# Homework8

软件13 杨楠

## 1

证明：由于所有256个字符中，出现频率最高的低于最低的2倍，所以第一次选择出现频率最低的2个对象合并之后，新的对象的频率之和将会比其他所有未合并的字符都高。以此类推，第一轮合并后，会生成128个新对象，即128棵子树。这是两两合并的过程。

这新的128个对象同样满足最高频率低于最低频率的2倍。继续以此类推，每一轮合并都是当前所有子树的根节点两两合并的过程，最终会生成一棵完整的满二叉树，所有256个叶子节点的深度都相等，都是8位编码，这并不比8位固定长度编码更高效。

## 2

### a.

设计贪心算法如下。每次找硬币时，尽可能多地优先使用面额最大的硬币进行找零，如果面额不够了，再改为次一个面额的硬币，直至找零结束。

下面证明该算法能得到最优解。

记 $\{x_i\}$ 为贪心算法所求得的找零序列，序列单调不增，每一项是具体的找零硬币面值，总项数 $m$ 即为所需硬币数。记 $\{y_i\}$ 为实际最优解序列，同样是单调不增，总项数为 $s$ ，那么有 $s \leq m$ 。显然两个序列所有项的总和相等。

假设 $s < m$ ，那么这两个序列就不是完全相同。记前 $a$ 项是相同的，第 $a+1$ 项开始出现不同，即 $x_{a+1} \neq y_{a+1}, a \leq s$ 。

由于贪心算法是优先选取尽可能大的面额，所以 $x_{a+1} > y_{a+1}$ ，又由于之前假设 $y$ 序列是单调不增，所以 $y_{a+1}$ 以及后面那些项也都小于 $x_{a+1}$ 。

对于本题，最优解序列 $y$ 中，1的个数最多只能有4个，否则可以将其中5个1换成1个5，与最优解序列的假设矛盾。

同理，5的个数最多只能有1个，否则可以将其中2个5换成1个10。

10的个数最多只能有2个，否则可以将其中3个10换成1个25和1个5。并且当10的个数达到最多即2个的时候，序列中不会再出现5了，否则可以将其中的2个10和1个5换成1个25。

从而可以依次算得，假设最优解序列 $y$ 中最大的面额不是25，那么该序列所能找的零钱的最大值，比“大于该最大面额的硬币”的面额小。这意味着 $y_{a+1}$ 以及后面那些项的总和小于 $x_{a+1}$ ，那么这两个序列所有项的总和不相等，矛盾。这意味着不存在这样的 $a$ ，即 $s < m$ 的假设不成立，所以这贪心算法的解即为最优解。

### b.

与题a的证明类似，可以证明，最优解序列中除了 $c^k$ 个数不限，其他面额的出现个数最多只能是 $c-1$ 个，否则可以用更高面额的硬币代替。从而 $(c-1)\sum_{i=0}^{j} c^i = (c-1)\frac{c^j-1}{c-1} = c^j - 1$ ，由此可得，如果最优解序列 $y$ 中最大的面额不是 $c^k$ ，那么该序列所能找的零钱的最大值，比“大于该最大面额的硬币”的面额小。从而贪心算法的解即为最优解。

### c.

设计一组硬币面额（美分）分别为1，4，5，6。

当n为9时，如果按照该贪心算法，所得的找零序列为 $\{6,1,1,1\}$ ，一共4个硬币。但是最优解序列为 $\{5,4\}$ ，一共2个硬币。所以这种贪心算法不能保证得到最优解。

### d.

设计算法如下。是用动态规划算法。

记 $f[i]$ 为当所需找零数为i的时候，需要最少硬币的数量，记 $c[i]$ 为这k种不同面额的硬币的面额序列。那么 $f$ 的最优子结构满足：

$$f[i] = \min\{f[i-c[j]] + 1, 1 \leq j \leq k, c[j] \leq i\}$$

且 $f[0] = 0$ ，那么可以用递推循环的方式求得 $f[n]$ 的值。每次循环的时间为 $\mathrm{O}(k)$ ，共循环n次，所以总时间复杂度为 $\mathrm{O}(nk)$ 。