University of Washington March 4, 2020
Department of Computer Science and Engineering
CSE 417, Winter 2020

Homework 8, Problem 3

Problem 3 (10 points) Counting solutions to the subset sum:

The subset sum counting problem is: Given a set of values $S = \{s_1, \ldots, s_n\}$, and an integer K, determine the number of subsets of S that sum to exactly K. Design an algorithm that solves the subset sum counting problem. Your algorithm should have runtime $O(nK)$.

Answer:

algorithm:

```
def count_subset_sum_to_K(S, K):
    sorted(S)
    Opt = [0 for i in range(K + 1)]
    Opt[0] = 1
    for s in S:
        for k in range(K, -1, -1):
            if k >= s:
                Opt[k] += Opt[k - s]
    return Opt[K]
```

proof:

base case:

By defination, when there is no value, the sum from 0 to K can not attain by no value, therefore the Opt value all equal to zero.

induction hypothesis:

For each value s_i in set S, assume the value $s_0 \ldots s_{i-1}$ has been correctly determined :
Then when k equal to each value from 0 to K: Opt[k] = Opt[k] + Opt[k - s]

The function apply to each inductive step since for each value k, there are only two possibilities:
(1) value k can be divide into s and k - s, s and k - s can be calculated from value $s_0 \ldots s_{i-1}$

(2) value k can be divide into s and k - s, s and k - s cannot be calculated from value $s_0 \ldots s_{i-1}$
In either case, the value of Opt[k] (number of solutions to get k from $s_0 \ldots s_i$) rely on the correctly precaclulated value Opt[k] and Opt[k - s], regardless k can be attaibed by values from value $s_0 \ldots s_{i-1}$ or not, if not, the Opt[k] and Opt[k - s] are all zero, if can attaibed, the value built upon the prior result from $s_0 \ldots s_{i-1}$.
Since the algorithm calculate Opt$[0 \ldots k]$ from s_0 cummulative to s_n, and for each adding s_i, the algorithm attaines result built opon from correctly precomputed value, then algorithm is garantees to find the number of solutions add up to K.

complexity:

Since on each subproblem of subset $s_0 \dots s_i$, the algorithm only traces the result from at most K result, so each subproblem will has K opearations. And there are n subproblem, therefore the time complexity is $O(nK)$.