

《计算机算法设计与分析》

第六章 回溯法

马丙鹏

2020年11月05日



第六章 回溯法

- 6.1 一般方法
- 6.2 8-皇后问题
- 6.3 子集和数问题
- 6.4 图的着色
- 6.5 0/1背包问题
- 6.6 哈密顿环
- 6.7 和最小
- 6.8 跳马问题



6.6 哈密顿环

■ 问题描述

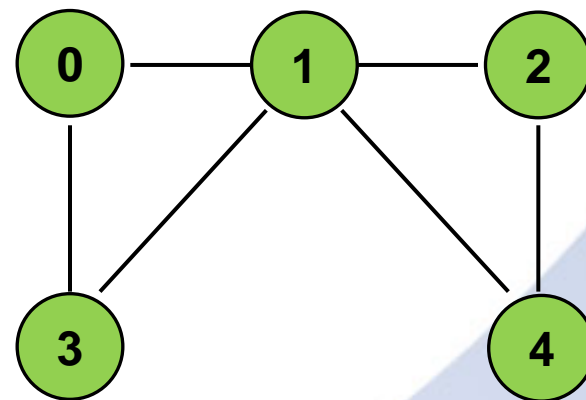
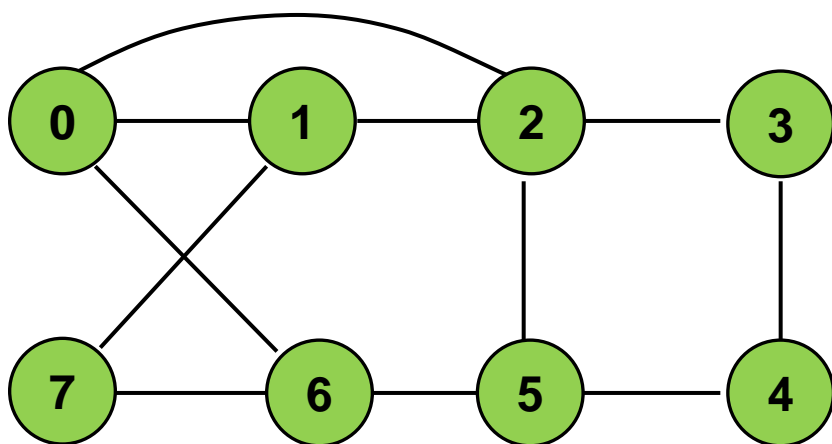
- 哈密顿环 (Hamiltonian cycle) : 连通图 $G=(V, E)$ 中的一个回路, 经过图中每个顶点, 且只经过一次。
- 一个哈密顿环就是从某个结点 v_0 开始, 沿着图 G 的 n 条边环行的一条路径 $(v_0, v_1, \dots, v_{n-1}, v_n)$ 。
 - 除 $v_0=v_n$ 外, 路径上其余结点各不相同。
 - $(v_i, v_{i+1}) \in E$ ($0 \leq i < n$)。
 - 它访问图中每个结点且仅访问一次, 最后返回开始结点。



6.6 哈密顿环

■ 问题描述

□ 并不是每个连通图都存在哈密顿环!



要确定一个连通图是否存在哈密顿环没有容易的办法。



6.6 哈密顿环

■ 算法说明

□ 采用 n -元组 (x_1, x_2, \dots, x_n) 表示哈密顿环问题的解。

➤ 显示约束: $x_i \in \{1, 2, \dots, n-1, n\}$, $0 < i \leq n$, 代表路径上一个结点的编号。

➤ 隐式约束: $x_i \neq x_j$ ($0 < i, j \leq n$, $i \neq j$), 且 $(x_i, x_{i+1}) \in E$ ($i=1, 2, \dots, n-2, n-1$), 又 $(x_n, x_1) \in E$ 。

➤ 解空间大小为 n^n 。



6.6 哈密顿环

■ 算法说明

- 如果已选定 x_1, x_2, \dots, x_{k-1} , 那么下一步要做的工作是如何找出可能 x_k 的结点集合。
- 若 $k=1$, 则 $X(1)$ 可以是这 n 个结点中的任一结点, 但为了避免将同一环重复打印 n 次, 可事先指定 $X(1)=1$ 。
- 若 $1 < k < n$, 则 $X(k)$ 可以是不同于 $X(1), X(2), \dots, X(k-1)$ 且和 $X(k-1)$ 有边相连的任一结点 v 。
- $X(n)$ 只能是唯一剩下的且必须与 $X(n-1)$ 和 $X(1)$ 皆有边相连的结点。
- 过程 NEXTVALUE 给出了在求哈密顿环的过程中如何找下一个结点的算法。



算法8.9 生成下一个结点

procedure NEXTVALUE(k)

//X(1),...,X(k-1)是一条有k-1个不同结点的路径。若X(k)=0，则表示再无结点可分配给X(k)。若还有与X(1),...,X(k-1)不同且与X(k-1)有边相连接的结点，则将其中标数最高的结点置于X(k)。若k=n，则还需要与X(1)相连接//

global integer n, X(1:n), **boolean** GRAPH(1:n, 1:n)

integer k, j

loop

X(k) \leftarrow (X(k)+1) mod (n+1) //下一个结点//

if X(k)=0 **then return endif**

if GRAPH(X(k-1), X(k)) //有边相连吗//

then for j \leftarrow 1 **to** k-1 **do** //检查与前k-1个结点是否相同//

if X(j)=X(k) **then exit endif** //有相同结点，退出此循环//

repeat

if j=k //若为真，则是一个不同结点//

then if k<n **or** (k=n **and** GRAPH(X(n), 1)) **then return**

endif

endif

repeat

end NEXTVALUE



中国科学院大学

University of Chinese Academy of Sciences 7

算法8.10 找所有的哈密顿环

procedure HAMILTONIAN(k)

//这是找出图G中所有哈密顿环的递归算法。图G用它的布尔邻接矩阵GRAPH(1:n,1:n)表示。每个环都从结点1开始//

global integer X(1:n)

local integer k, n

loop //生成X(k)的值//

call NEXTVALUE(k) //下一个合法结点分配给X(k)//

if X(k)=0 **then return endif**

if k=n

then print(X, '1') //打印一个环//

else call HAMILTONIAN(k+1)

endif

repeat

end HAMILTONIAN

这个过程首先初始化邻接矩阵GRAPH(1:n,1:n), 然后置 $X(2:n) \leftarrow 0$, $X(1) \leftarrow 1$, 在执行call HAMILTONIAN(2)



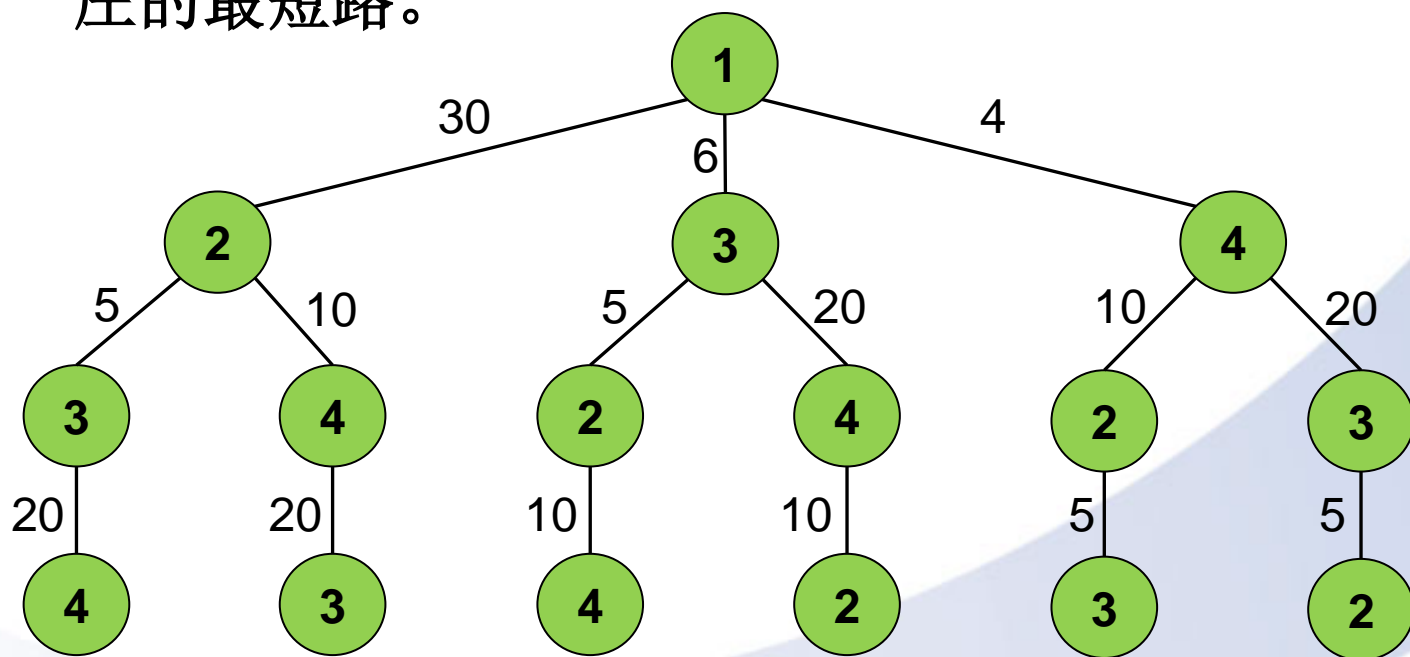
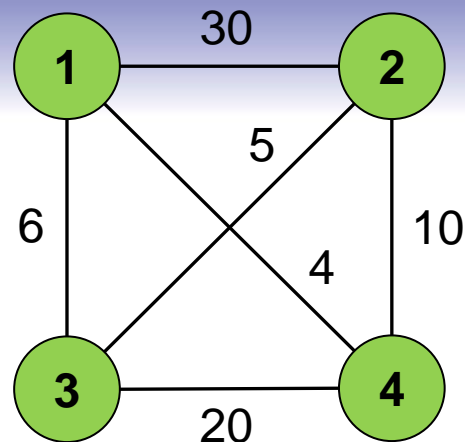
中国科学院大学

University of Chinese Academy of Sciences 8

6.6 哈密顿环

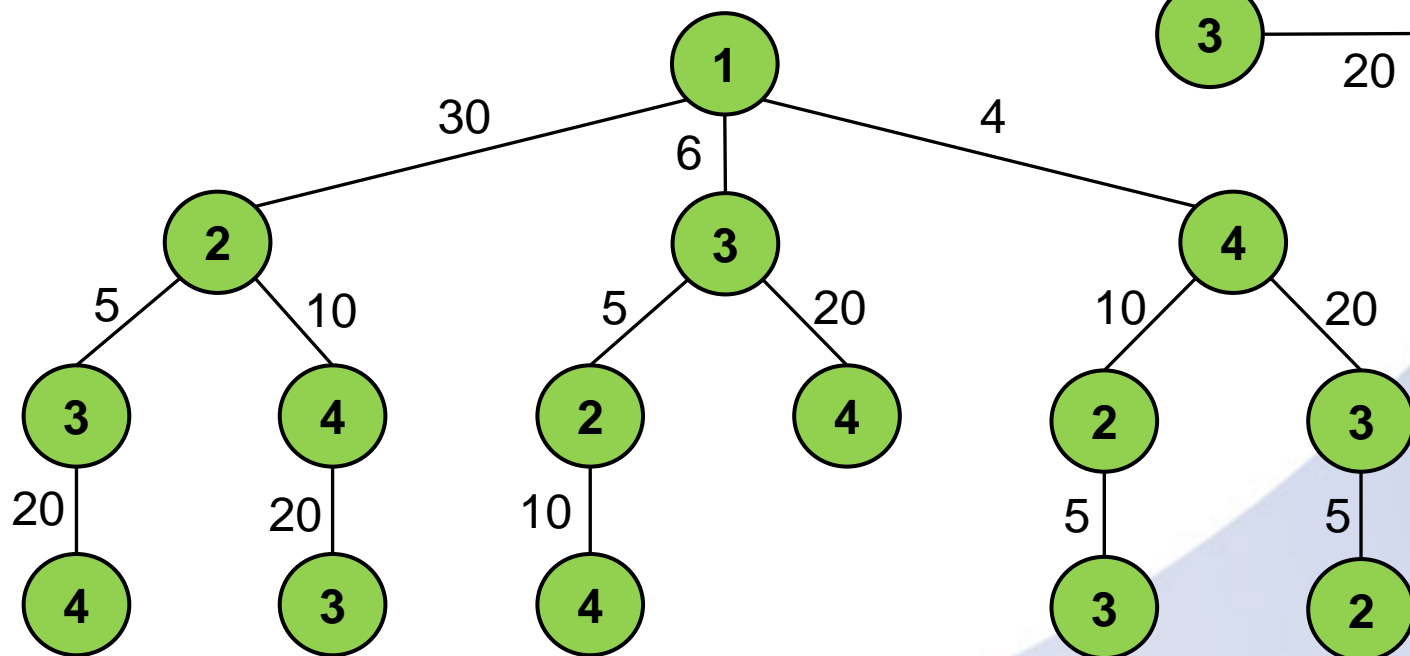
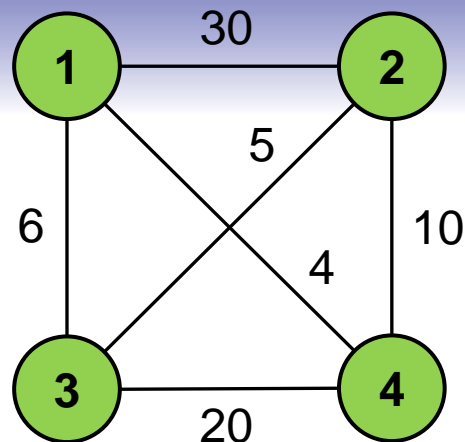
■ 实例分析

□有 n 个村庄，每个村庄必须经过一次，也只能经过一次，求一条走遍全部村庄的最短路。



6.6 哈密顿环

■ 实例分析



$$\begin{aligned} 30+5+20 \\ +4=59 \end{aligned}$$

$$\begin{aligned} 30+10+20 \\ +6=66 \end{aligned}$$

$$\begin{aligned} 6+5+10 \\ +4=25 \end{aligned}$$



中国科学院大学

University of Chinese Academy of Sciences 10

第六章 回溯法

- 6.1 一般方法
- 6.2 8-皇后问题
- 6.3 子集和数问题
- 6.4 图的着色
- 6.5 0/1背包问题
- 6.6 哈密顿环
- 6.7 和最小
- 6.8 跳马问题



6.7 和最小

■ 问题描述

□ 设有一个长度为 N 的数字串，要求使用 K 个加号将它分成 $K+1$ 个部分，找出一种分法，使得这 $K+1$ 个部分的和能够为最小。

□ 例：有一个数字串：312，当 $N=3$ ， $K=1$ 时会有以下两种分法：

① $3+12=15$

② $31+2=33$

这时，符合题目要求的结果是： $3+12=15$



6.7 和最小

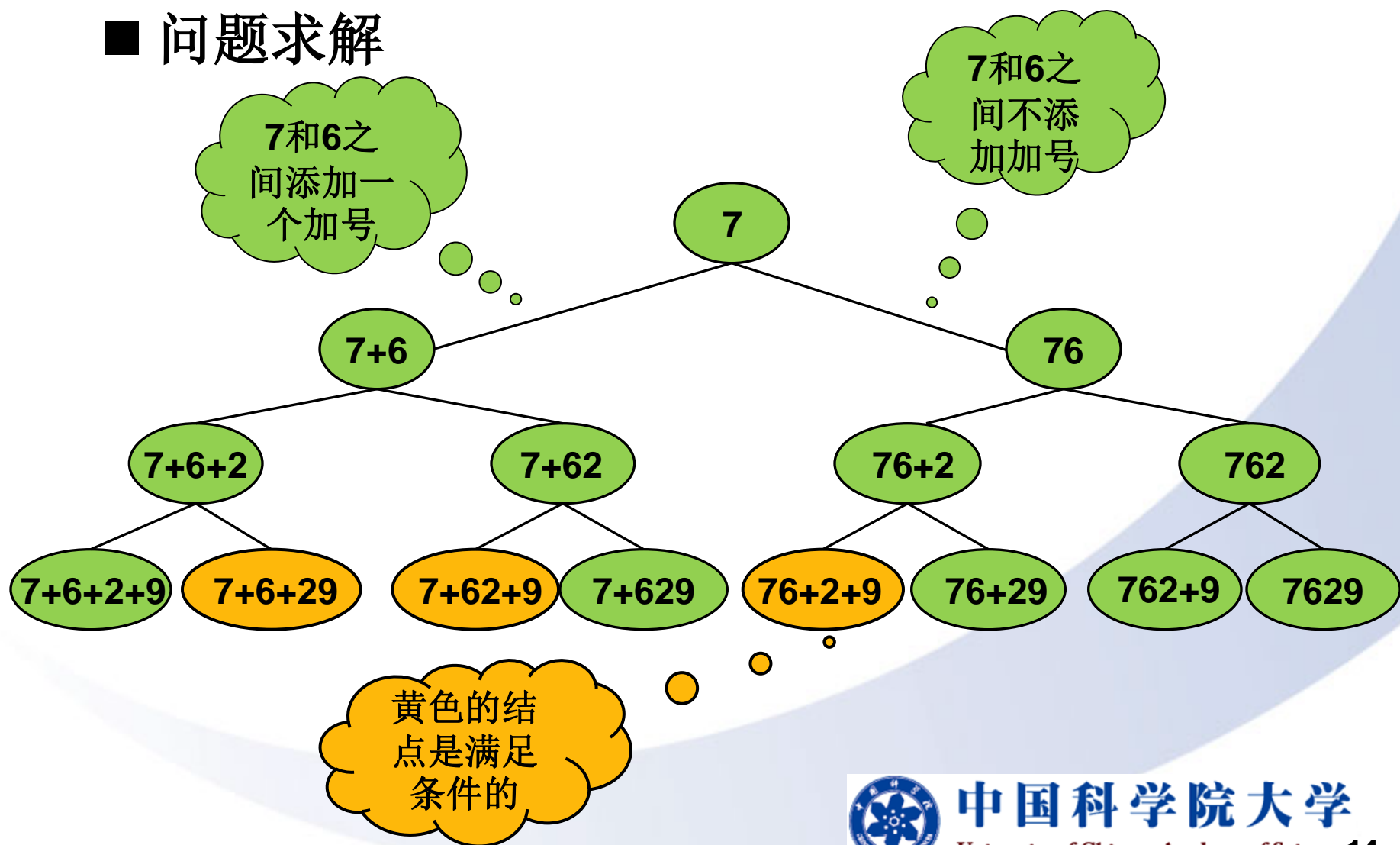
■ 问题分析

- 题目要求的就是在每个数字之间：或者填加号，或者什么都不填。
- 根据这个要求，我们可以从头开始扫描整个数字串，逐个考察是否要填加号，然后检查下一个数字间的位置，直到最后一个数字。
- 例：
 - 数字7629需要插入2个加号的
 - 完整的搜索树。



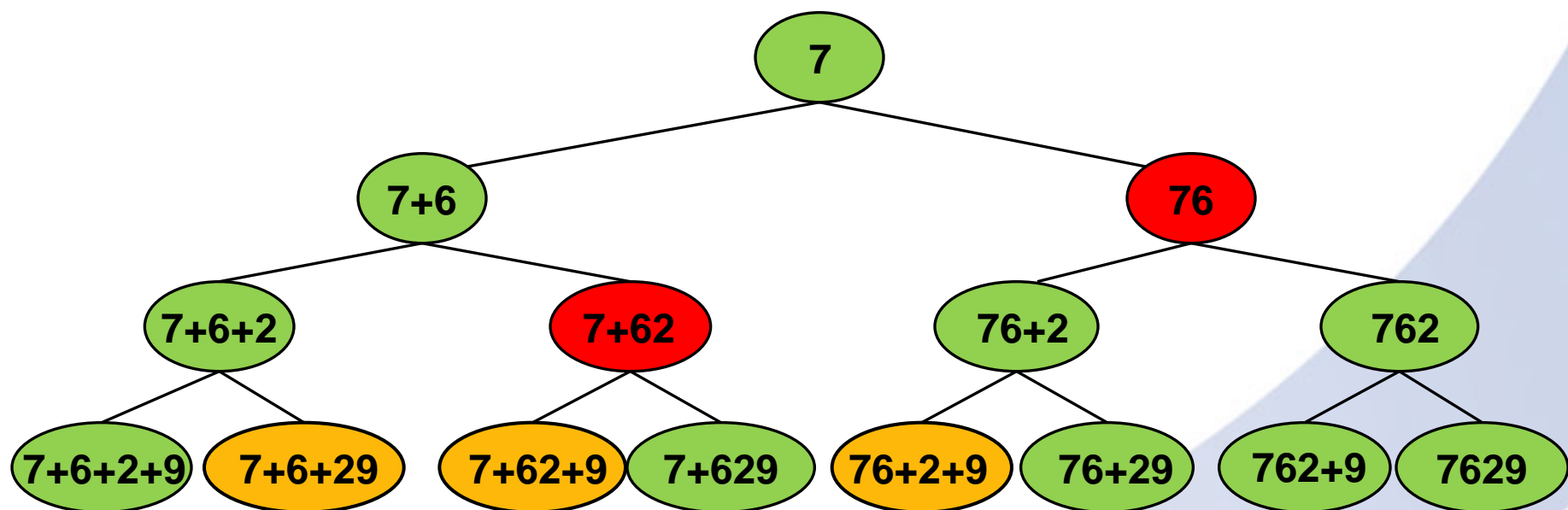
6.7 和最小

■ 问题求解



6.7 和最小

■ 问题求解

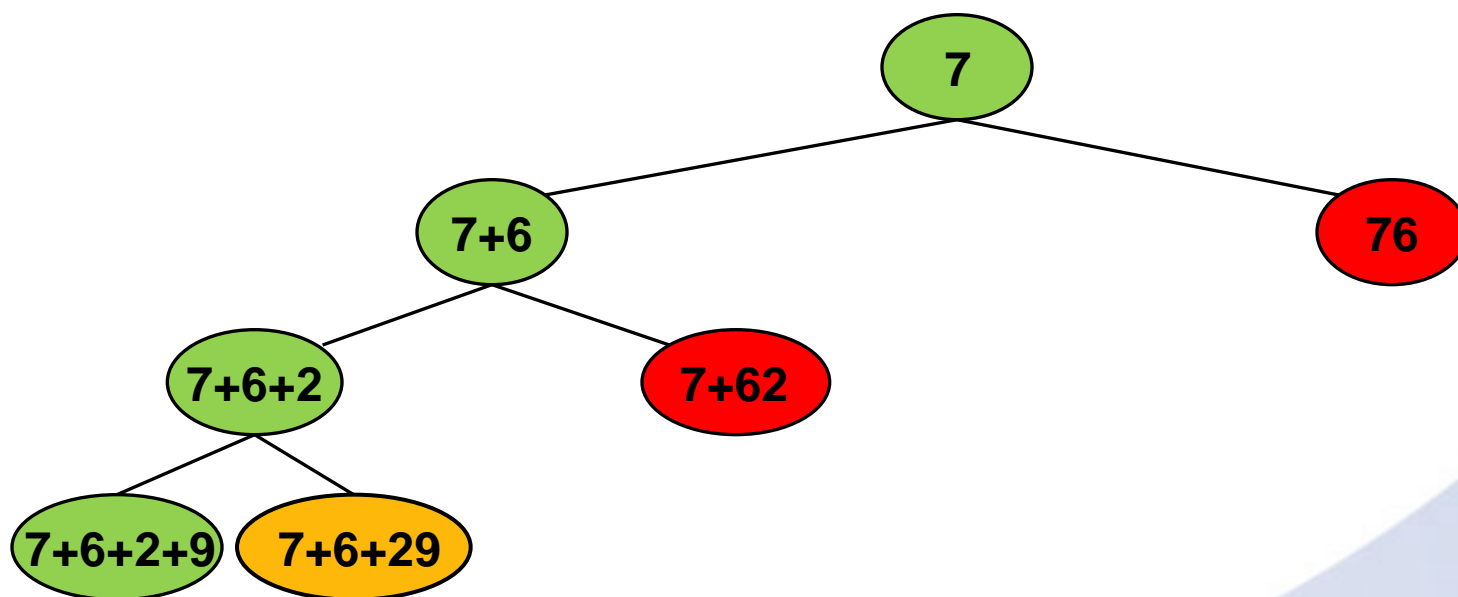


红色结点的子节点不可能有最优解!



6.7 和最小

■ 问题求解



第六章 回溯法

- 6.1 一般方法
- 6.2 8-皇后问题
- 6.3 子集和数问题
- 6.4 图的着色
- 6.5 0/1背包问题
- 6.6 哈密顿环
- 6.7 和最小
- 6.8 跳马问题



6.8 跳马问题

■ 问题描述

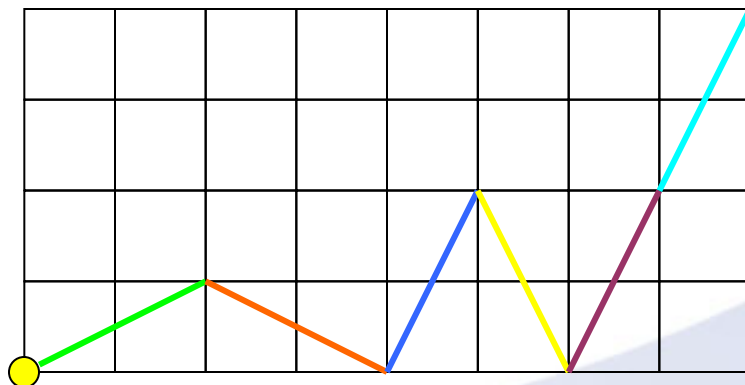
□ 在 $n \times m$ 棋盘上有一中国象棋中的马：

① 马走日字；

② 马只能往右走。

□ 请你找出一条可行路径，使得马可以从棋盘的左下角 $(1, 1)$ 走到右上角 (n, m) 。

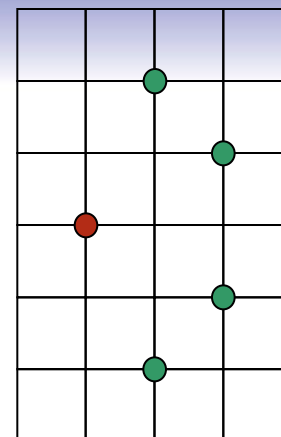
□ 例：



6.8 跳马问题

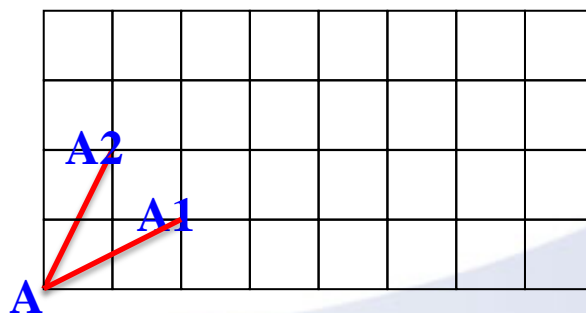
■ 问题分析

□ 按题意，马每一步可以有4种走法！



■ 搜索过程

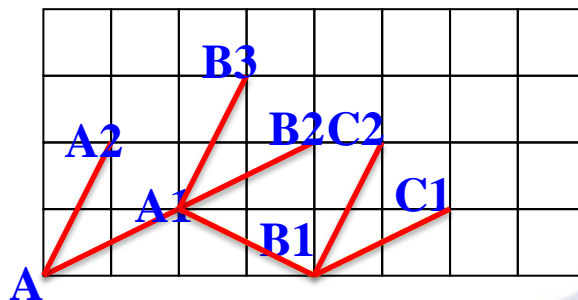
□ 当马一开始位于左下角的时候，根据规则，它只有两条线路可以选择（另外两条超出棋盘的范围），我们无法预知该走哪条，故任意选择一条，到达A1。



6.8 跳马问题

■ 搜索过程

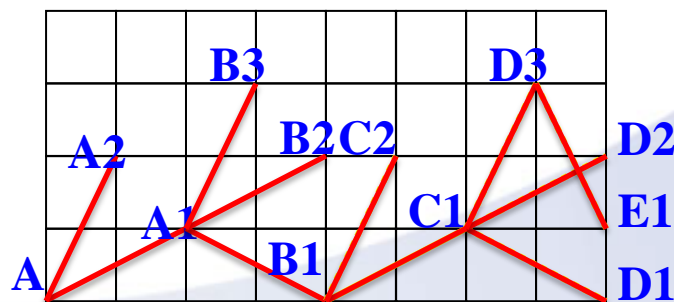
- 当到达A1点后，又有三条线路可以选择，于是再任意选择一条，到达B1。
- 从B1再出发，又有两条线路可以选择，先选一条，到达C1。



6.8 跳马问题

■ 搜索过程

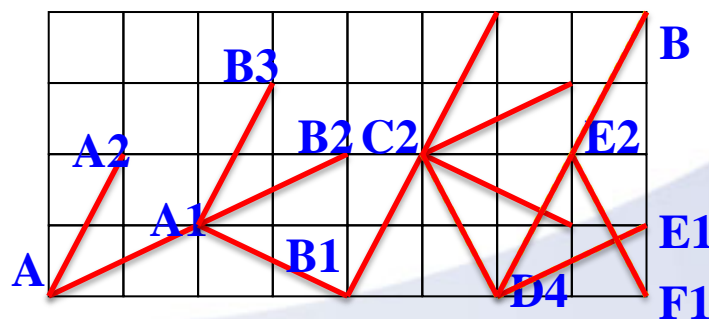
□ 从C1出发，可以有三条路径，选择D1。但到了D1以后，我们无路可走且D1也不是最终目标点，因此，选择D1是错误的，我们退回C1重新选择D2。同样D2也是错误的。再回到C1选择D3。D3只可以到E1，但E1也是错误的。返回D3后，没有其他选择，说明D3也是错误的，再回到C1。此时C1不再有其他选择，故C1也是错误的，退回B1，选择C2进行尝试。



6.8 跳马问题

■ 搜索过程

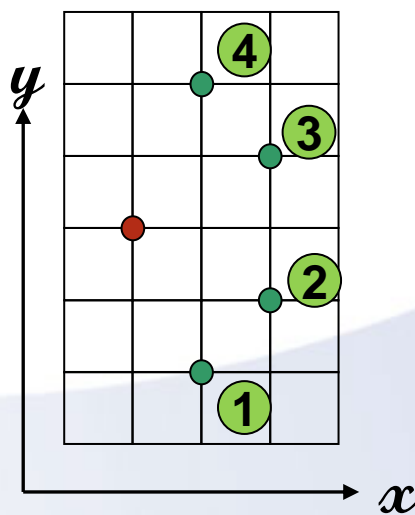
- 从C2出发，有四条路径可以选择，
- 选择D4，从D4出发又有两条路径，
- 选择E1错误，返回D4选择E2，
- 从E2出发有两条路径，先选择F1错误，
- 返回E2选择B，而B恰好是我们要到达的目标点，
- 至此，一条路径查找成功。



6.8 跳马问题

■ 算法实现

- 在无法确定走哪条线路的时候，任选一条线路进行尝试；为方便路径表示，对马可以走到的四个点（方向）都编上号；
- 当从某点出发，所有可能到达的点都不能到达终点时，说明此点是一个死节点，必须回溯到上一个点，并重新选择一条新的线路进行尝试。



6.8 跳马问题

■ 算法实现

□解空间：为了描述路径，我们最直接的方法就是记录路径上所有点的坐标。

□约束条件：不越界：

➤ $(x + dx[i] \leq n) \text{ and } (y + dy[i] > 0) \text{ and } (y + dy[i] \leq m)$

若马所处的位置为 (x,y) ，则其下一步可以到达的四个位置分别是 $(x+1, y-2)$ ， $(x+2, y-1)$ ， $(x+2, y+1)$ ， $(x+1, y+2)$ 。

增量数组：

$dx = (1, 2, 2, 1)$
 $dy = (-2, -1, 1, 2)$

`path:array[1...m] of integer;`
其中，`path[i]`：表示第*i*个节点所走的方向

方向 t ，下一步的位置就是
 $(x+dx[t], y+dy[t])$ 。

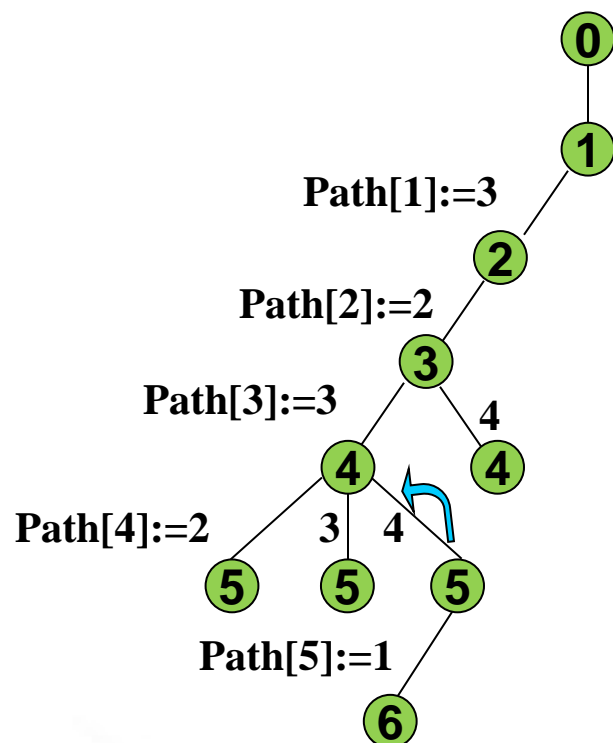


中国科学院大学

University of Chinese Academy of Sciences 24

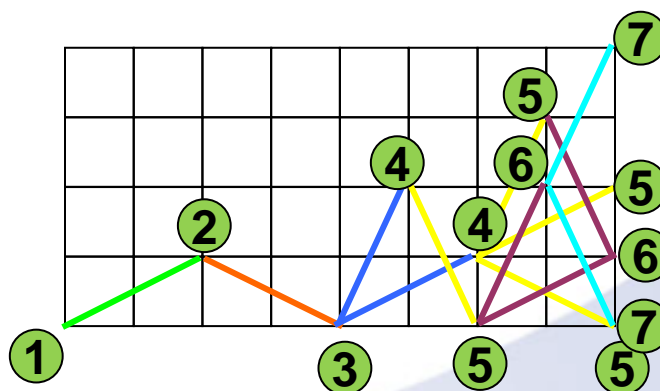
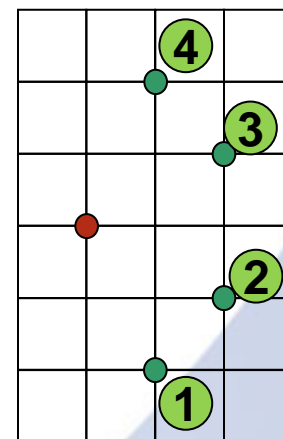
6.8 跳马问题

状态树:



算法描述:

1. 产生一种新走法
2. 越界, 继续用新走法, 直到找到一种走法不越界----不超过4种走法
3. if 不越界 then $k < n \rightarrow k+1$
 $k = n \rightarrow$ 一组解
4. if 越界 then 回溯



中国科学院大学

University of Chinese Academy of Sciences 25

跳马问题（递归）

Begin

procedure search(k: integer); // 递归查找

read(n, m);

x:=1; y:=1;

search(1);

begin

End.

for i := 1 **to** 4 **do** // 依次尝试四个方向

if (x + dx[i] ≤ n) **and** (y + dy[i] > 0) **and** (y + dy[i] ≤ m) **then**
//在棋盘上

begin

path[k] := i; // 记录下当前方向

x := x + dx[i]; y := y + dy[i]; // 修改扩展节点坐标

if (x = n) **and** (y = m) **then** // 是否是目标点

begin

output(k); **halt**; // 是目标点，输出结果并终止程序

end

else

search(k+1); // 不是目标点，继续尝试下一步

// 扩展出的点是死点，回溯

x := x - dx[i]; y := y - dy[i]; // 恢复扩展节点坐标，状态恢复

end;

end;



中国科学院大学

University of Chinese Academy of Sciences 26

作业-算法实现7

■ 问题描述-算24点

- 几十年前全世界就流行一种数字游戏，至今仍有人乐此不疲。在中国我们把这种游戏称为“算24点”。您作为游戏者将得到4个1~9之间的自然数作为操作数，而您的任务是对这4个操作数进行适当的算术运算，要求运算结果等于24。
- 您可以使用的运算只有： $+$ ， $-$ ， $*$ ， $/$ ，您还可以使用（）来改变运算顺序。注意：所有的中间结果须是整数，所以一些除法运算是不允许的（例如， $(2*2)/4$ 是合法的， $2*(2/4)$ 是不合法的）。下面我们给出一个游戏的具体例子：
- 若给出的4个操作数是：1、2、3、7，则一种可能的解答是 $1+2+3*7=24$ 。



作业-算法实现7

■ 问题描述-算24点

□ 输入

- 只有一行，四个1到9之间的自然数。

□ 输出

- 如果有解的话，只要输出一个解，输出的是三行数据，分别表示运算的步骤。其中第一行是输入的两个数和一个运算符和运算后的结果，第二行是第一行的结果和一个输入的数据、运算符、运算后的结果；第三行是第二行的结果和输入的一个数、运算符和“=24”。如果两个操作数有大小的话则先输出大的。
- 如果没有解则输出 “No answer!”



作业-算法实现7

■ 问题描述-算24点

□ 输入样例

➤ 1 2 3 7

□ 输出样例

➤ $2+1=3$

➤ $7*3=21$

➤ $21+3=24$

■ 要求

□ 作业提交到课程网站上

□ 用C(C++)或者matlab实现

□ 要有算法的求解说明



中国科学院大学

University of Chinese Academy of Sciences 29

END

