

Section 8.2 Coin Change

Determine the minimum number of coins needed to make change for A cents, with a given set of n denominations. The denominations are arranged in decreasing order, and the smallest denomination is 1. When A and n are relatively “small”, we can use dynamic programming.

A = the amount of change (in cents)
 denom = array of denominations (types of available coins)
 $\text{denom}[1] > \text{denom}[2] > \dots > \text{denom}[n] = 1$
 $C[i][j]$ = minimum number of coins needed to make j cents using only coins of denominations: $\text{denom}[i], \text{denom}[i+1], \dots, \text{denom}[n]$
 $\text{used}[i][j]$ = true if a coin of denomination $\text{denom}[i]$ is used in the smallest set of coins to make j cents

To use dynamic programming, we will build a solution to this problem by solving subproblems.

The (i, j) -subproblem: for $1 \leq i \leq n$, and $0 \leq j \leq A$
determine the minimum number of coins needed to make j cents using only the smallest denominations from i to n , that is, using only denomination set:
 $\text{denom}[i], \text{denom}[i+1], \dots, \text{denom}[n]$
Store the result, i.e., the minimum number of coins for the (i, j) -subproblem in $C[i][j]$.

For the (i, j) -subproblem, we must decide whether or not 1 coin of denomination $\text{denom}[i]$ should be used to make j cents. Thus, consider the following two options:

If we use one coin of $\text{denom}[i]$ to make j cents:

- o First, make sure that $j \geq \text{denom}[i]$ (is it even possible to use such a coin?)
- o Remaining change amount is: $j - \text{denom}[i]$
- o Use the same denomination set $\text{denom}[i], \text{denom}[i+1], \dots, \text{denom}[n]$ to make the remaining change amount $j - \text{denom}[i]$ with as few coins as possible:
 $C[i][j - \text{denom}[i]]$
- o Add 1 for the one coin of denomination $\text{denom}[i]$
 Total number of coins to make j cents would be: $1 + C[i][j - \text{denom}[i]]$

If we do not use one coin of $\text{denom}[i]$ to make j cents:

- o Remaining change amount is still: j
- o Use the denomination set $\text{denom}[i+1], \text{denom}[i+2], \dots, \text{denom}[n]$ to make the remaining change amount j . (Notice $\text{denom}[i]$ is not used.)
 $C[i+1][j]$
- o Number of coins to make j cents would be: $C[i+1][j]$

To find the solution for the (i, j) -subproblem, we determine which way is better, i.e.

which is smaller? $1 + C[i][j - \text{denom}[i]]$ or $C[i+1][j]$

The optimal substructure:

The solution to the (i, j) - subproblem can be summarized as follows:

$$C[i][j] = \begin{cases} C[i+1][j], & \text{if } \text{denom}[i] > j \\ \min\{C[i+1][j], 1 + C[i][j - \text{denom}[i]]\}, & \text{if } \text{denom}[i] \leq j \end{cases}$$

Once the array C has been filled, the solution to the problem for A cents and allowing all n denominations is in $C[\] [\]$.

Input: denom, A

Output:

```
dynamicCoinChange (denom, A)
{
    n = denom.last
    for j = 0 to A
    {
        C[n][j] = _____
        used[n][j] = _____
    }
    used[n][0] = false
    for i = n - 1 downto 1
    {
        for j = 0 to A
        {
            if (denom[i] > j || C[i+1][j] < 1 + C[i][j - denom[i]])
            {
                C[i][j] = _____
                used[i][j] = _____
            }
            else
            {
                C[i][j] = _____
                used[i][j] = _____
            }
        }
    }
}
```

running time:

