高等理工学院《算法设计与分析》
(2021 年秋季学期)

第四次作业参考答案

1 对下面的每个描述，请判断其是正确或错误，或无法判断正误。对于你判为错误的描述，请说明它为什么是错的。(每小题 5 分，共 20 分)

1. P 类问题为 NP 类问题的真子集。
2. 判定无向图中是否存在环这一问题属于 NP 问题。
3. 如果假设 $P \neq NP$，则 NP 完全问题可以在多项式时间内求解。
4. 已知一个问题是 NP 问题，如果该问题可以在多项式时间内求解，则可以证明 $P = NP$。

解：

1. 无法判定。
2. 正确。
3. 错误，如果 NP 完全问题可以在多项式时间内求解，则所有 NP 问题可以在多项式时间内求解，与假设 $P \neq NP$ 矛盾。
4. 错误，该问题可能为 P 问题。

2 班委评选问题 (20 分)

给定一个班级包括 n 位同学，编号分别为 $1, 2, \cdots, n$。班级成员可以推荐一些人担任班委，已知总共存在 m 个推荐关系，第 i 个推荐关系使用 (s_i, t_i) 表示编号为 s_i 的人推荐 t_i 担任班委。推选关系可以传递，例如 A 推荐 B，B 推荐 C，则可以推出 A 推荐 C。如果一个人被所有人推荐，则他当选班委。

请设计一个高效算法计算出所有班委的编号，写出该算法伪代码并分析时间复杂度。

1. 问题分析

首先考虑简单情况，使用节点表示同学，边表示推荐关系，输入数据表示为图 $G(V, E)$，当图 G 为有向无环图（DAG）时，如果仅有一个节点出度为 0，则该节点的同学当选班委，否则无人担任班委。

2. 问题求解

当输入图 G 不是有向无环图（DAG）时，对输入数据计算强连通分量，使用强连通分量构造新的有向无环图 G'，将强连通分量作为 G' 的节点，E 中跨越两个强连通分量之间的边作为 G' 的边。

若图 G' 中仅有一个出度为 0 的点，则该节点代表的强连通分量中的同学担任班委，否则无人担任班委。

伪代码见4。

3. 时间复杂度分析

计算强连通分量的复杂度为 $O(n+m)$，构造新图的复杂度为 $O(n+m)$，故总复杂度为 $O(n+m)$。

Algorithm 1 $scc(G)$

Input: 图 $\boldsymbol{G}$
Output: 强连通分量

$R \leftarrow \{\}$
$G^R \leftarrow G.reverse()$
$L \leftarrow$ DFS(G^R)
$color[1..V] \leftarrow WHITE$
for $i \leftarrow L.length()\ downto\ 1$ **do**
  $u \leftarrow L[i]$
  if $color[u] = WHITE$ **then**
    $L_{scc} \leftarrow$ DFS-Visit(G, u)
    $R \leftarrow R \cup set(L_{scc})$
  end if
end for
return R

Algorithm 2 $dfs(G)$

Input: 图 $\boldsymbol{G}$
Output: 数组 L

新建数组 $\boldsymbol{color[1..V], L[1..V]}$
for $v \in V$ **do**
  $color[v] \leftarrow WHITE$
end for
for $v \in V$ **do**
  if $color[v] = WHITE$ **then**
    $L' \leftarrow$ DFS-Visit(G, v)
    向 $\boldsymbol{L}$ 结尾追加 $\boldsymbol{L'}$
  end if
end for
return L

Algorithm 3 $dfs - visit(G)$

Input: 图 $\boldsymbol{G}$, 顶点 $\boldsymbol{v}$
Output: 按完成时刻从早到晚排列的顶点 $\boldsymbol{L}$

$color[v] \leftarrow GRAY$
初始化空队列 L
for $w \in G.Adj[v]$ **do**
  if $color[w] = WHITE$ **then**
    向 L 追加 DFS-Visit(G, w)
  end if
end for
$color[v] \leftarrow BLACK$
向 $\boldsymbol{L}$ 结尾追加顶点 $\boldsymbol{v}$
return L

Algorithm 4 $class(n, m, (s_i, t_i))$

使用推荐关系 (s_i, t_i) 构造图 $G(V, E)$
$\{s_1, s_2, \cdots, s_k\} \leftarrow scc(G)$
$V' \leftarrow \{s_1, s_2, \cdots, s_k\}$
$E' \leftarrow \{< s_a, s_b > \mid < u, v > \in E, u \in s_a, v \in s_b\}$
$pos \leftarrow -1$
$cnt \leftarrow 0$
for $u : 1 \to k$ **do**
    $out[s_u] \leftarrow |\{< s_u, s_i > \mid < s_u, s_i > \in E'\}|$
    if $out[s_u] = 0$ **then**
        $cnt \leftarrow cnt + 1$
        $pos \leftarrow u$
    end if
end for
if $cnt > 1$ **then**
    return $\emptyset$
end if
return s_{pos}

3 传递闭包问题 (20 分)

给定一个包含 n 个节点的有向图 $G = (V, E)$，其传递闭包定义为一个 $n \times n$ 的布尔矩阵 $T = \{t_{ij}\}$，其中矩阵第 i 行 $(i \le i \le n)$ 第 j 列 $(1 \le j \le n)$ 的元素 t_{ij} 表示图中是否存在从 i 到 j 的路径。如果从第 i 个顶点到第 j 个顶点之间存在一条有向路径，则 t_{ij} 为 1；否则 t_{ij} 为 0。

如图1所示，对于该有向图

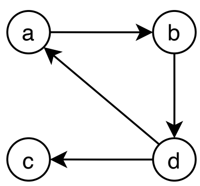

图 1: 有向图

其邻接矩阵为 $A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix}$，求出的传递闭包为 $T = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix}$。

现给定一个有向图的邻接矩阵 A，请设计高效算法求出其传递闭包 T，写出该算法伪代码并分析时间复杂度。

解：

1. 问题思路

求解传递闭包可以用 Floyd-Warshall 算法，即设 $D[i][j][k]$ 表示从 i 到 j 只经过 $1..k$ 中的节点为中间节点的是否可以到达。

则有:

1. 若 $D[i][j][k-1] = 1$ 则 $D[i][j][k] = 1$
2. 若 $D[i][k][k-1] = 1 \wedge D[k][j][k-1] = 1$ 则可以经过 k 为中间节点连通 i, j，即 $D[i][j][k] = 1$

2. 时间复杂度分析

故需要考察 $D[1..n][1..n][1..n]$ 的所有状态，才能求得传递闭包 $T[1..n][1..n] = D[1..n][1..n][n]$。而每个状态需要考察的都是上述两种选择，故总时间复杂度为 $T(n) = O(n^3)$

参考伪代码如5所示。

Algorithm 5 $transitive(n, A[1..n][1..n])$

```
D[1..n][1..n] = A[1..n][1..n]
for k : 1 → n do
    for i : 1 → n do
        for j : 1 → n do
            if D[i][k] = 1 ∧ D[k][j] = 1 then
                D[i][j] ← 1
            end if
        end for
    end for
end for
return D[1..n][1..n]
```

4 食物链问题 (20 分)

给定一个食物网，包含 n 个动物，m 个捕食关系，第 i 个捕食关系使用 (s_i, t_i) 表示，s_t 捕食者，t_i 表示被捕食者，根据生物学定义，食物网中不会存在环。

长度为 k 的食物链指包含 k 个动物的链：$a_1, a_2, \cdots, a_k$，其中 a_i 会捕食 a_{i+1}，一个食物链为**最大食物链**当且仅当 a_1 不会被任何动物捕食，且 a_k 不会捕食任何动物。

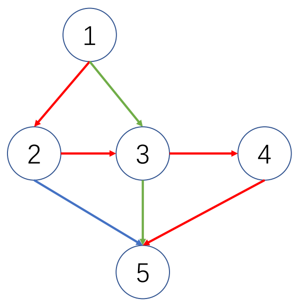

图 2: 食物网示例

如图2所示，该食物网存在 5 个动物，7 个捕食关系，其中红色和绿色均可以称为最大食物链，而蓝色食物链则不能成为最大食物链，图中一共包含 5 个最大食物链，分别是 1-2-5，1-2-3-5，1-2-3-4-5，1-3-5，1-3-4-5。

请设计一个高效算法计算食物网中最大食物链的数量，写出该算法伪代码并分析时间复杂度。

1. 问题分析

使用 n 个动物作为节点 V，m 个捕食关系作为边 E，构造图 $G(V, E)$。

最大食物链可以表示为图上的一条路径，该路径满足起点的入度为 0，终点的出度为 0，因此可以考虑在图 G 上进行动态规划以求解问题。

2. 问题求解

状态定义：使用 $dp[i]$ 表示编号为 i 的节点为路径终点的数量。

转移方程：对于前驱节点 $pre[v] = \{u| < u, v >\in E, u \in V\}$，$dp[v] = \sum_{u\in pre[v]} dp[u]$。

边界条件：对于所有入度为 0 的点 u，$dp[u] = 1$。

遍历顺序：以图 G 拓扑排序顺序进行遍历。

伪代码见算法7。

3. 时间复杂度分析

对于每个节点的转移复杂度为 $|pre[v]|$，总复杂度为 $O(n + m)$。

Algorithm 6 topo_sort

Input: 图 G
Output: 顶点拓扑序

初始化空队列 Q
for $v \in V$ **do**
  if $v.in\_degree = 0$ **then**
    $Q.Enqueue(v)$
  end if
end for
定义数组 ans
while $not\ Q.is\_empty()$ **do**
  $u \leftarrow Q.Dequeue()$
  在数组 ans 后追加 u
  for $v \in G.Adj(u)$ **do**
    $v.in\_degree \leftarrow v.in\_degree - 1$
    if $v.in\_degree = 0$ **then**
      $Q.Enqueue(v)$
    end if
  end for
end while
return ans

Algorithm 7 $foodchain(n, m, (s_i, t_i))$

使用推荐关系 (s_i, t_i) 构造图 $G(V, E)$
$topo \leftarrow topo\_sort(G)$
$ans \leftarrow 0$
for $i : 1 \to n$ **do**
  $pre[a_i] \leftarrow \{u| < u, a_i > \in E, u \in V\}$
  $out[a_i] \leftarrow |\{u| < a_i, u > \in E, u \in V\}|$
  $dp[a_i] \leftarrow \sum_{u \in pre[a_i]} dp[u]$
  if $out[a_i] = 0$ **then**
    $ans \leftarrow ans + dp[a_i]$
  end if
end for
return ans

5 骨牌覆盖问题 (20 分)

给定一个大小为 $n \times m$ 的棋盘，其中某些位置被损坏了，如图3所示。棋盘的信息通过矩阵 R 给出，$R[i][j] = 0$ 表示该位置是完好的，$R[i][j] = 1$ 表示该位置被损坏了。现请你使用大小为 1×2 的骨牌来覆盖整个棋盘（如图4所示，注意：必须**恰好**覆盖该棋盘，换言之，所有损坏的地方以及超出棋盘边界的地方均不能被骨牌覆盖）。请设计一个高效算法判断是否能恰好覆盖整个棋盘，写出该算法伪代码，分析该算法时间复杂度。

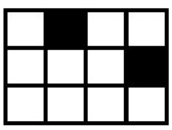

图 3: 一个 3×4 的棋盘，有两个位置被损坏了

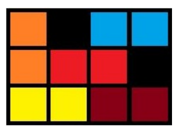

图 4: 该棋盘可以被恰好覆盖

解：

1. 问题求解

将棋盘中每一个没有被损坏的格子视为一个节点，每个节点和它上下左右没有被损坏的格子连一条无向边，这样我们可以得到一个无向图。

可以证明，该无向图为二分图。证明过程如下：

将棋盘第 i 行第 j 列的格子对应的节点编号为 (i,j)，并将节点按照 $i+j$ 的奇偶性进行分类。令

$$A=\{(i,j)|i+j\text{为奇数}\},B=\{(i,j)|i+j\text{为偶数}\}$$

可以发现所有的边连接的两个节点一定是一个属于集合 A，另一个属于集合 B。故该图为二分图。

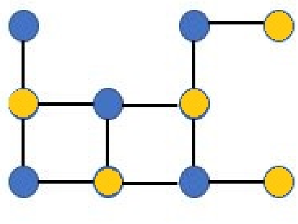

图 5: 将棋盘转化为一个二分图

而每个 1×2 的骨牌会覆盖该二分图中相连的两个节点。这样，判断该棋盘是否可以被恰好覆盖，等价于判断等式

$$\text{图中的最大匹配}=\text{节点数}/2$$

是否成立。

伪代码见算法11

2. 时间复杂度分析

求二分图的最大匹配可以使用匈牙利算法，其时间复杂度为 $O(|V|\times|E|)$，在本题中，图的节点个数与边数均为 $O(nm)$，故时间复杂度为 $O(n^2m^2)$

Algorithm 8 $hungarian(G)$

Input: 二分图 $\boldsymbol{G=<L,R,E>}$

Output: 匹配数组 $\boldsymbol{matched}$

新建一维数组 $\boldsymbol{matched[1..|R|],color[1..|R|]}$

for $u\in R$ **do**

  $matched[u]\leftarrow NULL$

end for

for $v\in L$ **do**

  //初始化 $color$ 数组

  for $u\in R$ **do**

    $color[u]\leftarrow WHITE$

  end for

  DFS-Find(G,v)

end for

return $matched$

Algorithm 9 $DFS - Find(G)$

Input: 二分图 $\boldsymbol{G} =< \boldsymbol{L}, \boldsymbol{R}, \boldsymbol{E} >$, 顶点 $\boldsymbol{v}$

Output: 是否存在从顶点 $\boldsymbol{v}$ 出发的交替路径

//深度优先搜索寻找以顶点 $\boldsymbol{v}$ 出发的交替路径
for $u \in G.Adj[v]$ **do**
    if $color[u] = BLACK$ **then**
        continue
    end if
    $color[u] \leftarrow BLACK$
    if $matched[u] = NULL$ or DFS-Find($G, matched[u]$) **then**
        $matched[u] \leftarrow v$
        return $True$
    end if
end for
return $False$

Algorithm 10 $get\_match(G)$

Input: 匹配数组 $matched$

Output: 最大匹配 M

for $u \in R$ **do**
    if $matched[u]! = NULL$ **then**
        $M \leftarrow M + \{(matched[u], u)\}$
    end if
end for
return M

Algorithm 11 $domino(n, m, R[i][j])$

```
V ← ∅
E ← ∅
for i : 1 → n do
    for j : 1 → m do
        if R[i][j] = 1 then
            continue
        end if
        V.insert(i * m + j)
        if i > 1 and R[i − 1][j] = 0 then
            E.insert((i * m + j, (i − 1) * m + j))
        end if
        if i < n and R[i + 1][j] = 0 then
            E.insert((i * m + j, (i + 1) * m + j))
        end if
        if j > 1 and R[i][j − 1] = 0 then
            E.insert((i * m + j, i * m + j − 1))
        end if
        if j < m and R[i][j + 1] = 0 then
            E.insert((i * m + j, i * m + j + 1))
        end if
    end for
end for
使用节点和 E 构造图 G(V, E)
match ← get_match(hungarian(G))
if |V| = |match| * 2 then
    return True
else
    return False
end if
```