Lab05-DynamicProgramming

CS214-Algorithm and Complexity, Xiaofeng Gao, Spring 2020.

$*$ If there is any problem, please contact TA Shuodian Yu.

$*$ Name:Yijia Diao Student ID:518030910146 Email: diao_yijia@sjtu.edu.cn

1. **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-P1.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-P1.** .You need to briefly describe your algorithm and find the result of `Data-P1.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. Result of test case: 2542.

   The original Algorithm of this problem is:

Algorithm 1: DP for Bookshelf (original)

```
Input: $n, w_i, t_i (1 \le i \le n)$.
Output: the minimum total thickness of the vertical books.
1  $SumThick \leftarrow \sum_{i=1}^{n} t_i$;
2  Allocate $OPT[n][SumThick]$, $w$ is the empty length for the horizontal books to
     put on;
3  $\forall w \le SumThick : OPT[0][w] \leftarrow SumThick$;
4  for $i = 1 \to n$ do
5      for $w = SumThick \to (w_i + t_i)$ do
6          if $OPT[i-1][w - w_i - t_i] - t_i \le OPT[i-1][w]$ then
7              $OPT[i][w] \leftarrow OPT[i-1][w - w_i - t_i] - t_i$;
8          else
9              $OPT[i][w] \leftarrow OPT[i-1][w]$;
10     for $w = (w_i + t_i - 1) \to 0$ do $OPT[i][w] \leftarrow OPT[i-1][w]$;
11 return $OPT[n][SumThick]$;
```

From the recurrence relation, I discover that when calculating row i, it only needs data from row $i-1$. So if we calculate one row from right to left, we can reduce the space complexity to $O(n)$. The optimized algorithm is:

Algorithm 2: DP for Bookshelf (optimized)

Input: $n, w_i, t_i (1 \le i \le n)$.
Output: the minimum total thickness of the vertical books.

```
1 $SumThick \leftarrow \sum_{i=1}^{n} t_i$;
2 $OPT[SumThick] \leftarrow 0$;
3 for $i = 1 \rightarrow n$ do
4     for $j = SumThick \rightarrow w_i + t_i$ do
5         if $OPT[j - w_i + t_i] + t_i >= OPT[j]$ then
6             $OPT[j] = OPT[j - w_i + t_i] + t_i$;
7 return $SumThick - OPT[SumThick]$;
```

□

2. 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-P2.**. 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-P2a.txt` and `Data-P2b.txt`.

Solution. Result of test case: 362. □

Notice of `Code-P2.cpp`: you can un-comment `#define ONLY_COST` so that the program won't do recurrence to find the paired sequence, and it only outputs the cost result. You can also un-comment `#define OUTPUT_HORIZONTAL` to change the paired sequences' output format.

Remark: You need to include your .pdf and .tex and 2 source code files in your uploaded .rar or .zip file.