

# 算法设计与分析第三章作业

1652270 冯舜

## 1. 算法分析题 4-1

1652270 冯舜

4-1. 答: (1) 证明按  $l_1 \leq l_2 \leq \dots \leq l_n$  次序考虑时, 无法产生最优解:

这里举出反例. 当  $l_{n-1}$  与  $l_n$  安排在不同的磁带上, 且  $l_n$  与  $l_1$  安排在同一磁带上, 且  $l_n - l_{n-1} > l_1$  时, 明显将  $l_1$  放在另一磁带时可使  $\max\{\sum_{i \in A} l_i, \sum_{i \in B} l_i\}$  更小, 故贪心算法无法给出最优解. 如:

|       |   |   |   |    |    |    |    |        |
|-------|---|---|---|----|----|----|----|--------|
| $T_1$ | 1 | 2 | 5 | 14 | 15 | 32 | 57 | (贪心算法) |
| $T_2$ | 1 | 3 | 9 | 14 | 17 | 33 |    |        |

|       |   |   |    |    |    |    |       |
|-------|---|---|----|----|----|----|-------|
| $T_1$ | 2 | 5 | 14 | 15 | 32 | 57 | (更优解) |
| $T_2$ | 1 | 1 | 3  | 9  | 14 | 17 | 33    |

(2) 证明按  $l_1 \geq l_2 \geq l_3 \geq \dots \geq l_n$  考虑时, 无法产生最优解:

这里给出反例. 当  $l_n$  序列为

120, 50, 50, 30, 29, 28 时,

|       |     |               |               |               |             |             |
|-------|-----|---------------|---------------|---------------|-------------|-------------|
| $T_1$ | 120 | <del>29</del> | <del>30</del> | <del>28</del> | (150) (149) | (贪心算法, 158) |
| $T_2$ | 50  | 50            | <del>29</del> | <del>28</del> | (158)       |             |

|       |     |    |    |    |       |            |
|-------|-----|----|----|----|-------|------------|
| $T_1$ | 120 | 30 |    |    | (150) | (更优解, 157) |
| $T_2$ | 50  | 50 | 29 | 28 | (157) |            |

(3) ① 回溯算法. 对于每个程序  $l_i$ , 可放入  $T_1$ , 或可放入  $T_2$ .

故所有可能的放法可构成  $(n+1)$  层二叉树. 对该二叉树进行深度优先搜索, 遇到叶节点时则有  $\max\{\sum_{i \in A} l_i, \sum_{i \in B} l_i\}$  的值, 是则更新最优解. 搜索完毕后即得最优解, 因为对所有情况已进行了搜索.

② 动态规划算法. 不妨令  $T_1$  程序总长更短, 则对于  $i \in A \cup B$ , 令  $x_i = \begin{cases} 1, & i \in A \\ 0, & i \in B \end{cases}$ . 则  $\sum_{i \in A} l_i = \sum_{i \in A \cup B} x_i l_i$ ,  $\sum_{i \in B} l_i = \sum_{i \in A \cup B} (1-x_i) l_i$ . 有:  $\sum_{i \in A \cup B} x_i l_i \leq \sum_{i \in A \cup B} (1-x_i) l_i$  即  $\sum_{i \in A \cup B} x_i l_i \leq \sum_{i \in A \cup B} l_i - \sum_{i \in A \cup B} x_i l_i$ , 同时使  $\sum_{i \in A \cup B} (1-x_i) l_i$  最小. 此即 0-1 背包问题, 用已有算法解决.

## 2. 算法实现题 4-1

### 2.1. 解法一

基本算法思想和课文中的活动安排问题一致，即对活动的完成时间进行从小到大排序，之后将会场数量设置为 0，开始遍历这些活动。若某一活动的开始时间在某会场的最晚结束时间之后，则将其安排入该会场。若没有会场满足要求，则启用一个新的会场，会场数量加一。

会场应以最晚结束时间为主键维持有序。这样，扫描符合要求的会场时只需用二分查找算法。在更新会场的最晚结束时间时，应重新用二分查找法确定该会场的位置。

具体过程可看 4-1.py。如果知道活动数为 $n$ ，时间复杂度为 $O(n\log n)$ （排序用 $O(n\log n)$ ，扫描 $n$ 个活动，每个活动扫描、插入会场用 $O(\log n)$ ，共 $O(n\log n)$ ）。

```
C:\Windows\system32\cmd.exe
原开始时间: [1, 12, 25, 27, 36]
原结束时间: [23, 28, 35, 80, 50]
排序后开始时间: [1, 12, 25, 36, 27]
排序后结束时间: [23, 28, 35, 50, 80]
考虑活动 0 其开始时间 1 结束时间 23
使用了新会场
更新后的会场 [(23, [0])]
考虑活动 1 其开始时间 12 结束时间 28
使用了新会场
更新后的会场 [(23, [0]), (28, [1])]
考虑活动 2 其开始时间 25 结束时间 35
使用了会场 0
更新后的会场 [(28, [1]), (35, [0, 2])]
考虑活动 3 其开始时间 36 结束时间 50
使用了会场 1
更新后的会场 [(28, [1]), (50, [0, 2, 3])]
考虑活动 4 其开始时间 27 结束时间 80
使用了新会场
更新后的会场 [(28, [1]), (50, [0, 2, 3]), (80, [4])]
会场 0 : 结束时间 28, 安排活动 [1]
会场 1 : 结束时间 50, 安排活动 [0, 2, 3]
会场 2 : 结束时间 80, 安排活动 [4]
结果: 3

运行完成。
请按任意键继续. . .
```

### 2.2. 解法二

基本算法思想仍和课文中的活动安排问题一致。每次都贪婪选择活动算法选择出可以加入的活动，此后把加入这个会场的活动剔除。之后如此循环，直到活动全部被剔除，之前用过的会场数就是可以安排所有活动的会场数。时间复杂度 $O(n^2)$ 。

### 2.3. 解法三

将活动的开始时间、结束时间看作 $2n$ 个端点，对这些断点从小到大排序。初始化一个大小为 $n$ 的会场栈，并维护一个变量记录会场栈中用过的会场数。从小到大扫描这些端点。遇到开始时间端点，新会场出栈，并用该新会场举办该活动；遇到结束时间端点，该活动的会场入栈。扫描完毕后，用过的会场数即最小所需会场数，同时每个活动都有了对应的会场。时间复杂度 $O(n\log n)$ （排序）。

### 3. 算法实现题 4-6

每次挑选服务时间最小的顾客服务即可，详细过程见 4-6.py，时间复杂度 $O(n\log n)$ 。

```
C:\Windows\system32\cmd.exe
0 : 第 2 位顾客
1 : 第 1 位顾客
2 : 第 6 位顾客
3 : 第 7 位顾客
4 : 第 0 位顾客
5 : 第 8 位顾客
6 : 第 3 位顾客
7 : 第 5 位顾客
8 : 第 9 位顾客
9 : 第 4 位顾客
平均等待时间: 532.0
请按任意键继续. . .
```

### 4. 算法实现题 4-9

每次使汽车驾驶到最远能到的加油站即可。详细过程见 4-9.py，时间复杂度 $O(n)$ 。

```
C:\Windows\system32\cmd.exe
成功从节点 0 到达节点 1 剩余油量 6 。
成功从节点 1 到达节点 2 剩余油量 4 。
成功从节点 2 到达节点 3 剩余油量 1 。
从节点 3 到节点 4 需要汽油 4，当前有 1，需要第 1 次加油
成功从节点 3 到达节点 4 剩余油量 3 。
从节点 4 到节点 5 需要汽油 5，当前有 3，需要第 2 次加油
成功从节点 4 到达节点 5 剩余油量 2 。
成功从节点 5 到达节点 6 剩余油量 1 。
从节点 6 到节点 7 需要汽油 6，当前有 1，需要第 3 次加油
成功从节点 6 到达节点 7 剩余油量 1 。
从节点 7 到节点 8 需要汽油 6，当前有 1，需要第 4 次加油
成功从节点 7 到达节点 8 剩余油量 1 。
加油次数: 4
请按任意键继续. . .
```