计算机学院《算法设计与分析》
(2022 年秋季学期)

第三次作业参考答案

1 最长空位问题 (20 分)

给定一长度为 n 的 01 串 $S =< s_1, s_2, \cdots, s_n >$，仅有一次机会挑选其中两个元素 $s_i, s_j (1 \leq i, j \leq n)$ 并交换他们的位置。请设计算法求出交换之后 S 中最多有几个连续的 0。

例如，串 $S =$“10010101” 通过交换 s_4 和 s_7 可以变为 “10000111”，连续的 0 的数量为 4。

请先**简要描述策略**，然后**写出伪代码**，最后**分析时间复杂度**。

解：

1. 贪心策略

记串 S 中 0 的个数为 sum，可知连续的 0 最多不会超过 sum 个。

考虑到可以进行一次交换操作，原问题可转化为找出串 S 中最多包含一个 1 的最长子串。该子串的长度即为答案。

若在串 S 的首尾分别添加一个字符 1，我们要找的即为在两个 1 之间且仅包含一个 1 的最长子串。枚举所有 1 的位置作为子串中包含的 1 所在的位置，计算第 $i-1$ 个 1 和第 $i+1$ 个 1 之间的长度，从中找出最大值即可。

需要注意的是边界情况：若串 S 为 ‘‘1001001″，我们发现在两个 1 之间且仅包含一个 1 的最长子串是 ‘‘00100″，但是这里包含的 1 并不能交换到这个子串之外。因此，原问题的答案应为上述最大值与 sum 的较小者。

2. 伪代码

伪代码如 Algorithm 1 所示。

Algorithm 1 $MinChange(S[1..n])$

Input: 长度为 n 的 01 串 S

Output: 交换之后 S 中连续 0 的个数的最大值

```
pos ←< 0 >
for i ← 1 to n do
    if s_i = 1 then
        pos.append(i)
    end if
end for
pos.append(n + 1)
sum ← n − pos.length + 2
ans ← 0
for i ← 1 to pos.length do
    ans ← max{ans, pos_{i+1} − pos_{i−1} − 1}
end for
return min{sum, ans}
```

3. 时间复杂度分析

该算法需枚举 n 个位置，时间复杂度为 $O(n)$

2 最大收益问题 (20 分)

某公司有一台机器，在每天结束时，该机器产出的收益为 X_1 元。在每天开始时，若当前剩余资金大于等于 U 元，则可以支付 U 元来升级该机器（每天最多只能升级一次）。从升级之日起，该机器每天可以多产出 X_2 元的收益。即是说，在执行 K 次升级之后，这台机器每天的产出为 $X_1 + K \times X_2$ 元。

该公司初始资金为 C 元，请你设计算法求出 n 天之后该公司拥有的总资金的最大值。

请先**简要描述策略**，然后**写出伪代码**，最后**分析时间复杂度**。

解：

1. 贪心策略

根据机器产出的公式可以看出，每次升级其实是相互独立的。我们可以将每次升级看作花费 U 元购买了一台新机器，这台新机器每天可以产出 X_2 元。

因此，在第 i 天升级（购买新机器），这台新机器的总产出为 $(n-i+1) \times X_2$ 元。只要该总产出大于升级的成本 U，我们就应该在第 i 天进行升级。当然，还需要根据第 i 天时的剩余资金来判断这一天是否可以进行升级。

2. 伪代码

伪代码如 Algorithm 2 所示。

Algorithm 2 $profit(X_1, X_2, U, C, n)$

Input: 机器初始产出 X_1，升级的收益 X_2，升级所需成本 U，初始资金 C 以及总天数 n
Output: 该公司 n 天后总资金的最大值

```
for i ← 1 to n do
    if (n − i + 1) × X_2 > U and C ≥ U then
        C ← C − U
        X_1 ← X_1 + X_2
    end if
    C ← C + X_1
end for
return C
```

3. 时间复杂度分析

该算法需模拟每天的收益情况，时间复杂度为 $O(n)$。

3 探险家分组问题 (20 分)

营地中共有 n 个探险家，第 i 个探险家的经验值为 $e_i (1 \le i \le n)$。现他们希望组成尽可能多的队伍前去探险。探险家组建队伍需满足如下规则：

1. 探险家可以不参加任何队伍，即留在营地；
2. 如果第 i 个探险家参加了某支队伍，那么该队伍的人数应不小于其经验值 e_i。

请设计一个尽可能高效的算法求出最多可组建几支队伍前去探险，并分析其时间复杂度。

例如有 $n = 5$ 名探险家，其经验值分别为 $e = \{2, 1, 2, 2, 6\}$，则可组建 $(1, 2)$,$(2, 2)$ 两支队伍，把经验值为 6 的探险家留在营地。

请先**简要描述策略**，然后**写出伪代码**，最后**分析时间复杂度**。

解：

1. 贪心策略

每个队伍可以挑选一名经验值最大的探险家担任这个队伍的队长，根据题意，只需要满足每个队伍的人数不小于队长的经验值即可。

从贪心策略上讲，应该使每个队伍的人数刚好与该队伍队长的经验值相同。（若人数更多，则可以把多余的人从本队移除加入其他队，或新生成一个队）

那么假设当前队伍的队长经验值为 e，则其可以让 $e-1$ 个经验值小于等于 e 的探险家不成为其他队伍的队长。我们应该尽可能选择经验值最靠近 e 的 $e-1$ 个探险家，从而避免这些探险家成为其他队伍的队长使得队伍人数过大导致队伍数量变小。

将所有探险家按照经验值 e_i 从小到大排序，遍历所有探险家：

1. 将探险家加入当前分组。

2. 如果当前探险家组内探险家人数与当前组内经验值最大的探险家相同，则将这些探险家分成一组，分组数量增加，当前分组重新置为空；

2. 伪代码

伪代码见 Algorithm3。

Algorithm 3 $group(n, e[1..n])$

Input:

长度为 n 的数列 e 表示探险家的经验值

Output:

最多分组数量

```
group ← 0
cnt ← 0
将 e 从小到大排序
for i : 1 → n do
    cnt ← cnt + 1
    if cnt = a[i] then
        group ← group + 1
        cnt ← 0
    end if
end for
return group
```

3. 时间复杂度分析

遍历整个数组的时间复杂度为 $O(n)$，而对所有元素排序的时间复杂度为 $O(n \log n)$，故总时间复杂度为 $T(n) = O(n \log n)$。

4 分店选址问题 (20 分)

某奶茶品牌想在全国投资开分店来扩大规模提高影响力，通过向新老顾客发放问卷的形式调研产生了 n 个备选地址，并实地考察到了两组数据 $flow$ 和 $cost$，其中 $flow[i]$ 表示第 i 个备选地址的人流量，$cost[i]$ 表示在该地址开店所需的最低资金。由于当前总资金有限，该奶茶品牌希望根据这两组数据，从 n 个备选地址中挑选出 k 个组成最终分店名单，使得能够在满足以下约束条件的前提下尽可能降低总投资成本：

1. 对每个被选中的地址，应当按照其人流量与其他 $k-1$ 个被选中地址人流量的比例来投入资金；

2. 被选中的每个地址的投入资金都不得低于其所需的最低资金。

请设计一个算法求满足上述条件的前提下，该奶茶品牌需要投入的总资金的最小值。
请先**简要描述策略**，然后**写出伪代码**，最后**分析时间复杂度**。

解：

1. 贪心策略

假设我们最终挑选出的分店名单为 $[h_1, h_2, ..., h_k]$，其中 h_i 表示编号为 h_i 的备选地址，这组选址的总人流量为 $totalFlow$，总投资成本为 $totalCost$。那么按照题目的要求，对于每个选址 h_i 需要满足：

$$totalCost \times \frac{flow[h_i]}{totalFlow} \geq cost[h_i]$$

即：

$$totalCost \geq totalFlow \times \frac{cost[h_i]}{flow[h_i]}$$

所以当总人流量固定时，总投资成本只与这 k 个选址的 $max\frac{cost[h_i]}{flow[h_i]}, 1 \leq i \leq k$ 有关。

因此**贪心思路**为：设每个选址 i 的权重为 $weight[i] = \frac{cost[i]}{flow[i]}$，那么当我们以某个选址 x 作为最终选址名单中权重最高的时，选址名单中的其他选址只需要从权重小于等于 $weight[x]$ 的集合中选择人流量最低的 $k-1$ 个地址来组成最终选址名单即可，此时便能达到以地址 x 为权重最高的最终选址名单的总人流量最小，从而达到总投资成本最小。我们可以枚举以每一个能成为最终选址名单中权重最大的地址来计算最少总投资成本，然后取其中的最小值即可。在处理过程中可以先将备选地址按照权重进行升序排列，然后在遍历过程中可以用优先队列维护之前人流量最低的 $k-1$ 名应届生。

2. 伪代码

伪代码见 Algorithm4。

Algorithm 4 $Address(flow[1..n], cost[1..n], k)$

Input:

$flow[1..n]$：各个备选地址的人流量
$cost[1..n]$：各个备选地址的最低投资成本
k：开设的分店数目

Output:

$minCost$：奶茶品牌需要投入的总资金的最小值

对备选地址按照最低投资成本与人流量的比值作为权重来进行升序排列，得到编号数组 $h[1..n]$
$priorityQueue \leftarrow$ 初始化一个基于大顶堆的优先队列
$totalFlow \leftarrow 0$
$minCost \leftarrow MAX\_VALUE$
for $i \leftarrow 1$ *to* $k-1$ **do**
    $totalFlow \leftarrow totalFlow + flow[h[i]]$
    $priorityQueue.push(flow[h[i]])$
end for
for $i \leftarrow k-1$ *to* n **do**
    $totalFlow \leftarrow totalFlow + flow[h[i]]$
    $priorityQueue.push(flow[h[i]])$
    $totalCost \leftarrow totalFlow \times \frac{cost[h[i]]}{flow[h[i]]}$
    $minCost \leftarrow \min\{minCost, totalCost\}$
    $totalFlow \leftarrow totalFlow - priorityQueue.top()$
    $priorityQueue.pop()$
end for
return $minCost$

3. 时间复杂度分析

时间复杂度为 $O(nlogn)$，主要为排序和每个元素进出优先队列的时间复杂度

5 交通建设问题 (20 分)

现有 n 个城市，初始时任意两个城市之间均不可互达。现有两种交通建设方案：

1. 花费 $c_{i,j}$ 的代价，在城市 i 和城市 j 之间建设一条道路，可使这两个城市互相可达。
2. 花费 a_i 的代价，在城市 i 建设一个机场，可以使得其与其他所有建设了机场的城市互相可达。

只要两个城市 i 和 j 之间存在一条可达的路径，则这两个城市也互相可达。请设计一个尽可能高效的算法求出使所有城市之间互相可达所需的最小花费。

请先**简要描述策略**，然后**写出伪代码**，最后**分析时间复杂度**。

解：

1. 问题思路

如果问题不存在建立机场，则为一个简单的最小生成树问题（$G(V,E)$ 其中 V 是 n 个城市,E 是每个城市之间建设道路的费用构成的集合），对于建立机场，相当于可以虚拟一个新点 sky，增加每个点 i 到 sky 存在一条 a_i 权值的边，为一个新图 G_{sky}。

即仅需要对原图 G 求一个最小生成树 T 和对新图 G_{sky} 求一个最小生成树 T_{sky}。比较两个树的大小，选较小的那个为最少花费即可。

2. 伪代码

伪代码如 Algorithm 5 所示。

Algorithm 5 $city(n, a[1..n], c[1..n][1..n])$

Input:

$n \times n$ 的矩阵 c，表示任意两点之间道路的费用，长度为 n 的数列 a，表示建设机场的代价

Output:

使得所有城市可以互相到达的最小花费

$V \leftarrow \{1, \cdots, n\}$
$E \leftarrow \{< u, v, w > | u, v \in V, w = c_{u,v0}\}$
$V' \leftarrow V \cup \{sky\}$
$E' \leftarrow E \cup \{< u, sky, w > | u \in V, w = a_u\}$
$T \leftarrow$ PrimMST(V,E)
$T' \leftarrow$ PrimMST(V',E')
if T 的边权和小于 T' **then**
    return T 的边权和,T 两个点之间的边表示建设道路
else
    return T' 的边权和,T' 其中连到 sky 的边表示建设机场
end if

3. 时间复杂度分析

本题仅需要求解两次最小生成树即可，注意到图是稠密图，采用 Prim 算法求解最小生成树的时间复杂度为 $O(n^2)$。故总时间复杂度为 $T(n) = O(n^2)$ 为所求。