HW 8

肖桐 PB18000037

2020 年 12 月 24 日

解 1. *(a).*

(i). 设 C 是图中的一条欧拉回路, 则每个顶点都将在该回路中出现, 且对于任意顶点 v, v 每在 C 中出现一次都会与一条入边和一条出边相对应. 假设 v 在 C 中共出现了 k 次, 则 $in-degree(v)=out-degree(v)=k$. 证毕.

(ii). 先证该图可表示为多个无公共边的圈的并.

因为该图中所有顶点的 $in-degree=out-degree$, 因此该图不是树 *(*若是树则必存在仅有入度或者仅有出度的顶点*)*, 从而该图中存在一个圈 C_1. 令 $G'=G-C_1$, 则显然 G' 中的顶点也是 $in-degree=out-degree$. 因此存在圈 C_2.

依此类推, 可知 G 可表示为 $C_1\cup C_2\cdots\cup C_n$.

因为 G 是连通图, 因此圈 C_1, C_2 存在公共顶点, 因此 $C_1\cup C_2$ 是一个闭行迹, 同理 $C_1\cup C_2\cup C_3$ 也是闭行迹. 依此类推, $C_1\cup C_2\cup\cdots\cup C_n$ 也是闭行迹, 即是 G 的一个欧拉回路. 证毕.

(b). 应该不存在找出一个图的欧拉回路、且复杂度为 $O(E)$ 的算法. 因为 $O(E)$ 的复杂度意味着查找过程中仅需对所有的边进行一次或常数次遍历即可找到欧拉回路.

但是实际上在找欧拉回路的过程中应当注意到要尽量不去选择图中的“桥”. 在判断一条边是否为桥的过程, 对应的复杂度不会是 $O(1)$. 该算法为 *Fleury* 算法, 复杂度为 $O(VE^2)$

另一个找出欧拉回路的一个算法是“逐步插入回路法”, 复杂度为 $O(V^2E)$. 在图论中有对“逐步插入回路法”的详细描述:

输入： *Euler*图$G=(V(G),E(G))$。
输出： 图G的一条*Euler*回路。

(1) $i\leftarrow 0$，$v^*=v_1$，$v=v_1$，$P_0=v_1$，$G_0=G$；

(2) 在G_i中取与v关联的任意一条边$e=vv'$，将e及v'加入P_i中得到$P_{i+1}=P_iev'$；

(3) 若$v'=v^*$，转*(4)*，否则$i\leftarrow i+1$，$v\leftarrow v'$，转*(2)*；

(4) 若$E(P_{i+1})=E(G)$，停止；否则，令$G_{i+1}=G-E(P_{i+1})$，在G_{i+1}中任取一条与P_{i+1}中某顶点v_k关联的边e，先将P_{i+1}改写成起点*(*终点*)*为v_k的简单回路，再置$v^*=v_k$，$v=v_k$，$i\leftarrow i+1$，转*(2)*。

解 2. *(a).* 将汇率表看作一个图, 若 $R[i,j]\neq 0$, 则该图中有一条 $i\to j$ 的有向边, 且该有向边的权重为 $R[i,j]$. 以 1 为起点, 使用 DFS 对该图进行遍历.

维护一个变量 var, 初始化 $var=1.0$, 在 DFS 执行的过程中, 若 i 已为灰色, 则在将 i 的某一个邻居顶点 j 变成灰色之前, 执行 $var*=R[i,j]$. 若从 i 出发有一条有向边指向起始的顶点 1, 则计算 $var*R[i,1]$, 若结果大于 1, 则找到了一条满足题意的路径.

算法执行过程中仅仅是在 DFS 中加入了一些复杂度为 $O(1)$ 的乘法操作，因此不会影响 DFS 的复杂度，总复杂度仍为 $O(V+E)$.

(b). 在上述的算法中，若找到了一条路径，假设为 $1 \to \cdots \to i \to 1$，则对顶点 i 执行函数：

```
print_path(G, s, v)
{
    if v == s
    {
        print s
    }
    else
    {
        print_path(G, s, v.pi)      //v.pi 为 v 的前驱节点
    }
    print v
}
```

执行：

```
print_path(G, 1, i);
print 1;
```

即可.

解 3. 由 *Johnson* 算法对图中的路径进行重新赋值权重的公式：$\hat{w}(i,j) = w(i,j) + h(i) - h(j)$，可知对原图中的一个权重为 0 的圈，有：

$\hat{w}(i_0,i_0) = w(i_0,i_0) + h(i_0) - h(i_0) = 0$. 假设该圈为 $i \to i_1 \to \cdots \to i_n \to i_{n+1}(i_0)$，则有 $\sum_{k=0}^{n} \hat{w}(i_k,i_{k+1}) = 0$.

因为对于任意 $i,j \in V(G)$，$\hat{w}(i,j) \geq 0$，因此有 $\hat{w}(i_k,i_{k+1}) = 0$，$k = 0,1,\ldots,n$. 证毕.

解 4. *(a).* 若原来有 $f(u,v) < c(u,v)$，则对 $(u,v) \in E$ 的容量增加一个单位之后最大流不会改变.

若原来有 $f(u,v) = c(u,v)$，则对 $(u,v) \in E$ 的容量增加一个单位之后最大流有可能会增加 1. 此时只需要迭代一个 $2F$ 算法，从源顶点通过 BFS 找到当前残存容量图中的一个增广路径，然后对当前流进行更新即可. 一次 BFS 的复杂度为 $O(V+E)$，满足要求.

(b). 若原来有 $f(u,v) < c(u,v)$，则对 $(u,v) \in E$ 的容量减少一个单位之后也不会改变最大流.

若原来有 $f(u,v) = c(u,v)$，则对 $(u,v) \in E$ 的容量减少一个单位之后最大流有可能会减小 1.

此时需要先从源顶点开始执行一次 BFS 找出从源顶点 s 出发，经过边 (u,v) 到达汇顶点 t 的一条路径，将这条路径上所有的边的流量都减一.

此时该图中任意边的流都不会超过其容量，且此时最大流减一.

再从 s 开始执行一次 BFS，以查找该图中是否存在增广路径. 若有则对当前流进行一次迭代，得到最大流.

因为仅将 (u,v) 的流进行减一，则在最大流减一之后至多只需要使用 $2F$ 算法进行一次迭代寻找增广路径. 否则意味着在将 $c(u,v)$ 减一之前图中还有更大的流存在，这与题意是矛盾的.

因此两次 BFS 的时间复杂度为 $O(V+E)$.