计算机学院《算法设计与分析》
(2021 年秋季学期)

第三次作业参考答案

1 吃糖果问题 (20 分)

给定 n 个糖果盒子，每个盒子糖果数量为 a_i，需要从一些盒子中挑选一些糖果吃掉，使任意两个编号相邻盒子中糖的数量之和小于 x。

请设计一个高效算法计算最少需要吃掉几颗糖，**写出该算法伪代码**并分析时间复杂度。

解：

1. 贪心策略

考虑三个盒子，每个盒子糖果数量 a_1, a_2, a_3 的简单情况，如果糖果数满足 $a_1+a_2+1-x=d$，很明显需要 a_1 和 a_2 减少 d，考虑如果对 a_1 减少 d，仅对第二个盒子有影响，如果对 a_2 减少 d，可以同时影响第一个和第三个盒子。因此应尽量吃靠后的糖果。

首先考虑第一个盒子糖果数量大于等于 x，则必须要吃到 $x-1$ 个糖果，如果小于 x 个糖果，则直接吃第二个盒子，使得第一个和第二个盒子糖果数量小于 x，对第二个盒子同理，依次向右遍历。

2. 时间复杂度分析

只会遍历一次，时间复杂度 $O(n)$。

伪代码见 Algorithm1。

Algorithm 1 $Delete(d)$

Input:
糖果数量 $a[n]$。
Output:
最少需要吃的糖果数量 ans。

$ans \leftarrow 0$
if $a[1] >= x$ **then**
  $delta \leftarrow a[1] + 1 - x$
  $ans \leftarrow ans + delta$
  $a[1] \leftarrow a[1] - delta$
end if
for $i : 2 \to n$ **do**
  if $a[i] + a[i-1] >= x$ **then**
    $delta \leftarrow a[i] + a[i-1] + 1 - x$
    $ans \leftarrow ans + delta$
    $a[i] \leftarrow a[i] - delta$
  end if
end for
return ans

---

2 排队接水问题 (20 分)

给定 n 个人排队接水，已知每个人接水时间为 t_i，仅有一个水龙头，因此他们依次排成一队接水，每个人接水等待时间记为 w_i（从第一个人接水开始，到自己接完为止），总等待时间为 $\sum_i w_i$。

请设计一个高效算法，安排这 n 个人的接水顺序，使得总等待时间最少，**写出该算法伪代码**并分析时间复杂度。

解：

1. 贪心策略

考虑两个人接水时间为 a 和 b 且 $a < b$，存在两种排列情况：

1. a 排在 b 前面总等待时间：$t1 = a + a + b$；
2. b 排在 a 前面总等待时间：$t2 = b + b + a$。

于是由 $a < b$ 得出 $t1 < t2$，得出当 a 在 b 前面时，总等待时间最小。
按照接水时间排序，接水时间少的人优先接水，直接计算答案即可。

2. 时间复杂度分析

排序复杂度 $O(n \log n)$，计算答案复杂度为 $O(n)$，总时间复杂度为 $O(n \log n)$。
伪代码见 Algorithm2。

Algorithm 2 $Water(t)$

Input:
接水时间 $t[n]$。
Output:
最少等待时间 ans。

```
sort(t)
for i = 1 → n do
    ans ← ans + (n − i + 1) ∗ t[i]
end for
return ans, t
```

---

3 数组分段问题 (20 分)

给定 n 个数的数组 $A = (a_1, a_2, \cdots, a_n)$，现需将其分为 k 段，并要求每段中的元素连续，且每段至少包含一个元素。每段取最小值记为 m_i，该分段方法的分值记为 $\max_{1 \le i \le k} m_i$。
请设计一个高效算法，计算分段方法的最大分值，**写出该算法伪代码**并分析时间复杂度。

解：

1. 贪心策略

讨论三种情况：

1. $k = 1$，只有一段，答案为数组最小值；
2. $k = 2$，只有两段，枚举中间切分点即可；
3. $k \ge 3$，有三段，让最大值单独一段，其他分段不影响答案。

$k = 1$ 和 $k = 2$，遍历了所有分组方式，正确性显然。
$k \ge 3$，将最大值单独分为一段，答案即为最大值，显然不可能得到更大的答案。

2. 时间复杂度分析

只需遍历一遍处理 $k = 2$ 的情况，时间复杂度 $O(n)$。
伪代码见 Algorithm3。

Algorithm 3 $Cut(a)$

Input:

接水时间 $a[n]$。

Output:

答案 ans。

```
初始化数组 minL[n] 和 minR[n]
for i : 1 → n do
    minL[i] ← a[i]
    minR[i] ← a[i]
end for
for i : 2 → n do
    minL[i] ← min(minL[i − 1], minL[i])
end for
for i : n − 1 → 1 do
    minR[i] ← min(minR[i + 1], minR[i])
end for
ans ← −INF
if k = 1 then
    ans ← max(ans, minL[n])
end if
if k = 2 then
    for i : 1 → n − 1 do
        ans ← max(ans, max(minL[i], minR[i + 1]))
    end for
end if
if k ≥ 3 then
    for i : 1 → n do
        ans ← max(ans, a[i])
    end for
end if
return ans
```

4 纪念品分组问题 (20 分)

给定 n 件纪念品，每件价值 v_i，需要把纪念品分组，但每组最多只能包括两件纪念品，并且每组纪念品的价格之和不能超过一个给定的整数 m。

请设计一个高效的算法，计算所有分组方案中，最少的分组数量，**写出该算法伪代码**并分析时间复杂度。

解：

1. 贪心策略

首先考虑价值最高的物品，如果无法和价值最低的物品分组，那么价值最高的物品只能单独一组，否则将其配对，重复此操作直至分组完成。

如果存在两组 (a_1, b_1) 和 (a_2, b_2)，满足 $a_1 < b_1$，$a_2 < b_2$，$a_1 < a_2$，$b_1 > b_2$。

如果交换 b_1 和 b_2，显然无法使 $max(a_1 + b_1, a_2 + b_2)$ 更小，因此为最优解。

2. 时间复杂度分析

首先对数组排序，复杂度为 $O(n \log n)$，贪心仅需遍历一遍数组，复杂度为 $O(n)$，因此总复杂度为 $O(n \log n)$。

伪代码见 Algorithm4。

Algorithm 4 $Souvenir(a)$

Input:

纪念品数量 n，价值 $v[n]$，最大分组价值 W。

Output:

```
sort(a)
l ← 1
r ← n
while l ≤ r do
    if a[l] + a[r] ≤ W then
        l ← l + 1
        r ← r − 1
        ans ← ans + 1
    else
        r ← r − 1
        ans ← ans + 1
    end if
end while
return ans
```

5 迷宫逃离问题 (20 分)

给定一个 $m \times n$ 的迷宫，其入口和出口分别为 $(1,1)$ 和 (m,n)。每个格子有两种状态：

1. $c_{i,j} = 0$，表示这个格子是空格子，可以通过；
2. $c_{i,j} = 1$，表示这个格子是障碍物，不可通过。

入口 $(1,1)$ 和出口 (m,n) 均为空格子。在迷宫中可从某个格子 (i,j) 移动到与其相邻的空格子 ($(i,j-1),(i,j+1),(i-1,j),(i+1,j)$ 其中之一)，消耗体力为 1。

现可将至多 1 个障碍物移除，使其对应的格子的变为空格子。请在此基础上设计一个尽可能高效的算法，求出从入口 $(1,1)$ 到出口 (m,n) 需消耗的最小体力，**写出该算法伪代码**并分析时间复杂度。

解：

1. 搜索状态

注意到经过每个格子时有尚未移除障碍物，和已经移除障碍物两种状态，故状态空间是 $n \times m \times 2$ 的。可以尝试使用广度优先搜索。考虑用状态 $(x,y,0)$ 表示在 (x,y) 点还没有移除障碍物，状态 $(x,y,1)$ 表示在 (x,y) 点已经移除了某一个障碍物。

2. 搜索顺序

在状态 $(x,y,0)$ 可以扩展到如下几个状态：

1. $(x,y-1,0),(x-1,y,0),(x,y+1,0),(x+1,y,0)$ 如果对应的新的格子是可以通过的空格子；
2. $(x,y-1,1),(x-1,y,1),(x,y+1,1),(x+1,y,1)$ 如果对应的新的格子是障碍物，则移除之。

而状态 $(x,y,1)$ 只能扩展到四个方向相邻的空格子中去:$(x,y-1,1),(x-1,y,1),(x,y+1,1),(x+1,y,1)$。

3. 时间复杂度分析

注意到每个状态至多只会被搜索一次，每次扩展时间是常数级别的，故总的时间复杂度为 $T(n,m) = O(n \times m)$。

该算法的伪代码如 Algorithm 5 所示。

Algorithm 5 $maze(m, n, c[1..m][1..n])$

Input:

给定的迷宫 $c_{m\times n}$

Output:

最小消耗的体力

$Q \leftarrow \emptyset$
$vis[1..m][1..n][0..1] \leftarrow 0$
将 $(1, 1, 0)$ 状态加入队列 Q
$vis[1][1][0] \leftarrow 1$
$eng[1][1][0] \leftarrow 0$
while Q 非空 **do**
    从 Q 队首取出状态 (x, y, w)
    for $(x', y') \leftarrow \{(x-1, y), (x+1, y), (x, y-1), (x, y+1)\}$ **do**
        if $c_{x',y'} = 0$ **then**
            if $vis[x'][y'][w] = 0$ **then**
                $vis[x'][y'][w] \leftarrow 1$
                $eng[x'][y'][w] \leftarrow eng[x][y][w] + 1$
                将状态 (x', y', w) 加入队列 Q
            end if
        else
            if $w = 0 \cap vis[x'][y'][1] = 0$ **then**
                $vis[x'][y'][1] \leftarrow 1$
                $eng[x'][y'][1] \leftarrow eng[x][y][w] + 1$
                将状态 $(x', y', 1)$ 加入队列 Q
            end if
        end if
    end for
end while
$ans \leftarrow +\infty$
if $vis[m][n][0] = 1$ **then**
    $ans = min(ans, vis[m][n][0])$
end if
if $vis[m][n][1] = 1$ **then**
    $ans = min(ans, vis[m][n][1])$
end if
return ans