计算机学院《算法设计与分析》
(2019 年秋季学期)

第三次作业参考答案

1 合并果子问题 (20 分)

现有 n 堆果子，第 i 堆果子的权值为 $w_i(w_i > 0)$。若把权值为 a 和 b 的两堆果子合并，则可得到权值为 $2\sqrt{ab}$ 的一堆果子。设计一个尽可能高效的算法，使得合并 $n-1$ 次后剩下的一堆果子的权值最小，并分析其时间复杂度。

解:

1. 贪心策略

考虑每次从所有果子中里面挑选权值最大的两堆果子合并，最后剩下的那堆即为权值最小的一堆。

2. 策略证明

不妨考虑任意三堆果子，其权值分别为 a, b, c（可不妨设 $a \geq b \geq c$）则需要判别 $2\sqrt{c \times 2\sqrt{ab}}$, $2\sqrt{b \times 2\sqrt{ac}}, 2\sqrt{a \times 2\sqrt{bc}}$。

因为有 $abc \times c \leq abc \times b \leq abc \times a$ 故可知 $2\sqrt{c \times 2\sqrt{ab}} \leq 2\sqrt{b \times 2\sqrt{ac}} \leq 2\sqrt{a \times 2\sqrt{bc}}$。

故先合并较大的两堆为此时的最优策略。

此外，考虑到对于任意的两堆 a, b，新增的一堆 c 尽可能的小，会使得最终合并的权值也尽可能的小。

故有上述两点归纳可得，原贪心策略正确。

3. 贪心步骤

实现上述策略分为以下几个步骤：

1. 先对原权值序列构建一个大根堆 H（键值为权值）
2. 若大根堆 H 中还有两个以上的元素则进行步骤3，否则进行步骤6
3. 每次从大根堆 H 里选出最大的两个元素 m_1,m_2，并合并得到 $m_3 = 2\sqrt{m_1 m_2}$
4. 将合并得到的 m_3 插入大根堆 H。
5. 跳转步骤2，尝试下一次合并
6. 合并结束，得到合并后的最小权值。

4. 时间复杂度分析

实现上述贪心算法需要做 $O(n)$ 次合并操作，每次合并需要 $O(1)$ 时间取出堆顶元素，和 $O(\log n)$ 时间将一个元素插入进堆。初始的建堆复杂度为 $O(n)$。故总的时间复杂度为 $T(n) = O(n \log n)$。参考伪代码如1所示。

2 分蛋糕问题 (20 分)

给定 n 块体积不同的蛋糕，其体积分别用 $a_1, \cdots, a_n$ 表示。现要从中挑选出 $k(k < n)$ 块蛋糕分给同学们。不妨记选出的蛋糕的编号为 $s_1, \cdots, s_k (1 \leq s_1 < \cdots < s_k \leq n)$，则这次分配的不公平度为

$$\max\{a_{s_1}, \cdots, a_{s_k}\} - \min\{a_{s_1}, \cdots, a_{s_k}\}$$

Algorithm 1 $merge(n, w[1..n])$

Input: n 堆果子，及其每一堆的权值 w_i
Output: 最优合并后的权值

```
以 w_i 为关键字对 w[1..n] 建立大根堆 H
while H 至少还有两个元素 do
    取出 H 中最大的两个元素 m_1, m_2
    m_3 = 2*sqrt(m_1, m_2)
    将 m_3 插入大根堆 H
end while
return 返回 H 中目前的元素
```

请设计一个尽可能高效的算法制定蛋糕的选取方案，使得选出蛋糕的不公平度最小。并分析其时间复杂度。

解:

1. 贪心策略

将蛋糕按照体积从小到大排序，$\min_{i=1}^{n-k+1}\{a_{i+k-1}-a_i\}$ 为最小不公平度。此时记 $p=\arg\min_{i=1}^{n-k+1}\{a_{i+k-1}-a_i\}$。则选取排序后的 $a_p \sim a_{p+k-1}$ 块为使得不公平最小的一个选取方案。

2. 策略证明

考虑以 a_i 为最小元素的集合 S_i，希望最小化 S_i 的不公平度即为最小化 S_i 集合的最大元素。此情况即为选择了大于等于 a_i 的最小的前 k 个元素。

故上述贪心策略实际上考察了每一个元素作为最小元素的所有集合，故策略正确。

3. 时间复杂度分析

注意到仅仅需要做一次排序 $O(n\log n)$ 和将整个数组扫描一趟 $O(n)$。故总的复杂度为 $T(n)=O(n\log n)$。参考伪代码如2所示。

Algorithm 2 $alloc(n, k, a[1..n])$

Input: n 块蛋糕 $a[1..n]$，及所选的块数 k
Output: 不公平度最小的方案

```
将 a[1..n] 从小到大排序
minval ← ∞
for i ← 1 → n − k + 1 do
    if a_{i+k−1} − a_i ≤ minval then
        minval ← a_{i+k−1} − a_i
        p ← i
    end if
end for
return minval, a[p..p + k − 1]
```

3 最小生成树问题 (20 分)

给定带权无向图 $G=(V,E)$，记边 $(u,v)\in E$ 的权重为 $w(u,v)$。现给定一个点集 $U\subset V$. 需寻找一棵生成树满足:

1. 所有 U 集合中的节点都是该树的叶子节点；
2. 该树是满足条件1的生成树中边权和最小的。

请设计一个尽可能高效的算法，求出满足上述条件的合法生成树，或判断不存在这样的生成树。并分析其时间复杂度。

解:

1. 贪心策略

先对点集 $V-U=\{x|x\in V, x\notin U\}$ 做最小生成树 T‘。

1. 若 T' 不为空，且 U 中所有点都存在与 T 相连的边，再将 U 中所有点用与 T' 相连的最小边与之相连即可。
2. 若 T' 不为空，U 中存在一点没有与 T 相连的边，则不存在这样的生成树。
3. 若 T 为空，则将 U 中所有点建立最小生成树 T_U，如果不满足题目条件1，则不存在这样的生成树，否则即为 T_U

2. 策略证明

先在 T' 不为空的情形下证明，注意到此时 U 集合中的每一个点都只能与一条边相连，且 U 集合要与 $V-U$ 集合连通，故 U 集合的点只能与 $V-U$ 集合的点相连。否则若 $p,q \in U$ 相连，又由 p,q 集合要与 $V-U$ 相连，则 p,q 存在一条到 $V-U$ 的边，这与 p,q 都是叶子节点矛盾（必有一个节点度大于 1）。

注意到 T' 已经是点集 $V-U$ 所对应的最小生成树，设用该贪心方法求出的生成树为 T_d，若存在一个更小的符合题意的生成树 T_s，不妨将 T_s 也划分为 $V-U$ 点集中互相连的边 E_1 和，U 集合与 $V-U$ 集合互相连的边 E_2:

1. 若 $\sum_{e\in E_1} w(e) < w(T')$ 这与 T' 是点集 $V-U$ 的最小生成树矛盾。
2. 若 $\sum_{e\in E_2} w(e) < w(T_d - T')$ 这与所选的边是于 T' 相连的最小边矛盾。

故原算法所得到的生成树是此条件下的最小生成树。

在 T' 为空的条件下，注意到此时即 $U=V$，而满足所有点度都为 1 的生成树只有点集小于等于 2 的生成树，故此时只需要检查当前最小生成树是否合法（即节点个数是否小于等于 2）。

3. 时间复杂度分析

若采用 `Prim` 算法做最小生成树，则第一步时间复杂度为 $O(E\log V)$，后续的连边过程是 $O(E)$ 的，总时间复杂度为 $T(V,E)=O(E\log V)$ 的。

若采用 `kruskal` 算法做最小生成树，则第一步复杂度为 $O(E\log E)$，后续的连边过程是 $O(E)$ 的，总时间复杂度为 $T(E)=O(E\log E)$ 的。

参考伪代码如3所示。

Algorithm 3 $restrict-mst(V,E,U)$

Input: 给定的图 $G(V,E)$ 以及点集 U
Output: 满足限制条件的最小生成树 U

```
V' = V - U
E' = {< u, v > | u, v ∈ V', < u, v > ∈ E}
求解图 G'(V', E') 的最小生成树 T'
if T' 为空 then
    求解图 G(V, E) 的最小生成树 T_U
    if U 点集中所有节点都是 T_U 的叶子节点 then
        return T_U
    else
        return 无解
    end if
else
    if U 点集中所有节点和 T' 都有连边 then
        E_U = {< u, v > | u ∈ U, v ∈ T', ∀v' ∈ T', w(u, v) ≤ w(u, v')}
        return T' ∪ E_U
    else
        return 无解
    end if
end if
```

4 老城区改造问题 (20 分)

最近某老城区建设了一个新型能源发电站，现希望该发电站给 n 个小区提供电能（目前发电站与任何小区都不存在传输电路）。已知在这些小区之间已有 m 条传输电路，第 i 条传输电路 (a_i,b_i) 表示第 a_i 个小区和第 b_i 个小区之间的电能可以互相传输。

每个小区可通过下述两种方式之一获取电能：

1. 政府投资 1000 万元，在该小区和新型能源发电站之间建设新的传输电路。
2. 该小区可以通过已有的传输电路从其他已被供电的小区获得电能。

请设计一个尽可能高效的算法，使得政府投资建设传输电路的费用尽可能的少。并分析其时间复杂度。

解:

1. 贪心策略

把 n 个小区看做点，m 条传输线路看做无向边，则对于该图的每一个连通分量，任选一个点（小区）和新型能源发电站建设新的传输电路即可。故总代价为 1000 万 $\times$ 连通分量个数。

2. 策略证明

注意到在连通分量内部建设新的传输电路没有意义，因为连通分量内任意两点已经存在路径。那么问题变为：最少新增多少边，能使得每个连通分量都与新型能源发电站相连。

而实际上，一张 k 个点的连通图至少需要 $k-1$ 条边，故不可能用少于连通分量个数的边使得其均与新型能源发电站相连。

而把每个连通分量都任选其中一个点与新型能源发电相连恰为一个使得此图连通的方案，故贪心策略正确。

3. 贪心步骤

求解连通分量个数的问题用简单的深度优先搜索就能完成:

1. 遍历所有点，找到一个未被搜索过的 i，如果没有这样的点跳转步骤3
2. 从点 i 开始沿着边搜索所有未被搜索过的点，搜到一个点 i 所在的连通分量。
3. 将所有连通分量都与新型能源发电站之间连一条边，总代价为连通分量个数 $\times$1000 万。

4. 时间复杂度分析

注意到每个点和每条边都只会在深度优先搜索的过程中考察一次，故总的时间复杂度为 $T(n,m)=O(n+m)$。参考伪代码如4所示。

Algorithm 4 老城区改造问题

Input: 给定的 n 个小区和 m 条传输电路 (a_i, b_i)
Output: 建设电路的最小花费

```
Procedure dfs(i)
vis[i] ← 1
for j : to[i] do
    if vis[j] == 0 then
        dfs(j)
    end if
end for
EndProcedure
Function rebuild(n, m, a[1..n], b[1..n])
vis[1..n] ← 0
to[x] ← {(x, y)|x = a_i ∪ x = b_i}
cost ← 0
for i ← 1 → n do
    if vis[i] = 0 then
        dfs(i)
        cost ← cost + 10,000,000
    end if
end for
return cost
EndFunction
```

5 货物运输问题 (20 分)

给定一个有向无环图 $G=(V,E)$，图中仅有一个入度为零的点 S（起点），和一个出度为零的点 T（终点）。现需要从起点将一定量的货物运输到终点。

图中的每个点 v_i 有一个属性值 k_i：

1. 若 $k_i=0$，则进入点 v_i 时要支付 1 个货物作为代价。
2. 若 $k_i=1$，则进入点 v_i 时要支付当前货物的 $1/20$（向上取整）作为代价。

（初始时货物已在起点，不需要支付任何代价。）

现希望达到终点后至少有 m 个货物，请设计一个尽可能高效的算法，求出在起点时最少要携带多少货物才能满足需求。并分析其时间复杂度。

解：

1. 基本思路

逆向思考问题，若已知某一个节点的所有后继节点最少需要的货物数量，则可以知道当前节点的所需要的货物数量。故可以建立图 $G'=(V,E')$，$E'=\{<u,v>\mid <v,u>\in E\}$（即将 E 中所有边反向）。根据图 G' 的拓扑序逐一确定每个点需要的最少货物数量。

2. 算法步骤

1. 先对所有节点求出出度值 $out_i=|\{<i,u>\mid <i,u>\in E\}|$，对于终点 T，有最少货物需求 $lim_T=m$，其他任意一点 i 为 $lim_i=\infty$
2. 将当前所有出度为 0 的节点加入集合 S
3. 从集合 S 中任意取出元素 x，如果不存在这样的元素跳转5。此时 x 点所对应的最少货物需求 lim_x 已经确定。对于 x 的每一条入边 $<u,x>$：
   + 更新其最少货物需求 $\lim_u=\begin{cases}\min(lim_u,lim_x+1) & \text{if } k_x=0\\ \min(lim_u,\lceil lim_x*\frac{20}{19}\rceil) & \text{if } k_x=1\end{cases}$
   + 更新当前出度值 $out_u=out_u-1$。（因为 x 对 u 的约束条件已经更新）。
4. 跳转步骤2
5. 说明目前所有节点已经按照拓扑序确定其最少货物需求，输出 lim_S 即可。

3. 时间复杂度分析

在确定最少货物需求的同时，相当于在做一次拓扑排序，故时间复杂度与拓扑排序的复杂度相同为 $T(|V|,|E|)=O(|V|+|E|)$。参考伪代码如5所示。

Algorithm 5 $getlim(G(V,E),S,T,k[1..|V|],m)$

Input: 给定有向无环图 $G(V,E)$，以及起点 S 和终点 T，以及属性值数组 k 和终点至少需要的货物数量 m

Output: 起点 S 最少需要的货物数量

$\forall u \in V, lim_u = \infty, out_u = |\{< u,v > | < u,v > \in E\}|$
$lim_T = m$
$Q \leftarrow \emptyset$
将 T 插入队列 Q
while Q 非空 **do**
  从 Q 中取出元素 u
  for $< v,u > \in E$ **do**
    if $k_u = 0$ **then**
      $lim_v = \min(lim_v, lim_u + 1)$
    else
      $lim_v = \min(lim_v, \lceil lim_u \times \frac{20}{19} \rceil)$
    end if
    $out_v = out_v - 1$
    if $out_v = 0$ **then**
      将 v 插入队列 Q
    end if
  end for
end while
return lim_S