一、题目说明

题目494. Target Sum，给定一列非负整数，一个目标数S，给定每个数一个`+,-`号，计算有多少组合可以生成S的值。难度是Medium！

二、我的解答

最直接的方案就是对每一个数`num[i]`，可以正，可以负，通过递归就可以枚举所有情况。

```
class Solution{
    public:
        //recursive
        int findTargetSumWays(vector<int>& nums,int S){
            dfs(nums,0,0,S);
            return count;
        }
        void dfs(vector<int>& nums,int i,int sum,int S){
            if(i==nums.size()){
                if(sum == S){
                    count++;
                }
            }else{
                dfs(nums,i+1,sum+nums[i],S);
                dfs(nums,i+1,sum-nums[i],S);
            }
        }
    private:
        int count = 0;
};
```

性能如下：

```
Runtime: 1084 ms, faster than 20.25% of C++ online submissions for Target Sum.
Memory Usage: 8.6 MB, less than 61.54% of C++ online submissions for Target Sum.
```

三、优化措施

设`dp[i][j]`表示用数组中的前 i 个元素，组成和为 j 的方案数。`dp[i][j] = dp[i - 1][j - nums[i]] + dp[i - 1][j + nums[i]]`

用P表示所有正数的和，用N表示所有负数的和，用T表示目标和，用All表示整个集合的和：

```
P-N = T    --->    N = P-T
P+N=All   --->    P+(P-T) = 2P-T = ALL
P = (T+ All)/2    总和All已知，T已知，求P即可，这是一个典型的背包问题。
```

问题转换为：存在一个容量为P 的背包，从 nums 中任意抽取一定数量的数，使得背包恰好被放满，有多少种放法。

```
class Solution{
    public:
        //dp  P = (T+ All)/2
```

```
        int findTargetSumWays(vector<int>& nums,int S){
            long sum = 0;
            for(auto it:nums){
                sum += it;
            }
            if((S+sum)%2==1 || S>sum){
                return 0;
            }
            S = (S+sum)/2;
            vector<int> dp(S+1,0);
            dp[0] = 1;
            for(const int & it:nums){
                for(int j=S;j>=it;j--){
                    dp[j] += dp[j-it];
                }
            }
            return dp[S];
        }
};
```

性能如下：

```
Runtime: 4 ms, faster than 97.66% of C++ online submissions for Target Sum.
Memory Usage: 8.8 MB, less than 46.15% of C++ online submissions for Target Sum.
```