Greedy Algorithms

Instead of considering all sequences of steps, the algorithm selects the best choice in each step.

Cashier's Algorithm.

Let n be a positive integer. Using quarters, dimes, nickels, and pennies, find the combination that uses the least number of coins that sum to n cents.

procedure change($c_1, \ldots, c_r$: values of coins, $c_1 > \cdots > c_r$
n: positive int)

```
for i=1 to r
    d_i := 0
    while n >= c_i
        d_i := d_i + 1
        n := n - c_i
```

Ex. 67 (cents) $= 25 + 25 + 10 + 5 + 1 + 1$.

6 coins are used: 2 quarters, 1 dime, 1 nickel
2 pennies.

46 (cents) $= 25 + 10 + 10 + 1$

4 coins are used: 1 quarter, 2 dimes, 1 penny.

Remark) The greedy algorithm may not be optimal for some values of coins.

Ex. US coin system without nickels (5)

40 (cents) $= 25 + 10 + 5 \cdot 1$ (7 coins)

$= 4 \cdot 10$ (4 coins)

The upper one is the result of cashier's algorithm whi's is not optimal.

Thus, greedy algorithm does not always find the optimal solution.

<u>Theorem</u> For US coin system, the greedy algorithm is optimal

<u>Lemma1.</u> If n is a positive integer, then n cents in quarters dimes, nickels, and pennies using the fewest coins possible has at most 2 dimes, at most 1 nickel, at most 4 pennies and cannot have 2 dimes and 1 nickel. The amount of change in dimes, nickels, and pennies cannot exceed 24 cents.

proof) (Proof by contradiction, exchange argument) If we had more than specified,

3 dimes → 1 quarter, 1 nickel. (fewer)
2 nickel → 1 dime (")
5 pennies → 1 nickel (")
2 dimes, 1 nickel → 1 quarter

We can have at most 2 dimes, 1 nickel, 4 pennies, but we cannot have 2 dimes and 1 nickel. Thus, 24 cents is the most money in dimes, nickels, pennies when we make change using the fewest number of coins for n cents

Note) Exchange argument proves non-optimality if one deviates from the proposed algorithm.

Proof of the optimality of cashier's algorithm for US coin system

(Proof by contradiction, exchange argument)

Suppose for some n, there is a way to make n cents using fewer than the greedy algorithm does. Let q' be the number of quarters in the optimal way to make n cents. Let q be the number of quarters used in the greedy algorithm to make n cents. Since the greedy algorithm uses the most number of quarters possible, $q' \le q$.

If $q' < q$, then $n - 25q' \geq 25$ and this amount is to be made with dimes, nickels, pennies. By Lemma 1, the optimal way uses dimes, nickels, pennies only up to 24. This is a contradiction. Hence $q' = q$ and $n - 25q' \le 24$

Noting that up to 24 cents, the greedy gives the optimal number of coins in each denomination, dime, nickel, penny, the optimal way agrees with the greedy.

Remark) When the greedy agrees with the optimal, such a coin system is called <u>canonical</u>. Thus, US coin system is canonical.

Pearson's algorithm: Decide whether a coin system is canonical in $O(n^3)$ time where $n = $ # of coin denominations.

Division Algorithm.

If n is an integer, d is a nonzero integer, then there are unique integers q, r such that

$$n = dq + r, \quad 0 \le r < |d|.$$

Cashier's Algorithm revisited (Python3)

```
def greedy_rep(coins, amount):
    rep=[]
    rem = amount              # remaining amount
    for c in coins:           # instead of repeating subtraction
        cnt = rem // c        # we may find # of coins directly
        rep.append(cnt)       # by ⌊rem/c⌋ = q, new rem = r.
        rem -= cnt * c
    return rep
```