算法作业理论题

徐浩宇

2022 年 10 月 10 日

1 理论题第三题

- 获得有向图 G 的全部强连通分量集合 SCCLIST
- 在每个强连通分量 SCC 中搜索环
  - 对 SCC 深度遍历
  - 每访问一条边 (v,w) 就把边压进栈 stack 中，并用字典 dict 记录顶点 w 在栈的第几位，如果不在栈中就记录为-1
  - 对于边 (v,w)，如果 dict[w] 的值不是-1，则 (v,w) 是一条回边，即遇到了环
  - 用 length=dict[v]-dict[w]+1 就得到环的长度 length
  - if length 是奇数
    - 将栈弹出至 w 顶点所在，弹出的边与 (v,w) 即构成了路径长度为奇数的环
  - 反之继续运行
  - if 顶点 w 的所有邻接点都已经访问且程序未退出
    - 栈顶弹出边 (v,w)，并将 v 作为当前顶点继续遍历，dict[w]=-1

Algorithm 1 设计一个线性时间算法，找出有向图中一条路径的长度为奇数的环

Input: 有向图 G(V,E)，顶点 v

Output: 有向图中一条路径的长度为奇数的环中的边

```
SCCLIST = Kasarsju(G)

for SCC in SCCLIST do
    dict; stack; v; //dict: 字典保存顶点 v 在 DFS 树中的层数
    Explore(SCC, v, dict, stack)
    if 已找到 then
        return 环的所有边
    end if
end for
return xxx
```

Algorithm 2 Explore(G,v,dict,stack)

Input: 有向图 G(V,E)，顶点 v , dict, stack

Output: 有向图中一条路径的长度为奇数的环中的边

```
for (v,w) in G.adjedge(v) do
    if dict[w]!=-1 then
        if (dict[v]-dict[w]+1) 是奇数 then
            return (v,w) 与栈顶往下到 w 出现且包括此边的的所有边
        end if
        break
    end if
    push(stack,(v,w)); 置 dict[w]
    Explore(G,w,dict,stack)
end for
(v,w)=pop(s); dict[w]=-1;//退栈
return NULL
```

2 给定有向无环图 G 以及图中两个顶点 s 和 t，计算 G 中从 s 到 t 的所有路径的数量

如算法 2

- 对于图 G，记顶点 s 到 t 的是所有路径为数为 path(s,t)
- V 为边（v，t）的所有 v 的集合
- 根据观察有 $path(s,t) = \sum_{v_i \in V} path(s, v_i)$
- 于是此问题可以划分成若干个子问题

Algorithm 3 getPathNums(G, s, t)//给定有向无环图 G 以及图中两个顶点 s 和 t，计算 G 中从 s 到 t 的所有路径的数量

Input: 有向图 G(V,E)，顶点 s、t

Output: G 中从 s 到 t 的所有路径的数量

```
对于图 G，记顶点 s 到 t 的是所有路径为数为 path(s,t)
V 为边（v，t）的所有顶点 v 的集合
for v in V do
    if s == v then
        return 1
    end if
    path(s,v) = getPathNums(G, s, v)
    path(s,t) += path(s,v)
end for
return path(s, t)
```