National Taiwan University
Department of Electrical Engineering
Algorithms, Spring 2022

March 27, 2022

Programming Assignment #2 (Dynamic Programming)

Problem: Cell Legalization

Given is a row with a positive integer width $W \in \mathbb{N}\backslash\{0\}$ and n cells $\{C_1, ..., C_n\}$ (see Figure 1). Each cell C_i has a positive width $w_i \in \mathbb{N}\backslash\{0\}$, a non-negative initial position $x_i \in \mathbb{N}$, and a positive weight $e_i \in \mathbb{N}\backslash\{0\}$. Note that we have $x_1 \leq x_2 \leq ... \leq x_n$. The cell legalization problem can be formulated as the following optimization problem. (1) is the objective function, which is the summation of the weighted displacements of all cells. (2) means that the final position x'_i of each cell C_i is an integer between 0 and $W-1$. (3) means that the cells cannot overlap, and the order of the cells cannot be changed. (4) means that the last cell cannot exceed the width of the row W. You are asked to write a program that computes the minimum cost and the corresponding final position x'_i of each cell C_i. Note that we always have $\sum_{i=1}^{n} w_i \leq W$ to ensure the existence of a valid solution.

$$\min_{x'_i} \sum_{i=1}^{n} e_i |x'_i - x_i| \tag{1}$$

$$s.t.\ x'_i \in \{0, 1, ..., W-1\} \tag{2}$$

$$x_i + w_i \leq x_{i+1} \tag{3}$$

$$x_n + w_n \leq W \tag{4}$$

Figure 1 is an example with three cells. The row has a width $W = 5$, and the widths of the three cells are $w_1 = 1$, $w_2 = 2$, and $w_3 = 1$, respectively. The initial positions of the three cells are $x_1 = 0$, $x_2 = 0$, and $x_3 = 4$, respectively. The weights of the three cells are $e_1 = 1$, $e_2 = 1$, and $e_3 = 1$, respectively. There are four possible arrangements. The second one, which places C_1 at 0, C_2 at 1, and C_3 at 4, has the minimum cost among all the solutions. Thus, the minimum cost is 1, and the optimal positions are $x'_1 = 0$, $x'_2 = 1$, and $x'_3 = 4$.

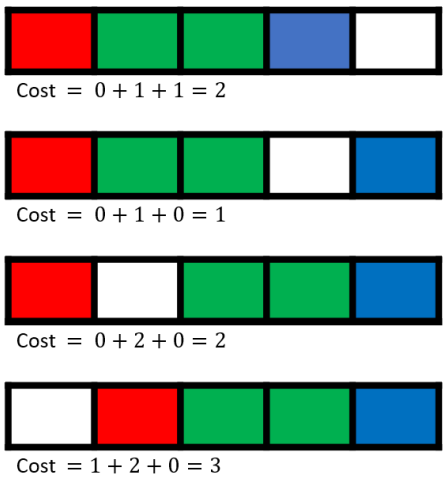

Figure 1: Cell Legalization.

Input

The input file consists of the following.

1. An integer W, $1 \leq W \leq 600000$, denoting the width of the row.
2. An integer n, $1 \leq n \leq 150000$, denoting the number of the cells.

3. One line, containing n integers $(w_1, ..., w_n)$, $1 \leq w_i \leq 600000$, denoting the width of each cell.
4. One line, containing n integers $(x_1, ..., x_n)$, $0 \leq x_i \leq 600000$, denoting the initial position of each cell.
5. One line, containing n integers $(e_1, ..., e_n)$, $1 \leq e_i \leq 9$, denoting the weight of each cell.

Output

The output file consists of the following.

1. An integer, denoting the minimum cost.
2. n lines, each line is an integer, denoting the position of each cell.

Here is an input/output example (see Table 1 and Figure 1):

Sample Input	Sample Output
5	1
3	0
1 2 1	1
0 0 4	4
1 1 1	

Table 1: Input/output example.

Command-line Parameter:

The executable binary must be named as **"cl"** and use the following command format.

```
./cl <input_file_name> <output_file_name>
```

For example, if you would like to run your binary for the input file `input_3.txt` and generate a solution named `output_3.txt`, the command is as follows:

```
./cl input_3.txt output_3.txt
```

Compilation:

We expect that your code can be compiled and run as follows. Type the following commands under <**student_id**>**_pa2/** directory,

```
make
./bin/cl inputs/<input_file_name> outputs/<output_file_name>
```

Required Files:

You need to create a directory named <**student_id**>**_pa2/** (*e.g.* b07901000_pa2/) **(the student ID should start with a lowercase letter)** which must contain the following documents:

- A directory named **src/** containing your source codes (*e.g.* `CellLegalization.cpp`): only `*.h`, `*.hpp`, `*.c`, `*.cpp` are allowed in src/, and no directories are allowed in src/;
- A directory named **bin/** containing your executable binary named **cl**;
- A directory named **doc/** containing your report;
- A makefile named **makefile** that produces an executable binary from your source codes by simply typing "make": the binary should be generated under the directory <student_id>_pa2/bin/;

- A text readme file named **README** describing how to compile and run your program.
- A report named **report.pdf** on the data structures used in your program and your findings in this programming assignment.

We will use our own test cases, so you do NOT need to submit the input files. Nevertheless, you should have at least the following items in your *.tgz file.

```
src/<all your source code>
bin/cl
doc/report.pdf
makefile
README
```

Submission Requirements:

1. Your program must be compilable and executable on EDA union servers.
2. The runtime limit for each test case is **1 minute**.
3. Please use the following command to compress your directory into a *.tgz* file:

   ```
   tar zcvf <filename>.tgz <your directory>
   ```

   The submission file should be named as <**student_id**>_**pa2.tgz** (*e.g.* b07901000_pa2.tgz). For example, if your student ID is *b07901000*, then you should use the command below.

   ```
   tar zcvf b07901000_pa2.tgz b07901000_pa2/
   ```

4. Please submit your *.tgz* file to the NTU COOL system before **1pm, April 17, 2022 (Sunday)**.
5. You are required to run the `checkSubmitPA2.sh` script to check if your *.tgz* submission file is correct. Suppose you are in the same level as the PA1 directory,

   ```
   bash checkSubmitPA2.sh <your submission>
   ```

   For example,

   ```
   bash checkSubmitPA2.sh b07901000_pa2.tgz
   ```

Language/Platform:

1. Language: C or C++.
2. Platform: Unix/Linux

Evaluation:
The individual score per test case is determined by the correctness of the output result and the file format. Bonus will be given to the most efficient programs. (Note that there are more hidden test cases for evaluating your program.) For any questions, please email Pan-Yang at b07901093@ntu.edu.tw. Thank you!