CS 401 Programming Assignment I Report

Participants

Sai Nadkarni- 672756678

Ivan Madrid- 677607014

Source Code

Github: https://github.com/sai-n20/subsetsum-DP

Data structures used:-

2D vector distinctSubsets- DP table for housing count of distinct subsets

2D vector minCount- DP table for housing minimum size among all possible subsets

distinctSmallestSubsetCounter- Unsigned integer to count distinct smallest subsets

Number of distinct subsets

For distinct subsets we create a *n* * sum+1 sized DP table. We bootstrap the table in the usual way for 0 and 1 cases. Then we iterate on the table while performing a constant amount of work, which covers and establishes our runtime complexity at O(n*Sum). The DP optimal substructure here is basically accounting for 2 cases, one case where we figure out the subset count for including the current (i^{th}) element into consideration, while another case of not including the current element in the subset calculation. For include case, if we don't include the current element in our distinct subsets count, then we have to adjust the target sum accordingly, and consequently the DP table entry we're accessing. For the exclude case we simply refer to the previous row which tells us the subset count at 1 element less. For an entry of the DP table, we add the values of both include and exclude case and we get the distinct subset count for that combination.

```
if (elems.at(i).x <= x) {
                    includingCurrentValue = distinctSubsets[i - 1][x - elems.at(i).x];
                }
                excludingCurrentValue = distinctSubsets[i - 1][x];
                distinctSubsets[i][x] = includingCurrentValue + excludingCurrentValue;
```

After the table is built, we will return the last value within the table which holds the value of the total distinct subsets that have a sum of the target sum.

Size of smallest subset

We proceed in a similar manner as the previous problem, except here the "function" is to minimize among the include case and the exclude case. We bootstrap the table with a minor change that since an empty set {} will have size 0, the first column will be filled with 0's and not 1's. Also another optimization possible here is if coincidentally the target matches with some element of our input, we can trivially just return 1 as the lowest sized subset. Our include and exclude variables are initialized to max values of the data type possible, with some room for error (removed a digit so one order of magnitude lesser here). The exclude case is same as before but with the include case, we need to look in the DP table for a similar sized input but with a sum lesser than the current sum in consideration, where its lesser by the magnitude of the element we're perusing to add. Since we're increasing the size of the subset, we add 1 to the include value. Finally, we take a min() of both include and exclude values to make a decision on that iteration on whether to include or exclude that element.

In practice for the given tasks, we're not getting the correct output. While developing, we used a smaller test file which included data for 12 randomly picked states. Output mentioned below. This text file is also included in my source files for reference.

```
F:\Projects\subsetsum-DP>ssum 21 < test.txt
The target sum of 21 is FEASIBLE!

Number of distinct solutions: 15
Size of smallest subset: 2
Amount of smallest subsets: 3
F:\Projects\subsetsum-DP>
```

Number of distinct minimum size subsets

Not attempted.

Lexicographic first minimum size subset

Not attempted.

Data on various runs

Scenario: Electoral college tie scenario, T = 269

```
F:\Projects\subsetsum-DP>ssum 269 < electoral.txt
The target sum of 269 is FEASIBLE!

Number of distinct solutions: 16976480564070
Size of smallest subset: 6

Amount of smallest subsets: 0
F:\Projects\subsetsum-DP>
```

Scenario: Republican victory among purple states

```
F:\Projects\subsetsum-DP>ssum 220 < purple.txt
The target sum of 220 is FEASIBLE!

Number of distinct solutions: 9958625
Size of smallest subset: 7

Amount of smallest subsets: 0
```

Scenario: Democratic victory among purple states

```
F:\Projects\subsetsum-DP>ssum 121 < purple.txt
The target sum of 121 is FEASIBLE!

Number of distinct solutions: 9958625
Size of smallest subset: 4

Amount of smallest subsets: 0
```