

Lab05-DynamicProgramming

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

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

* Name: Futao Wei Student ID: 518021910750 Email: weifutao@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 $\{(t_i, w_i) | 1 \leq i \leq 5\} = \{(1, 12), (1, 3), (2, 15), (2, 5), (2, 1)\}$. The algorithm should return 5.

Solution. The minimum total thickness is 2542 for the case in `Data-P1.txt`. The description of my algorithm is as follows.

We use a two-dimensional array $dp[0 \dots m][0 \dots 2n]$ to store the results. $dp[i][j]$ denotes the minimum total width for the first i books with a total thickness j . All of the elements in dp are initialized to be a large enough number, say 999999, except that $dp[0][0] = 0$.

For the i -th book, OPT puts it either vertically or horizontally. Hence we have

$$dp[i][j] = \begin{cases} \min(dp[i-1][j-t[i]], dp[i-1][j] + w[i]), & j \geq t[i] \\ dp[i-1][j] + w[i], & j < t[i] \end{cases}$$

We can derive $dp[0 \dots m][0 \dots 2n]$ following the recurrence above. To obtain the minimum total thickness, we need to check $dp[n][j]$ for $1 \leq j \leq 2n$ until we find the minimum j such that $j \geq dp[n][j]$. \square

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. My program works fine for small data, but fails to run with the data provided either on my Macbook or on online C++ platforms. □

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