Session 33: Optimization Algorithms

- Greedy Algorithms
- Cashier’s Algorithm

Optimization Problems

Optimization problems minimize or maximize some parameter over all possible inputs

Examples

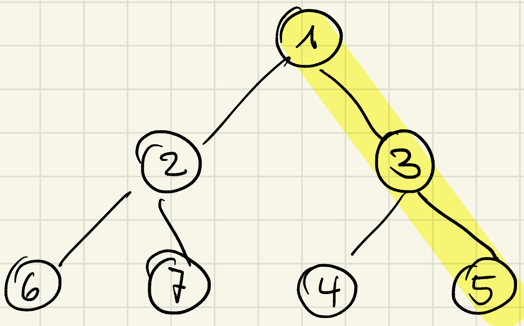
- Finding a route between two cities with the smallest total mileage.
- Determining how to encode messages using the fewest possible bits.
- Finding the fiber links between network nodes using the least amount of fiber.

Greedy Algorithms

Optimization problems can often be solved using a *greedy algorithm*, which makes the “best” choice at each step.

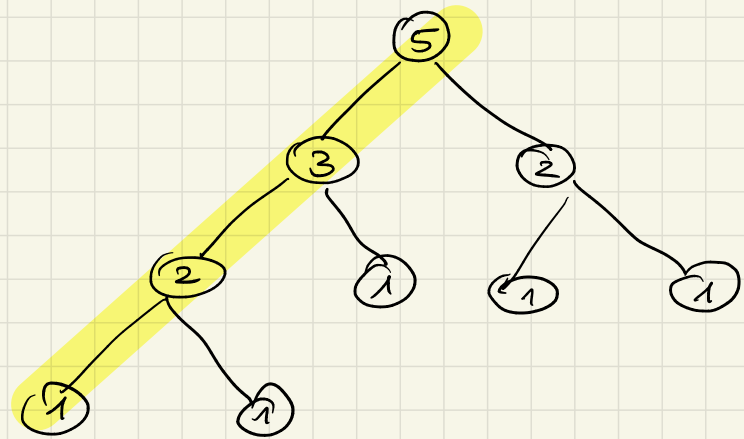
- Making the “best choice” at each step does not necessarily produce an optimal solution to the overall problem
  - but in many instances, it does.
- After specifying the greedy algorithm,
  - Either we prove that this approach always produces an optimal solution
  - or we find a counterexample to show that it does not.

find the largest leaf!



greedy fails!

find a longest path!



greedy works!

Cashier’s Algorithm

Task: Find for an amount of any n cents the least total number of coins using the following coins: quarters (25 cents), dimes (10 cents), nickels (5 cents), and pennies (1 cent).

(Greedy) Idea: At each step choose the coin with the largest possible value that does not exceed the amount left.

Example

Change for n = 67 cents.

$25 - 10 - 5 - 1$

choice	rest
25	42
25	17
10	7
5	2
1	1
1	

Smallest number of coins (?)

Cashier's Algorithm

The algorithm works with any coin denominations $c_1, c_2, \ldots, c_r$

procedure *change*($c_1, c_2, \ldots, c_r$: values of coins, where $c_1 > c_2 > \ldots > c_r$; n: a positive integer)

for $i := 1$ **to** r *{start with the largest coins}*

$d_i := 0$ *{d_i counts the coins of denomination c_i}*

while $n \geq c_i$

$d_i := d_i + 1$ *{add a coin of denomination c_i}*

$n = n - c_i$ *{remove the value of the coin}*

return $d_1, d_2, \ldots, d_r$

For the example of U.S. currency, we have quarters, dimes, nickels and pennies, with $c_1 = 25$, $c_2 = 10$, $c_3 = 5$, and $c_4 = 1$.

Proving Optimality

Show that the change making algorithm for *U.S.* coins is optimal.

Lemma 1: If n is a positive integer, then n cents in change using quarters, dimes, nickels, and pennies, using the fewest coins possible

1. has at most 2 dimes, 1 nickel, 4 pennies
2. cannot have 2 dimes and 1 nickel
3. the total amount of change in dimes, nickels, and pennies cannot exceed 24 cents.

Proof of Lemma 1

Proof:

Property 1: By contradiction (not the fewest coins)

- If we had 3 dimes, we could replace them with a quarter and a nickel.
- If we had 2 nickels, we could replace them with 1 dime.
- If we had 5 pennies, we could replace them with a nickel.

Property 2: By contradiction (not the fewest coins)

- If we had 2 dimes and 1 nickel, we could replace them with a quarter.

Property 3: is a consequence

- The largest allowable combination, 2 dimes and 4 pennies, has a maximum value of 24 cents.

Theorem: The greedy change-making algorithm for U.S. coins produces change using the fewest coins possible.

Proof: Let d_{25}, d_{10}, d_5, d_1 the number of coins in the optimal solution and

$\bar{d}_{25}, \bar{d}_{10}, \bar{d}_5, \bar{d}_1$ be the number of coins chosen by the algorithm.

From lemma we have: $d_{10} \leq 2$, $d_5 \leq 1$, $d_1 \leq 4$, $10 d_{10} + 5 d_5 + d_1 \leq 24$

$\bar{d}_{25} \geq d_{25}$ since alg. chooses the maximal number of 25

(*) next page

Assume $\bar{d}_{25} > d_{25}$, then $10 d_{10} + 5 d_5 + d_1 > 24$. Contradiction! Thus $\bar{d}_{25} = d_{25}$

$\bar{d}_{10} \geq d_{10}$ since alg. chooses the maximal number of 10 from remaining amount

Assume $\bar{d}_{10} > d_{10}$, then $5 d_5 + d_1 > 9$. Contradiction, since $d_5 \leq 1, d_1 \leq 4$, Thus $\bar{d}_{10} = d_{10}$

$\bar{d}_5 \geq d_5$ since alg. chooses max. nb. of 5 from remaining amount

Assume $\bar{d}_5 > d_5$, then $d_1 > 4$. Contradiction since $d_1 \leq 4$, Thus $\bar{d}_5 = d_5$

and $\bar{d}_1 = d_1$ $\blacktriangle$

* We want to show that $\overline{d}_{25} = d_{25}$.

The greedy algorithm chooses the largest possible number of coins of value 25

Therefore $\overline{d}_{25} \geq d_{25}$, no other solution can have more coins.

Thus, the only possibility that $\overline{d}_{25} \neq d_{25}$ is when $\overline{d}_{25} > d_{25}$.

Let us assume $\overline{d}_{25} > d_{25}$

Then in the optimal solution the coins of value 10, 5, 1 have to

cover an amount of at least 25, to replace at least one coin of 25

Therefore, $10 d_{10} + 5 d_5 + d_1 \geq 25$.

This contradicts property 3 of lemma 1, which states $10 d_{10} + 5 d_5 + d_1 \leq 24$

Therefore, by contradiction, it is not possible that $\overline{d}_{25} > d_{25}$

and thus $\overline{d}_{25} = d_{25}$

Cashier’s Algorithm Discussion

Optimality depends on the denominations available.

If we allow only quarters (25 cents), dimes (10 cents), and pennies (1 cent), the algorithm no longer produces the minimum number of coins.

Counterexample: Consider the example of 31 cents.

- The optimal number of coins is 4, i.e., 3 dimes and 1 penny. (4 coins)
- Result of algorithm: 1 quater, 6 dimes (7 coins)

Summary

- Greedy Algorithms
- Cashier’s Algorithm
- Optimality Proof

CH coin denominations : 1, 5, 10, 20, 50

What would be lemma 1?

Lemma 1_{CH} : n cents in change with fewest coins possible would have

1. at most 4×1, 1×5, 1×10, 2×20
2. cannot have 2×20 and 1×10
3. total amount of change using 1, 5, 10, 20 cannot exceed 49

Can we drop coins in CH?

Theorem:

For denominations $c_1, \dots, c_n$ where $c_i \mid c_{i+1}$, the greedy algorithm is optimal (sufficient condition).

Therefore 1, 5, 10, 50 works!

But also 1, 10, 20, 50 works (the theorem is not a necessary condition)

We cannot drop further coins: e.g. 1, 20, 50 does not work.

Counter-Example: For 61: optimal $3 \times 20 + 1 \times 1$

greedy $1 \times 50 + 11 \times 1$