CS222 Homework 4

Algorithm Analysis & Deadline: 2020-11-04 Wednesday 23:59

Exercises for Algorithm Design and Analysis by Li Jiang, 2020 Autumn Semester

1. Given a integer set S, which has n elements in it, you need to divide this set into 2 subarray, where the subarray S1 has m elements, and the subarray S2 has (n-m) elements. the target is minimizing $abs(sum(S1) - sum(S2))$.

   Implement your algorithm with C/C++/Python. Please attach your source code named as *Code-P1.**. The file `Data-P1.txt` is a test case, includes an 1-D array of integer with random length, and your program needs to output the final sum. You need to briefly describe your algorithm and find the final sum of `Data-P1.txt` by your program.

   Example:

   Given input array: $arr[] = 1, 6, 11, 5$. The algorithm should return 1. The subset $S1 = [1, 5, 6], S2 = [11]$.

   Solution.
   Def.

$$f(i, sum) = \begin{cases} True\ , & if\ \exists\ I \subseteq \{1, 2, 3, \cdots, i\}\ s.t.\ \sum_{j \in I} S_j = sum \\ False\ , & otherwise \end{cases}$$

   Recursion:

$$f(i + 1, sum) = f(i, sum) \vee f(i, sum - S_i)$$

   Initialization:

$$f(0, 0) = True$$

   Traverse $f(n, s)$ to check whether S can be divide to two parts S_1 and S_2 s.t. $Sum(S_1) = s$.
   With proper recursion order, the component i is not necessary. We can finish it in $\Theta(nS)$ **time** and $\Theta(S)$ **space**.

   Result of my program on test data: 1
   Code for P1: read from "./code/Data-P1.txt", and print the minimal $|sum_1 - sum_2|$ to the screen. It is recommended to turn on the '-O2' option. It takes about $30s$ to run the example test data on my computer. I used fixed length array, so the test data's length should not be larger than 1000000.

2. **Bookshelf:** Tim has n books and he wants to make a bookshelf to them. The pages' width of the i-th book is w_i and the thickness is t_i.

   Tim puts the books on the bookshelf in the following way. He selects some books and put them vertically. Then the rest of the books are put horizontally above the vertical books. Obviously, the total thickness of the books put vertically must be greater than the sum of widths of the horizontal books. As long as tim wants to make the bookshelf as small as possible, please help him to find the minimum total thickness of the vertical books.

   To simplify the problem, the thickness of each book is either 1 or 2. And all the numbers in this problem are positive integers.

   Design an algorithm based on dynamic programming and implement it in C/C++/Python. The file `Data-P2.txt` is a test case, where the first line contains an integer n. Each of the next n lines contains two integers t_i and w_i denoting two attributes of the i-th book. Source code should be named as *Code-P2.** .You need to briefly describe your algorithm and find the result of `Data-P2.txt` by your program.

   Example:

   Given $n = 5$ books, and $\{(t_i, w_i) | 1 \le i \le 5\} = \{(1, 12), (1, 3), (2, 15), (2, 5), (2, 1)\}$. The algorithm should return 5.

Solution.
Def. $f(i,t)$:= the minimal width w of books when the thickness is exactly t. i.e.

$$f(i,t) := \min_{S\subseteq\{1,2,\cdots,i\}} \sum_{i\in S} w_i \ , \ s.t. \sum_{i\in\{1,2,\cdots,i\}\setminus S} t_i = t$$

Recursion:

$$f(i,t) = \min\{f(i-1,t-t_i), f(i-1,t) + w_i\}$$

For the $i-th$ book, there are two options:

1) Put vertically, then the optimal value is $f(i-1,t-t_i)$.
2) Put horizontally, then the optimal value is $f(i-1,t)+w_i$.

Initialization:

$$f(0,t) = \begin{cases} 0, & if\ t=0 \\ +\infty, & otherwise \end{cases}$$

The thickness t is feasible if and only if $f(n,t) \le t$. So after calculate $f(.)$, we only need to find

$$\min t \ \ s.t.\ f(n,t) \le t$$

With proper recursion order, the component i is not necessary. **Time** $O(n^2 t_{max})$, **Space:** $O(nt_{max})$.
Result of my program on test data: 2542
Code for P2: read from "./code/Data-P2.txt", and print the minimal thickness to the screen. It is recommended to turn on the '-O2' option. It takes less than $1s$ to run the example test data on my computer.

3. Recall the *String Similarity* problem in class, in which we calculate the edit distance between two strings in a sequence alignment manner.

   You are to find the lowest aligning cost between 2 DNA sequences, in which the cost matrix is defined as

	-	A	T	G	C
-	0	1	2	1	3
A	1	0	1	5	1
T	2	1	0	9	1
G	1	5	9	0	1
C	3	1	1	1	0

   where `(-, A)` means adding (or removing) one `A`, etc.

   (a) Implement Hirschberg's algorithm with C/C++/Python. Please attach your source code named as *Code-P3.**. Your program will be tested against random inputs. Your program should be able to output two sequences after editing.

   (b) Using your program, find the edit distance between the two DNA sequences found in attachments `Data-P3a.txt` and `Data-P3b.txt`.

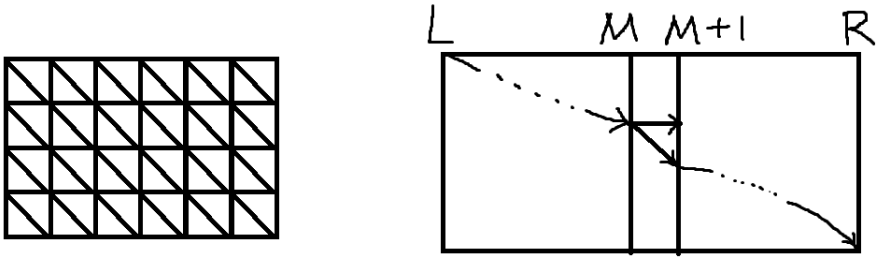
Solution.
Def. $f(i,j)$:= the minimal aligning cost between $s_{1\cdots i}$ and $t_{1\cdots j}$, where s and t are the two strings.
Recursion:

$$f(i,j) = \min \begin{cases} f(i-1,j) + cost(s_i, -) \\ f(i,j-1) + cost(-, t_j) \\ f(i-1,j-1) + cost(s_i, t_j) \end{cases}$$

Time: $\Theta(length_s \cdot length_t)$ acceptable. **Space:** $\Theta(length_s \cdot length_t)$ (to record the optimal path) **not** acceptable.

Use the divide and conquer algorithm talked in the class:

Figure 1: Skecth for problem 3



Abstract the origin problem to a shortest path problem on a grid. When $R - L \leq 2$, divide the problem to 2 subproblems. Find the shortest length of path from (L, top) to $(M, *)$, and the shortest length of path from $(M + 1, *)$ to $(R, bottom)$, and determine which point is on the shortest path. In the class, we have proved that this algorithm can solve the problem within $\Theta(length_s \cdot length_t)$ **time** and $\Theta(length_s + length_t)$ **space**, acceptable.
Code for P3: read from "./code/Data-P3a.txt" and "./code/Data-P3b.txt", write the result s' to "./code/Data-P3-out-a.txt" and t' to "./code/Data-P3-out-b.txt". And print the minimal edit distance to the screen. It is recommended to turn on the '-O2' option. It takes about $40s$ to run the example test data on my computer. I used fixed length arrays, so the test data should no longer than 1000000 letters. If you want to test larger data, you should modify the 'MAXN' value in my code.

4. Considering you are playing a game with your friend, there are n coins placed in one row with the value of v1, v2, ..., vn, respectively. You and your friend can take one coin from row head or row tail sequentially. If you are the first to choose the coin, write an algorithm to ensure that you can get the maximum profit. Suppose that your friend is as smart as you.
   Implement your algorithm with C/C++/Python. Please attach your souce code named as *Code-P4.**. The file `Data-P4.txt` is a test case, includes an 1-D array of unsigned integer with random length, and your program needs to output the the coin value list that you choose, and the maximum profit you will get. You need to briefly describe your algorithm and find the final answer of `Data-P4.txt` by your program.

   textbfExample:

   Given input array: $arr[] = 8, 15, 3, 7$. The algorithm should return your choice list [7, 15], and the final profit 22.

Solution.
Game problem. Max-min. Your policy is to maximize the estimated value of the game state, while the other play's policy is to minimize the estimated value of the game state. The estimated value of the game state is defined from your point of view:

Def. $f(i, j) :=$ the maximum profit you can get from the sequence $(v_i, v_{i+1}, \cdots, v_j)$ if you play first.
Equation:

$$f(i,j) = \begin{cases} v_i, & if\ i = j \\ \max\{v_i, v_j\}, & if\ i+1 = j \\ \max\{v_i + \min\{f(i+2,j), f(i+1,j-1)\}, v_j + \min\{f(i,j-2), f(i+1,j-1)\}\}, & if\ i+1 < j \end{cases}$$

If $i + 1 < j$, then there are two choices for you:

1) Take one from row head: you get v_i. Then the other player will take an action to minimize your left profit. So your profit is $v_i + \min\{f(i+2, j), f(i+1, j-1)\}$
2) Take one from row tail: you get v_j. Then the other player will take an action to minimize your left profit. So your profit is $v_j + \min\{f(i, j-2), f(i+1, j-1)\}$

Your maximum profit is the larger one of the two choices above.
Result of my program on test data: 250072
Code for P4: read from "./code/Data-P4.txt" and write the answer (both the optimal step and value) to the file "./code/Data-P4-out.txt". It is recommended to turn on '-O2' option. The length of test data should be no larger than 100000. If you do want to test larger data, you should modify 'MAXN' value in my code.

Remark: You need to upload your .pdf file and write the pseudocode.