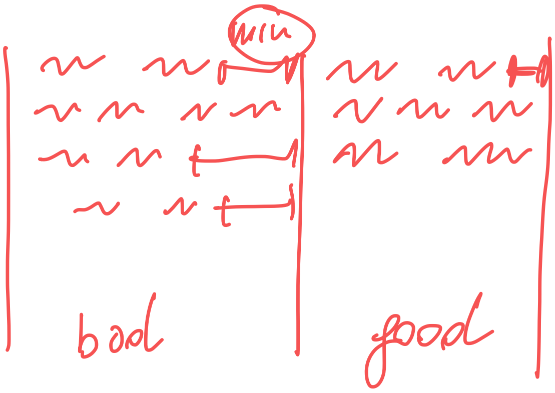
Dynamic programming : Text justification

1. Sequence of words
2. Limit on # char per line = page_width
3. put line breaks in the given sequence



Input: A given array of words $\ell = w[0, .., n-1]$

$|\ell| = n$

Scoring rule: suppose we are considering the words from i to j => $w[i]$ to $w[j]$

$w[i .. j]$ then we define the <u>badness</u> of a justification as

$$\text{badness}(i,j) = \begin{cases} +\infty & \text{if } \overbrace{j+1-i}^{\text{total length}} > \text{page\_width} \\ \underbrace{[\text{page\_width} - (j+1-i)]^2, \ \text{otherwise}}_{\text{latex rule (uses cubes)}} & \end{cases}$$

GOAL: split words into lines

$\ell_1 = w[0 .. i_1 - 1]$

$\ell_2 = w[i_1 .. i_2 - 1]$ to minimize $\sum_i$ badness$(\ell_i$

brute force algorithm $O(2^n)$

Subproblem structure

1. subproblem $c[i] = \min \underbrace{\text{badness}(i, n-1)}_{\substack{\text{suffix word} \\ w[i, \dots n-1]}}$

$\#\text{subproblem} = \Theta(n)$ where $n = \#\text{words}$

2. guessing = where to end the first line in the optimal justification of words $w[i, .., n-1]$

$\Rightarrow \#\text{choices} = n - i + 1 = O(n)$

3. Recurrence

$$c[i] = \min\big(\text{badness}(i, j) + c[j+1]\big)$$
$$\forall \; j \in [i, n-1]$$

Sequence: diamonds are girls best friends

page_width = 12

i		length
0	diamonds	8
1	are	3
2	girls	5
3	best	4
4	friends	7

Subproblem structure

1. subproblem $c[i] = \min \underbrace{\text{badness}(i, n-1)}_{\text{suffix word } w[i, \ldots n-1]}$

# subproblem = $\Theta(n)$ where $n = \#$ words

2. guessing = where to end the first line in the optimal justification of words $w[i, \ldots, n-1]$
$\Rightarrow$ #choices = $n - i + 1 = O(n)$

3. Recurrence

$$c[i] = \min\left(\text{badness}(i, j) + c[j+1]\right)$$
$$\forall \quad j \in [i, n-1]$$

Sequence: diamonds are girls best friends

i		length
0	diamonds	8
1	are	3
2	girls	5
3	best	4
4	friends	7

page_width = 12

i		length
0	diamonds	8
1	are	3
2	girls	5
3	best	4
4	friends	7

page_width = 12

$b(0,0)$ = badness if we store only $w[0]$ in l_1

Greedy approach

	badness
diamonds are	$\emptyset^2$
girls best	2^2
friends	5^2
	29

DP approach

diamonds	4^2
are girls	3^2
best friends	$\emptyset$
	25

$i \downarrow$ \ $j \rightarrow$	0	1	2	3	4
0	16	0	+∞	+∞	+∞
1	–	81	9	+∞	+∞
2	–	–	49	4	+∞
3	–	–	–	64	0
4	–	–	–	–	25

$i\ j$ ← first iter

min_costs c

0	1	2	3	4
25	9	29	0	25

index of c $c[\overset{\cancel{i}}{j}]$

0	1	2	3	4
1	3	4	5	5

1 3 5

0 to 1-1 → only $w[0]$

$w[1]$ to $w[3-1]$ → $w[1], w[2]$

$w[3]$ to $w[5-1]$ ⇒ $w[3], w[4]$

$b(0,4) = +\infty$ $\quad b(0,1) + c[2] = 29$

$b(0,3) = +\infty$ $\quad b(0,0) + c[1]$

$b(0,2) = +\infty$ $\quad = 16 + 9 = 25$

→ 2nd iter

$i \leftarrow i - 1$

$i = 3 \quad j = n - 1 = 4$

badness$(i,j) = b(3,4) = \emptyset$

→ 3rd iter

$i \leftarrow i - 1$

$i = 2$

$j = n - 1 = 4$

$j -= 1$

$b(2,4) = +\infty$

$b(2,3) + c(4) = 4 + 25 = 29$

$b(2,2) + c(3) = 49 + 0 = 49$

→ 4th iter

$b(1,4) = +\infty$ $\quad b(1,2) + c[3] = 9 + 0 = 9$

$b(1,3) = +\infty$ $\quad b(1,1) + c[2] = 81 + \quad >$

```
TEXT_JUSTIFICATION (W, page-width)
input: an array of words and p_w
output: minCost and index

let badness [0 .. len(w)-1 , 0 .. len(w)-1] be an empty
matrix
for i ← 0 to len(w)-1 do
    badness[i,i] ← page_width − len(w[i])
    for j ← i+1 to len(w)-1 do
        badness[i,j] ← badness[i, j-1] − len(w[j]) − 1

for i ← 0 to len(w)-1 do
    for j ← i to len(w)-1 do
        if badness[i,j] < 0 then
            badness[i,j] ← +∞
        else
            badness[i,j] ← $badness[i,j]^2$
// find the minCost by using the given recurrence
// min (badness (i,j) + c[j+1]), $\forall j \in [i, n-1]$
let minCost[0, .. n-1] ∧ index[0, .. n-1] be 2 empty
    arrays
for i ← n-1 to 0 do
    minCost[i] ← badness[i, n-1]
    index[i] ← n-1
    for j ← n-1 to i+1 do
        if badness[i, j-1] != +∞ do
            if minCost[i] > badness[i, j-1] + minCost[j] then
                minCost[i] ← badness[i, j-1] + minCost[j]
                index[i] ← j
return(minCosts, index)
```