Solution Review: Problem Challenge 2

Target Sum (hard)

You are given a set of positive numbers and a target sum 'S'. Each number should be assigned either a '+' or '-' sign. We need to find the total ways to assign symbols to make the sum of the numbers equal to the target 'S'.

Example 1:

```
Input: {1, 1, 2, 3}, S=1
Output: 3
Explanation: The given set has '3' ways to make a sum of '1': {+1-1-2+3} & {-1+1-2+3} & {+1+1+2
-3}
```

Example 2:

```
Input: {1, 2, 7, 1}, S=9
Output: 2
Explanation: The given set has '2' ways to make a sum of '9': {+1+2+7-1} & {-1+2+7+1}
```

Solution

This problem follows the **0/1 Knapsack pattern** and can be converted into Count of Subset Sum. Let's dig into this.

We are asked to find two subsets of the given numbers whose difference is equal to the given target 'S'. Take the first example above. As we saw, one solution is {+1-1-2+3}. So, the two subsets we are asked to find are {1, 3} & {1, 2} because,

```
    (1 + 3) - (1 + 2 ) = 1
```

Now, let's say 'Sum(s1)' denotes the total sum of set 's1', and 'Sum(s2)' denotes the total sum of set 's2'. So the required equation is:

```
    Sum(s1) - Sum(s2) = S
```

This equation can be reduced to the subset sum problem. Let's assume that 'Sum(num)' denotes the total sum of all the numbers, therefore:

```
    Sum(s1) + Sum(s2) = Sum(num)
```

Let's add the above two equations:

```
    => Sum(s1) - Sum(s2) + Sum(s1) + Sum(s2) = S + Sum(num)
    => 2 * Sum(s1) =  S + Sum(num)
    => Sum(s1) = (S + Sum(num)) / 2
```

Which means that one of the set 's1' has a sum equal to `(S + Sum(num)) / 2`. This essentially converts our problem to: "Find the count of subsets of the given numbers whose sum is equal to `(S + Sum(num)) / 2`"

Code

Let's take the dynamic programming code of Count of Subset Sum and extend it to solve this problem:

Java | Python3 | C++ | JS

```java
class TargetSum {

  public int findTargetSubsets(int[] num, int s) {
    int totalSum = 0;
    for (int n : num)
      totalSum += n;

    // if 's + totalSum' is odd, we can't find a subset with sum equal to '(s + totalSum) / 2'
    if(totalSum < s || (s + totalSum) % 2 == 1)
      return 0;

    return countSubsets(num, (s + totalSum) / 2);
  }

```

Mark Course as Completed

```java
  // this function is exactly similar to what we have in 'Count of Subset Sum' problem.
  private int countSubsets(int[] num, int sum) {
    int n = num.length;
    int[][] dp = new int[n][sum + 1];

    // populate the sum=0 columns, as we will always have an empty set for zero sum
    for(int i=0; i < n; i++)
      dp[i][0] = 1;

    // with only one number, we can form a subset only when the required sum is equal to the number
    for(int s=1; s <= sum ; s++) {
      dp[0][s] = (num[0] == s ? 1 : 0);
    }

```

RUN SAVE RESET

Close

Output 2.809s

```
3
2
```

Time and Space complexity

The above solution has time and space complexity of $O(N * S)$, where 'N' represents total numbers and 'S' is the desired sum.

We can further improve the solution to use only $O(S)$ space.

Space Optimized Solution

Here is the code for the space-optimized solution, using only a single array:

Java | Python3 | C++ | JS

```java
class TargetSum {

  public int findTargetSubsets(int[] num, int s) {
    int totalSum = 0;
    for (int n : num)
        totalSum += n;

    // if 's + totalSum' is odd, we can't find a subset with sum equal to '(s + totalSum) / 2'
    if(totalSum < s || (s + totalSum) % 2 == 1)
        return 0;

    return countSubsets(num, (s + totalSum) / 2);
  }

  // this function is exactly similar to what we have in 'Count of Subset Sum' problem.
  private int countSubsets(int[] num, int sum) {
    int n = num.length;
    int[] dp = new int[sum + 1];
    dp[0] = 1;

    // with only one number, we can form a subset only when the required sum is equal to the number
    for(int s=1; s <= sum ; s++) {
      dp[s] = (num[0] == s ? 1 : 0);
    }

    // process all subsets for all sums
    for(int i=1; i < num.length; i++) {
      for(int s=sum; s >= 0; s--) {
```

RUN SAVE RESET

Close

Output 1.399s

```
3
2
```


Back
Problem Challenge 2

Next
Introduction

Mark as Completed

Report an Issue
Ask a Question

Explore
Tracks
My Courses
Edpresso
Refer a Friend
Create