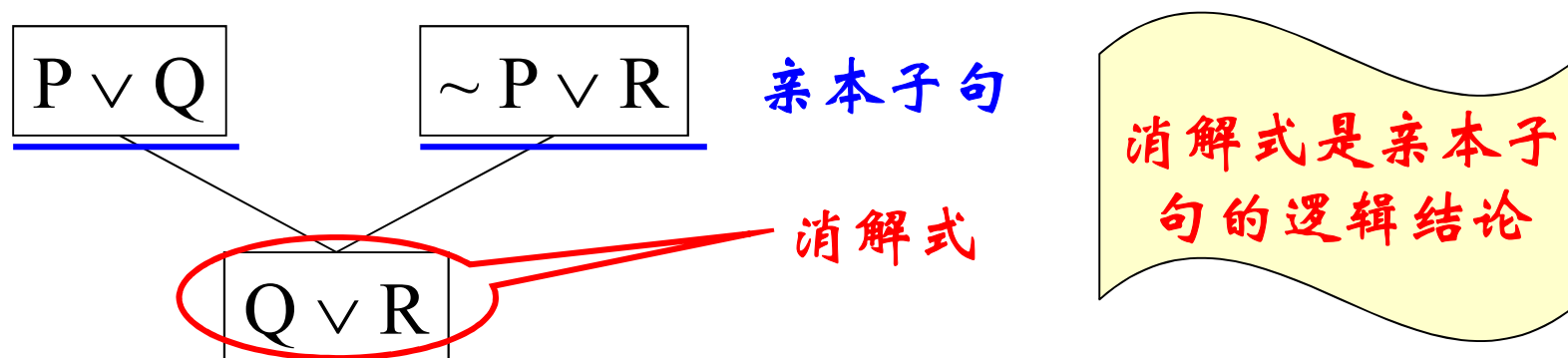
## 例2:

如果今天不下雨，我就去你家

今天没有下雨



# 什么叫消解



- ❖ 消解只能在仅含否定和析取联接词的公式（子句）间进行
- ❖ 必须先把公式化成规范的形式（范式，子句集）

# 含变量的消解

## 例：苏格拉底论断

凡人都会死.  $(\forall x) (\text{Man}(x) \Rightarrow \text{Mortal}(x))$

苏格拉底是人.  $\text{Man}(\text{Socrates})$

如何得到结论：苏格拉底会死.  $\text{Mortal}(\text{Socrates})$

## 要完成消解还面临几个问题

❏ “ $\Rightarrow$ ” 和 “ $\forall$ ” 必须去掉

❏  $\text{Man}(x) \Rightarrow \text{Mortal}(x) \Leftrightarrow \sim \text{Man}(x) \vee \text{Mortal}(x)$

❏ “ $\forall$ ” 怎么办？

} 化为子句集

❏ 如果能去掉 “ $\forall$ ”， $\sim \text{Man}(x)$  和  $\text{Man}(\text{Socrates})$

也不能构成互补对，形式不一样，怎么办？

} 置换与合一

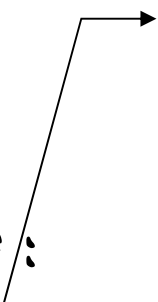
### 3.4.1 子句集的求取

步骤：共9步。

例：将下列谓词演算公式化为一个子句集

$$(\forall x) \{ P(x) \Rightarrow \{ (\forall y) [P(y) \Rightarrow P(f(x,y))] \wedge \\ \sim(\forall y) [Q(x,y) \Rightarrow P(y)] \} \}$$

开始：


$$(1) (\forall x) \{ \sim P(x) \vee \{ (\forall y) [\sim P(y) \vee P(f(x,y))] \} \\ \wedge \sim(\forall y) [\sim Q(x,y) \vee P(y)] \}$$

(1) 消去蕴涵符号

只应用  $\vee$  和  $\sim$  符号，以  $\sim A \vee B$  替换  $A \rightarrow B$ 。

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

(2) 减少否定符号的辖域

每个否定符号 $\sim$ 最多只用到一个谓词符号上，并反复应用狄·摩根定律。

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

(3) 对变量标准化

对哑元（虚构变量）改名，以保证每个量词有其自己唯一的哑元。

$$(4) (\forall x) \{ \sim P(x) \vee \{ (\forall y) [\sim P(y) \vee P(f(x,y))] \} \\ \wedge [Q(x,g(x)) \wedge \sim P(g(x))] \} \}$$

式中,  $w=g(x)$  为一Skolem函数。

#### (4) 消去存在量词

以Skolem函数代替存在量词内的约束变量, 然后消去存在量词

#### (5) 化为前束形

把所有全称量词移到公式的左边, 并使每个量词的辖域包括这个量词后面公式的整个部分:

前束形 = {前缀} {母式}  
           全称量词串 无量词公式

$$(5) (\forall x)(\forall y) \{ \sim P(x) \vee \{ [\sim P(y) \vee P(f(x,y))] \\ \wedge [Q(x,g(x)) \wedge \sim P(g(x))] \} \}$$

$$(6) (\forall x)(\forall y) \{ [\sim P(x) \vee \sim P(y) \vee P(f(x,y))] \wedge [\sim P(x) \vee Q(x,g(x))] \wedge [\sim P(x) \vee \sim P(g(x))] \}$$

(6) 把母式化为合取范式

任何母式都可写成由一些谓词公式和(或)谓词公式的否定的析取的有限集组成的合取。

$$(7) \{ [\sim P(x) \vee \sim P(y) \vee P(f(x,y))] \wedge [\sim P(x) \vee Q(x,g(x))] \wedge [\sim P(x) \vee \sim P(g(x))] \}$$

(7) 消去全称量词

所有余下的量词均被全称量词量化了。消去前缀，即消去明显出现的全称量词。

$$\begin{aligned}
 (8) \quad & \sim P(x) \vee \sim P(y) \vee P(f(x,y)) \\
 & \sim P(x) \vee Q(x,g(x)) \\
 & \sim P(x) \vee \sim P(g(x))
 \end{aligned}$$

(8) 消去连词符号  $\wedge$

用  $\{A,B\}$  代替  $(A \wedge B)$ ，消去符号  $\wedge$ 。最后得到一个有限集，其中每个公式是文字的析取。

(9) 更换变量名称

可以更换变量符号的名称，使一个变量符号不出现在一个以上的子句中。

$$\begin{aligned}
 (9) \quad & \sim P(x_1) \vee \sim P(y) \vee P[f(x_1,y)] \\
 & \sim P(x_2) \vee Q[x_2,g(x_2)] \\
 & \sim P(x_3) \vee \sim P[g(x_3)]
 \end{aligned}$$

## 3.4.2 消解推理规则 (Resolution Inference Rules)

### ✿ 消解式的定义

- ✦ 令 $L_1, L_2$ 为两任意原子公式； $L_1$ 和 $L_2$ 具有相同的谓词符号，但一般具有不同的变量。
- ✦ 已知两子句 $L_1 \vee \alpha$ 和 $\sim L_2 \vee \beta$ ，如果 $L_1$ 和 $L_2$ 具有最一般合一者 $\sigma$ ，那么通过消解可以从这两个父辈子句推导出一个新子句 $(\alpha \vee \beta)\sigma$ 。这个新子句叫做消解式。



## ❖ 证明

Resolution:

$$\left. \begin{array}{l} C_1 = L \vee \alpha \\ C_2 = \sim L \vee \beta \end{array} \right\} \Rightarrow C_{12} = \alpha \vee \beta$$

**Proof:**

$$\because C_1 = \alpha \vee L \Leftrightarrow \sim \alpha \Rightarrow L \quad \text{且}, C_2 = \sim L \vee \beta \Leftrightarrow L \Rightarrow \beta$$

$$\therefore C_1 \wedge C_2 \Leftrightarrow (\sim \alpha \Rightarrow L) \wedge (L \Rightarrow \beta)$$

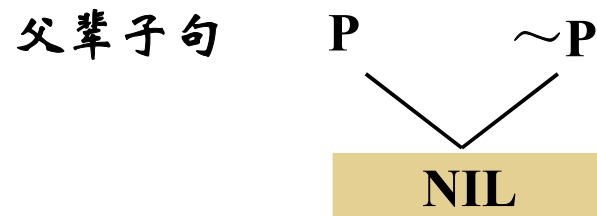
$$[(\sim \alpha \Rightarrow L) \wedge (L \Rightarrow \beta)] \Rightarrow [\sim \alpha \Rightarrow \beta]$$

$$\sim \alpha \Rightarrow \beta \Leftrightarrow \alpha \vee \beta = C_{12}$$

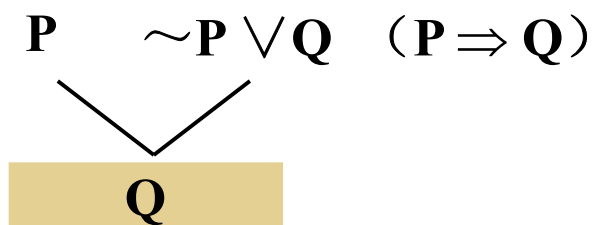
$$\therefore C_{12} = C_1 \wedge C_2$$

## 消解式例子

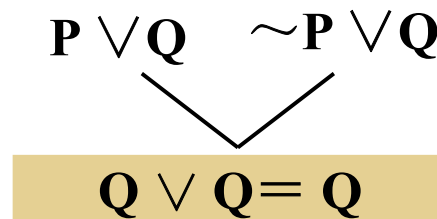
(a) 空子句 NIL Clause



(b) 假言推理 Modus ponens



(c) 合并 Combination



(d) 重言式 Tautologies

$$\begin{array}{c} P \vee Q \quad \sim P \vee \sim Q \\ \swarrow \quad \searrow \\ Q \vee \sim Q \end{array}$$

$$\begin{array}{c} P \vee Q \quad \sim P \vee \sim Q \\ \swarrow \quad \searrow \\ P \vee \sim P \end{array}$$

(e) 链式 (三段论) Chain

$$\begin{array}{c} \sim P \vee Q \quad \sim Q \vee R \\ \swarrow \quad \searrow \\ \sim P \vee R \end{array}$$

### 3.4.3 含有变量的消解式 (Resolvent with Variable)

#### ❖ 含有变量的子句之消解式

要把消解推理规则推广到含有变量的子句，必须找到一个作用于父辈子句的置换，使父辈子句含有互补文字。

#### ❖ 例子

$$\begin{array}{ccc} P[x, f(y)] \vee Q(x) \vee R[f(a), y] & & \sim P[f(f(a)), z] \vee R(z, w) \\ & \searrow \quad \swarrow & \\ & \sigma = \{f(f(a))/x, f(y)/z\} & \\ & Q[f(f(a))] \vee R(f(a), y) \vee R(f(y), w) & \end{array}$$

### 3.4.4 消解反演求解过程 (Solving Process of Resolution Refutation)

#### ✚ 消解反演

给出公式集  $\{S\}$  和目标公式  $L$

- ✚ 否定  $L$ , 得  $\sim L$ ;
- ✚ 把  $\sim L$  添加到  $S$  中去;
- ✚ 把新产生的集合  $\{ \sim L, S \}$  化成子句集;
- ✚ 应用消解原理, 力图推导出一个表示矛盾的空子句

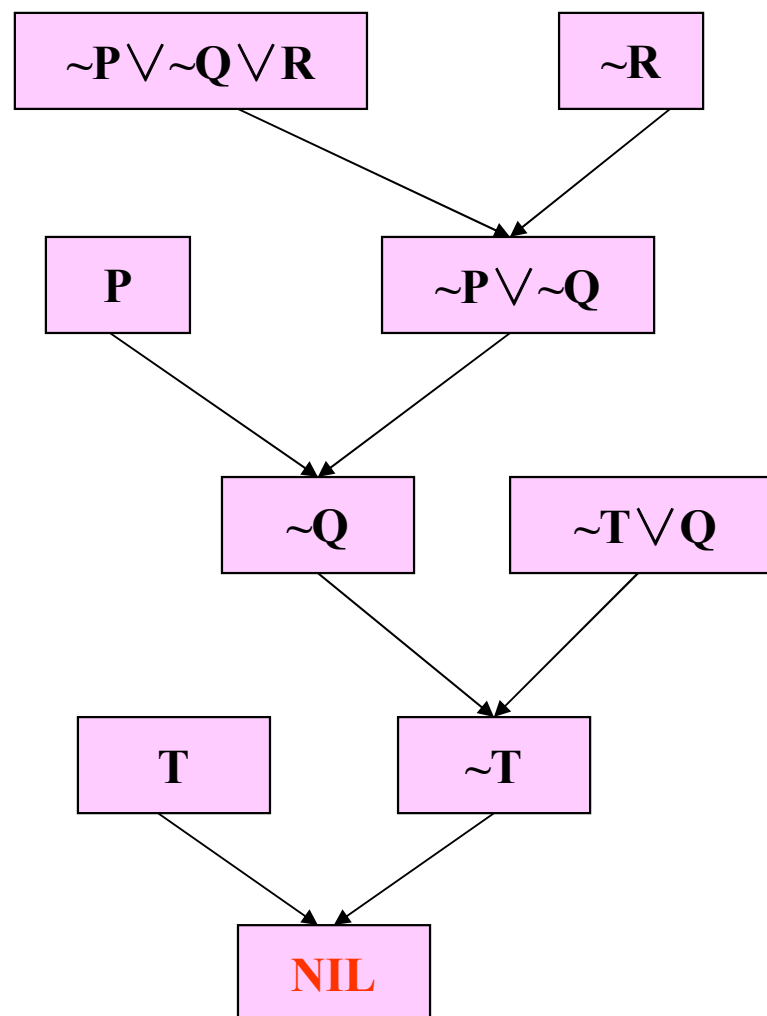
## 例 1

- 设事实的公式集合  
 $\{P, (P \wedge Q) \Rightarrow R,$   
 $(S \vee T) \Rightarrow Q, T\},$   
证明:  $R$

否定结论，将公式化为子句，  
得子句集：

- $\{P, \sim P \vee \sim Q \vee R,$   
 $\sim S \vee Q, \sim T \vee Q, T,$   
 $\sim R\}$

消解反演树



## Example2: Happy student

- ⊕ “Happy Student”: Everyone who pass the computer test and win the prize is happy. Everyone who wish study or is lucky can pass all tests. Zhang doesn’t study, but he is lucky. Every lucky person can win the prize.
- ⊕ Prove: Zhang is happy

## ✚ Solution

### ▣ Step1: first-order logic representation of the problem

Facts or Knowledge:

$$(\forall x) (\text{Pass}(x, \text{computer}) \wedge \text{Win}(x, \text{prize})) \Rightarrow \text{Happy}(x)$$

$$(\forall x) (\forall y) (\text{Study}(x) \vee \text{Lucky}(x) \Rightarrow \text{Pass}(x, y))$$

$$\sim \text{Study}(\text{zhang}) \wedge \text{Lucky}(\text{zhang})$$

$$(\forall x) (\text{Lucky}(x) \Rightarrow \text{Win}(x, \text{prize}))$$

Negation of the conclusion:  $\sim \text{Happy}(\text{zhang})$



## Step2: Convert the sentence above into clauses

(1)  $\sim \text{Pass}(x, \text{computer}) \vee \sim \text{Win}(x, \text{prize}) \vee \text{Happy}(x)$

(2)  $\sim \text{Study}(y) \vee \text{Pass}(y, z)$

(3)  $\sim \text{Lucky}(u) \vee \text{Pass}(u, v)$

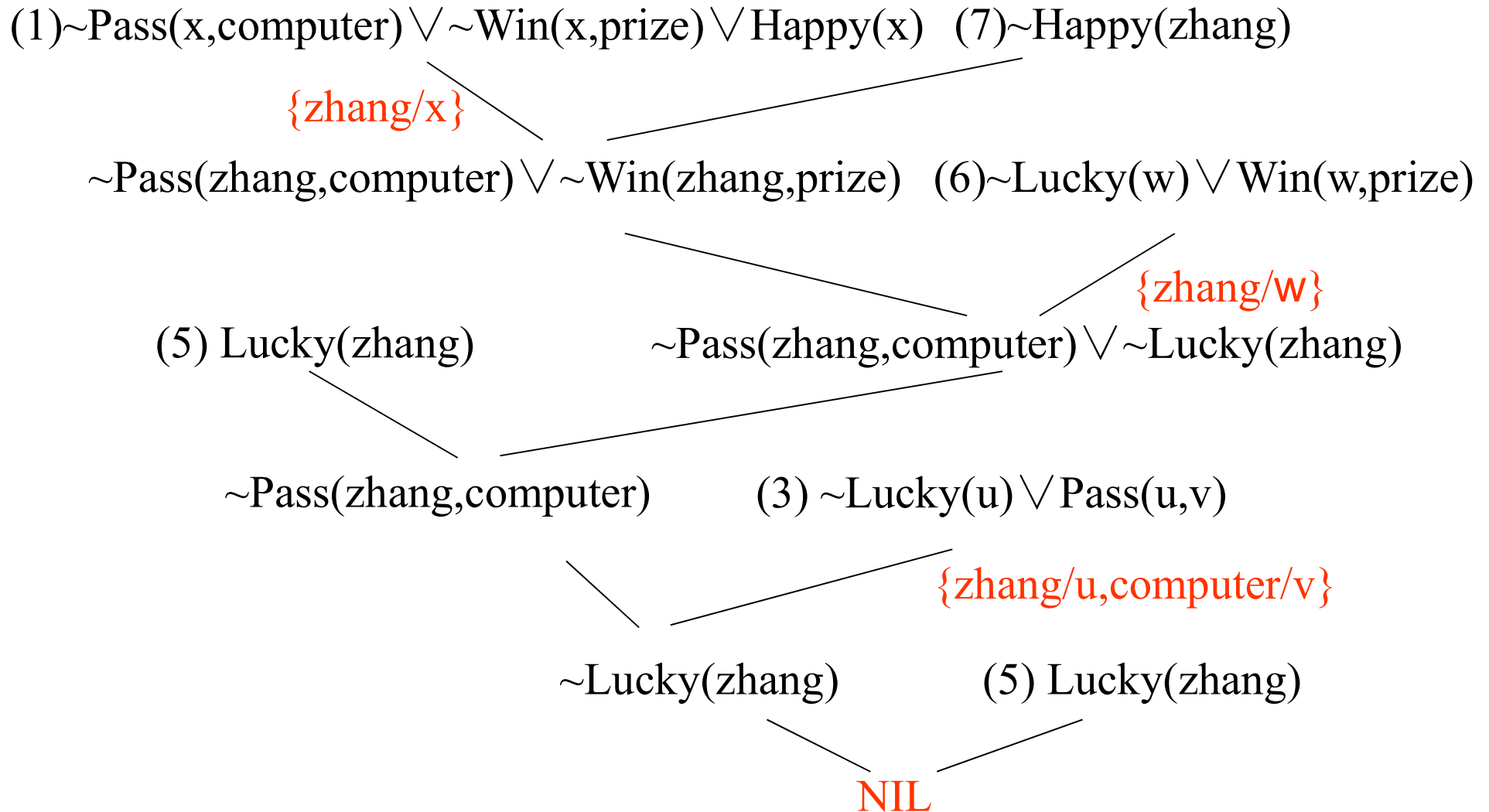
(4)  $\sim \text{Study}(\text{zhang})$

(5)  $\text{Lucky}(\text{zhang})$

(6)  $\sim \text{Lucky}(w) \vee \text{Win}(w, \text{prize})$

(7)  $\sim \text{Happy}(\text{zhang})$

### Step3: Resolve these clauses



### 例3：储蓄问题

前提：每个储蓄钱的人都获得利息。

结论：如果没有利息，那么就没有人去储蓄钱

证明：

(1) 规定原子公式：

$S(x, y)$  表示 “ $x$ 储蓄 $y$ ”

$M(x)$  表示 “ $x$ 是钱”

$I(x)$  表示 “ $x$ 是利息”

$E(x, y)$  表示 “ $x$ 获得 $y$ ”

(2) 用谓词公式表示前提和结论：

前提：  $(\forall x)[(\exists y)(S(x, y)) \wedge M(y)] \Rightarrow [(\exists y)(I(y) \wedge E(x, y))]$

结论：  $[\sim(\exists x)I(x)] \Rightarrow [(\forall x)(\forall y)(M(y) \Rightarrow \sim S(x, y))]$

### (3) 化为子句形

把前提化为子句形：

$$1) \sim S(x,y) \vee \sim M(y) \vee I(f(x))$$

$$2) \sim S(x,y) \vee \sim M(y) \vee E(x,f(x))$$

把结论的否定化为子句形：

$$3) \sim I(z)$$

$$4) S(a,b)$$

$$5) M(b)$$

#### (4) 消解反演求空子句 (NIL)

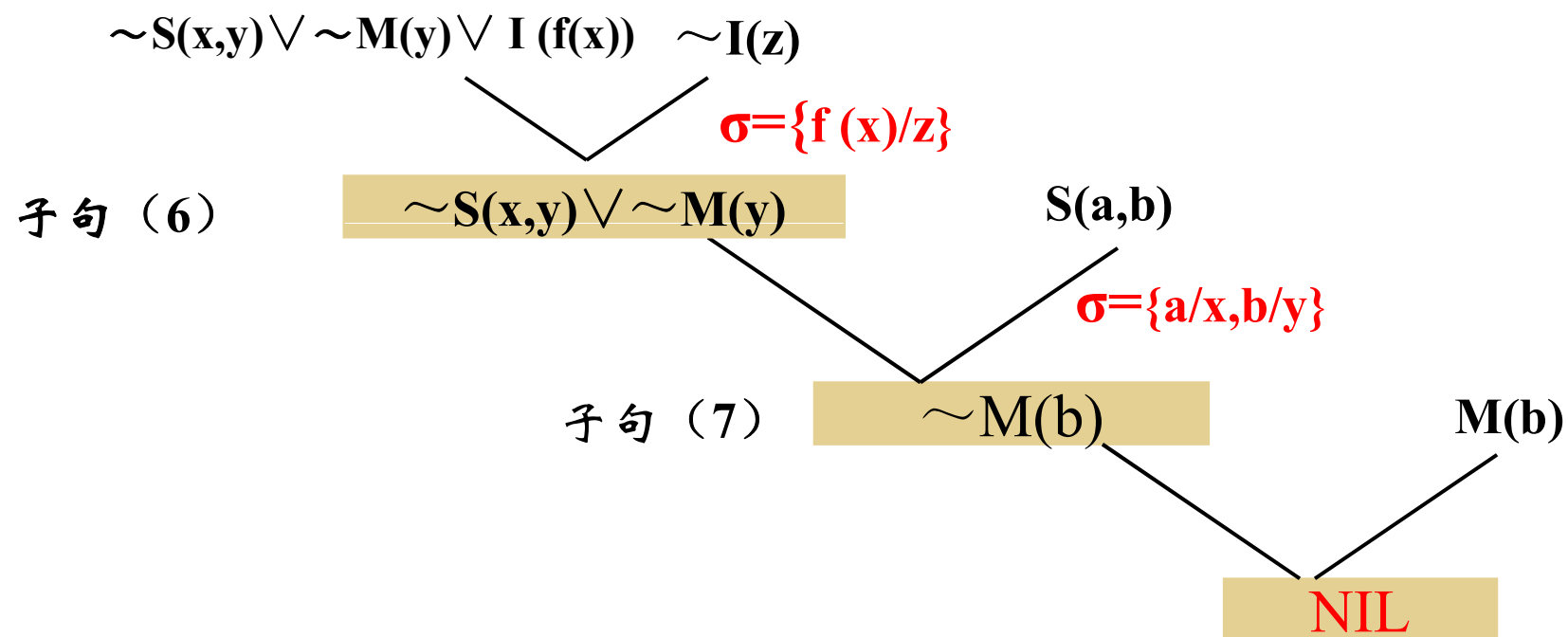


图3.13 储蓄问题反演树

## ❁ 反演求解过程

- ❁ 把由目标公式的否定产生的每个子句添加到目标公式否定之否定的子句中去。
- ❁ 按照反演树，执行和以前相同的消解，直至在根部得到某个子句止。
- ❁ 用根部的子句作为一个回答语句

## ❁ 实质

- ❁ 根部为NIL 变换为 根部为回答语句

## Example 4

✿ **Given:** “Zhang and Li are classmates; If  $x$  and  $y$  are classmates, then the classroom of  $x$  is the one of  $y$ ; Now Zhang is at 302 classroom.”    **Question:** “Now which classroom is Li at?”

✿ **Answer:**

✦ Define the predicates:

✿  $C(x, y)$      $x$  and  $y$  are classmates ;

✿  $At(x, u)$      $x$  is at  $u$  classroom.

✦ Represent the given facts as *wffs* :

✿  $C(\text{zhang}, \text{li})$

✿  $(\forall x) (\forall y) (\forall u) (C(x, y) \wedge At(x, u) \Rightarrow At(y, u))$

✿  $At(\text{zhang}, 302)$

✦ Represent the Question's negation as *wffs* :

✿  $\sim(\exists v) At(\text{li}, v)$     即求  $v = ? ? ?$

❏ Convert the given facts' *wffs* to clauses:

●  $C(\text{zhang}, \text{li})$

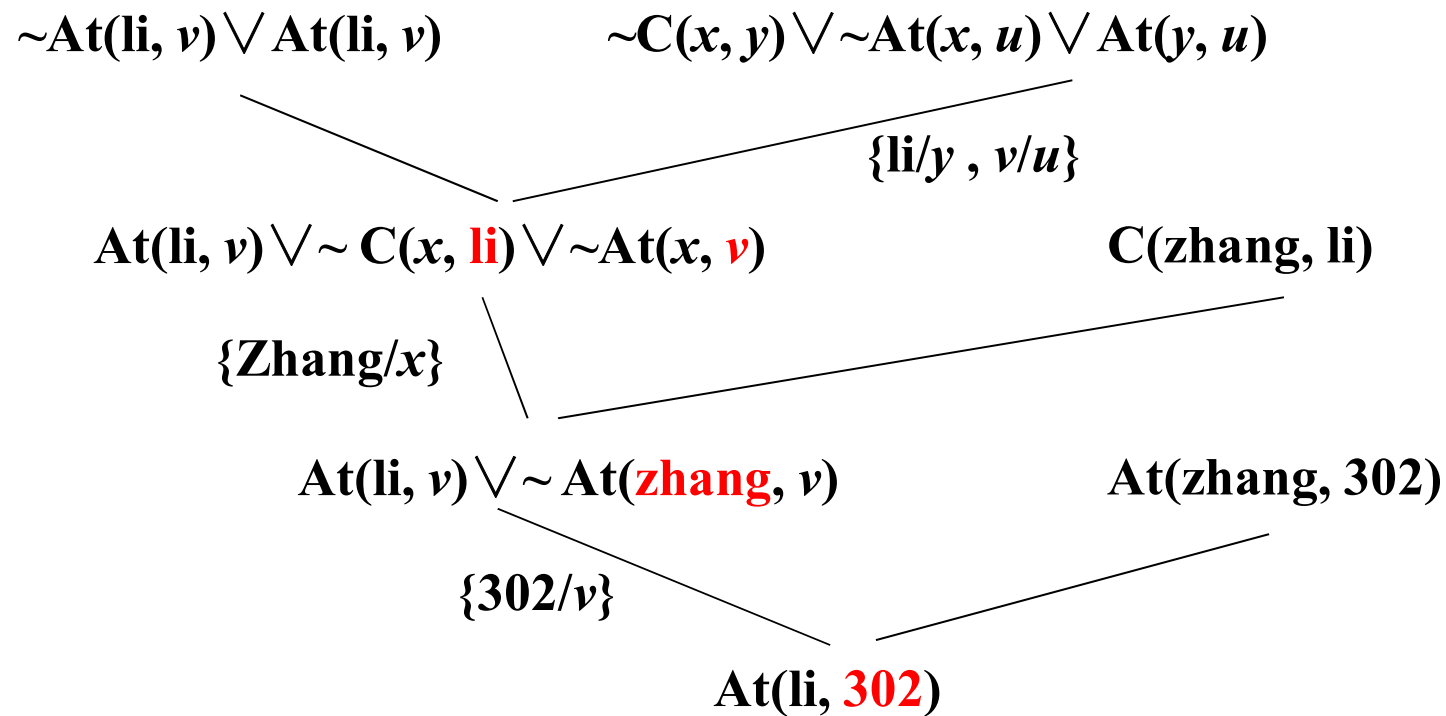
●  $\sim C(x, y) \vee \sim \text{At}(x, u) \vee \text{At}(y, u)$

●  $\text{At}(\text{zhang}, 302)$

❏ Convert the question's negation to clauses and add its negation, as  $\sim \text{At}(\text{li}, v) \vee \underline{\text{At}(\text{li}, v)}$



## Resolution



## Example 5

“如果无论John到哪里去，Fido也就去那里，那么如果John在学校里，Fido在哪里呢？”

解：(1) 用谓词公式表示命题和事实：

事实：  $(\forall x)[AT(JOHN, x) \Rightarrow AT(FIDO, x)]$   
 $AT(JOHN, SCHOOL)$

目标：  $(\exists x) AT(FIDO, x)$  求 $x$

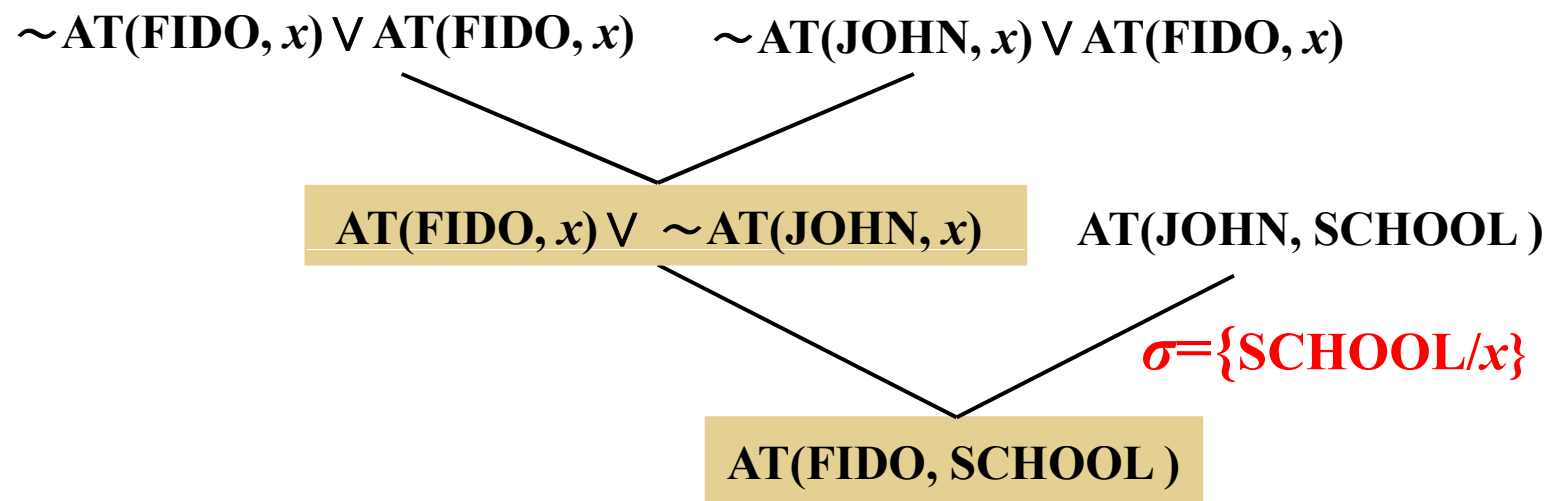
(2) 反演求解过程(1)：

目标否定子句：  $\sim AT(FIDO, x)$

将其添加到目标否定之否定的子句中：

$\sim AT(FIDO, x) \vee AT(FIDO, x)$

(3) 反演求解过程(2) :



(4) 用从根部求得答案  $\text{AT}(\text{FIDO}, \text{SCHOOL})$