Paragraph Layout

input: A pair $I = (N, w[1..n])$ where N is positive integer, and $w[1..n]$ is an array of words (strings), each of length at most N.

output: A minimum-cost *layout* of w, of width at most N.

A layout breaks the sequence $w[1], w[2], \ldots, w[n]$ of words into *lines*, by inserting line breaks after some of the words. For example, suppose $N = 6$ and the given sequence of words is $W[1] =$ `aaa`, $W[2] =$ `bb`, $W[3] =$ `cc`, $W[4] =$ `dddd`.

Two possible layouts (each with three lines) are shown to the right. The *width* of each line is the number of characters in the line, including the spaces that separate the words. The three lines in Layout 1 have widths 6, 2, and 4. The three lines in Layout 2 have widths 3, 5, and 4.

layout 1	layout 2
`aaa.bb`	`aaa...`
`cc....`	`bb.cc.`
`dddd..`	`dddd..`

Each line in the layout must have width at most N. We define the *cost* of a layout as follows. For each line, define the *gap* of the line to be N minus the width. The three lines in Layout 1 have gaps 0, 4, and 2. The three lines in Layout 2 have gaps 3, 1, and 2. The cost of a layout is the sum, over its lines, of the gap of the line *squared*. Layout 1 has cost $0^2 + 4^2 + 2^2 = 20$. Layout 2 has cost $3^2 + 1^2 + 2^2 = 14$. The goal is to compute a layout of minimum cost.

Minimizing the cost tends to reduce the variation in the gaps, giving a better-looking layout.

Next we describe the algorithm. Fix any instance (N, w).

subproblems. For each $i \in \{0, 1, \ldots, n\}$, let $M[i]$ be the optimal cost for the subproblem $(N, w[1..i])$ formed by the first i words. The solution to the given problem $(N, w[1..n])$ is $M[n]$.

recurrence. First, define $\mathsf{width}(w[i..j])$ (for $i \le j$) to be the width of a line formed by words $w[i], w[i+1], \ldots, w[j]$. That is, $\mathsf{width}(w[i..j]) = \sum_{h=i}^{j}(|w[i]| + 1) - 1$. Then the gap of such a line is $N - \mathsf{width}(w[i..j])$.

Here is the recurrence for M. For $j = 0$, $M[j] = 0$. For $j \ge 1$, we have

$$M[j] = \min\left\{M[i-1] + \left(N - \mathsf{width}(w[i..j])\right)^2 : i \in \{1, \ldots, j\},\ \mathsf{width}(w[i..j]) \le N\right\}.$$

The intuition is as follows. Each possible layout for words $w[1..j]$ consists (for some i) of a layout for the words in $w[1..i-1]$, followed by one last line with the words in $w[i..j]$. Any $i \in \{1, 2, \ldots, j\}$ with $\mathsf{width}(w[i..j]) \le N$ gives such a layout.

time. There are $n + 1 = O(n)$ subproblems, and for each, evaluating the right-hand side of the recurrence takes time proportional to the maximum number of words that fit in the line. (This is at worst $O(\min(n, N))$.) So the total time is $O(n \min(n, N))$. In practice, the number of words that fit on a line is typically constant, in which case the run time will be $O(n)$.

correctness. To prove correctness, we would give a more careful proof that the recurrence relation is correct. Correctness of the algorithm would follow by induction (on the size of the subproblem).

This algorithm is equivalent to running the Shortest Path algorithm on a graph with a vertex v_j for each subproblem $M[j]$, and appropriately chosen weighted edges.

External resources on Paragraph Layout

- CLRS Chapter 15 Problem 15–4
- MIT lecture video

  https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-006-introduction-to-algorithms-fall-2011/lecture-videos/lecture-20-dynamic-programming-ii-text-justification-blackjack/