算法与复杂性　作业八

516021910528 - SHEN Jiamin

2020 年 4 月 9 日

1　0409

1. $G = (V, E)$ 是一个无向图，每个顶点的度数都为偶数。设计线性时间算法，给 G 中每条边一个方向，使每个顶点的入度等于出度。（请先简单说明算法思想，再给出伪代码，然后证明其时间复杂性符合要求）

> 因为无向图每个顶点的度数都为偶数，所以该图是欧拉图，即一定存在欧拉回路能经过每一条边且每条边仅经过一次。沿欧拉回路标记每一条边的方向，即可保证每个顶点的入度等于出度。

2. 连连看游戏中用户可以把两个相同的图用线连到一起，如果连线拐的弯小于等于两个则表示可以消去。设计算法，判断指定的两个图形能否消去。

> 分以下三种情况讨论：两个图案通过一条直线、两段折线、三段折线相连。
>
> 伪代码见算法1、算法2。

3. 证明任意连通无向图中必然存在一个点，删除该点不影响图的连通性。用线性时间找到这个点。

> 以图中任意一节点为根节点做深度优先搜索，一定存在搜索到某个节点时，该节点的所有相邻节点都已经被标记了。即该节点的所有相邻节点都可以从根节点通过不经过该节点的路径到达。所以删除该节点一定不影响图的连通性。
>
> 进行深度优先搜索的时间复杂度为 $O(|V| + |E|)$。因为该算法不需要完成对整个图的遍历，所以该算法的时间复杂度不超过 $O(|V| + |E|)$。
>
> 伪代码见算法3。

算法 1 判别两图案是否可以消除 (3)

输入: 矩阵 $M[0..m+1][0..n+1]$，其中 $M[1..m][1..n]$ 为游戏图案；矩阵上两个元素 $M[r_1][c_1], C[r_2][c_2]$

输出: 两元素是否可以消除

```
1: if OneSegment(M, r1, c1, r2, c2) then
2:     return True
3: else if TwoSegment(M, r1, c1, r2, c2) then
4:     return True
5: else if ThreeSegment(M, r1, c1, r2, c2) then
6:     return True
7: else
8:     return False
9: end if

10: procedure Expands(M[0...m+1][0...n+1], row, col)
11:     找到 d ≤ row ≤ u，使得 M[d...u][col] 范围内除了 M[row][col] 全为空
12:     找到 l ≤ col ≤ r，使得 M[row][l...r] 范围内除了 M[row][col] 全为空
13:     return (u, d, l, r)
14: end procedure

15: procedure ThreeSegment(M[0...m+1][0...n+1], r1, c1, r2, c2)
16:     (u1, d1, l1, r1) ← Expands(M, r1, c1), (u2, d2, l2, r2) ← Expands(M, r2, c2)
17:     d ← max(d1, d2), u ← min(u1, u2), l ← max(l1, l2), r ← min(r1, r2)
18:     for r ← d to u (r ≠ r1 ∧ r ≠ r2) do
19:         if OneSegment(M, r, c1, r, c2) then
20:             return True
21:         end if
22:     end for
23:     for c ← l to r (c ≠ c1 ∧ c ≠ c2) do
24:         if OneSegment(M, r1, c, r2, c) then
25:             return True
26:         end if
27:     end for
28:     return False
29: end procedure
```

算法 2 判别两图案是否可以消除 (1)(2)

procedure OneSegment($M[0\ldots m+1][0\ldots n+1], r_1, c_1, r_2, c_2$)
    if $r_1 = r_2$ **then** ▷ 判断是否能通过一条水平线消除
        for $c \leftarrow \min(c_1, c_2) + 1$ to $\max(c_1, c_2) - 1$ **do**
            if $\neg M[r_1][c].isEmpty$ **then**
                return False
            end if
        end for
        return True
    else if $c_1 = c_2$ **then** ▷ 判断是否能通过一条竖直线消除
        for $r \leftarrow \min(r_1, r_2) + 1$ to $\max(r_1, r_2) - 1$ **do**
            if $\neg M[r][c_1].isEmpty$ **then**
                return False
            end if
        end for
        return True
    else ▷ 两图案不在同一条直线上
        return False
    end if
end procedure

procedure TwoSegment($M[0\ldots m+1][0\ldots n+1], r_1, c_1, r_2, c_2$)
    if OneSegment$(M, r_1, c_1, r_1, c_2) \wedge$ OneSegment(M, r_2, c_2, r_1, c_2) **then**
        return True
    else if OneSegment$(M, r_1, c_1, r_2, c_1) \wedge$ OneSegment(M, r_2, c_2, r_2, c_1) **then**
        return True
    end if
    return False
end procedure

算法 3 寻找割点

输入: 图 $G = (V, E)$

输出: 割点 $v \in V$，使图中删除点 v 时不影响图的连通性

```
v ← SELECT(V)                                  ▷ 任选一个节点作为根节点
v.marked ← True                                ▷ 标记已经搜索过 v_0
repeat
    continue ← False
    for (v, w) ∈ E do
        if ¬w.marked then                      ▷ 如果还存在一个与 v 相邻的节点 w 未被标记
            w.marked ← True                    ▷ 标记该点
            v ← w
            continue ← True                    ▷ 从该点开始继续搜索
            break
        end if
    end for
until ¬continue                                ▷ continue 为 False 时，节点 v 不存在未被标记的邻点
return v
```