算法设计与分析第五章作业
苏亦凡　　计科12班　　200111229

1

（30 分）假定我们不再一直选择最早结束的活动，而是选择最晚开始的活动，前提仍然是与之前选出的所有活动兼容。描述如何利用这一方法设计贪心算法，并证明算法会产生最优解。

设 $S=\{a_1,a_2,\ldots,a_n\}$ 是 n 个活动(a_n)的集合，各个活动使用同一个资源，资源在同一时间只能为一个活动使用每个活动 i 有起始时间 s_i ，终止时间 f_i ， $s_i \leq f_i$ 。

若 $s_j \leq f_i$ 或 $s_i \leq f_j$,活动 i 和 j 是相容的。

定义: S_{ij} 为在第i个活动结束后开始，在第j个活动开始前结束的事件集合。

设计贪心算法如下：

每次从 S_{ij} 中选取 s 最大的活动 a_k ，然后对 S_{ik} 继续进行此操作。

若 s 已排序，则描述算法为

```
n <- length(S)
A <- {1}
j <- 1
for i <- 2 to n do
    if f_i <= S_j
        then A <- A + {i}
             j <- i
return A
```

正确性证明：

首先证明以下几个引理：

引理1：

设 $S=\{a_1,a_2,\ldots,a_n\}$ 是 n 个活动的集合，$[s_i,f_i]$ 是活动 a_n 的起始终止时间，且 $s_1 \geq s_2 \geq \cdots \geq s_n$，$S$ 的活动选择问题的某个优化解包括a_1。

证明：

设 A 是一个优化解，按起始时间排序 A 中活动，设其第一个活动为 k，第二个活动为 j。

- 如果 $k=1$，引理成立。
- 如果 $k\neq 1$，令 $B=A-a_k+a_1$，由于 A 中活动相容，$s_1\geq s_k\geq f_j$，B 中活动相容。因为 $|B|=|A|$，所以 B 是一个优化解，且包括a_1。

> 引理1证明了贪心策略的正确性，即问题的贪心选择性。

引理2：在引理1的基础上

设 $S=\{a_1,a_2,\dots,a_n\}$ 是 n 个活动的集合，$[s_i,f_i]$ 是活动的起始终止时间，且 $s_1\geq s_2\geq\cdots\geq s_n$，设 A 是 S 的调度问题的一个优化解且包括a_1，则 $A'=A-a_1$ 是 $S'=\{i\in S|f_i\leq s_1\}$ 的调度问题的优化解。

证明：

显然，$A'=A-a_1$ 中的活动是相容的。 仅需要证明 A' 是 S' 最大的。

设不然，于是有：

- 存在一个 S' 的活动选择问题的优化解 B'，$|B'|>|A'|$。
- 令 $B=a_1+B'$，若 $\forall i\in S'$，$f_i<s_1$，B 中活动相容。B 是 S 的一个解。由于 $|A|=|A'|+1$，$|B|=|B'|+1>|A'|+1=|A|$。
- 与 A 最大矛盾。

于是原命题得证。

> 引理2证明了问题具有优化子结构。

下证算法正确性：

算法正确性的具体描述可为：

设 $S=\{a_1,a_2,\dots,a_n\}$ 是 n 个活动的集合，$s_0=\infty$，a_{l_i} 是 $S_i=\{j\in S|f_j\leq s_{i-1}\}$ 中具有最大起始时间 s_{l_i} 的活动，设 A 是 S 的包含 a_1 的优化解，则

$$A=\bigcup_{i=1}^{k}a_{l_i}$$

证明：

对 $|A|$ 作归纳法。

- 当 $|A|=1$ 时，由引理 1，命题成立。
- 设 $|A|<k$ 时，命题成立。当 $|A|=k$ 时，由引理 2，$A=1+A_1$，A_1 是 $S_2=\{j\in S|f_j\leq s_1\}$ 的优化解。

- 由归纳假设，$A_1 = \bigcup_{i=2}^{k} a_{l_i}$。有 $l_0 = 1$，于是，$A_1 = \bigcup_{i=1}^{k} a_{l_i}$。

算法正确性得证。

2

（30 分）考虑用最少的硬币找 n 美分零钱的问题。假定每种硬币的面额都是整数。

a. 设计贪心算法求解找零问题。假定有 25 美分、10 美分、5 美分和1 美分 4 种面额的硬币。

b. 设计一组硬币面额，使得贪心算法不能保证的到最优解。这组硬币面额中应该包含 1 美分，使得对每个零钱值都存在找零方案。

a

由于有面额为1美分的硬币，故对任意零钱值都存在解。于是设计贪心算法如下：每次选取不大于剩余找零值的面额的硬币，从剩余找零值中减去该面额，然后重复以上操作直到剩余找零值为 0 。

b

硬币面额为 $\{100, 52, 1\}$ 。

例如找零 104 美分。

- 贪心求解：{100,1,1,1,1}
- 最优解：{52,52}

3

（40 分）编程题：柠檬水找零

题目描述：

在柠檬水摊上，每一杯柠檬水的售价为 `5` 美元。顾客排队购买你的产品，（按账单 `bills` 支付的顺序）一次购买一杯。

每位顾客只买一杯柠檬水，然后向你付 `5` 美元、 `10` 美元或 `20` 美元。你必须给每个顾客正确找零，也就是说净交易是每位顾客向你支付 `5` 美元。

注意，一开始你手头没有任何零钱。

给你一个整数数组 `bills` ，其中 `bills[i]` 是第 `i` 位顾客付的账。如果你能给每位顾客正确找零，返回 `true` ，否则返回 `false` 。

要求：运用贪心思想作答，请写出分析过程，并用一种语言（最好是 C++ 或 JAVA ）实现你的思路，复杂度尽可能低。

> leetcode 原题 柃檬水找零

算法设计思路：

在本题中选取的贪心策略为优先使用 `10` 美元的零钱，因为 `5` 美元的泛用性比 `10` 美元的泛用性高，而 `20` 美元的零钱完全没用。故建立两个变量 `nfive` 与 `nten` 记录这两种零钱的张数。

```
bool lemonadeChange(int *bills, int billsSize)
{
    int nfive = 0;
    int nten = 0;
    for (int i = 0; i < billsSize; ++i)
    {
        switch (bills[i])
        {
        case 5:
            ++nfive;
            break;
        case 10:
            --nfive;
            ++nten;
            break;
        case 20:
            --nten;
            --nfive;
            if (nten < 0)
            {
                ++nten;
                nfive -= 2;
            }
        }
        if (nfive < 0) return false;
    }
    return true;
}
```

leetcode提交记录截图

59 / 59 个通过测试用例
执行用时: **104 ms**
内存消耗: **12 MB**

状态: **通过**
提交时间: **1 分钟前**