算法设计与分析

作业（六）

姓　　名	熊恪峥
学　　号	22920202204622
日　　期	2022年3月31日
学　　院	信息学院
课程名称	算法设计与分析

作业（六）

目录

1 题7.3

首先，要使用尽可能少的教室来调度所有的活动，等价于求最大兼容子集的个数。所以可以反复调用$GreedyTaskSelec$
然后将选出的兼容子集从原集合中减去。直到所有的活动都被减去。如算法1

算法 1 安排教室

Input: 活动集合A

```
procedure AssignClassRoom(A)
    n ← 0
    while A is not empty do
        S ← GreedyTaskSelect(A)
        A ← A − S
        n ← n + 1
    return n
```

由于$GreedyTaskSelect$有时间复杂度$\mathcal{O}(n)$，设想一个最坏情况，即每一个活动有相同的开始和结束时间，那么$AssignClassRoom$的时间复杂度就是$\mathcal{O}(n^2)$。因为它调用了$GreedyTaskSelect$共n次。

证：$AssignClassRoom$有贪心选择性质和最优子结构性质. 如果一个教室分配不是最优的，那么存在至少两个教室C_i和C_j，使得C_i和C_j的活动集合A_i和A_j不是最大兼容子集。设C_i较最大兼容子集少，C_j包含缺少的元素，那么将C_j的元素加入C_i，C_i就包含最大兼容子集，而教师总数减少1

因此任何一个合法的解都能改造成贪心解，使得教室的分配是最优的。该问题有贪心选择性质。

如果在所有活动中移除一个活动$\{S - a_i\}$，假设这个活动被分配到了C_i，那么移除之后C_i依然包含一个最大兼容子集可见教室分配的方案依然是每个教室分配一个$\{S - a_i\}$的最大兼容子集的最优解。

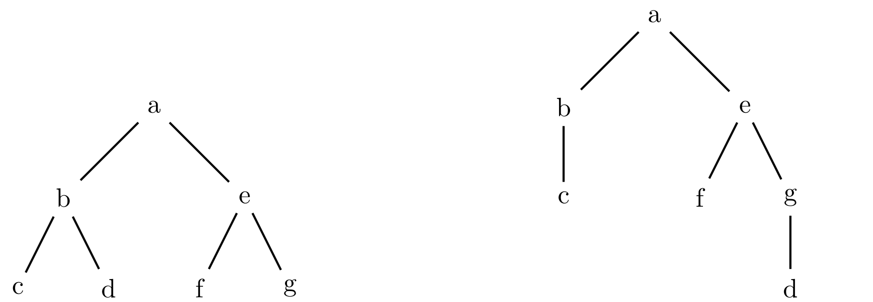
因此任何一个解都是由一次贪心选择和一个子问题构成的。该问题有最优子结构性质。

□

2 题7.6

设一个编码二叉树不是满二叉树，那么不妨设在满二叉树中的某一子树中，应该出现在i层的节点n出现在了第$i+1$层，如图 1。

图 1: 前缀编码二叉树的某个子树



(a) 满编码二叉树的一个子结构　　(b) 对应的非满二叉树的一个子结构

如果图 1(b)对应了一种最优编码,那么n节点的编码长度l是最小的。这时，将节点n向上移动到第i层，节点n的编码长度将变为$l-1$，这与图 1(b)对应最优编码的假设矛盾。注意到将节点n向上移动到第i层，二叉树会变为满二叉树，如图 1(a)，因此没有一棵非满二叉树对应一个最优前缀编码。

3 题7.9

硬币的面值是$1, 5, 10$，满足

$$2 \times 5 < 1 + 10$$

所以贪心算法可以给出最优解。先对硬币面值进行排序，从最大面值起依次进行兑换，如算法 2。

算法 2 硬币兑换

Input: 面值X，硬币$coins$

```
procedure ChangeCoins(X, coins)
    Sort(coins)
    results ← ∅
    for coin ∈ coins do
        while X > 0 do
            X ← X − coin
            results ← results ∪ coin
    return results
```

4 题7.10

为了使得等待时间最少，可以使用贪心算法。从需要服务时间最少的客户开始服务，可以使得等待时间最少

- 贪心选择性质

Proof. 假设有服务序列S的一个最优解为A，g_k是A中服务时间最短的客户，g_m是S中服务客户时间最短的客户。如果$g_k = g_m$，则$g_m \in S$，否则，如果$g_k \neq g_m$，则构造A'，将a_k与A中满足$a_i = a_m$的元素交换由于a_m服务时间最短，所以每一个活动的等待时间不会更长。因此A'是一个最优解。 □

- 最优子结构

Proof. 如过$A = \{g_1, g_2, \ldots, g_k, g_{k+1}, \ldots\}$是一个最优解且有$g_k > g_{k+1}$，那么交换$g_k$与$g_{k+1}$可以使$g_{k+1}$之后的元素等待时间更少这与$A$是最优解矛盾。 □

因此可以得到算法 3

算法 3 客户服务

Input: 客户等待时间集合G

```
procedure Serve(G)
    Sort(G)
    w ← G
    for i = 1 → |G| do
        w[i] = w[i] + w[i − 1]
    total ← 0
    for i = 1 → |G| do
        total ← w[i] − w[0]
    return total/|Q|
```

该算法可以得出最小的平均等待时间。

5 题7.12

对于正整数，$y = a^x$和$y = x^a$都有单调不减性。因此将A, B倒序排列可以获得最大的回报，如算法 4。

算法 4 获取最大回报

Input: 集合A, B

procedure MaxReturn(A, B)
    Sort(A, B)
    $ret \leftarrow 0$
    for $i = 0 \rightarrow |A|$ **do**
        $ret \leftarrow ret + A[i]^{B[i]}$
    return ret

该贪心算法可以得到最大回报。

Proof. 如果一个得到最大回报的排序，对于B中的元素存在$b_i < b_j$，不妨设A中的元素单调不增，那么在

$$\prod\{a_1^{b_1}, a_2b^2, \dots, a_i^{b^i}, a_j^{b_j}, \dots, a_n^{b_n}\}$$

中交换b^i和b^j，那么对其他元素幂的乘积大小无影响，而$a_i > a_j, b_j > b_i$，则$a_i^{b^j} \cdot a_j^{b_i} > a_i^{b^i} \cdot a_j^{b_j}$，由$y = a^x$的单调不减性，这样整体乘积不会减少，并且将上式改造为了一贪心解，因此贪心算法是正确的。 □

6 题7.14

*GreedyCaching*算法具有贪心选择性质。

Proof. 设一个置换出元素在置换出缓存之后再一次访问的时间点的最优序列为

$$E_1, E_2, \dots, E_n$$

设贪心解为

$$F_1, F_2, \dots, F_n$$

因为贪心策略选择之后尽可能迟访问的页面，那么$F_1 \geq E_1$

如果$F_1 = E_1$，那么不需要改造

如果$F_1 > E_1$，那么将满足$E_1 < E_k \leq E_k$的元素E_k与E_1交换，那么由于$E_k > E_1$，在整个过程中缓存的置换次数不会增加。进行类似的改造，完全将最优解改造成贪心解，而置换次数不会增加，因此贪心解也是一个最优解。它具有贪心选择性质。 □