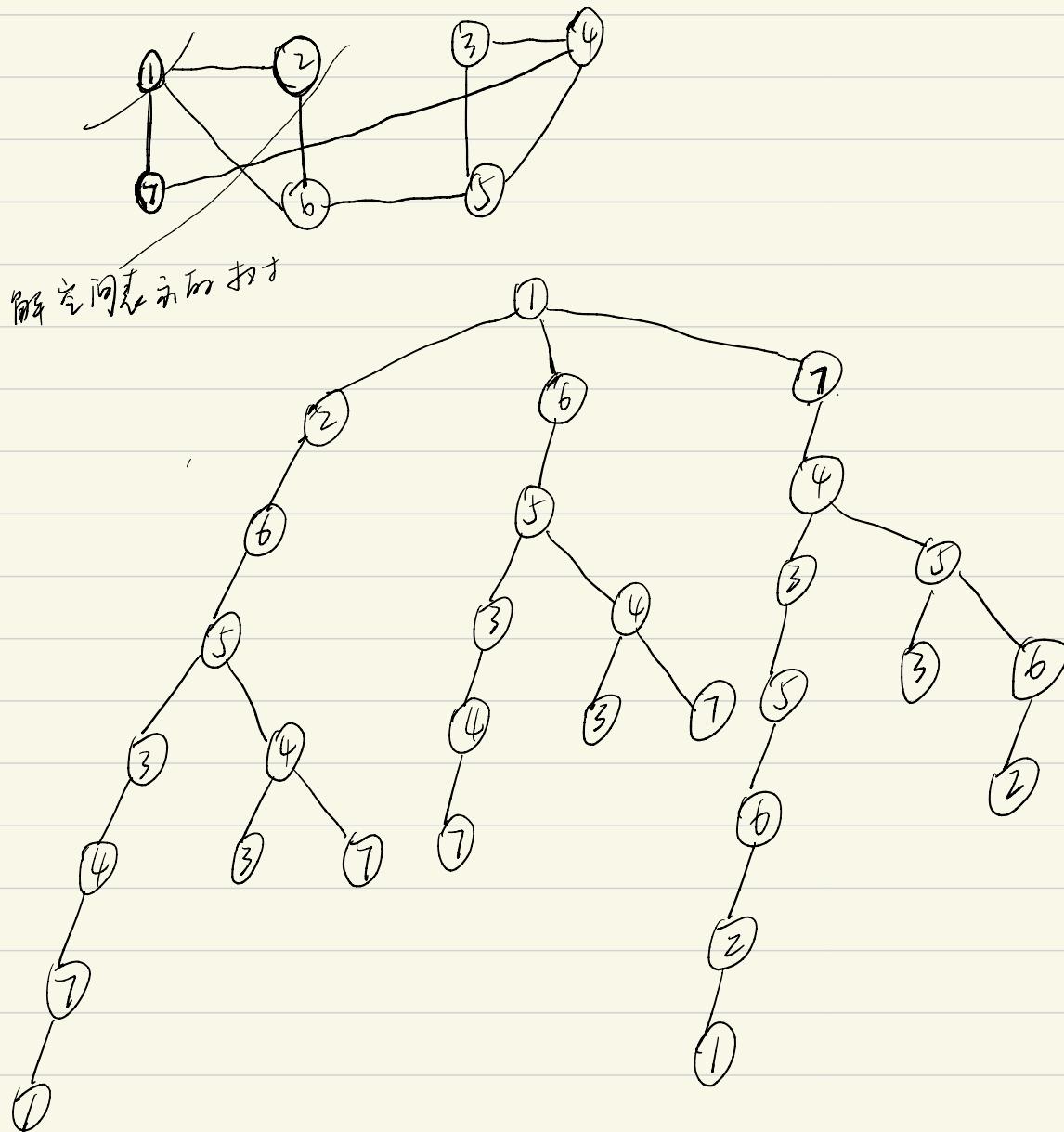


第7章

7.1



深度优先：

Stack S:

S.push(0);

while (!S.empty()) {

 de = S.top(); ele.visited[de] = true;

 设置 ele 为已访问 ← 会出问题

 S.pop(); 出栈

 将 S 的未访问的子结点入栈，

 如果所有的结点已访问，且将要入栈节点为①，

 则返回 true。

}

返回 false.

广度优先：

bool visited[]

queue q;

q.push(0);

while (!q.empty()) {

 de = q.top();

 设置 ele 为已访问 ←

 q.pop(); 出栈

 将 S 的未访问的子结点入栈，

 如果所有的结点已访问，且将要入栈节点为①，

 则返回 true。

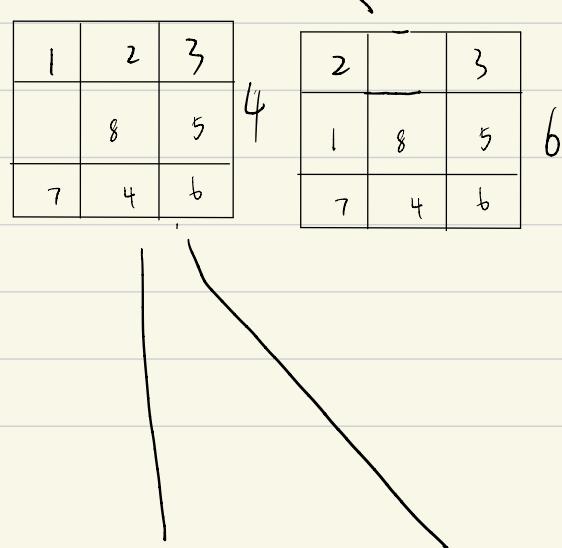
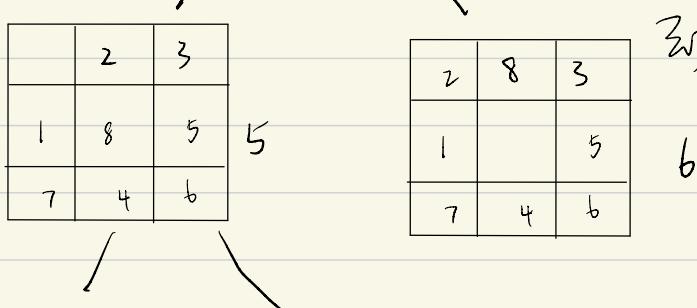
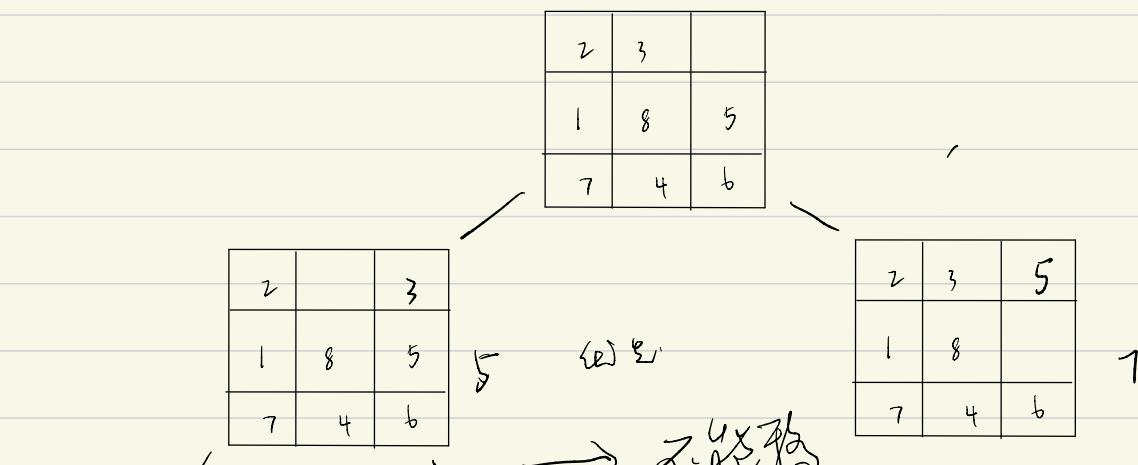
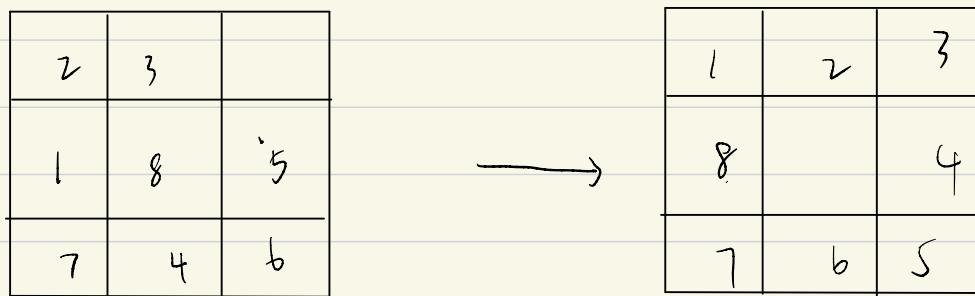
}

返回 false.

也需要把当前已访问放到队列中。

7.2 爬山法，最佳优先方法
 ↓
 使用贪心来确定搜索方向。
 (最佳优先搜索)

$f(n)$ 位于错误的方块数



1	2	3
8		5
7	4	6

1	2	3
7	8	5
	4	6

3

1	2	3
9	5	
7	4	6

1	2	3
8	4	5
7		6

3

1	2	3
8	5	6
7	4	

3

1	2	3
8		5
7	4	6

2

1	2	3
8	4	5
7	6	

1

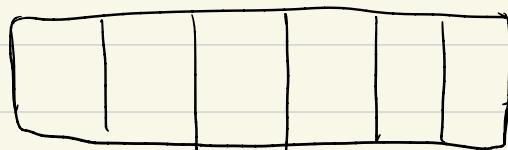
1	2	3
8	4	
7	6	5

↑

1	2	3
8		4
7	6	5

最佳优先判定.

输出数，在错误位置的元素的数量。



堆

最小堆

2	3	
1	8	5
7	4	6

① 入堆

2		3
1	8	5
7	4	6

2	3	5
1	8	
7	4	6

7 在堆中

如何判断之前是否
已经出现过相同的
排序

	2	3
1	8	5
7	4	6

2	8	3
1	.	5
7	4	6

6 在堆中

1	2	3
	8	5
7	4	6

4

1	2	3
8		5
7	4	6

3

1	2	3
7	8	5
	4	6

5 在堆中

1	2	3
8	4	5
7		6

3

1	2	3
8	5	
7	4	6

3 在堆中

1	2	3
8	4	5
7	6	

2

/

1	2	3
8	4	
7	6	5

{

/

1	2	3
8		4
7	6	5

0

'

Best-First

4.

每个节点 - n.

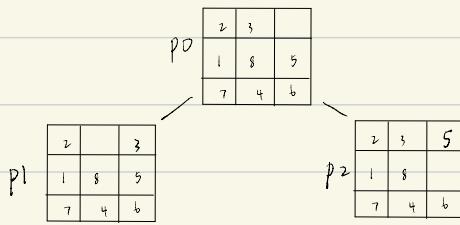
$$g(n) = \text{从根节点到 } n \text{ 的代价}.$$

$$h^*(n) = \text{从 } n \text{ 到 目标节点的代价} \rightarrow h(n) = h^*(n) \text{ 为估计. 和用 } f(n) \text{ 一样, 只是 } h(n) \text{ 不需要加 } g(n).$$

$$f^*(n) = g(n) + h^*(n) \text{ 是节点 } n \text{ 的代价.} \rightarrow f(n) = f^*(n) \text{ 为估计.}$$

$$f(n) = \boxed{g(n) + h(n)}$$

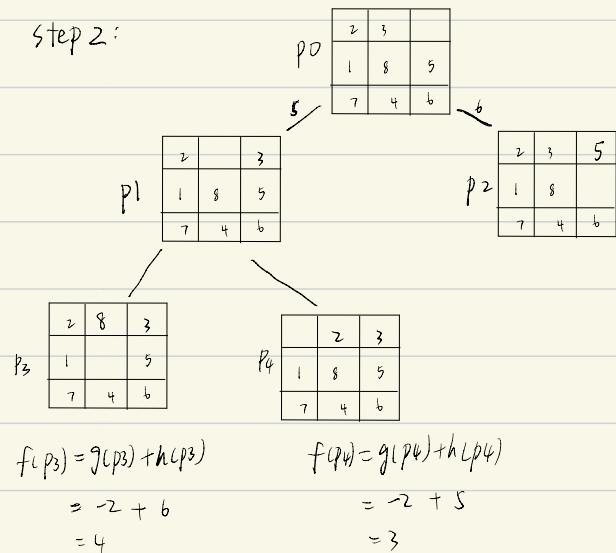
Step 1:



$$\begin{aligned} f(p_1) &= g(p_1) + h(p_1) \\ &= -1 + 6 \\ &= 5 \end{aligned}$$

$$\begin{aligned} f(p_2) &= g(p_2) + h(p_2) \\ &= -1 + 7 \\ &= 6 \end{aligned}$$

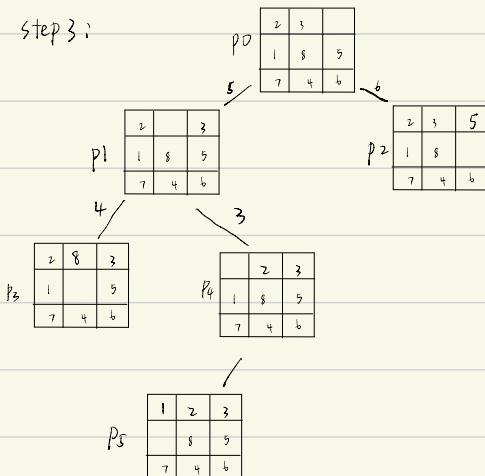
Step 2:



$$\begin{aligned} f(p_3) &= g(p_3) + h(p_3) \\ &= -2 + 6 \\ &= 4 \end{aligned}$$

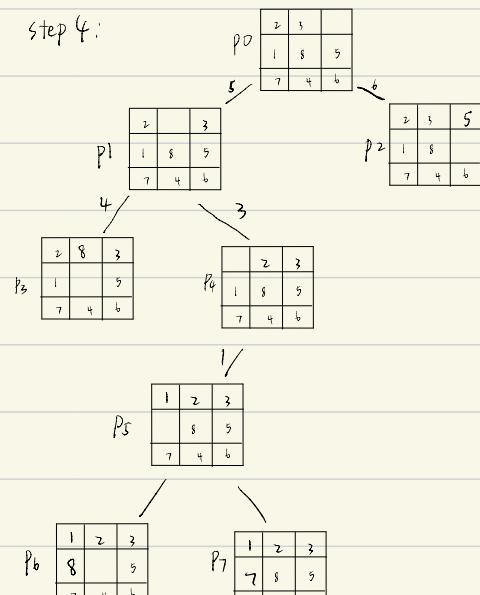
$$\begin{aligned} f(p_4) &= g(p_4) + h(p_4) \\ &= -2 + 5 \\ &= 3 \end{aligned}$$

Step 3:



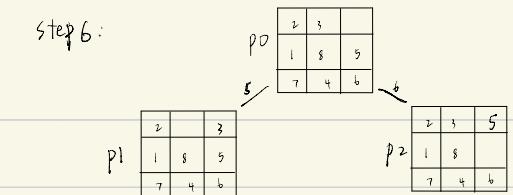
$$\begin{aligned} f(p_5) &= g(p_5) + h(p_5) \\ &= -3 + 4 = 1 \end{aligned}$$

Step 4:

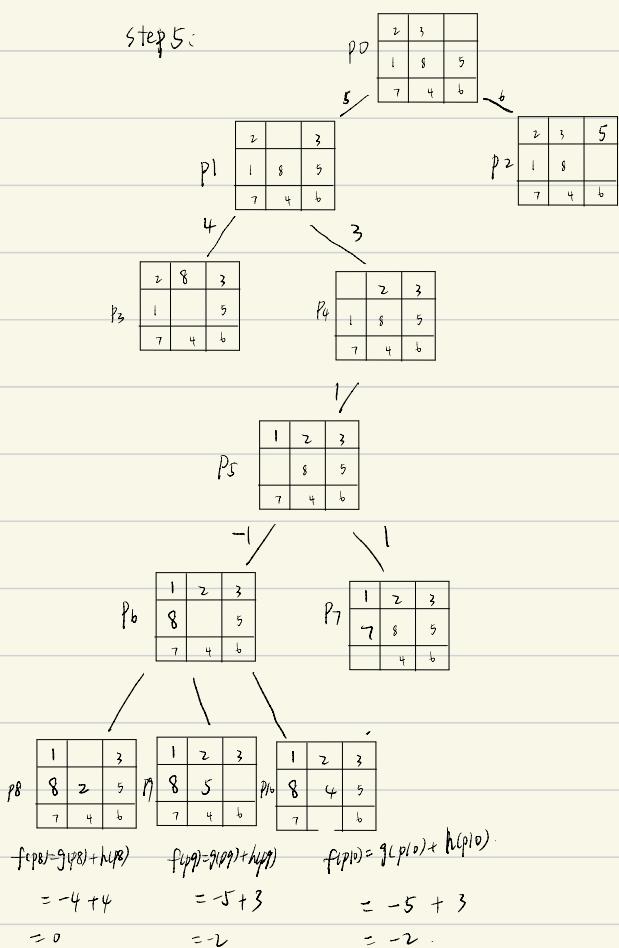


$$\begin{aligned} f(p_6) &= g(p_6) + h(p_6) \\ &= -4 + 3 \\ &= -1 \end{aligned} \quad \begin{aligned} f(p_7) &= g(p_7) + h(p_7) \\ &= -4 + 5 \\ &= 1 \end{aligned}$$

Step 6:



Step 5:

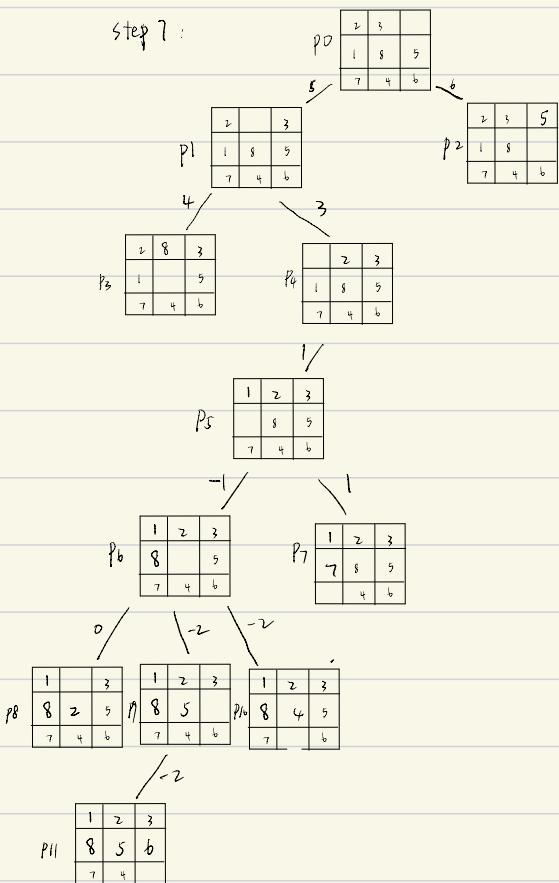


$$f(p11) = g(p11) + h(p11)$$

$$= -5 + 3$$

$$= -2$$

Step 7:

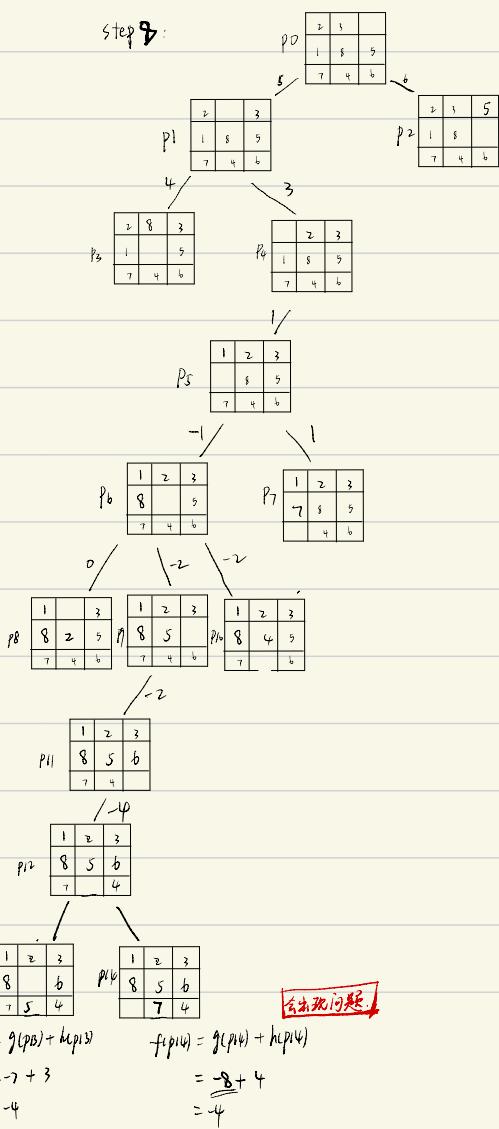


$$f(p12) = g(p12) + h(p12)$$

$$= -7 + 3$$

$$= -4$$

Step 8:



$$f(p13) = g(p13) + h(p13)$$

$$= -7 + 3$$

$$= -4$$

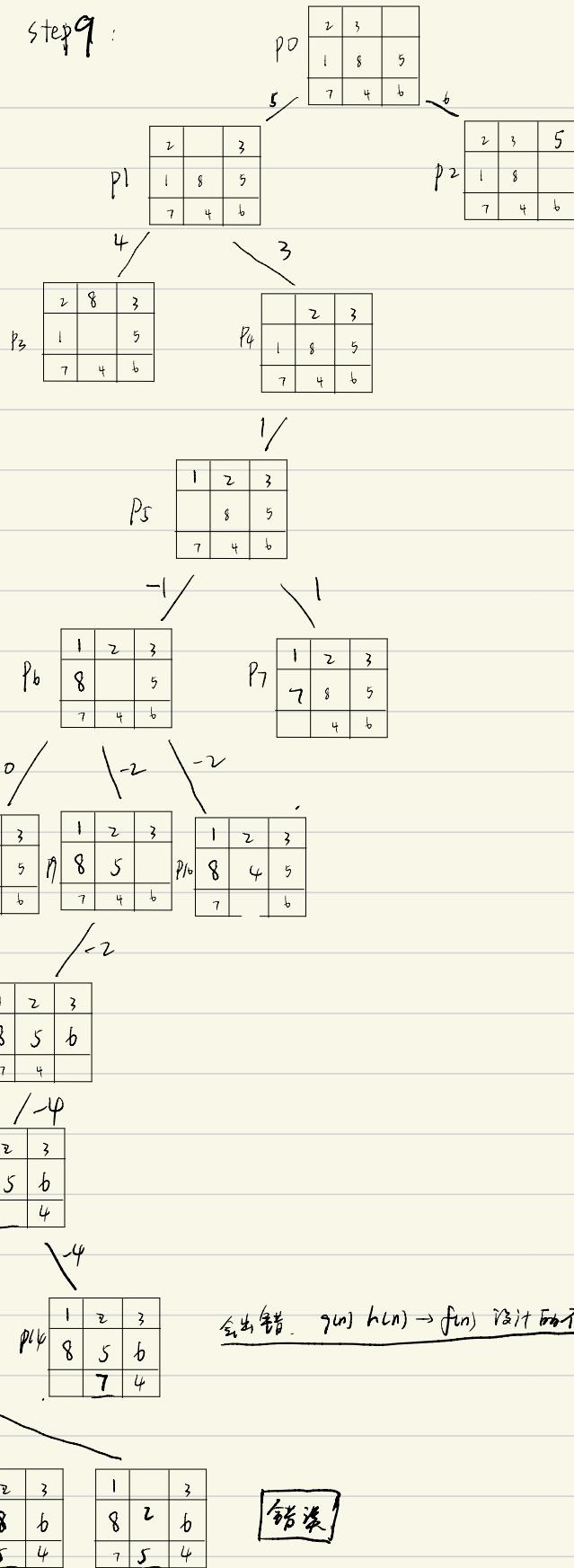
$$f(p14) = g(p14) + h(p14)$$

$$= -8 + 4$$

$$= -4$$

会出现问题

Step 9:



7.5

(1) $g(n)$: 从树根到 n 的代价.

$h(n)$: 由前节点出发到下一节点的代价.

$h^*(n)$: 从 n 到目标节点的代价.

$f(n) = g(n) + h(n)$ 为节点 n 的代价

(2)