4. GREEDY ALGORITHMS I

- *coin changing*
- *interval scheduling*
- *interval partitioning*
- *scheduling to minimize lateness*
- *optimal caching*

Coin changing

Goal. Given U. S. currency denominations { 1, 5, 10, 25, 100 }, devise a method to pay amount to customer using fewest coins.

Ex. 34¢.

Cashier's algorithm. At each iteration, add coin of the largest value that does not take us past the amount to be paid.

Ex. $2.89.

Cashier's algorithm

At each iteration, add coin of the largest value that does not take us past the amount to be paid.

CASHIERS-ALGORITHM $(x, c_1, c_2, \ldots, c_n)$

SORT n coin denominations so that $0 < c_1 < c_2 < \ldots < c_n$.

$S \leftarrow \emptyset$. ← multiset of coins selected

WHILE $(x > 0)$

  $k \leftarrow$ largest coin denomination c_k such that $c_k \leq x$.

  IF (no such k)

    RETURN *"no solution."*

  ELSE

    $x \leftarrow x - c_k$.

    $S \leftarrow S \cup \{ k \}$.

RETURN S.

Greedy algorithms I: quiz 1

Is the cashier's algorithm optimal?

A. Yes, greedy algorithms are always optimal.

B. Yes, for any set of coin denominations $c_1 < c_2 < \ldots < c_n$ provided $c_1 = 1$.

C. Yes, because of special properties of U.S. coin denominations.

D. No.

Cashier's algorithm (for arbitrary coin denominations)

Q. Is cashier's algorithm optimal for any set of denominations?

A. No. Consider U.S. postage: 1, 10, 21, 34, 70, 100, 350, 1225, 1500.

- Cashier's algorithm: 140¢ = 100 + 34 + 1 + 1 + 1 + 1 + 1 + 1.
- Optimal: 140¢ = 70 + 70.

A. No. It may not even lead to a feasible solution if $c_1 > 1$: 7, 8, 9.

- Cashier's algorithm: 15¢ = 9 + ?.
- Optimal: 15¢ = 7 + 8.

Properties of any optimal solution (for U.S. coin denominations)

Property. Number of pennies ≤ 4.

Pf. Replace 5 pennies with 1 nickel.

Property. Number of nickels ≤ 1.

Property. Number of quarters ≤ 3.

Property. Number of nickels + number of dimes ≤ 2.

Pf.

- Recall: ≤ 1nickel.
- Replace 3 dimes and 0 nickels with 1 quarter and 1 nickel;
- Replace 2 dimes and 1 nickel with 1 quarter.

dollars (100¢)

quarters (25¢)

dimes (10¢)

nickels (5¢)

pennies (1¢)

Optimality of cashier's algorithm (for U.S. coin denominations)

Theorem. Cashier's algorithm is optimal for U.S. coins { 1, 5, 10, 25, 100 }.

Pf. [by induction on amount to be paid x]

- Consider optimal way to change $c_k \le x < c_{k+1}$: greedy takes coin k.
- We claim that any optimal solution must take coin k.
  - if not, it needs enough coins of type $c_1, \ldots, c_{k-1}$ to add up to x
  - table below indicates no optimal solution can do this
- Problem reduces to coin-changing $x - c_k$ cents, which, by induction, is optimally solved by cashier's algorithm. ▪

k	c_k	all optimal solutions must satisfy	max value of coin denominations $c_1, c_2, \ldots, c_{k-1}$ in any optimal solution
1	1	$P \le 4$	–
2	5	$N \le 1$	4
3	10	$N + D \le 2$	4 + 5 = 9
4	25	$Q \le 3$	20 + 4 = 24
5	100	*no limit*	75 + 24 = 99