3.3 习题解析

1. 对 $N=5, k\leqslant 3$ 的传教士和野人问题，定义两个 h 函数（非零），并给出用这两个启发式函数的 A 算法搜索图。讨论用这两个启发函数求解该问题时是否得到最佳解。

参考答案：

1）传教士和野人问题描述

有 N 个传教士和 N 个野人来到河边渡河，河岸有一条船，每次至多可供 k 人乘渡。问传教士为了安全起见，应规划摆渡方案，使得任何时刻，传教士和野人在一起时河两岸及船上的野人数目总是不超过传教士的数目（否则传教士有可能被野人吃掉）。求解传教士和野人从左岸全部摆渡到右岸的过程中，任何时刻传教士和野人在一起时，满足传教士数大于或等于野人数，并且两者人数之和小于 k 的摆渡方案。

2）求解过程

设 M、C 为某一时刻传教士人数和野人人数，k 为船能够承载的人数。要求 $M\geqslant C$ 且 $M+C\leqslant k$。又设 L 表示船在左岸，R 表示船在右岸，并假设传教士和野人都为 5 时，该问题的状态空间可表示如下。

初始状态

	L	R
M	5	0
C	5	0
B	1	0

目标状态

	L	R
M	0	5
C	0	5
B	0	1

其中，$B=1$ 代表在左岸，$B=0$ 代表不在左岸。

由于野人和传教士的总数是一定的，我们可以只考虑左岸或右岸就可以了（这就减少了状态变量），因此，可以用如下方式考虑状态空间。

（1）定义状态空间。用三元组 $S_k=$(ML,CL,BL)来表示传教士、野人和船是否在左岸的状态，其中 ML 表示传教士在左岸的实际人数，CL 表示野人在左岸的实际人数，BL 用来指示船是否在左岸。

条件：$0\leqslant$ML,CL$\leqslant 5$,BL$\in\{0,1\}$

问题即转化为：初始状态为(5,5,1)，目标状态为(0,0,0)

（2）定义操作规则。为了表述的简洁，减少操作规则的罗列，这里假设 $N=3, k=2$。按每次渡河的人数分别写出每一个规则，共(3 0)、(0 3)、(2 1)、(1 1)、(1 0)、(0 1)、(2 0)、(0 2) 8 种渡河的可能（其中(x y)表示 x 个传教士和 y 个野人一起上船渡河），因此共有 16 个规则（从左岸到右岸、右岸到左岸各 8 个）。注意：这里没有(1 2)，因为该组合在船上的传教士人数少于野人人数。

规则集如下。

P01　if(ML,CL,BL=1)　then(ML,CL−1,BL−1)
P02　if(ML,CL,BL=1)　then(ML,CL−2,BL−1)
P03　if(ML,CL,BL=1)　then(ML,CL−3,BL−1)
P10　if(ML,CL,BL=1)　then(ML−1,CL,BL−1)
P11　if(ML,CL,BL=1)　then(ML−1,CL−1,BL−1)
P20　if(ML,CL,BL=1)　then(ML−2,CL,BL−1)
P21　if(ML,CL,BL=1)　then(ML−2,CL−1,BL−1)
P30　if(ML,CL,BL=1)　then(ML−3,CL,BL−1)

Q01　if(ML,CL,BL=0)　then(ML,CL+1,BL+1)
Q02　if(ML,CL,BL=0)　then(ML,CL+2,BL+1)
Q03　if(ML,CL,BL=0)　then(ML,CL+3,BL+1)
Q10　if(ML,CL,BL=0)　then(ML+1,CL,BL+1)
Q11　if(ML,CL,BL=0)　then(ML+1,CL+1,BL+1)

Q20　if(ML,CL,BL=0)　then(ML+2,CL,BL+1)

Q21　if(ML,CL,BL=0)　then(ML+2,CL+1,BL+1)

Q30　if(ML,CL,BL=0)　then(ML+3,CL,BL+1)

为了理解以上规则的含义，下面以规则 P03 来说明。

P21　if(ML,CL,BL=1)　then(ML−2,CL−1,BL−1)

表示从左岸渡了两个传教士和一个野人到右岸，因此，左岸的传教士和野人的人数分别为 ML−2 和 CL−1，而此时 BL−1=0，表示船在右岸。

Q21 表示的意义正好相反，即从右岸渡了两个传教士和一个野人到左岸。

(3) 启发式搜索策略。启发式函数：这里的启发式函数是使用 A 算法中的启发式函数表示的，即

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

其中，$g(n)$是搜索到节点 n 时已经花费的代价；$h(n)$指估计函数里的启发式部分，是从节点 n 到目标节点的最优路径的估计代价，搜索的启发信息主要由 $h(x)$来体现。

① 启发式函数 1：

$$f(n) = \begin{cases} g(n), & \text{两岸中传教士数目} \geqslant \text{野人数目} \\ \infty, & \text{其他} \end{cases}$$

其中，$g(n)$表示搜索树的深度，也是船只来回的次数；f 函数中启发函数为零，即 $h(n)=0$。此时对应的搜索图如图 3.1 所示。

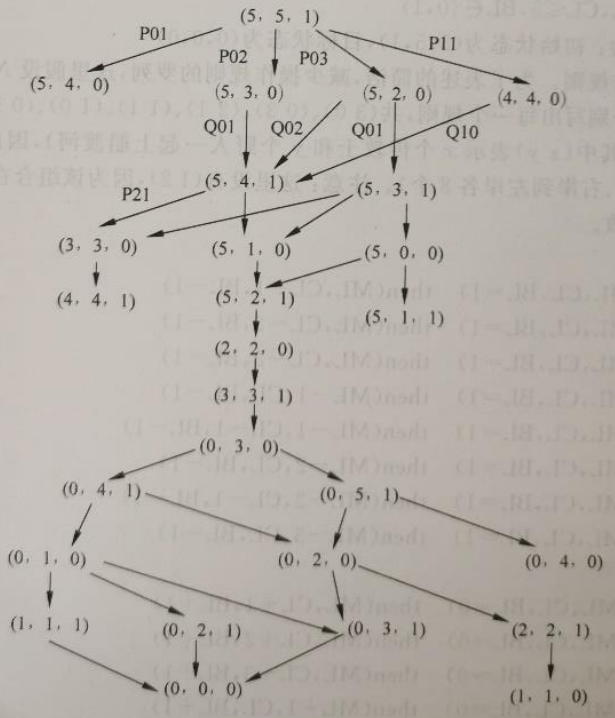

图 3.1　启发式函数 1 对应的搜索图

② 启发式函数 2：

$$f(n)=\begin{cases} g(n)+M+C, & \text{两岸中传教士数目} \geqslant \text{野人数目} \\ \infty, & \text{其他} \end{cases}$$

即启发函数为 $h(n)=M+C$。此时对应的搜索图如图 3.2 所示。

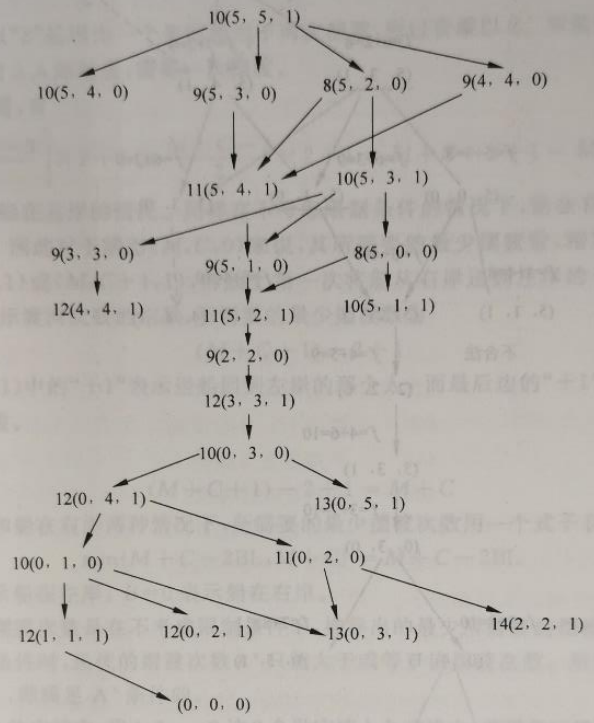

图 3.2　启发式函数 2 对应的搜索图

搜索图中的初始节点 10(5,5,1)中，表示 f 值为 $10(f(n)=g(n)+M+C=0+5+5=10)$，其中(5,5,1)表示有 5 个传教士、5 个野人在左岸、船在左岸。

③ 定义启发式函数 3：

$$f(n)=\begin{cases} g(n)+\mathrm{ML}+\mathrm{CL}-2\mathrm{BL}, & \text{两岸中传教士数目} \geqslant \text{野人数目} \\ \infty, & \text{其他} \end{cases}$$

其中，$g(n)$表示搜索树的深度，也是船只来回的次数。启发函数为 $h(n)=M+C-2\mathrm{BL}$，此时对应的搜索图如图 3.3 所示。

④ 关于 $h(n)=\mathrm{ML}+\mathrm{CL}-2\mathrm{BL}$ 的分析。该 $h(n)$是满足 A* 算法条件的，即 $h \leqslant h^*$，而 $h(n)=M+C$ 是不满足。要说明 $h(n)=M+C$ 不满足 A* 条件是很容易的，只需要给出一个反例就可以了。例如，对于状态(1,1,1)，$h(n)=M+C=1+1=2$，而实际上只要一次摆渡就可以达到目标状态，其最优路径的耗费值为 1，不满足 $h \leqslant h^*$，所以不满足 A* 的条件。

下面证明 $h(n)=M+C-2\mathrm{BL}$ 满足 A* 条件，即证明 $h \leqslant h^*$。以下分两种情况考虑。

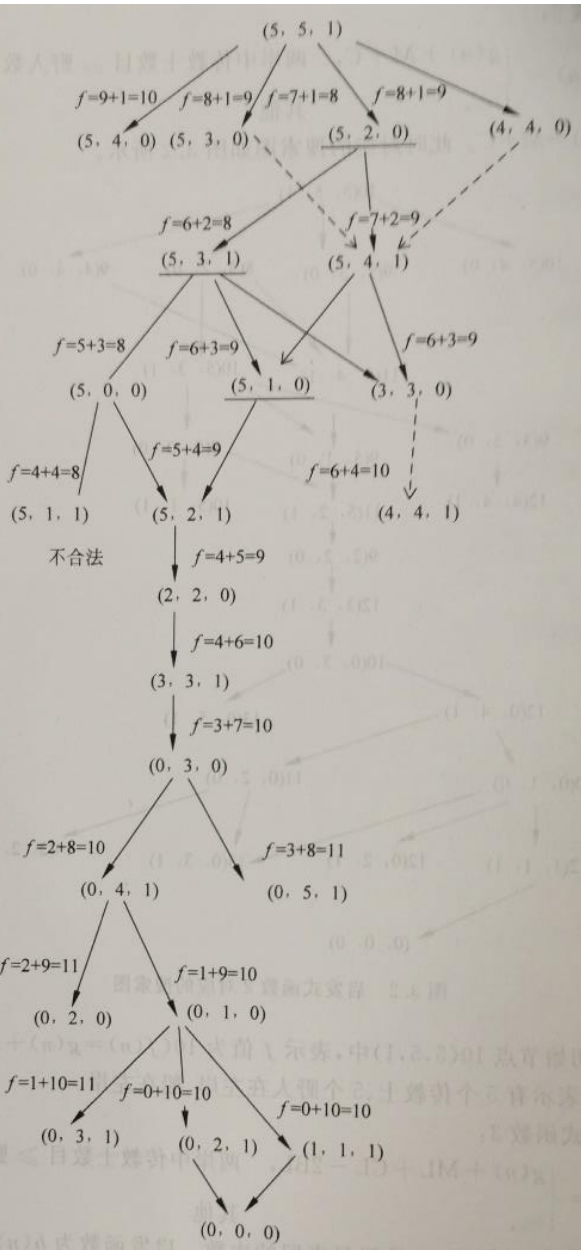

图 3.3　启发式函数 3 对应的搜索图

首先，考虑船在左岸的情况。如果不考虑限制条件（当野人和传教士在一起时，传教士人数不少于野人的人数），也就是说，船一次可以将 3 人从左岸运到右岸，然后再由 1 人将船送回来。这样，船一个来回可以运过河 2 人，而船仍然在左岸。而到最后渡河阶段剩下 3 人或少于 3 人时，则可以一次将他们全部从左岸运到右岸。所以，在不考虑限制条件的情况

下，也至少需要摆渡$\left\lceil\frac{M+C-3}{2}\right\rceil\times2+1$次。

其中，分子上的“-3”表示剩下3个留待最后一次运过去；除以“2”是因为一个来回可以运过去2人，需要$\frac{M+C-3}{2}$个来回，而“来回”数不能是小数，需要向上取整，取整用符号$\lceil\ \rceil$表示；而乘以“2”是因为一个来回相当于两次摆渡，所以要乘以2。而最后的“$+1$”，则表示将最后剩下的3人运过去，需要一次摆渡。

将$h(n)$化简，有

$$\left\lceil\frac{M+C-3}{2}\right\rceil\times2+1\geqslant\frac{M+C-3}{2}\times2+1=M+C-3+1=M+C-2$$

其次，考虑船在右岸的情况。同样在不考虑限制条件的情况下，船在右岸，需要一个人将船运到左岸。因此对于状态$(M,C,0)$来说，其所需要的最少摆渡数，相当于船在左岸时状态$(M+1,C,1)$或$(M,C+1,1)$，再加上第一次将船从右岸送到左岸的一次摆渡数。因此，利用前面左岸渡河次数的结果，所需要的最少摆渡数为

$$(M+C+1)-2+1$$

其中，$(M+C+1)$中的“$+1$”表示送船回到左岸的那个人；而最后边的“$+1$”，表示送船到左岸时的一次摆渡。

化简后，有

$$(M+C+1)-2+1=M+C$$

综合船在左岸和船在右岸两种情况下，所需要的最少摆渡次数用一个式子表示为

$$\min\{M+C-2\mathrm{BL},M+C\}=M+C-2\mathrm{BL}$$

其中，$B=1$表示船在左岸；$B=0$表示船在右岸。

由于这个摆渡次数是在不考虑限制条件下，推导出的最少所需要的摆渡次数，即$h(n)$。而考虑有限制条件时，最优的摆渡次数h^*只能大于或等于该摆渡次数。所以该启发函数h是满足：$h\leqslant h^*$，即满足A^*条件的。

2. 在3×3的宫格内，用1,2,…,9这9个数字填入九宫格内，使得每行数字组成的十进制数平方根为整数。试用启发式搜索算法求解，分析问题空间的规模和有用的启发式信息。

参考答案：

1）求解方法1

（1）问题的状态空间可表示如下。

a_1	a_2	a_3
a_4	a_5	a_6
a_7	a_8	a_9

首先对要考查的对象进行分析，是数字1～9还是100～999？抑或是从题意衍生出的某个集合？如果是1～9，那么整个集合会有$9\times8\times7\times6\times5\times4\times3\times2\times1$个对象，即问题空间的规模是1～9这9个数填入9个空格中的排列9!。

（2）有用的启发式信息。根据题意，某个数的平方需要构成3位数，又不能大于3位数，这个数必须大于10，可推导出这个数的考查集合为10～31（平方根范围）。

然后计算出这个数的平方相应的三位数对应表如下。

10	100	21	441
11	121	22	484
12	144	23	529
13	169	24	576
14	196	25	625
15	225	26	676
16	256	27	729
17	289	28	784
18	324	29	841
19	361	30	900
20	400	31	961

因为1～9的限制和互斥性(不能重复),首先可过滤掉如下一批对象。

100　超出1～9的限制
121　不满足互斥条件
225　不满足互斥条件
400　超出1～9的限制
441　不满足互斥条件
484　不满足互斥条件
676　不满足互斥条件
900　超出1～9的限制

得出可能满足条件的对象列表如下。

169　196　256　289　324　361　529　576　625　729　784　841　961

① 启发式信息1:因为数字之间互斥的要求,所以对对象中各数字出现的频率进行统计,频率小的,满足互斥条件的可能性越大(这就是启发式信息)。因此,统计如下。

1　5次
2　6次
3　2次
4　3次
5　4次
6　7次
7　3次
8　3次
9　6次

优先考查次数最少的为3、7、8、4。我们首先考查3、8:

324　289　361　784　841

因为3只在两组数中出现,所以324和361必具其一。假设是324被选中,那么289、784、841这一组将无法出现,所以只能是361被选中。

然后来考查5:因为6的排斥作用令529直接选取。

最后来考查 7：由于选取了 361 和 529，所以在{576，729，784}中，只有 784 可选取。因此，结果为 361、529、784。

② 启发式信息 2：由平方数的性质可知，3 个十进制数的个位只能为 1、4、5、6、9，由此可知个位数组合的总数为 $C_5^3=10$。然后定义启发式函数 $f(S_n)$=1，4，5，6，9 中已使用数的个数，S_n 为已经完成的状态，则 $f(S_n)$越小越好。

整个启发式过程如下。

(1) 每一步为从左向右，从上向下得向九宫格内放一个数，记下当时的状态 S_n，$n\geqslant 0$。

(2) 利用 $f(S_n)$判断下一步。当 $f(S_n)$最小时，选择此分支。

(3) 每次走到第 3 列时，判断当前行是否平方根为整数，若不是则抛弃此行，并回溯。

其结果如下。

3	6	1
5	2	9
7	8	4

2) 求解方法 2

(1) 定义状态空间。1，2，…，9 这 9 个数的一个排列为一个状态$(a_1,a_2,a_3,a_4,a_5,a_6,a_7,a_8,a_9)$。其目标状态为$(b_1,b_2,\cdots,b_9)$，其中 $100b_1+10b_2+b_3$，$100b_4+10b_5+b_6$，$100b_7+10b_8+b_9$ 为完全平方数。

(2) 定义操作规则。a_i 和 a_j 对换$(1\leqslant i\leqslant 9,1\leqslant j\leqslant 9)$。

(3) 采用启发式搜索必须缩小状态空间。其状态空间定义如下。

$$(X_1,X_2,X_3)(100\leqslant X_1\leqslant 999,100\leqslant X_2\leqslant 999,100\leqslant X_3\leqslant 999)$$

其中，X_1，X_2，X_3 为完全平方数且各位数字各不相同。

(4) 采用启发式搜索算法如下。

① 求出 10^2，11^2，12^2，…，31^2 三位数字各不相同的数放入数组 A 中。可得取值范围为 169，196，256，289，324，361，529，576，625，729，784，841，961。

② 从 A 中选取 3 个数，使这 3 个数恰好用到 1～9 这 9 个数字。且尾数必须是 1，4，5，6，9 中的一位，放入 3×3 宫格内。

具体操作步骤为：若要放入的数字与已放入的数字无冲突，则放入；若找不到可以放入的数，则将空格内的数字清空。

最后求出其中一组解如下。

3	6	1
5	2	9
7	8	4

3. 试给出爬山法和最佳优先搜索算法搜索图 3.4 所示的搜索路径。

参考答案：

(1) 爬山法：对新的可能解与至今测试过的解进行比较，如果最接近解，则保留为最佳解；否则舍弃。

(2) 最佳优先搜索：将新生成的解集加入到可能解集中，从所有解集中挑选最好的元

4. 找出深度优先搜索算法与广度优先搜索方法在文字叙述上的区别。

参考答案：

仅差一字，将新节点加入 OPEN 表，分别是加入“首”部和“尾”部。

(1) 深度搜索：若 n 不是目标，则将 n 的后继节点加入到 OPEN 表的首部。

(2) 广度搜索：若 n 不是目标，则将 n 的后继节点加入到 OPEN 表的尾部。

5. 在哪种问题空间，深度优先搜索优于广度优先搜索？

参考答案：

(1) 在有多条路径通向解且其中每条都很长的情况下。

(2) 当求解问题不要求找到所有的解及最优解，且扩展的节点较多并容易出现多分支，造成组合爆炸问题时，用深度优先搜索算法较好。

6. 在什么时候最佳优先搜索比广度优先搜索糟？

参考答案：

最佳优先搜索算法的评价函数定义得不好，往往是对达到目标的费用估计过低，导致在搜索图乱跳。此时，最佳优先搜索比广度优先搜索更差，特别是最佳优先搜索的启发式函数为 $f'=s'+h'$ 是真实的评价函数 $f=s+h$ 的估计函数。h' 为当前节点到目标节点的耗费的估计，当 h' 过高估计 h 时，最佳优先搜索比广度优先搜索糟，有可能找不到解。例如，在图 3.6 所示的搜索图中，对 D 节点的估计过高，就会将搜索引导到 B 或 C 节点所在的分支。

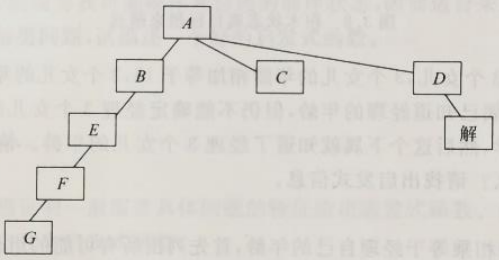

图 3.6　解离起始点比较近的搜索图

7. 考虑将图 3.7(a)中的积木形式转换成图 3.7(b)中的积木形状这一问题，可以使用的操作符有 PICKUP、PUTDOWM、STACK 和 UNSTACK。用爬山法求解这一问题有理吗？为什么？

参考答案：

(1) 定义机器人操作如下。

PICKUP(X)：从地上取出 X。

PUTDOWN(X)：把 X 放在地上。

STACK(X,Y)：把 X 放在 Y 上。

UNSTACK(X,Y)：把 X 从 Y 上取下。

注意，有一个约束条件是机器人一次只能搬动一块积木。在这个限制之下，机器人一种可行的求解此问题的方法如下。

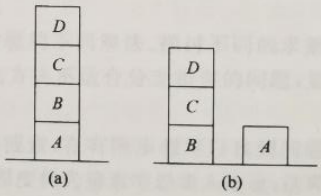

图 3.7　积木的搬动

4. 找出深度优先搜索算法与广度优先搜索方法在文字叙述上的区别。

参考答案：

仅差一字，将新节点加入 OPEN 表，分别是加入“首”部和“尾”部。

(1) 深度搜索：若 n 不是目标，则将 n 的后继节点加入到 OPEN 表的首部。

(2) 广度搜索：若 n 不是目标，则将 n 的后继节点加入到 OPEN 表的尾部。

5. 在哪种问题空间，深度优先搜索优于广度优先搜索？

参考答案：

(1) 在有多条路径通向解且其中每条都很长的情况下。

(2) 当求解问题不要求找到所有的解及最优解，且扩展的节点较多并容易出现多分支，造成组合爆炸问题时，用深度优先搜索算法较好。

6. 在什么时候最佳优先搜索比广度优先搜索糟？

参考答案：

最佳优先搜索算法的评价函数定义得不好，往往是对达到目标的费用估计过低，导致在搜索图乱跳。此时，最佳优先搜索比广度优先搜索更差，特别是最佳优先搜索的启发式函数为 $f'=s'+h'$ 是真实的评价函数 $f=s+h$ 的估计函数。h' 为当前节点到目标节点的耗费的估计，当 h' 过高估计 h 时，最佳优先搜索比广度优先搜索糟，有可能找不到解。例如，在图 3.6 所示的搜索图中，对 D 节点的估计过高，就会将搜索引导到 B 或 C 节点所在的分支。

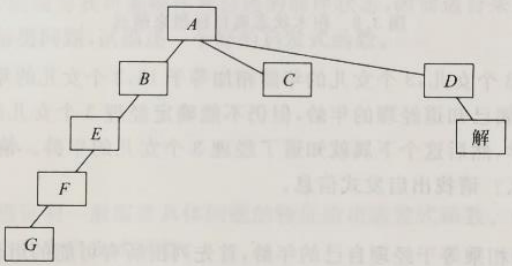

图 3.6　解离起始点比较近的搜索图

7. 考虑将图 3.7(a)中的积木形式转换成图 3.7(b)中的积木形状这一问题，可以使用的操作符有 PICKUP、PUTDOWM、STACK 和 UNSTACK。用爬山法求解这一问题有理吗？为什么？

参考答案：

(1) 定义机器人操作如下。

PICKUP(X)：从地上取出 X。

PUTDOWN(X)：把 X 放在地上。

STACK(X,Y)：把 X 放在 Y 上。

UNSTACK(X,Y)：把 X 从 Y 上取下。

注意，有一个约束条件是机器人一次只能搬动一块积木。在这个限制之下，机器人一种可行的求解此问题的方法如下。

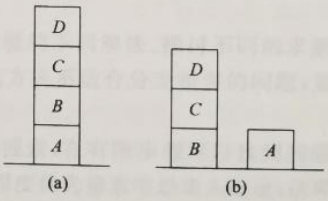

图 3.7　积木的搬动

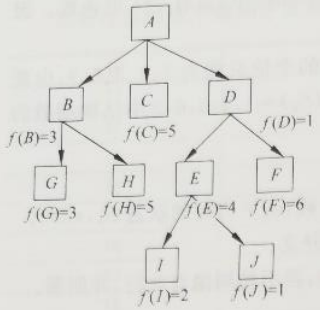

图 3.4　搜索图

素作为起点。

两者的区别是：前者从当前后继中找最好的；后者从所有已搜索过的路径找最佳。

搜索过程如图 3.5 所示。

解路径：爬山法为 $A\rightarrow D\rightarrow E\rightarrow J$；最佳优先搜索为 $A\rightarrow D\rightarrow B\rightarrow G$。

对于这个问题中，若解在 G 节点，则 $f(G)=0$，爬山法没有办法找到解；若解在 I 节点，则 $f(I)=0$，此时爬山法比最佳优先搜索更快地找到解。因为对最佳优先搜索来说，必须从 B 分支开始搜索，由于 G、H 不是解，因此 f 值必然较大，最后还是必须转移到 $D\rightarrow E\rightarrow I$ 的分支上来。

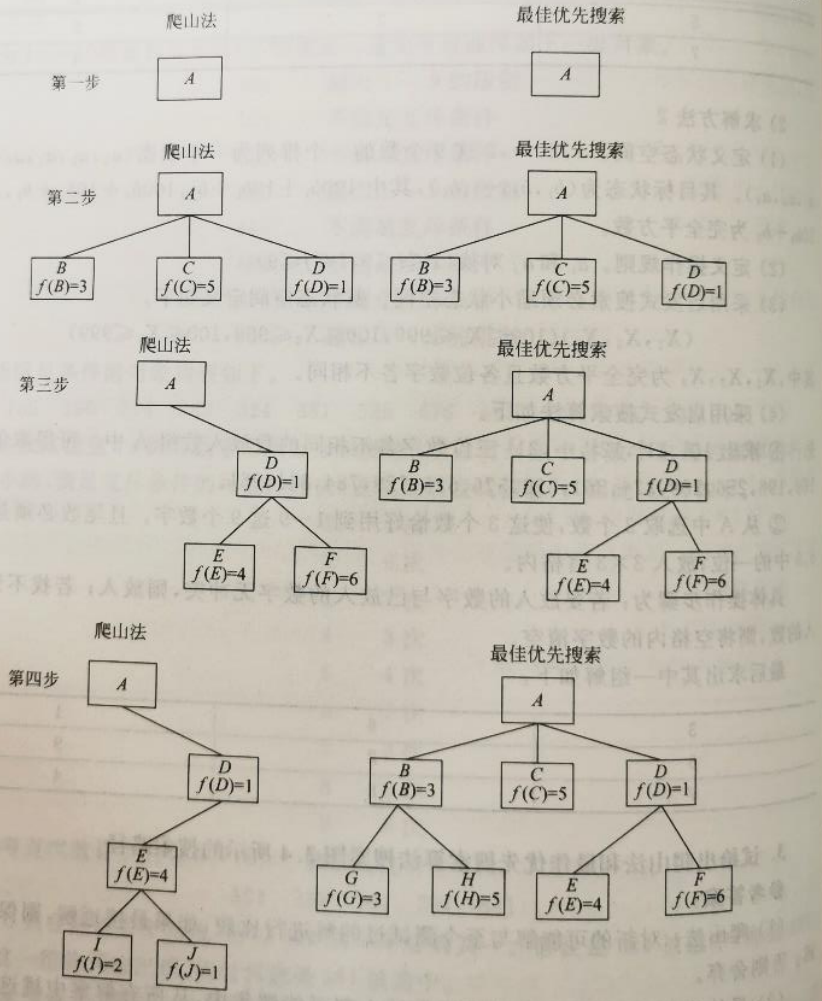

图 3.5　爬山法和最佳优先搜索算法对应的搜索图

(1) UNSTACK(D,C)　　(2) PUTDOWN(D)
(3) UNSTACK(C,B)　　(4) PUTDOWN(C)
(5) UNSTACK(B,A)　　(6) PUTDOWN(B)
(7) PICK(C)　　(8) STACK(C,B)
(9) PICK(D)　　(10) STACK(D,C)

(2) 使用爬山法求解这一问题不合理。因为爬山法搜索时局部最优，而该问题要用爬山法求解时，会做很多无用的搜索。

因为开始时，状态为 on(D,C)、on(C,B)、on(B,A)、on(A,table)，已经呈现出局部最优，4 个子目标 on(D,C)、on(C,B)、on(A,table)、on(B,table)中，只有 on(B,table)未达到。要在限制条件下，求解此问题，一种必需的动作是 Unstack(D,C)，若定义启发式函数 $f=$〈达到子目标的个数〉，则 $f(S)\leqslant 3$，在启发式函数引导下，又会将 D 搬到 C 之上，这显然是走了回头路。若进行了 Unstack(D,C)后，再 Unstack(C,B)，此时进入状态 1，再用爬山法求解局部最优时，为了取得启发式函数的最大值，又会选择动作 stack(C,D)，得到如图 3.8 所示的积木状态，显然这离目标越来越远。

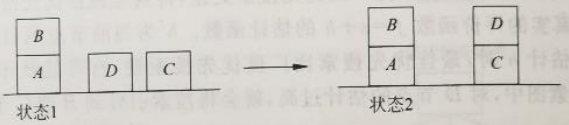

图 3.8　积木状态离目标越来越远

8. 一个经理有 3 个女儿，3 个女儿的年龄相加等于 13，3 个女儿的年龄相乘等于经理自己的年龄，有一个下属已知道经理的年龄，但仍不能确定经理 3 个女儿的年龄，这时经理说只有一个女儿在上学，然后这个下属就知道了经理 3 个女儿的年龄。请问 3 个女儿的年龄分别是多少？为什么？请找出启发式信息。

参考答案：

3 个女儿的年龄相乘等于经理自己的年龄，首先列出所有可能的组合，如下所示。

$1\times 1\times 11=11$
$1\times 2\times 10=20$
$1\times 3\times 9=27$
$1\times 4\times 8=32$
$1\times 5\times 7=35$
$1\times 6\times 6=36$
$2\times 2\times 9=36$
$2\times 3\times 8=48$
$2\times 4\times 7=56$
$2\times 5\times 6=60$
$3\times 3\times 7=63$
$3\times 4\times 6=72$
$3\times 5\times 5=75$
$4\times 4\times 5=80$

然后寻找启发式信息，根据题目中“有一个下属已知道经理的年龄，但仍不能确定经理3个女儿的年龄”这一句话可知，相乘得到与经理年龄相等的组合应该有两组或两组以上，这样才无法确定，否则就可以直接给出了。例如，如果经理年龄是80(假设)，那只有一组，即

4,4,5

因此从这个角度入手，可以从上面14组中筛选出两组(只有经理年龄为36岁时，员工才无法确定其女儿的年龄)，即

1,6,6

2,2,9

最后是那句“这时经理说只有一个女儿上学了”给出了启发式信息，得知1,6,6这组不可能，因为只有一个女儿上学，不可能是1岁的女儿上学，又不可能是两个6岁的女儿同时上学。所以可以得到最后答案：经理年龄36，他的3个女儿年龄是2,2,9。

9. 假设要写一道问题求解搜索程序去求解下面的每一类问题，试确定搜索是正向进行还是逆向进行：模式识别、积木世界、语言理解。

参考答案：

一般来说，模式识别、自然语言理解领域的问题求解，应该从当前已有数据和信息出发，通过搜索技术去寻找合理的目标模式，因而应该采用正向搜索；而积木世界则有可能从目标状态出发，一步步向前寻找可能操作和合理的前序状态，因而适合采用逆向搜索。

10. 对下面的每类问题，试描述一个好的启发式函数。

a. 积木世界。

b. 定理证明。

c. 传教士和野人。

参考答案：

积木世界和定理证明一般需要具体问题的特征给出启发式函数。传教士和野人问题在前面的习题中已给出几个启发式函数。

11. 分析宽度优先搜索和深度优先搜索的优缺点，举出它们的正例和反例。

参考答案：

宽度优先搜索的优点是：若问题有解，则可找出最优解；其缺点是：效率低，组合爆炸问题难以解决。

深度优先搜索的优点是：节省大量时间和空间；其缺点是：不一定能找到解。因为在深度无限搜索树的情况下，最坏的情况可能是不能停机。

小型的问题适合用宽度优先搜索方法，如求解数学题的不同解法、探讨不同的求解方法，若问题有解，一定可以用宽度搜索求得。但宽度优先方法不适合分支很多的问题，如果分支多，用宽度优先容易产生组合爆炸问题。

深度优先算法有限空间里的搜索，像走迷宫这样的搜索，在有限步骤可以找到问题的解，一旦找不到，可以寻找其他分支。在无限图的情况，深度优先搜索唯恐走入歧途，这样可能沿着某个路径无限搜索下去。

12. 有一个农夫带一只狐狸、一只小羊和一篮菜过河。假设农夫每次只能带一样东西过河，考虑安全，无农夫看管时，狐狸和小羊不能在一起，小羊和菜篮不能在一起。试设计求

解该问题的状态空间，并画出状态空间图。

参考答案：

以状态变量 m、f、s、v 分别表示农夫、狐狸、小羊、菜篮，且每个变量只可取值 1(表示在左岸)或 0(表示在右岸)。问题状态空间可以用四元组(m、f、s、v)描述，设初始状态下农夫、狐狸、小羊、菜篮均在左岸，目标状态为农夫、狐狸、小羊、菜篮都到达右岸。从而，问题求解任务可描述为

$$(1,1,1,1)\rightarrow(0,0,0,0)$$

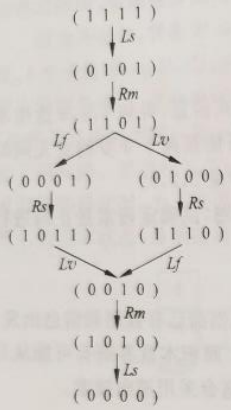

图 3.9　两条最短路径

由于问题空间简单，状态空间中可能的状态总数为 2×2×2×2=16，因此要遵从安全限制，合法的状态只有(除初始、目标状态外)。

1110,1101,1011,1010,0101,0001,0010,0100;

不合法状态有如下。

0111,1000,1100,0011,0110,1001

设计两类操作算子：Lx、Rx，x 为 m、f、s、v，L 表示农夫独自或带狐狸、带小羊、带菜蓝从左岸到右岸的操作，R 表示农夫独自或带狐狸、带小羊、带菜篮从右岸到左岸的操作，状态空间图如图 3.9 所示。由于 Lx 和 Rx 是互逆操作，故而解答路径可有很多条，但最近的路径只有两条，都由 7 个操作步骤构成。

(m、f、s、v)分别为农夫、狐狸、小羊、菜篮。

3.4　补充习题

1. 广度优先搜索与深度优先搜索各有什么特点？

参考答案：

广度优先搜索就是先在同一级节点中考查，只有当同一级节点考查完之后，才考查下一级节点；或者说，是以初始节点为根节点，向下逐级扩展搜索树。所以，广度优先策略的搜索树是自顶向下一层一层逐渐生成的。

深度优先搜索就是在搜索树的每一层先只扩展一个子节点，不断地向纵深前进，直到不能再前进(到达叶子节点或受到深度限制)时，才从当前节点返回到上一级节点，沿另一方向又继续前进。这种方法的搜索树是从树根开始一枝一枝逐渐形成的。深度优先搜索亦称为纵向搜索。由于一个有解的问题树可能含有无穷分枝，深度优先搜索如果误入无穷分枝(即深度无限)，则不可能找到目标节点。所以，深度优先搜索策略是不完备的。另外，应用此策略得到的解不一定是最佳解(最短路径)。

广度优先搜索与深度优先搜索都属于盲目搜索。

2. 图 3.10 是 5 个城市间的交通路线图，A 城市是出发地，E 城市是目的地，两个城市间的交通费用(代价)如图 3.10 中数字所示。求从 A 到 E 的最小费用交通路线。

参考答案：

先将交通图转换为搜索树，如图 3.11 所示。

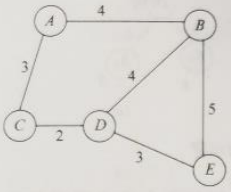

图 3.10　5 个城市间的交通路线图

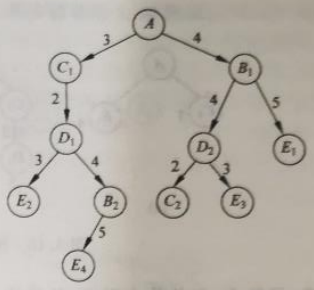

图 3.11　交通图转换为搜索树

若用 $g(x)$ 表示从初始节点 S_0 到节点 x 的代价，用 $c(x_1,x_2)$ 表示从父节点 x_1 到子节点 x_2 的代价，则有

$$g(x_2)=g(x_1)+c(x_1,x_2)$$

方法一：用广度优先进行搜索，扩展节点 n，并将其子节点放入 OPEN 表中，计算各子节点的代价，并按各节点的代价对 OPEN 表中全部节点按从小到大的顺序进行排序（队列）。其搜索步骤如图 3.12 所示。

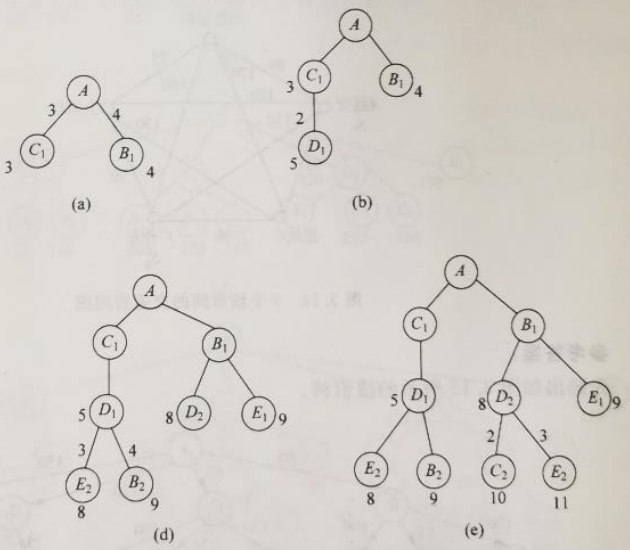

图 3.12　广度优先搜索的步骤

所以，最优路径为 $A\rightarrow C\rightarrow D\rightarrow E$。

方法二：用深度优先进行搜索（不一定是最优解），扩展节点 n，并将其子节点按代价从小到大的顺序放到 OPEN 表的首部（栈）。其步骤如图 3.13 所示。

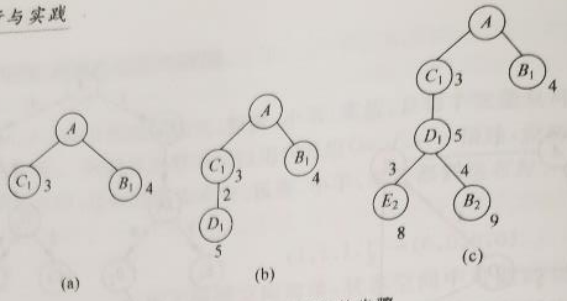

图 3.13　深度优先搜索的步骤

注意，虽然 D_1 的代价大于 B_1 的代价，但按照代价树的深度优先搜索策略，要对 D_1 进行扩展，放入 CLOSED 表中。若按最佳优先搜索，要对 B_1、D_1 排序，排序后应先扩展 B_1。

E 为目标节点，倒推的路径为 $E_2 \rightarrow D_1 \rightarrow C_1 \rightarrow A$，所以解路径为 $A \rightarrow C \rightarrow D \rightarrow E$。

此题中，深度优先搜索与搜索树的广度优先搜索的结果相同，但这只是巧合。一般情况下，这两种方法得到的结果不一定相同。另外，由于搜索树的深度优先搜索有可能进入一个无穷分枝的路径，因此它有可能找不到问题的真正解。

3. 图 3.14 是 5 个城市间的交通费用图，若从西安出发，要求把每个城市都访问一遍，最后到达广州，请找一条最优路线。路线边上的数字是两城市间的交通费用。

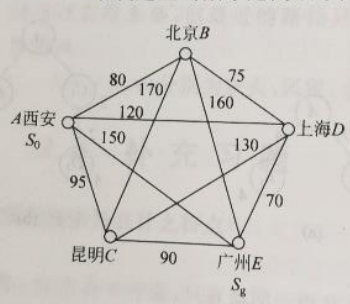

图 3.14　5 个城市间的交通费用图

参考答案：

先画出如图 3.15 所示的搜索树。

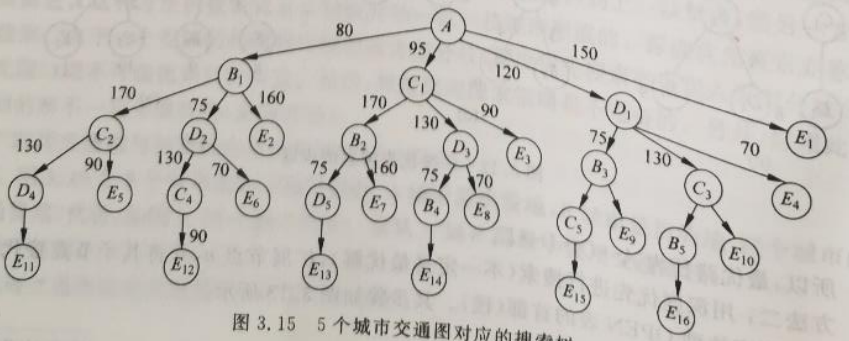

图 3.15　5 个城市交通图对应的搜索树

按搜索树的广度优先搜索即可得出最优路线，其步骤如图 3.16 所示。

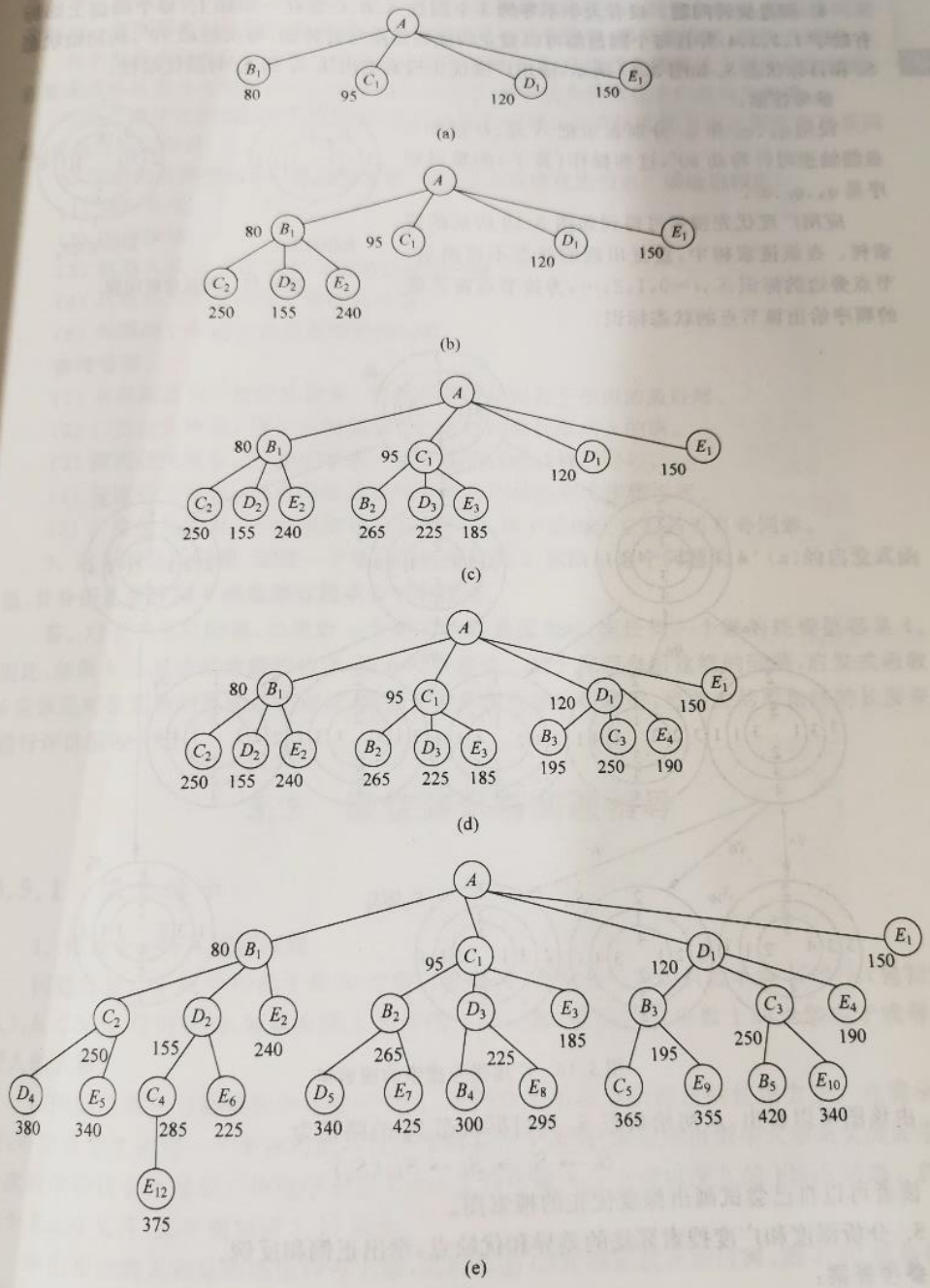

图 3.16　按搜索树求最优路线的步骤

因此可得出最优路线为 $A \to B_1 \to D_2 \to C_4 \to E_{12}$，即 $A \to B \to D \to C \to E$，解路径的费用为 375。

4. 圆盘旋转问题。设有大小不等的 3 个圆盘 A、B、C 套在一根轴上，每个圆盘上都标有数字 1、2、3、4，并且每个圆盘都可以独立的绕轴做逆时针转动，每次转动 90°，其初始状态 S_0 和目标状态 S_g 如图 3.17 所示，请用广度优先搜索求出从 S_0 到 S_g 的最优路径。

参考答案：

设用 q_A、q_B 和 q_C 分别表示把 A 盘、B 盘和 C 盘绕轴逆时针转动 90°，这些操作（算子）的排列顺序是 q_A、q_B、q_C。

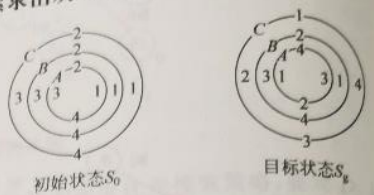

图 3.17　圆盘旋转问题

应用广度优先搜索可得到如图 3.18 所示的搜索树。在该搜索树中，重复出现的状态不再画出，节点旁边的标识 S_i，$i=0,1,2,\cdots$，为按节点被扩展的顺序给出该节点的状态标识。

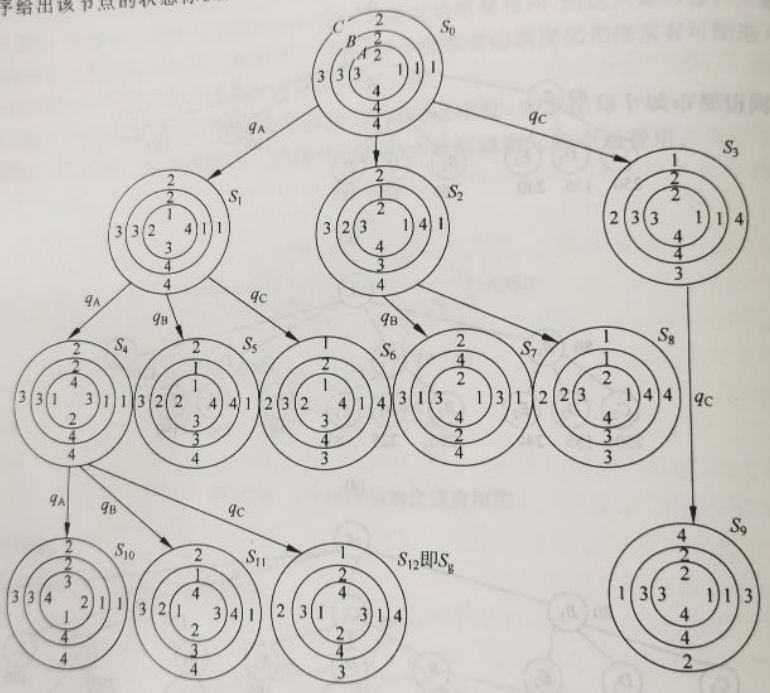

图 3.18　广度优先搜索的搜索树

由该图可以看出，从初始状态 S_0 到目标状态 S_g 的路径为

$$S_0 \to S_1 \to S_4 \to S_{12}(S_g)$$

读者可以自己尝试画出深度优先的搜索图。

5. 分析深度和广度搜索算法的差异和优缺点，举出正例和反例。

参考答案：

深度优先搜索与广度优先搜索的区别在于：在对节点 n 进行扩展时，其后继节点在

OPEN 表中的存放位置不同。广度优先搜索是将后继节点放入 OPEN 表的尾部，而深度优先搜索则是将后继节点放入 OPEN 表的首部。广度优先搜索是一种完备搜索，即只要问题有解就一定能够找到问题的解，而深度优先搜索是不完备搜索，即可能有解，但找不到解。

在不要求求解速度且目标节点的层次较深的情况下，广度优先搜索优于深度优先搜索；在要求求解速度且目标节点的层次较浅的情况下，深度优先搜索优于广度优先搜索。

支持广度优先的正例有积木问题，反例有迷宫问题；支持深度优先的正例有邮递员问题，反例有国际象棋。

6. 下面问题哪些适合广度优先搜索，哪些适合深度优先搜索？请给出理由。

(1) 国际象棋。

(2) 医疗诊断。

(3) 机器人从 *A* 点走到 *B* 点的路径规划问题。

(4) 从原料到产品的生产步骤的计划。

(5) 判断两个命题公式是否相等的问题。

参考答案：

(1) 有限深度的广度优先搜索：该程序必须找到若干步内的最好解。

(2) 广度优先搜索：医生要根据病人的各种病症判断病人的病。

(3) 深度优先搜索：该程序要求一定要找到到达目标的路径。

(4) 深度优先搜索：该程序要求找到生产出产品的最优步骤序列。

(5) 广度优先搜索：不能确定它们是否等同，即不能确定它们是否有等同解。

7. 对于四皇后问题，设放一个皇后的耗费值为 1，试给出这个问题的 $h^*(x)$ 的启发式函数，并分析是否任何 h 函数都对搜索有引导作用。

答：对于四皇后问题，如果放一个皇后的耗费值为 1，则任何一个解的耗费值都是 4。因此，如果 h 是对该耗费值的估计，则 h 没有意义。对于像四皇后这样的问题，启发式函数 h 应该是对找到解的可能性的评价，根据一个位置放置皇后之后，可消去的对角线的长度来进行评价值的设定。

3.5 课堂演示与实践指导

3.5.1 课堂演示

1. 传教士—野人过河问题

问题背景：有 N 个传教士和 N 个野人要过河，现在有一条船只能承载 K 个人(包括野人)，$K<N$，在任何时刻，如果有野人和传教士在一起，必须要求传教士的人数多于或等于野人的人数。

该问题已经在习题解析中详细讨论过，此处只是给出演示程序的使用方法。在演示之前，可以让学生使用一个手动的演示程序来体验这个游戏(该程序由清华大学吴文虎先生在一次教学研讨会上提供)，该程序是在 Excel 文件中嵌入一个可以交互的 Flash 动画。打开这个 Excel 文件，其界面如图 3.19 所示。

单击相应的人和鬼的图像即可上船，然后单击 GO! 图标就开始过河，图 3.20 就是过河情景之一。