分支限界法

1. 概　述
2. 图问题中的分支限界法
3. 组合问题中的分支限界法

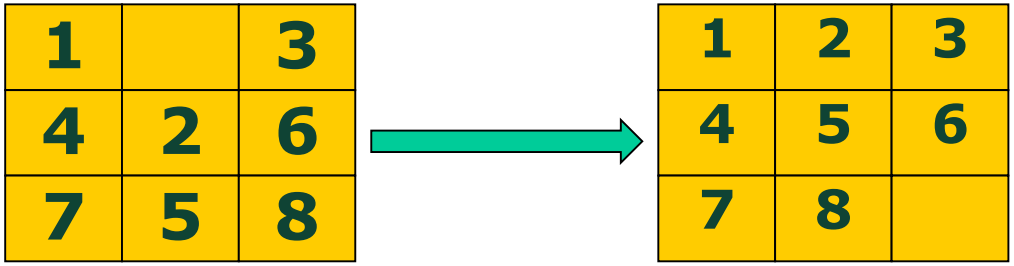
思考题

.5

使用什么方法求解八数码问题的最优解

初始状态

目标状态



- 我们已经知道，在现实世界中，当问题的规模变得很大时，穷尽搜索是不可能的，所以有必要找到某种启发式方法去掉那部分我们明知道不可能在其中找到最优解的搜索空间。

- 分支限界法就是这样的一种启发式方法，它建立在对搜索空间的基础上。将整个搜索空间想象成树状结构，根据计算出的特定解的代价剪去不感兴趣的分支。

解空间树的动态搜索（2）

确定一个合理的限界函数及界[*down, up*]

按照广度优先策略遍历问题的解空间树，估算孩子结点的评估函数的可能取值

如果某孩子结点的评估函数可能取得的值超出评估函数的界，则将其丢弃；否则，将其加入待处理结点表中。

依次从表中选取使评估函数的值取得极值的结点成为当前扩展结点

重复上述过程，直到找到最优解。

例：0/1背包问题。假设有4个物品，其重量分别为(4, 7, 5, 3)，价值分别为(40, 42, 25, 12)，背包容量W=10。首先，将给定物品按单位重量价值从大到小排序，结果如下：

物品	重量(w)	价值(v)	价值/重量(v/w)
1	4	40	10
2	7	42	6
3	5	25	5
4	3	12	4

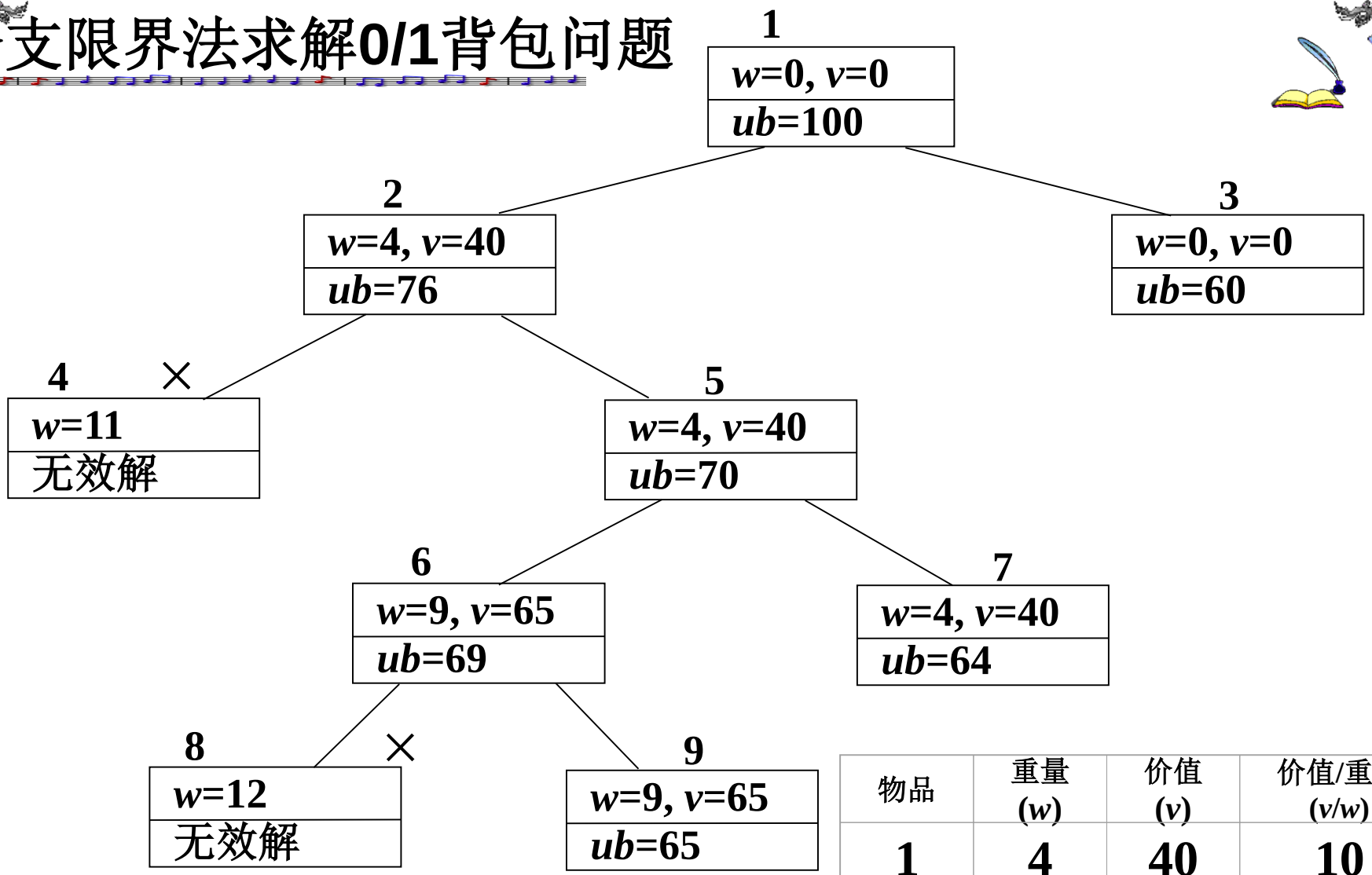
应用贪心法求得近似解为(1, 0, 0, 0)，获得的价值为40，这可以作为0/1背包问题的下界。

考虑最好情况，背包中装入的全部是第1个物品且可以将背包装满：$ub=W\times(v_1/w_1)=10\times10=100$。得到评估函数的界[40, 100]。

限界函数为：

$$ub = v + (W - w) \times (v_{i+1} / w_{i+1})$$

分支限界法求解0/1背包问题



物品	重量 (w)	价值 (v)	价值/重量 (v/w)
1	4	40	10
2	7	42	6
3	5	25	5
4	3	12	4

$$ub = v + (W - w) \times (v_{i+1} / w_{i+1})$$

分支限界法的设计思想

假设求解最大化问题

解向量为$X=(x_1, x_2, \ldots, x_n)$

部分解应满足：

$$bound(x_1)\geq bound(x_1, x_2)\geq \ldots \geq bound(x_1, x_2, \ldots, x_k)\geq \ldots \geq bound(x_1, x_2, \ldots, x_n)$$

若某孩子结点的目标函数值超出目标函数的界，则将该孩子结点丢弃；

否则，将该孩子结点保存在待处理结点表PT中。

分支限界法求解最大化问题的一般过程

分支限界法的一般过程

1．根据限界函数确定目标函数的界[down, up]；

2．将待处理结点表PT初始化为空；

3．对根结点的每个孩子结点x执行下列操作

3.1 估算结点x的目标函数值value;

3.2 若(value>=down)，则将结点x加入表PT中；

4．循环直到某个叶子结点的目标函数值在表PT中最大

4.1 i=表PT中值最大的结点；

4.2 对结点i的每个孩子结点x执行下列操作

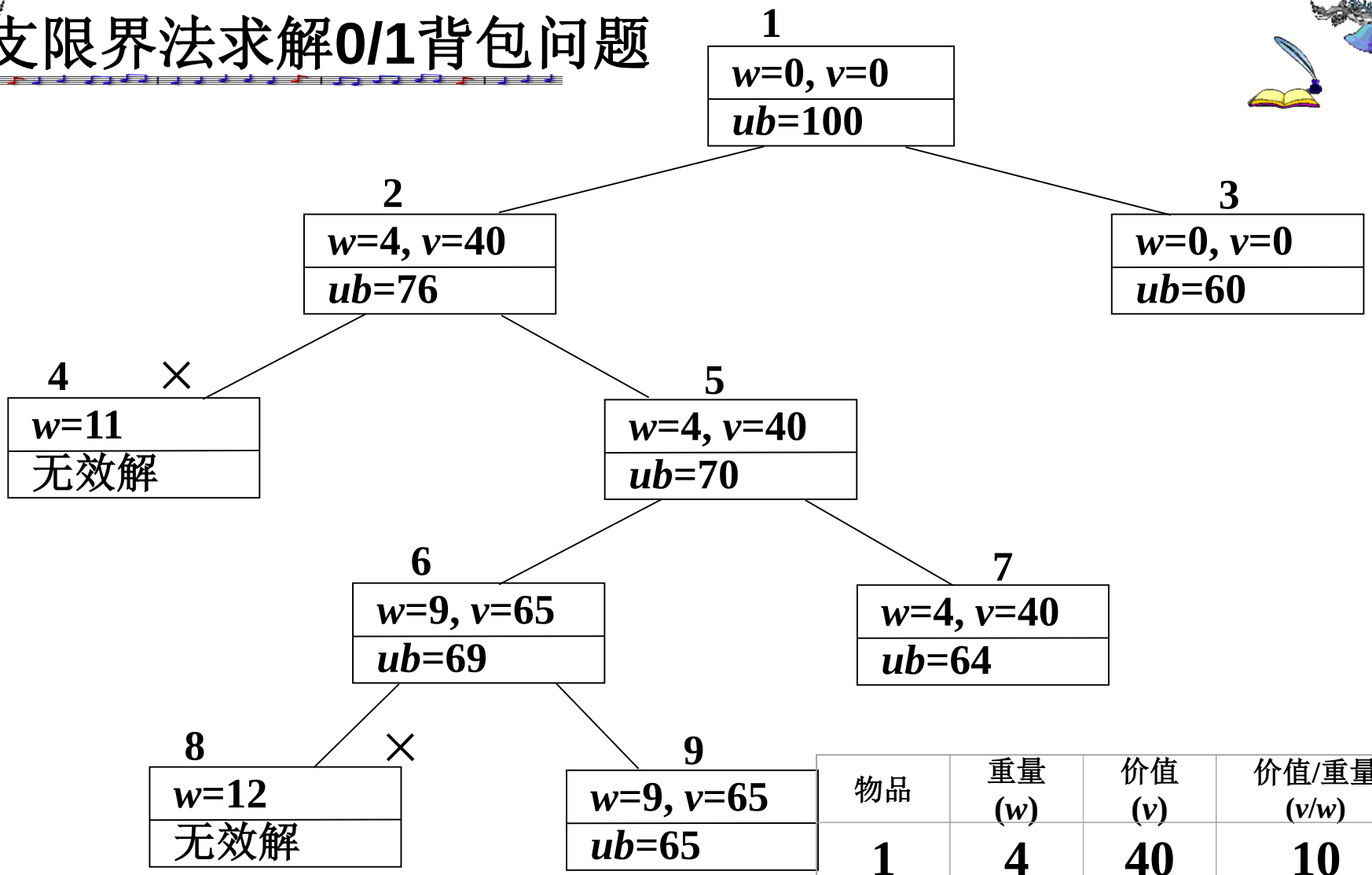
4.2.1 估算结点x的目标函数值value;

4.2.2 若(value>=down)，则将结点x加入表PT中；

4.2.3 若(结点x是叶子结点且结点x的value值在表PT中最大)，
则将结点x对应的解输出，算法结束；

4.2.4 若(结点x是叶子结点但结点x的value值在表PT中不是最大)，
则令down=value，并且将表PT中所有小于value的结点删除；

分支限界法求解0/1背包问题



物品	重量 (w)	价值 (v)	价值/重量 (v/w)
1	4	40	10
2	7	42	6
3	5	25	5
4	3	12	4

$$ub = v + (W - w) \times (v_{i+1} / w_{i+1})$$

应用分支限界法的关键问题

（1）如何确定合适的限界函数

（2）如何组织待处理结点表

（3）如何确定最优解中的各个分量

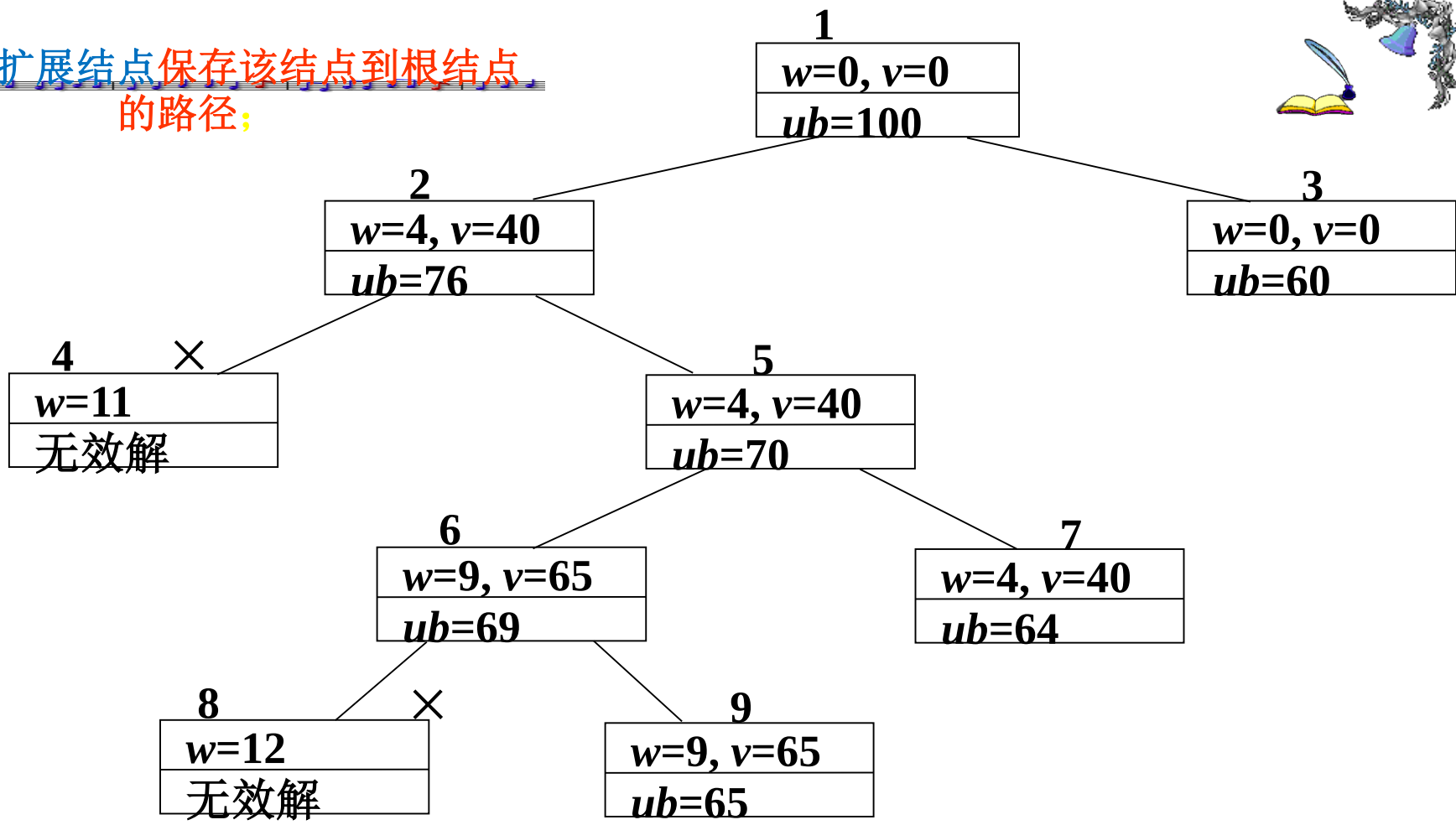
分支限界法对问题的解空间树中结点的处理是跳跃式的，回溯也不是单纯地沿着双亲结点一层一层向上回溯，

无法求得该叶子结点对应的最优解中的各个分量。这个问题可以用如下方法解决：

① 对每个扩展结点保存该结点到根结点的路径；

② 在搜索过程中构建搜索经过的树结构，在求得最优解时，从叶子结点不断回溯到根结点，以确定最优解中的各个分量。

对每个扩展结点保存该结点到根结点的路径；



(1)76	(0)60	

(a) 扩展根结点后表PT状态

(0)60	(1,0)70	

(b) 扩展结点2后表PT状态

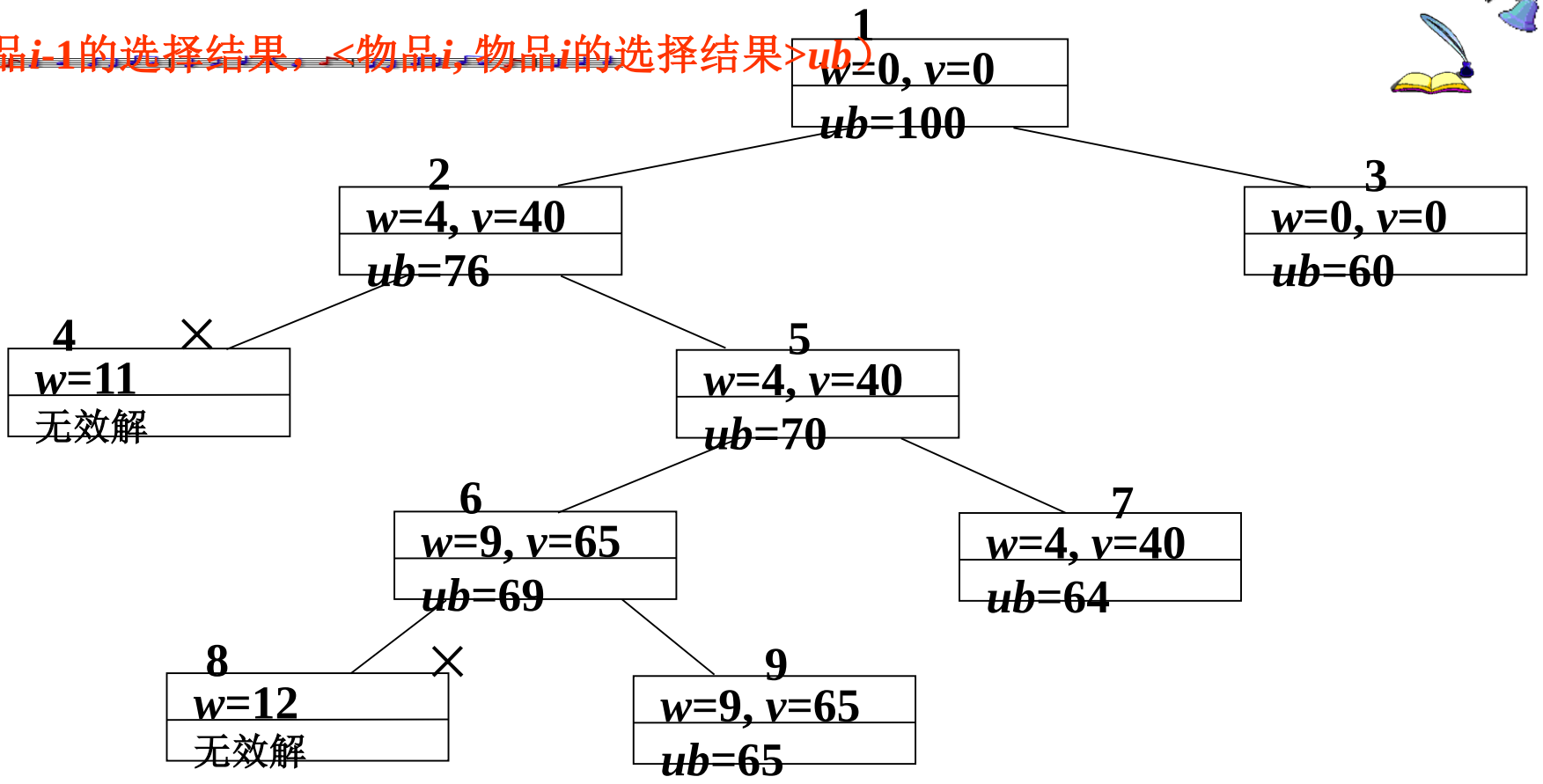
(0)60	(1,0,1)69	(1,0,0)64

(c) 扩展结点5后表PT状态

(0)60	(1,0,0)64	(1,0,1,0)65

(d) 扩展结点6后表PT状态，最优解为(1,0,1,0)65

(物品*i*-1的选择结果，<物品*i*，物品*i*的选择结果>*ub*)



PT	(0,<1,1>76)	(0,<1,0>60)	
ST			

(a) 扩展根结点后

PT	(0,<1,0>60)	(1,<2,0>70)	
ST	(0,<1,1>76)		

(b) 扩展结点2后

PT	(0,<1,0>60)	(0,<3,1>69)	(0,<3,0>64)
ST	(0,<1,1>76)	(1,<2,0>70)	

(c) 扩展结点5后

PT	(0,<1,0>60)	(0,<3,0>64)	(1,<4,0>65)
ST	(0,<1,1>76)	(1,<2,0>70)	(0,<3,1>69)

(d) 扩展结点6后，最优解为(1,0,1,0)65

分支限界法的时间性能

分支限界法和回溯法实际上都属于蛮力穷举法，在最坏情况下，时间复杂性肯定为指数阶。

如果选择了结点的合理扩展顺序以及设计了一个好的限界函数，分支界限法可以快速得到问题的解。

分支限界法的较高效率是以付出一定代价为基础的。

1：一个更好的限界函数通常需要花费更多的时间计算相应的目标函数值；

2：由于分支限界法对解空间树中结点的处理是跳跃式的，需要对每个扩展结点保存该结点到根结点的路径，使得算法的设计较为复杂；

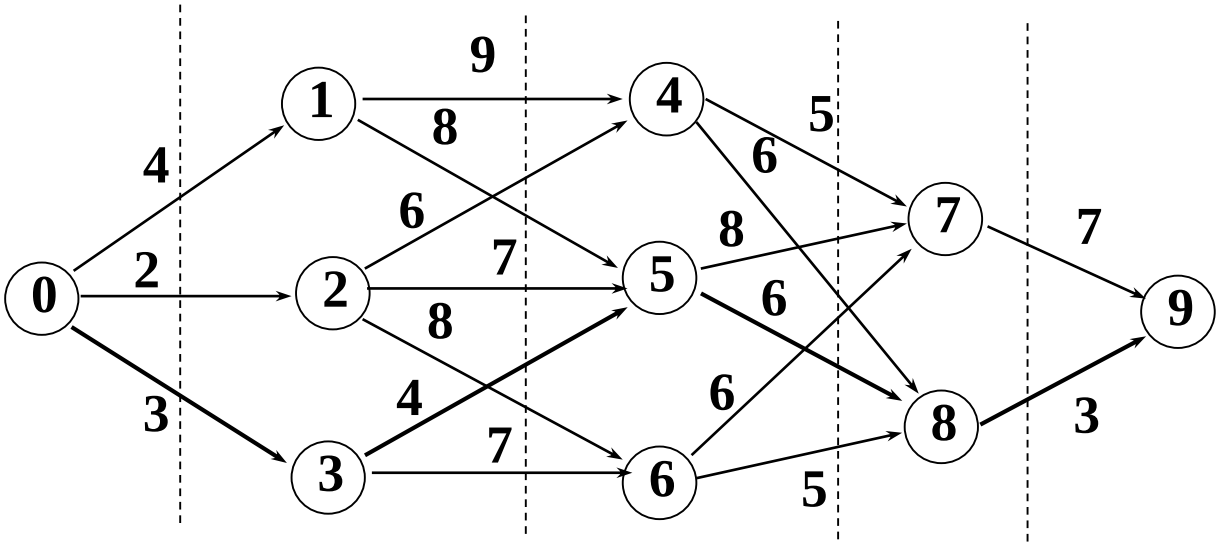
3：算法要维护一个待处理结点表PT，并且需要在表PT中快速查找取得极值的结点。这都需要较大的存储空间，在最坏情况下，分支限界法需要的空间复杂性是指数阶。

2. 图问题中的分支限界法

多段图的最短路径问题

多段图的最短路径问题

设图$G=(V, E)$是一个带权有向连通图，如果把顶点集合V划分成k个互不相交的子集V_i（$2\le k\le n$，$1\le i\le k$），使得E中的任何一条边(u, v)，必有$u\in V_i$，$v\in V_{i+m}$（$1\le i<k$，$1<i+m\le k$），则称图G为多段图，称$s\in V_1$为源点，$t\in V_k$为终点。



多段图的最短路径问题是求从源点到终点的最小代价路径。

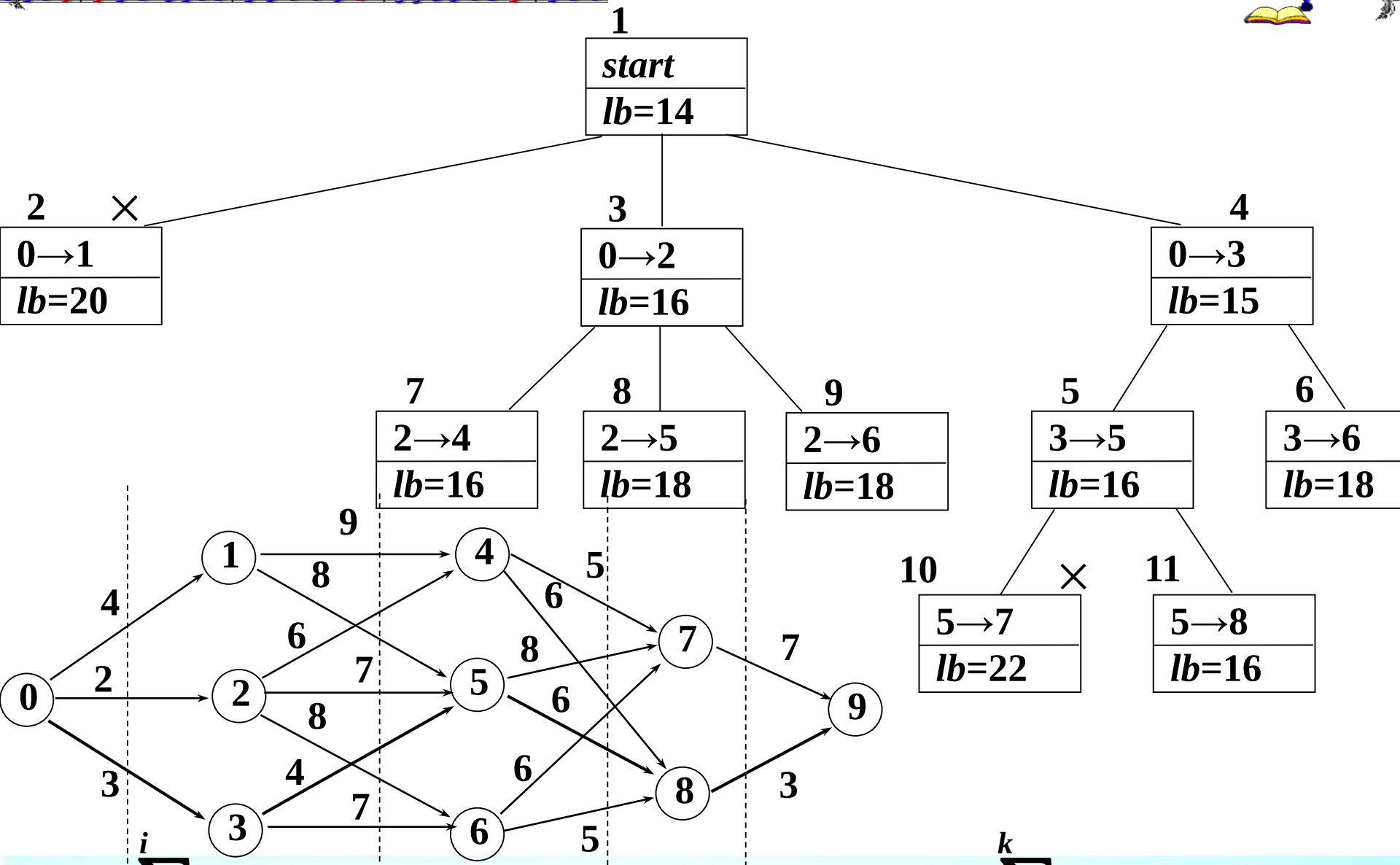
得到评估函数的界[14, 20]。

一般情况下，对于一个正在生成的路径，假设已经确定了i段（$1 \le i \le k$），其路径为$(r_1, r_2, \ldots, r_i, r_{i+1})$，此时，该部分解的评估函数值的计算方法即限界函数如下：

$$lb = \sum_{j=1}^{i} c[r_j][r_{j+1}] + \min_{<r_{i+1}, v_p> \in E} \{c[r_{i+1}][v_p]\} + \sum_{j=i+2}^{k} \text{第} j \text{ 段的最短边}$$

分支限界法求解多段图的最短路径问题示例

(×表示该结点被丢弃，结点上方的数组表示搜索顺序)



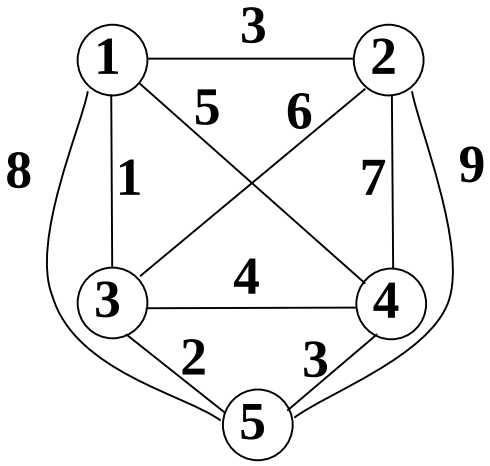
$$lb = \sum_{j=1}^{i} c[r_j][r_{j+1}] + \min_{<r_{i+1}, v_p> \in E} \{c[r_{i+1}][v_p]\} + \sum_{j=i+2}^{k} \text{第} j \text{段的最短边}$$

算法——多段图的最短路径问题

1．根据限界函数计算评估函数的下界down；采用贪心法得到上界up；
2．将待处理结点表PT初始化为空；
3．for (i=1; i<=k; i++)
    x[i]=0;
4．i=1; u=0;　　//求解第i段
5．while (i>=1)
    5.1 对顶点u的所有邻接点v
        5.1.1 根据式9.3计算评估函数值lb;
        5.1.2 若lb<=up，则将i,<u,v>,lb存储在表PT中；
    5.2 如果i= =k-1且叶子结点的lb值在表PT中最小，
        则输出该叶子结点对应的最优解；
    5.3 否则，如果i= =k-1且表PT中的叶子结点的lb值不是最小，则
        5.3.1 up=表PT中的叶子结点最小的lb值;
        5.3.2 将表PT中评估函数值超出up的结点删除；
    5.4 u=表PT中lb最小的结点的v值；
    5.5 i=表PT中lb最小的结点的i值；i++;

TSP问题

TSP问题是指旅行家要旅行n个城市，要求各个城市经历且仅经历一次然后回到出发城市，并要求所走的路程最短。



(a) 一个无向图

$$C=\begin{pmatrix} \infty & 3 & 1 & 5 & 8 \\ 3 & \infty & 6 & 7 & 9 \\ 1 & 6 & \infty & 4 & 2 \\ 5 & 7 & 4 & \infty & 3 \\ 8 & 9 & 2 & 3 & \infty \end{pmatrix}$$

(b) 无向图的代价矩阵

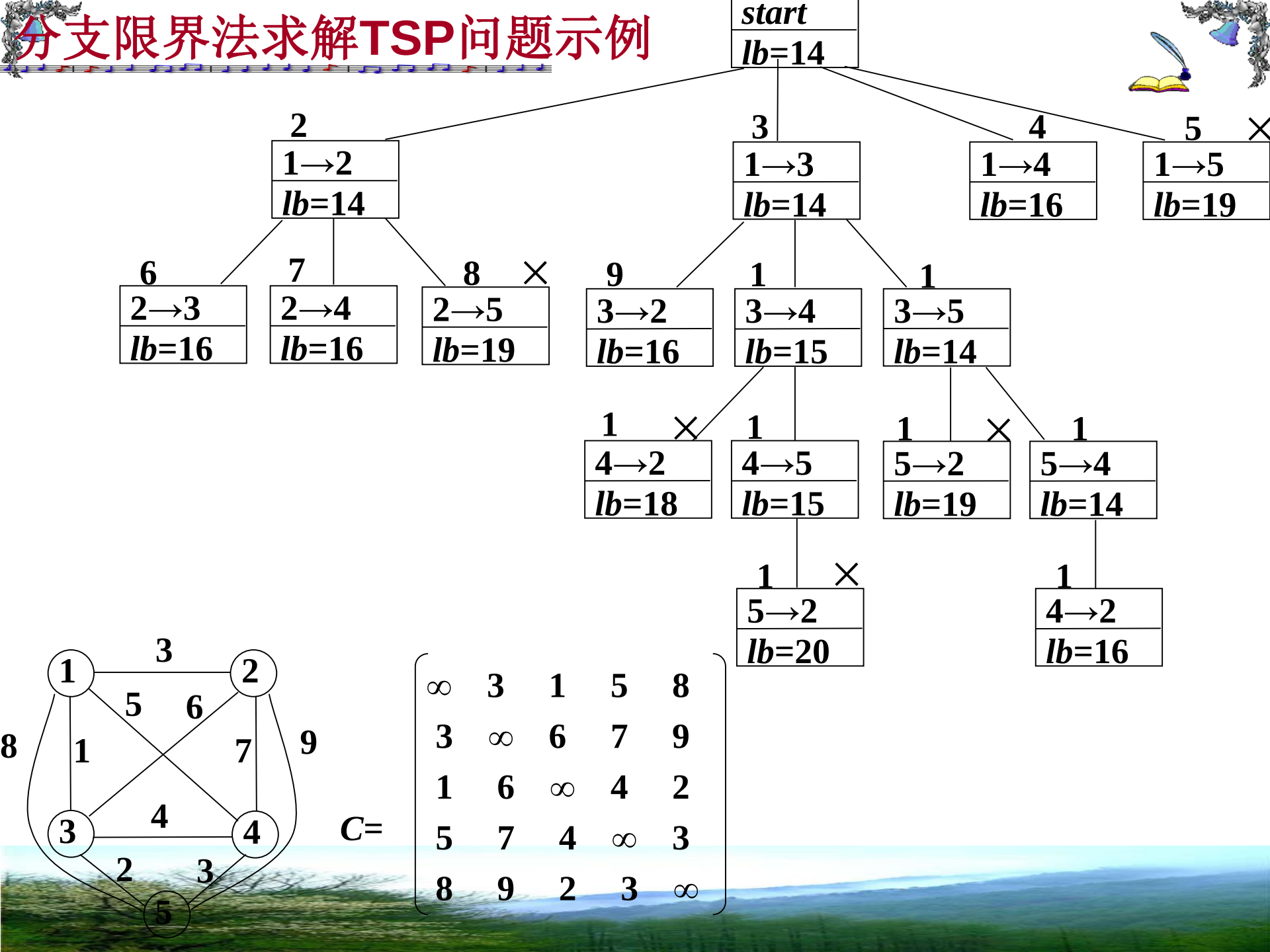
图9.4 无向图及其代价矩阵

部分解的目标函数值的计算方法

$$lb = (2\sum_{i=1}^{k-1} c[r_i][r_{i+1}] + \sum_{r_i \in U} r_i\text{行不在路径上的最小元 素} + \sum_{r_j \notin U} r_j\text{行最小的两个元素}) / 2$$

例如图所示无向图，如果部分解包含边(1，4)，则该部分解的下界是*lb*=((1+5)+(3+6)+(1+2)+(3+5)+(2+3))/2=16。

分支限界法求解TSP问题示例



$$C=\begin{pmatrix} \infty & 3 & 1 & 5 & 8 \\ 3 & \infty & 6 & 7 & 9 \\ 1 & 6 & \infty & 4 & 2 \\ 5 & 7 & 4 & \infty & 3 \\ 8 & 9 & 2 & 3 & \infty \end{pmatrix}$$

(1, 2)14	(1, 3)14	(1, 4)16

(a) 扩展根结点后的状态

(1, 3)14	(1, 4)16	(1, 2, 3)16	(1, 2, 4)16

(b) 扩展结点2后的状态

(1, 4)16	(1, 2, 3)16	(1, 2, 4)16	(1, 3, 2)16	(1, 3, 4)15	(1, 3, 5)14

(c) 扩展结点3后的状态

(1, 4)16	(1, 2, 3)16	(1, 2, 4)16	(1, 3, 2)16	(1, 3, 4)15	(1, 3, 5, 4)14

(d) 扩展结点11后的状态

(1, 4)16	(1, 2, 3)16	(1, 2, 4)16	(1, 3, 2)16	(1, 3, 4)15	(1, 3, 5, 4, 2)16

(e) 扩展结点13后的状态

(1, 4)16	(1, 2, 3)16	(1, 2, 4)16	(1, 3, 2)16	(1, 3, 5, 4, 2)16	(1, 3, 4, 5)15

(f) 扩展结点10后的状态

(1, 4)16	(1, 2, 3)16	(1, 2, 4)16	(1, 3, 2)16	(1, 3, 5, 4, 2)16	(1, 3, 4, 5)15

(g) 扩展结点16后的状态，最优解为1→3→5→4→2→1

图9.6 TSP问题最优解的确定

算法——TSP问题

伪代码

1．根据限界函数计算目标函数的下界down；采用贪心法得到上界up；
2．将待处理结点表PT初始化为空；
3．for (i=1; i<=n; i++)
    x[i]=0;
4．k=1; x[1]=1; //从顶点1出发求解TSP问题
5．while (k>=1)
    5.1 i=k+1;
    5.2 x[i]=1;
    5.3 while (x[i]<=n)
        5.3.1 如果路径上顶点不重复，则
            5.3.1.1 根据式9.2计算目标函数值lb；
            5.3.1.2 if (lb<=up) 将路径上的顶点和lb值存储在表PT中；
        5.3.2 x[i]=x[i]+1;
    5.4 若i=n且叶子结点的目标函数值在表PT中最小
        则将该叶子结点对应的最优解输出；
    5.5否则，若i=n，则在表PT中取叶子结点的目标函数值最小的结点lb，
        令up=lb，将表PT中目标函数值lb超出up的结点删除；
    5.6 k=表PT中lb最小的路径上顶点个数；

组合问题中的分支限界法

任务分配问题

任务分配问题

任务分配问题要求把n项任务分配给n个人，每个人完成每项任务的成本不同，要求分配总成本最小的最优分配方案。如图9.10所示是一个任务分配的成本矩阵。

$$C=\begin{array}{c} \begin{matrix} \text{任务1} & \text{任务2} & \text{任务3} & \text{任务4} \end{matrix} \\ \begin{pmatrix} 9 & 2 & 7 & 8 \\ 6 & 4 & 3 & 7 \\ 5 & 8 & 1 & 8 \\ 7 & 6 & 9 & 4 \end{pmatrix} \end{array} \begin{matrix} \text{人员}a \\ \text{人员}b \\ \text{人员}c \\ \text{人员}d \end{matrix}$$

图9.10 任务分配问题的成本矩阵

求最优分配成本的上界和下界

考虑任意一个可行解。

例如矩阵中的对角线是一个合法的选择；

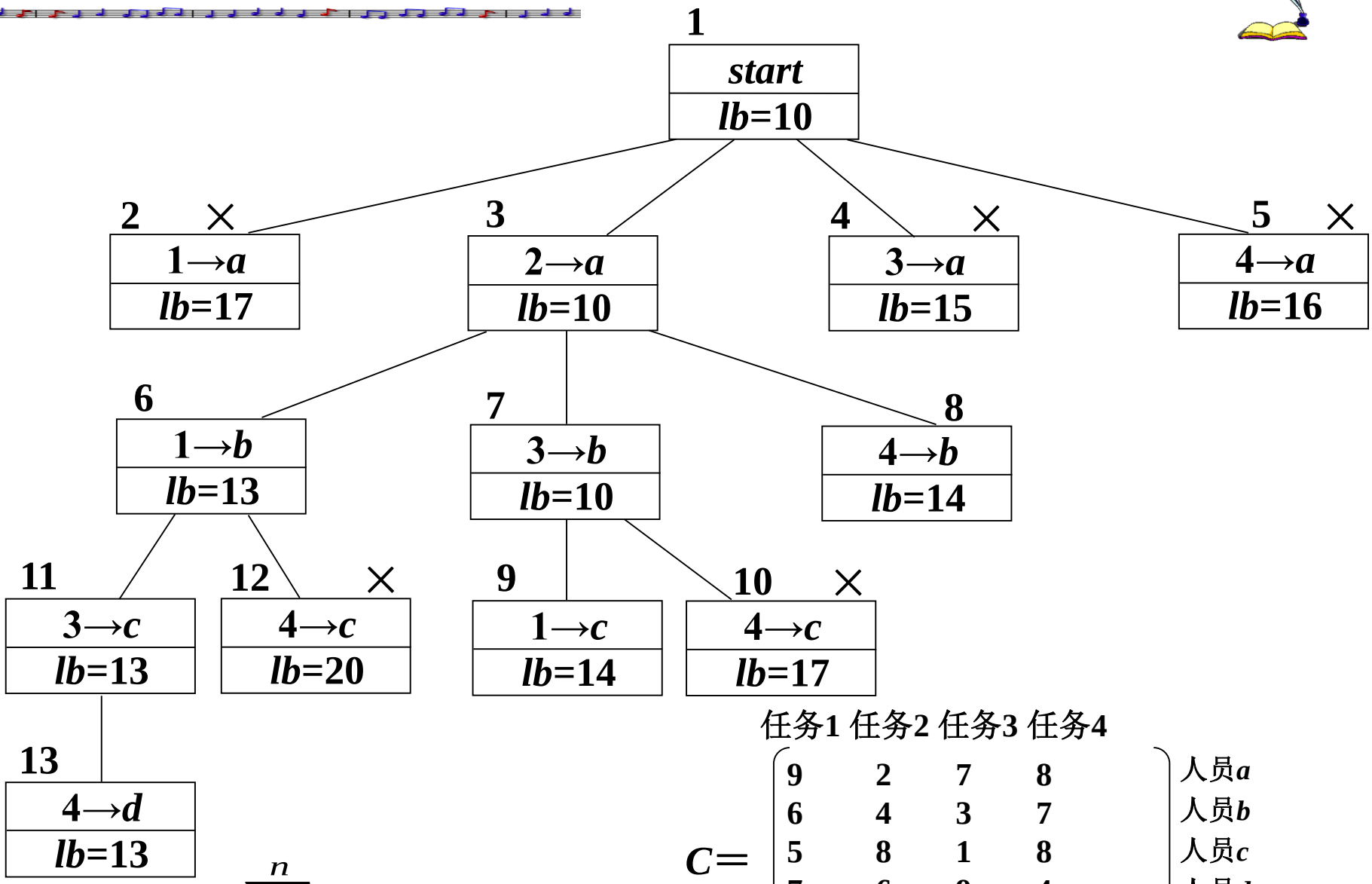
将每一行的最小元素加起来就得到解的下界，其成本是2+3+1+4=10。需要强调的是，这个解并不是一个合法的选择（3和1来自于矩阵的同一列），它仅仅给出了一个参考下界。

最优值一定是[10, 14]之间的某个值。

设当前已对人员1～i分配了任务，并且获得了成本v，则限界函数可以定义为：

$$lb = v + \sum_{k=i+1}^{n} \text{第}k\text{行的最小值}$$

分支限界法求解任务分配问题示例(×表示该结点被丢弃，结点上方的数组表示搜索顺序)



$$lb = v + \sum_{k=i+1}^{n} \text{第}k\text{行的最小值}$$

$C=$

任务1	任务2	任务3	任务4	
9	2	7	8	人员*a*
6	4	3	7	人员*b*
5	8	1	8	人员*c*
7	6	9	4	人员*d*

伪代码

算法——任务分配问题

1．根据限界函数计算评估函数的下界down；采用贪心法得到上界up；
2．将待处理结点表PT初始化为空；
3．for (i=1; i<=n; i++)
    x[i]=0;
4．k=1; i=0;　　//为第k个人分配任务，i为第k-1个人分配的任务
5．while (k>=1)
    5.1 x[k]=1;
    5.2 while (x[k]<=n)
        5.2.1 如果人员k分配任务x[k]不发生冲突，则
            5.2.1.1 根据式9.4计算评估函数值lb;
            5.2.1.2 若lb<=up，则将i,<x[k], k>lb存储在表PT中；
        5.2.2 x[k]=x[k]+1;
    5.3 如果k= =n且叶子结点的lb值在表PT中最小，
        则输出该叶子结点对应的最优解；
    5.4 否则，如果k= =n且表PT中的叶子结点的lb值不是最小，则
        5.4.1 up=表PT中的叶子结点最小的lb值;
        5.4.2 将表PT中超出评估函数界的结点删除；
    5.5 i=表PT中lb最小的结点的x[k]值；
    5.6 k=表PT中lb最小的结点的k值； k++;

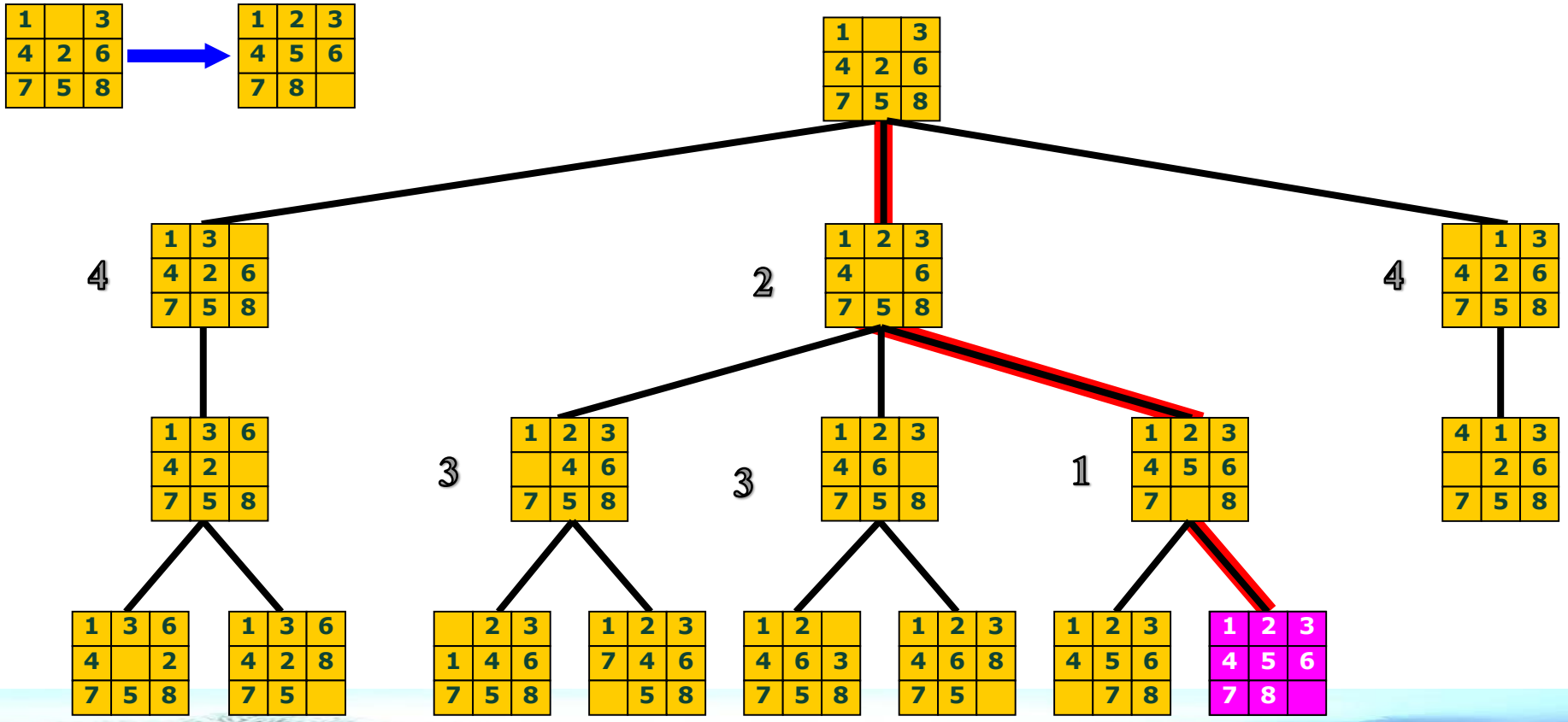

八数码问题

什么是八数码问题

- 八数码问题就是在一个3×3的九宫格棋盘上，分别将标有数字1~8的八个棋子摆放其中，摆放时要求棋子不能重叠。
- 允许空格周围的某一个棋子向空格移动。
- 假设给定一个初始的棋子布局（初始状态）和一个目标布局（目标状态），要求移动棋子，实现从初始状态到目标状态的转变，给出一个合法的走步序列。

八数码问题解析

.5 使用什么方法求解八数码问题的最优解



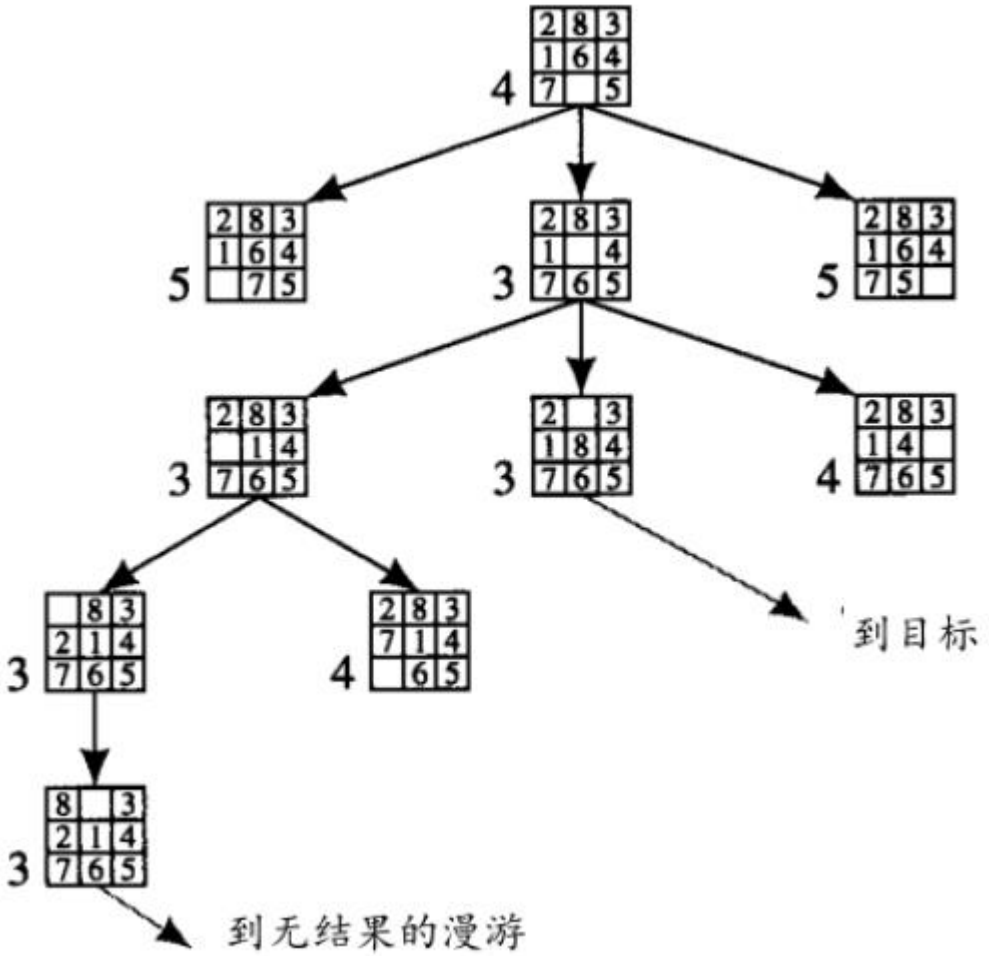
八数码问题解析

.5 使用什么方法求解八数码问题的最优解

1	2	3
8		4
7	6	5

评估函数：

f（n）= 位置不正确的数字个数（和目标相比）



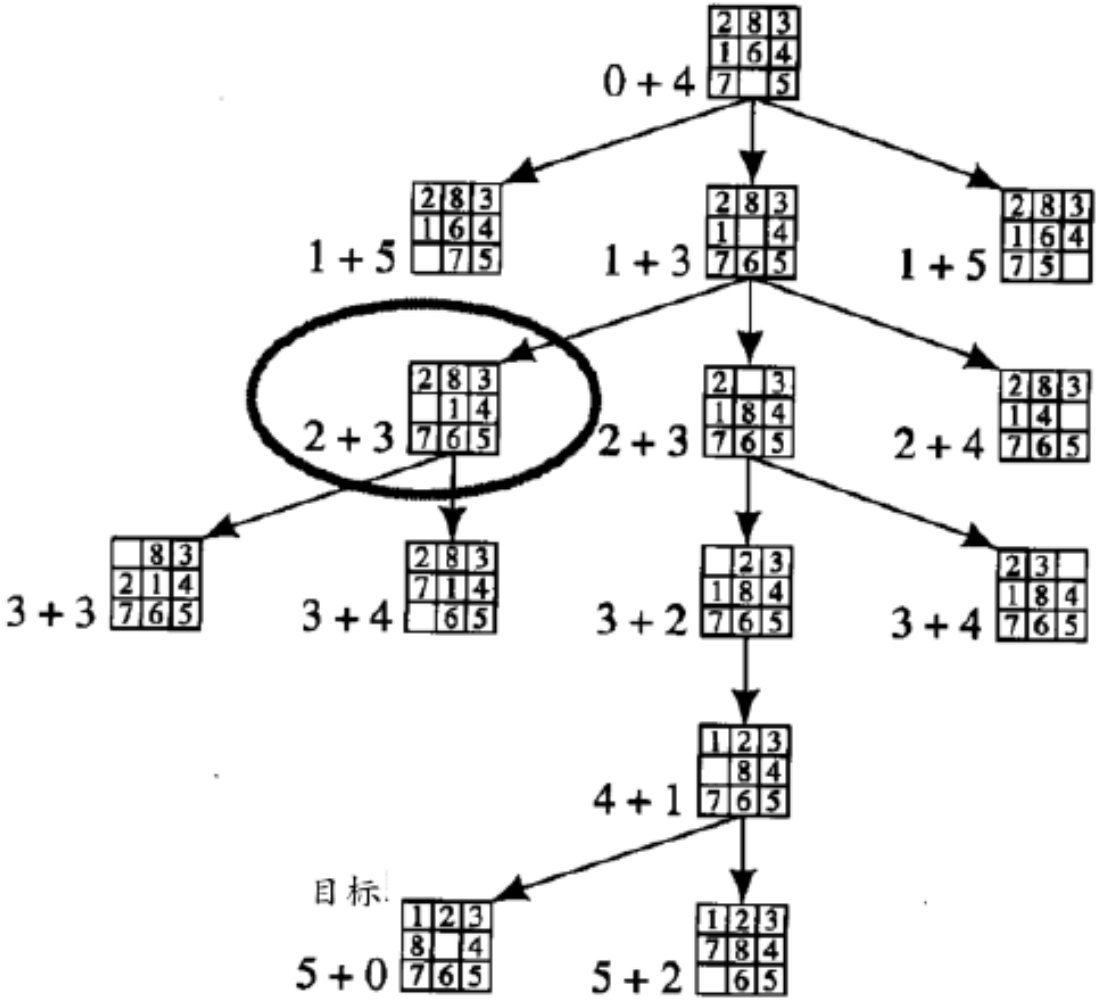
八数码问题解析

.5 使用什么方法求解八数码问题的最优解

1	2	3
8		4
7	6	5

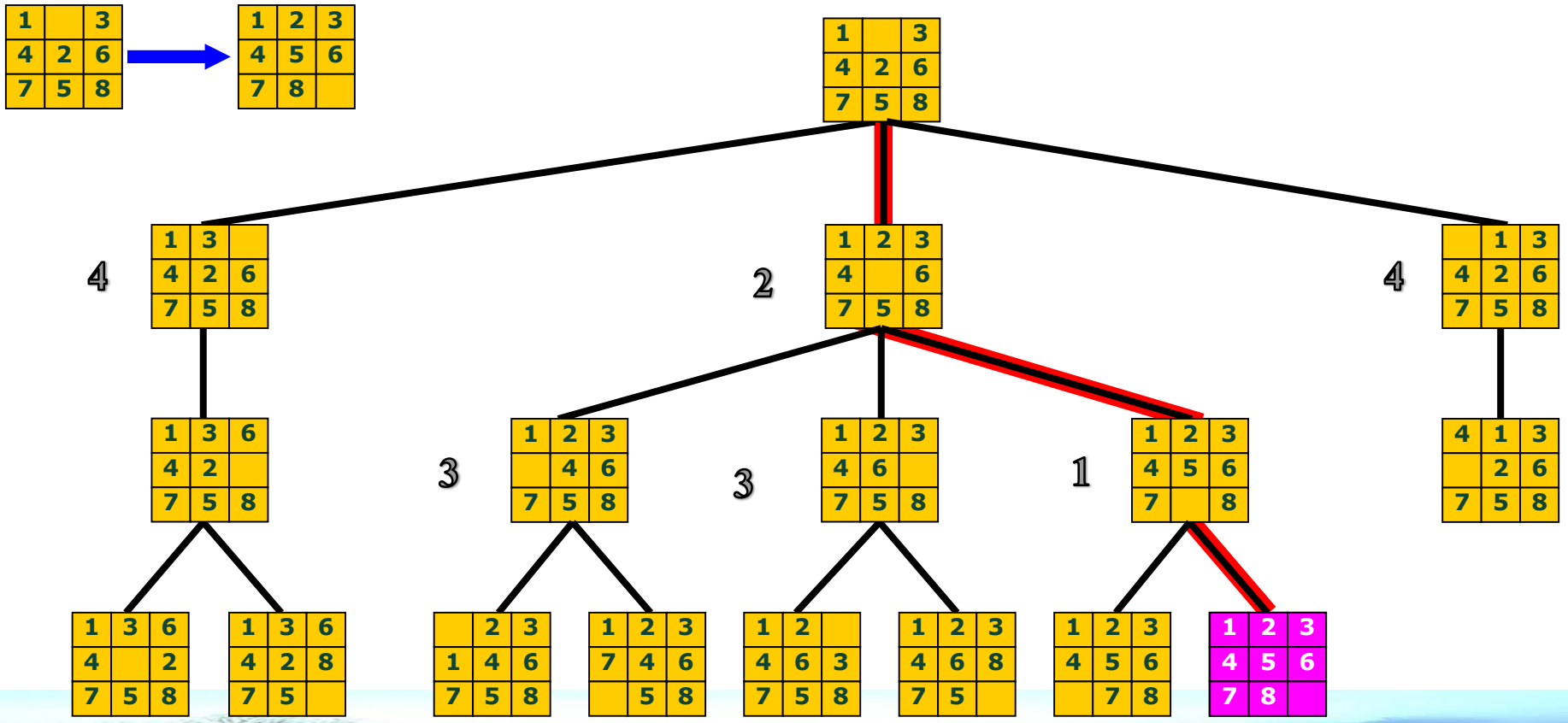
$$f(x) = g(x) + h(x)$$

h（x）为不正确的数字个数，g（x）为深度



八数码问题解析

.5 使用什么方法求解八数码问题的最优解



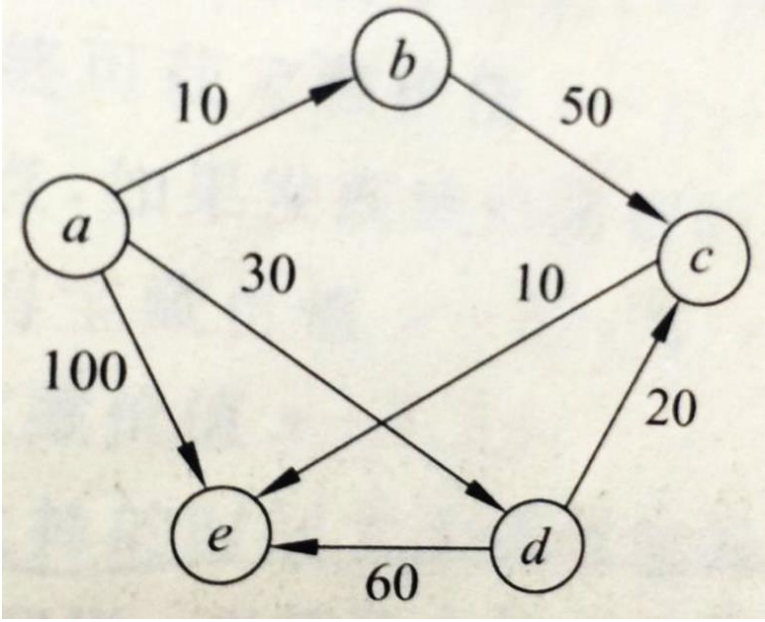
习题

- 有3个物品，其体积分别为（4, 3 , 2); 价值分别为(36,24,20),背包容量为6，请写出求最优解的过程。
- 分别用蛮力法、回溯法、分支限界法解如上的0/1背包问题：

主观题 10分

设置

作业



最短路径长度是_____

作答

作业

- 用分支限界法求解如下的任务分配问题

$$C=\begin{array}{c} \begin{matrix} \text{任务1} & \text{任务2} & \text{任务3} & \text{任务4} \end{matrix} \\ \begin{pmatrix} 3 & 8 & 4 & 12 \\ 9 & 12 & 13 & 5 \\ 8 & 7 & 9 & 3 \\ 12 & 7 & 6 & 8 \end{pmatrix} \end{array} \begin{matrix} \\ \text{人员}a \\ \text{人员}b \\ \text{人员}c \\ \text{人员}d \end{matrix}$$

分配方案为： 最佳成本是：

作答