人工智能基础 1

傅申 PB20000051

2023 年 3 月 28 日

3.7.

a. **初始状态** 没有地区被染色的地图;

目标测试 地图中所有地区都被染色, 且任何两个相邻的地区都没有染成相同的颜色;

后继函数 为一个未被染色的区域染色, 且该颜色与该相邻区域的颜色不同;

耗散函数 染色次数.

b. **初始状态** 猴子和两个箱子均在屋子的地面上, 香蕉挂在房间的房顶上;

目标测试 猴子拿到了香蕉;

后继函数 移动, 爬上箱子, 爬下箱子, 移动箱子, 将一个箱子堆叠在另一个上, 拿取香蕉;

耗散函数 操作次数.

d. **初始状态** 三个壶均为空;

目标测试 某个壶中刚好有 1 加仑水;

后继函数 装满某个壶, 倒空某个壶, 将某个壶中的水倒入另一个壶;

耗散函数 操作次数.

3.9.

a. 使用一个 3 元组描述状态, 其中三个元素分别代表原河岸的传教士, 野人和船的数目. 显然元组 (x, y, z) 满足 $0 \leqslant x, y \leqslant 3$, $z \in \{0, 1\}$.

初始状态 (3, 3, 1);

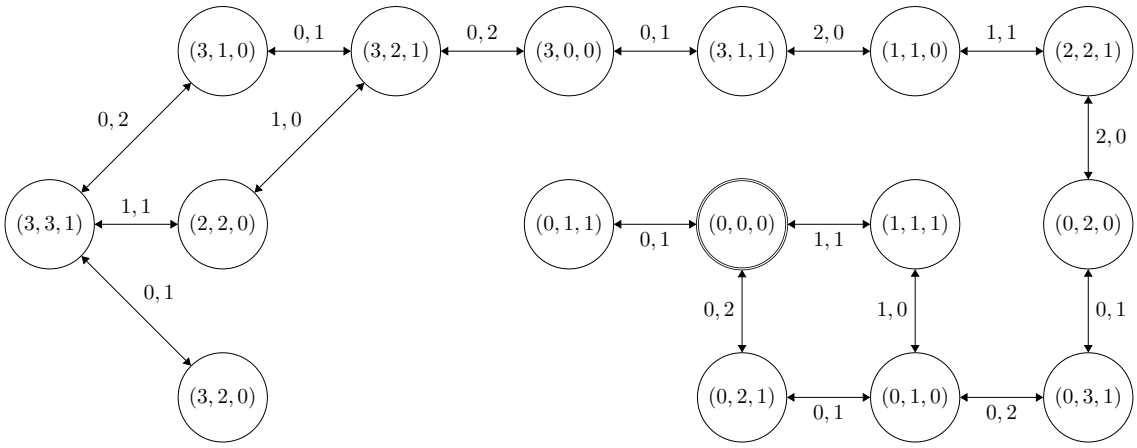
目标函数 (0, 0, 0);

后继函数 $actions : (x, y, z) \mapsto (x', y', z')$, 其中满足:

- $x' = y'$ 或 $x' = 0$ 或 $x' = 3$;
- $z + z' = 1$;
- 如果 $z = 1$ 则 $(x + y - x' - y') \in \{1, 2\}$, 否则 $(x' + y' - x - y) \in \{1, 2\}$;
- $(x - x') \times (y - y') \geqslant 0$;

耗散函数 每次操作代价均为 1.

按照上面的形式化描述, 完全状态空间图如下, 其中, 如果一个状态能转换到另一个状态, 则反之也成立, 故图中用双向箭头 $\leftrightarrow$ 代替两状态间相互转换的两个箭头以简化状态空间图. 箭头上的标识代表转换所对应的船上的传教士和野人人数.



b. 因为完全状态空间很小, 所以任何可获得最优解的搜索策略都是有效的, 比如 BFS, UCS, IDS 和双向搜索. 这里采用 BFS, 并且在扩张时不选择父节点和初始结点, 如下

```
solve():
    frontier = Queue((3, 3, 1))
    while
        if frontier.empty() then return failure
        head = frontier.dequeue()
        for state in succ(head) do
            if goal_test(state) then return the corresponding solution
            if state = (3, 3, 1) or state = head then continue
            frontier.enqueue(state)

goal_test((x, y, z)):
    return x = y = z = 0

succ((x, y, z)):
    actions = [(1, 1), (0, 2), (0, 1), (1, 0), (2, 0)]
    successors = []
    for (x, y) in actions do
        if z == 1 then
            successor = (x + y, x + y, 0)
            if is_valid(successor) then successors.append(successor)
        else
            successor = (x - y, x - y, 1)
            if is_valid(successor) then successors.append(successor)
    return successors

is_valid((x, y, z)):
    return 3 >= x, y >= 0 and (x = y or x = 0 or x = 3)
```

找到的一个可行的解为: $(3,3,1) \to (3,1,0) \to (3,2,1) \to (3,0,0) \to (3,1,1) \to (1,1,0) \to (2,2,1) \to (0,2,0) \to (0,3,1) \to (0,1,0) \to (1,1,1) \to (0,0,0)$.

我认为检查重复状态是没有必要的, 因为虽然每条边都是双向的, 且完全状态空间中出现了环 (只出现在了初始结点和目标结点处), 但是这两种情况导致的重复遍历完全可以通过在扩张时不选择父节

点和初始结点来避免.

c. 可能的原因是人很难判断一个动作是否是有效的, 因为有效的动作需要满足的条件很多.