

*LD 5800*

feb 11/14 2022

shelat

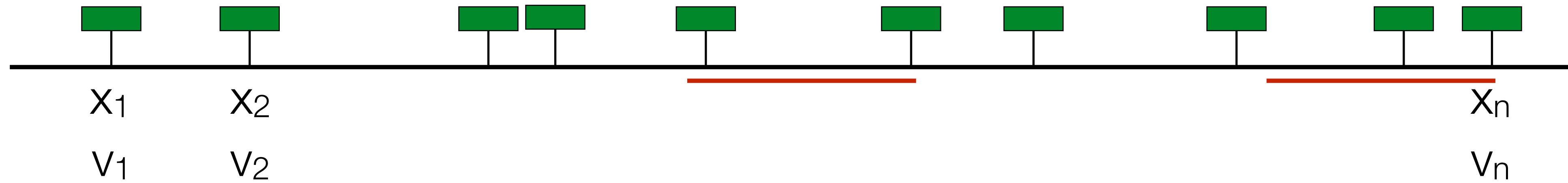


# Billboard problem





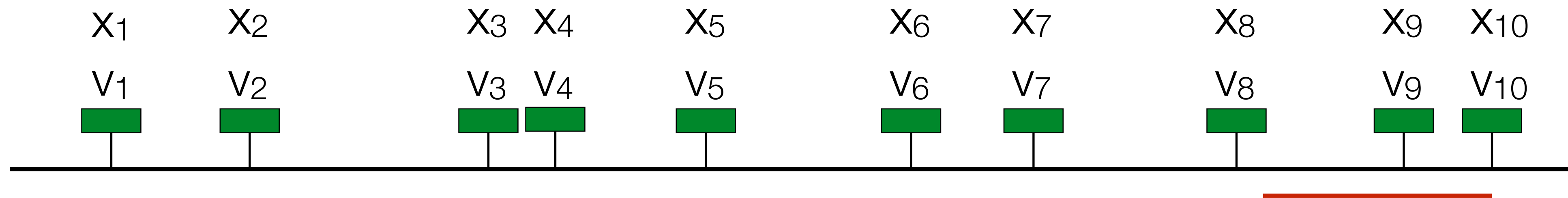
I-93



 distance parameter  
D Cannot place ads that are closer than D miles apart

I-93

————— D



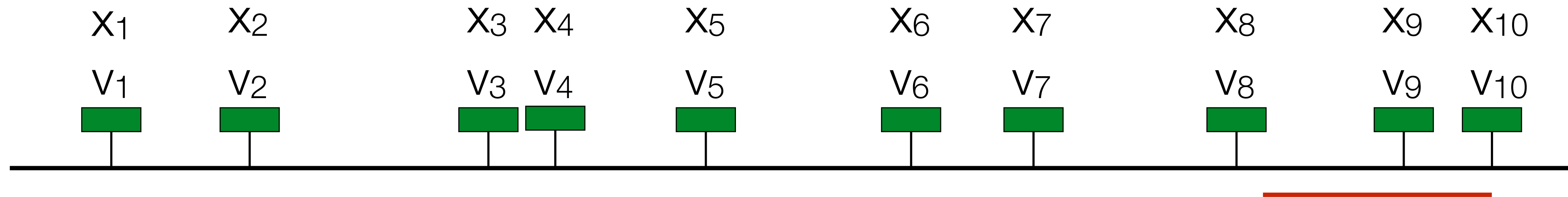
Input is  $((x_1, \dots, x_n)(v_1, \dots, v_n), D)$

Best<sub>n</sub> =



I-93

— D

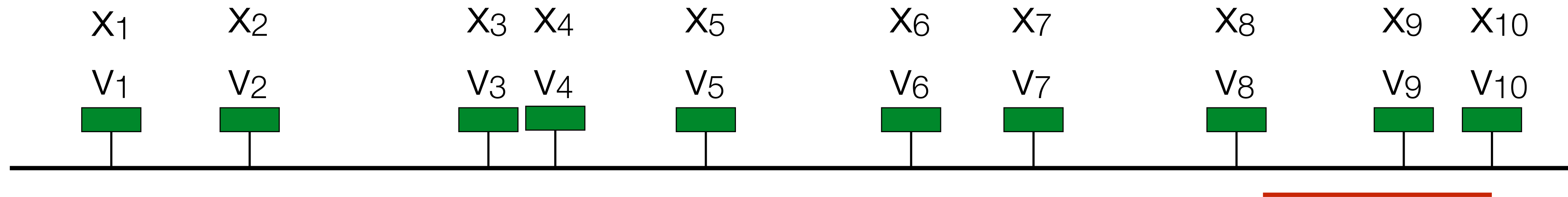


Input is  $((x_1, \dots, x_n)(v_1, \dots, v_n), D)$

Best<sub>n</sub> = Max viewers for a campaign that uses billboards {1...n} with separation D.

I-93

— D



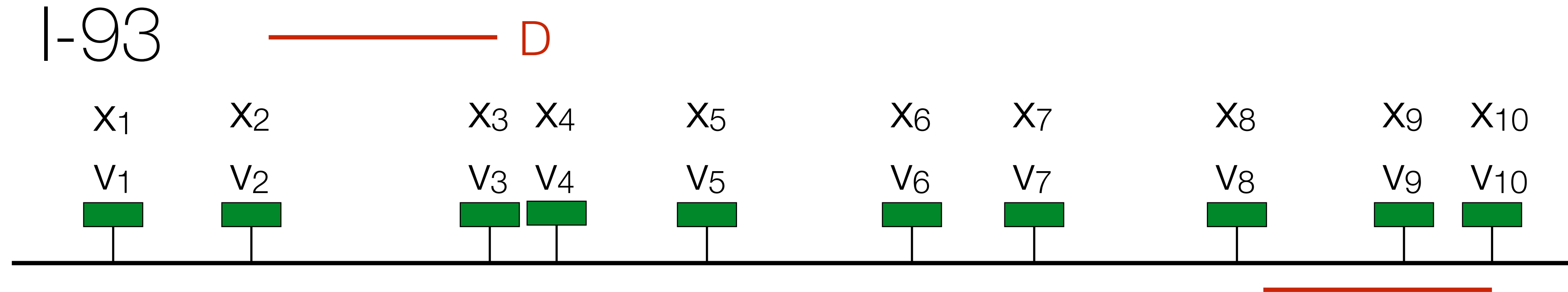
Input is  $((x_1, \dots, x_n)(v_1, \dots, v_n), D)$

$Best_n =$  Max viewers for a campaign that uses billboards  $\{1 \dots n\}$  with separation  $D$ .

$Best_n =$



| -93



Input is  $((x_1, \dots, x_n), (v_1, \dots, v_n), D)$

$Best_n =$  Max viewers for a campaign that uses billboards  $\{1 \dots n\}$  with separation  $D$ .

$$Best_n = \max \begin{cases} Best_{n-1} \\ v_n + Best_{closest_D(n)} \end{cases}$$

# Familiar?



# Familiar?

$Best_n =$

# Familiar?

$$Best_n = \max \begin{cases} Best_{n-1} \\ v_n + Best_{closest_D(n)} \end{cases}$$



# Familiar?

$$Best_n = \max \begin{cases} Best_{n-1} \\ v_n + Best_{closest_D(n)} \end{cases}$$

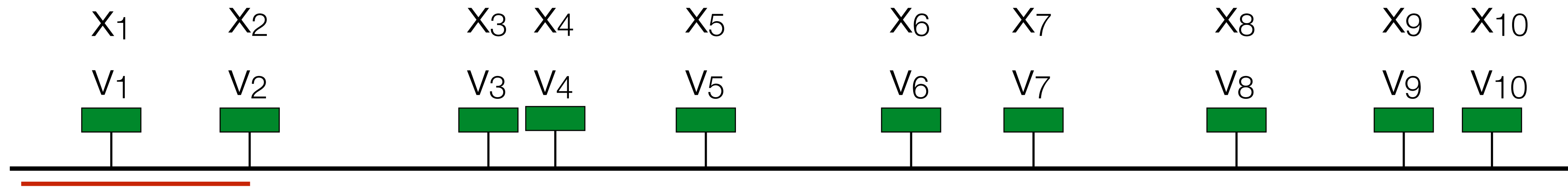
This equation is very similar to the log-cutter equation, with one difference.

We cannot simply use the price to pick the sub-problem, we have to use D:



I-93

————— D



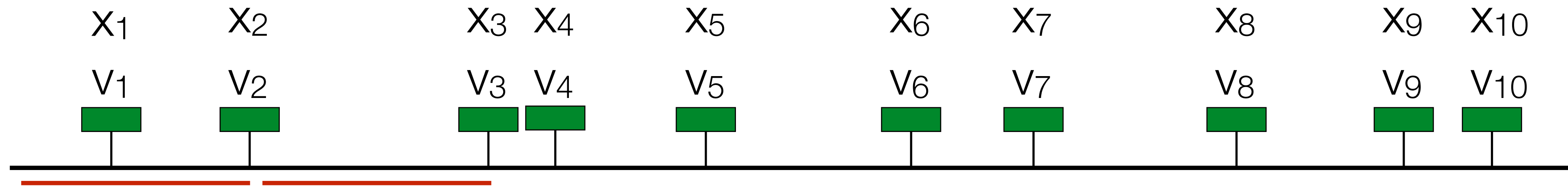
Best<sub>1</sub> =

Best<sub>2</sub> =



I-93

————— D



Best<sub>1</sub> =

Best<sub>2</sub> =

Best<sub>3</sub> =

# Billboard Problem

$$\text{BEST}_j = \max \begin{cases} \text{BEST}_{j-1} \\ v_j + \text{BEST}_{cl(j)} \end{cases}$$

`best[0] = 0`

`for i=1 to n`

`return best[n]`

# Billboard Problem

$$\text{BEST}_j = \max \begin{cases} \text{BEST}_{j-1} \\ v_j + \text{BEST}_{cl(j)} \end{cases}$$

```
best[0] = 0
```

```
for i=1 to n
```

```
    cl = i-1
```

```
    while( (x[i]-x[cl]) < D && cl > 0) cl=cl-1
```

```
    best[i] = max(best[i-1], v_i+best[cl])
```

```
return best[n]
```



# Billboard Problem

$$\text{BEST}_j = \max \begin{cases} \text{BEST}_{j-1} \\ v_j + \text{BEST}_{cl(j)} \end{cases}$$

```
best[0] = 0
```

```
for i=1 to n
```

```
    cl = i-1
```

```
    while( (x[i]-x[cl]) < D && cl > 0) cl = cl-1
```

```
    best[i] = max(best[i-1], v_i + best[cl])
```

```
return best[n]
```

This line can take  $\Theta(i)$  steps in the worst case.

Running time (worst case):  $\Theta(n^2)$

# Billboard Problem

$$\text{BEST}_j = \max \begin{cases} \text{BEST}_{j-1} \\ v_j + \text{BEST}_{cl(j)} \end{cases}$$

```
best[0] = 0
```

```
for i=1 to n
```

```
  cl = i-1
```

```
  while( (x[i]-x[cl]) < D && cl > 0) cl = cl-1
```

```
  best[i] = max(best[i-1], v_i + best[cl])
```

```
return best[n]
```

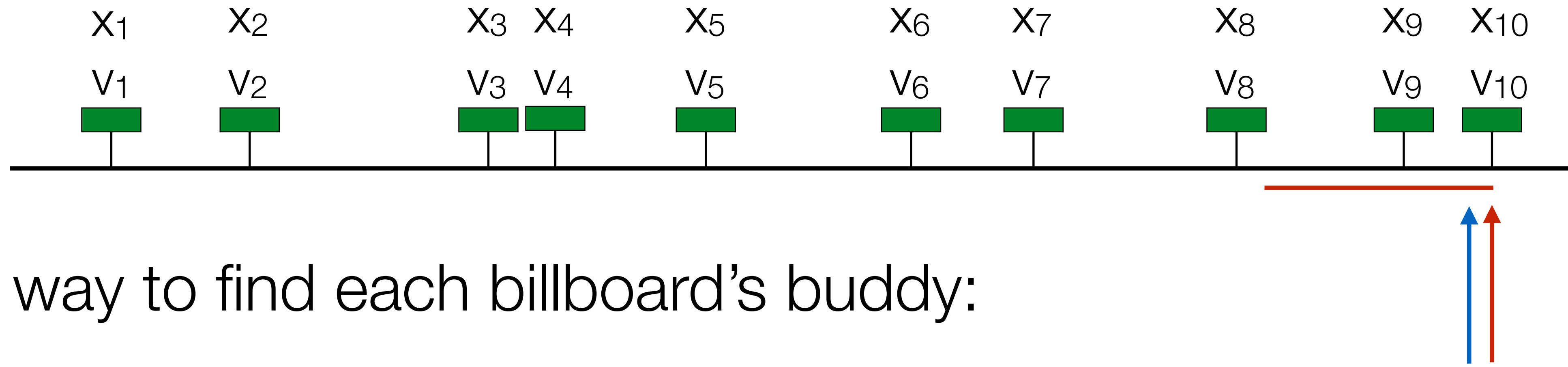
This line can take  $\Theta(i)$  steps in the worst case.

How can we improve?

Running time (worst case):  $\Theta(n^2)$

I-93

————— D



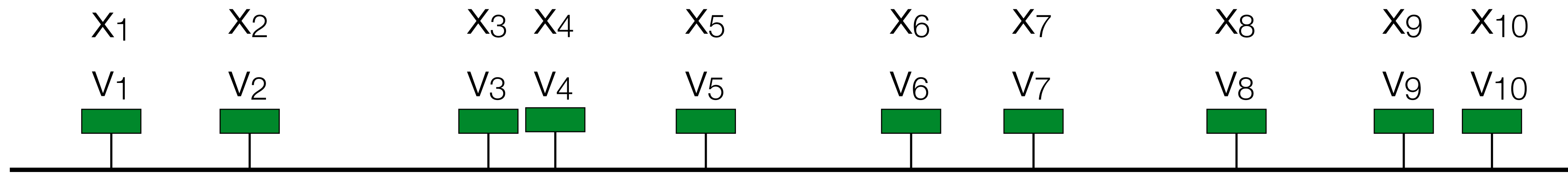
Faster way to find each billboard's buddy:

Pre-process to find every board's buddy.

right = n, left = n

I-93

————— D



Faster way to find each billboard's buddy:

Pre-process to find every board's buddy.

right = n, left = n

move left until  $\text{dist}(x[\text{right}], x[\text{left}]) > D$

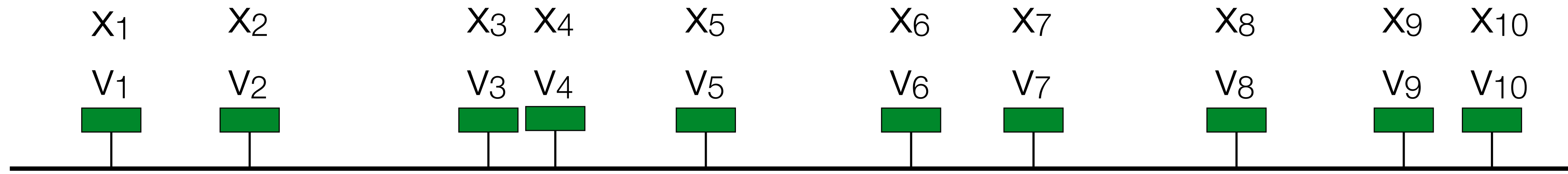
buddy[right] = left

b[10]=8



1-93

————— D



Faster way to find each billboard's buddy:

Pre-process to find every board's buddy.

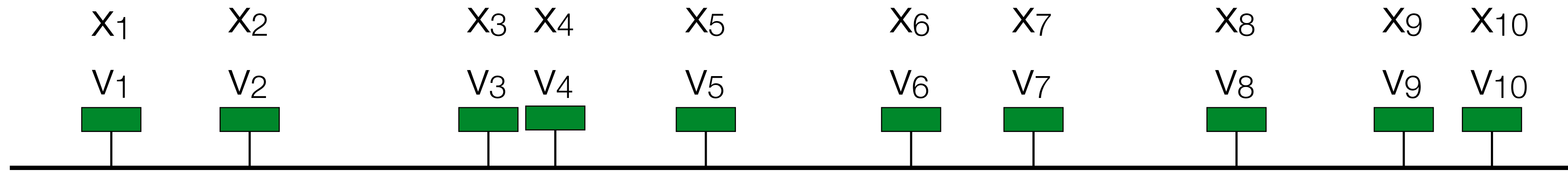
right = n, left = n

- move left until  $\text{dist}(x[\text{right}], x[\text{left}]) > D$
- buddy[right] = left
- move right to right

b[10]=8

I-93

————— D



Faster way to find each billboard's buddy:

Pre-process to find every board's buddy.

$right = n, left = n$

while  $right$  and  $left$  are valid

move  $left$  until  $dist(x[right], x[left]) > D$

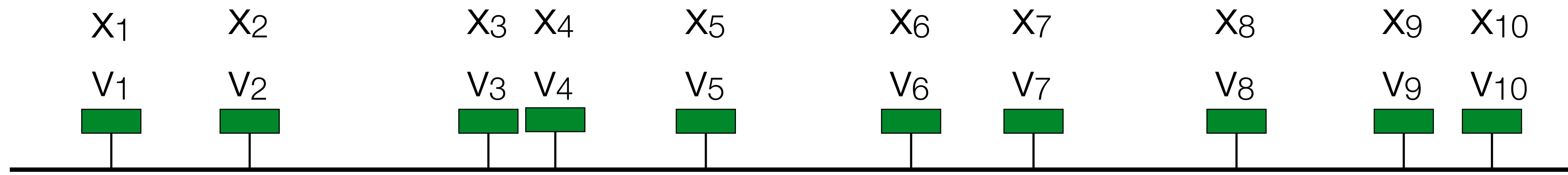
$buddy[right] = left$

move  $right$  to right

$b[10]=8$

I-93

————— D



Faster way to find each billboard's buddy:

Pre-process to find every board's buddy.

right = n, left = n

while right and left are valid

move left until  $\text{dist}(x[\text{right}], x[\text{left}]) > D$

buddy[right] = left

move right to right

handle all of the remaining buddies for right

b[10]=8

# Better Billboard

$$\text{BEST}_j = \max \begin{cases} \text{BEST}_{j-1} \\ v_j + \text{BEST}_{cl(j)} \end{cases}$$

<Preprocess buddies>

best[0] = 0

for i=1 to n

~~cl = i-1~~

~~while( (x[i]-x[cl]) < D && cl > 0) cl=cl-1~~



best[i] = max(best[i-1], v[j]+best[buddy[i]])

return best[n]

# Typesetting

It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity, it was the season of Light, it was the season of Darkness, it was the spring of hope, it was the winter of despair, we had everything before us, we had nothing before us, we were all going direct to heaven, we were all going direct the other way - in short, the period was so far like the present period, that some of its noisiest authorities insisted on its being received, for good or for evil, in the superlative degree of comparison only.



It was the best of times, it was the  worst of times, it was the age of wisdom, it was the  age of foolishness, it was the epoch of belief, it was the epoch of incredulity, it was the season of Light, it was the season of Darkness, it was the spring of hope, it was the winter of despair, we had everything before us, we had nothing before us, we were all going direct to heaven, we were all going direct the other way - in short, the period was so far like the present period, that some of its noisiest authorities insisted on its being received, for good or for evil, in the superlative degree of comparison only.

# First rule of typesetting

never print in the margin!

 are simply not allowed

It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity, it was the season of Light, it was the season of Darkness, it was the spring of hope, it was the winter of despair, we had everything before us, we had nothing before us, we were all going direct to heaven, we were all going direct the other way - in short, the period was so far like the present period, that some of its noisiest authorities insisted on its being received, for good or for evil, in the superlative degree of comparison only.

\_\_\_\_\_ is

It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity, it was the season of Light, it was the season of Darkness, it was the spring of hope, it was the winter of despair, we had everything before us, we had nothing before us, we were all going direct to heaven, we were all going direct the other way - in short, the period was so far like the present period, that some of its noisiest authorities insisted on its being received, for good or for evil, in the superlative degree of comparison only.

\_\_\_\_\_ is

Penalty is the square of the total slack.

It was the best of times, it was the worst  
 of times, it was the age of wisdom, it was  
 the age of foolishness, it was the epoch\_\_\_  
 of belief, it was the epoch of \_\_\_\_\_  
 incredulity, it was the season of Light, \_\_  
 it was the season of Darkness, it was the\_  
 spring of hope, it was the winter of \_\_\_\_\_  
 despair, we had everything before us, we \_\_  
 had nothing before us, we were all going \_\_  
 direct to heaven, we were all going direct  
 the other way - in short, the period was  
 so far like the present period, that some  
 of its noisiest authorities insisted on  
 its being received, for good or for evil,  
 in the superlative degree of comparison only

0	0
0	0
2	4
12	144
2	4
1	1
6	36
2	4
2	4
0	0
	197

Greedy fails: The first  
 two lines are perfect,  
 but the 4th line has  
 large slack.



It was the best of times, it was the \_\_\_\_\_  
 worst of times, it was the age of wisdom, \_\_\_\_\_  
 it was the age of foolishness, it was the \_\_\_\_\_  
 epoch of belief, it was the epoch of \_\_\_\_\_  
 incredulity, it was the season of Light, \_\_\_\_\_  
 it was the season of Darkness, it was the \_\_\_\_\_  
 spring of hope, it was the winter of \_\_\_\_\_  
 despair, we had everything before us, we \_\_\_\_\_  
 had nothing before us, we were all going \_\_\_\_\_  
 direct to heaven, we were all going direct  
 the other way - in short, the period was  
 so far like the present period, that some  
 of its noisiest authorities insisted on  
 its being received, for good or for evil,  
 in the superlative degree of comparison only

6	36
1	1
1	1
6	36
2	4
1	1
6	36
2	4
2	4
0	0
	123

A better solution evens out the slack between the first and 4th line.

# Typesetting problem

input:

output:

such that

# Typesetting problem

input:  $W = \{w_1, w_2, w_3, \dots, w_n\}$   $M$

output:  $L = (w_1, \dots, w_{\ell_1-1}), (w_{\ell_1}, \dots, w_{\ell_2-1}), (w_{\ell_2}, \dots, w_{\ell_3-1}), \dots, (w_{\ell_k}, \dots, w_n)$

such that

# Typesetting problem

input:  $W = \{w_1, w_2, w_3, \dots, w_n\}$   $M$   
Length of each word

output:  $L = (w_1, \dots, w_{\ell_1-1}), (w_{\ell_1}, \dots, w_{\ell_2-1}), (w_{\ell_2}, \dots, w_{\ell_3-1}), \dots, (w_{\ell_k}, \dots, w_n)$   
First words of each line

such that  $c_i = \left( \sum_{j=\ell_i}^{\ell_{i+1}-1} w_j \right) + (\ell_{i+1} - \ell_i - 1)$  Chars on each line

$c_i \leq M \quad \forall i$  No line over margin

$\min \sum (M - \overline{c_i})^2$  Minimize the slack<sup>2</sup>

# how to solve

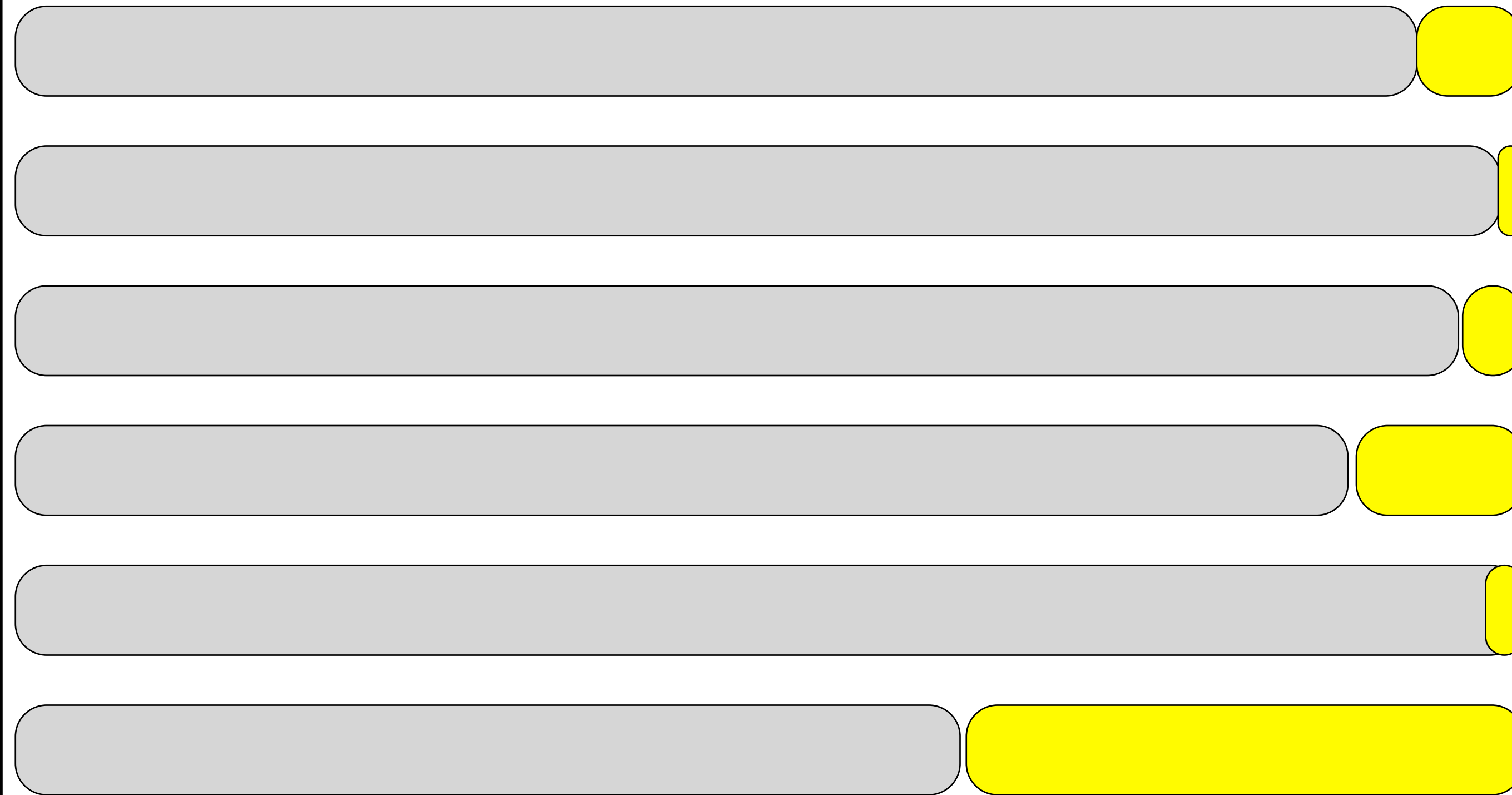
define the right variable:

Imagine optimal solution





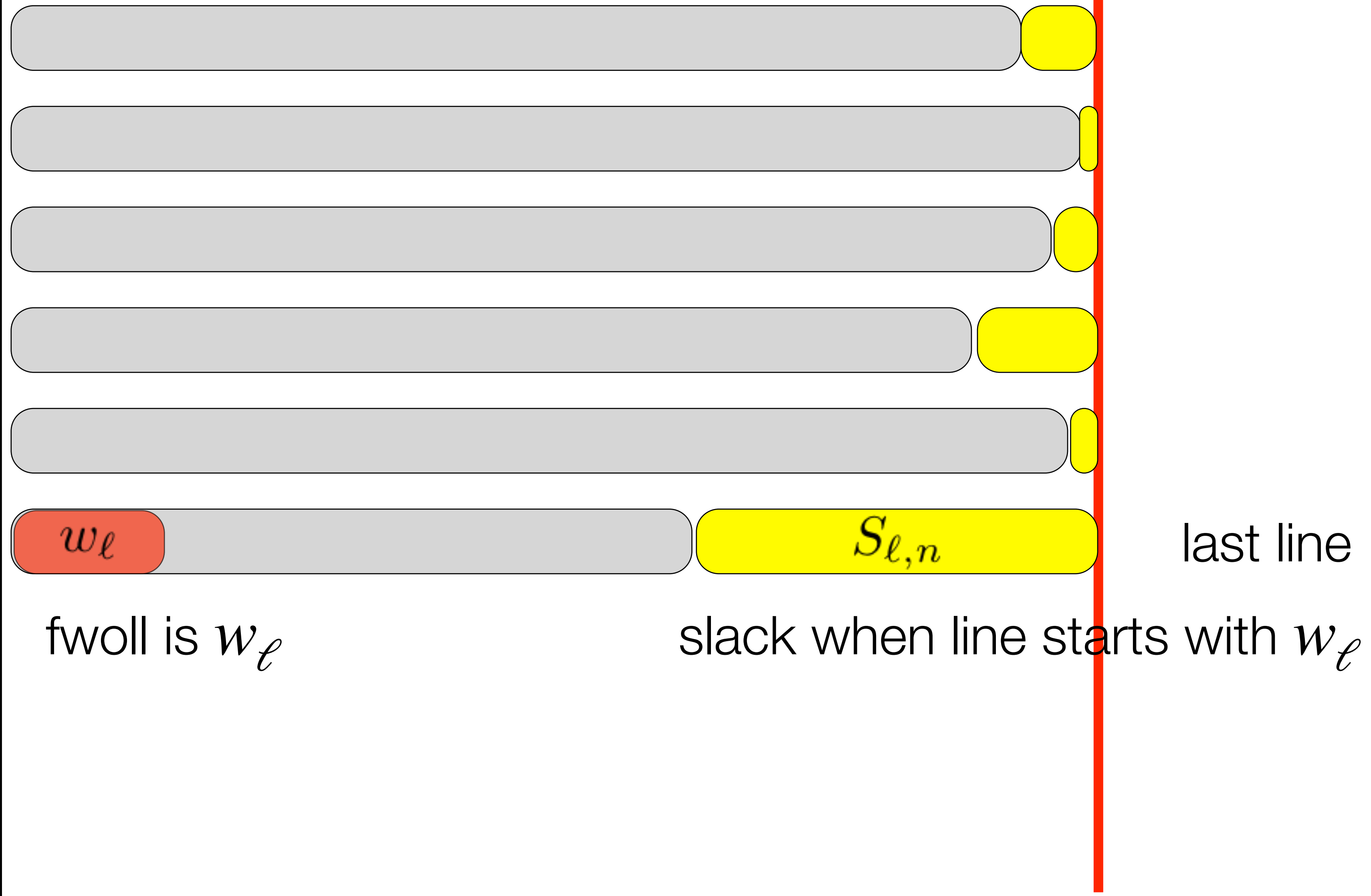
# Imagine optimal solution



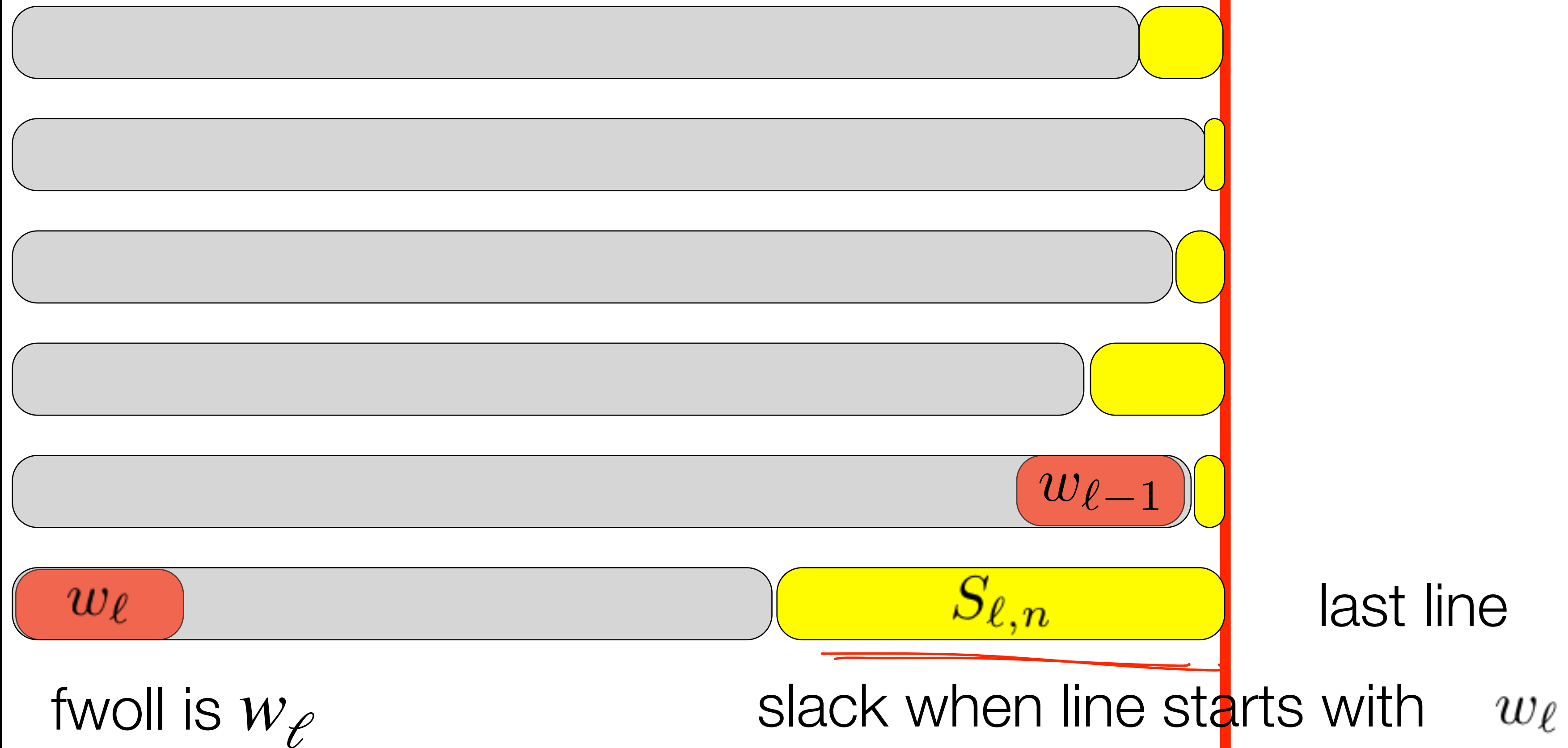
last line

Some word has to be  
the first-word-of-last-line  
(fwoll)

# Imagine optimal solution



# Imagine optimal solution



$$\text{BEST}_n = \text{BEST}_{\ell-1} + S_{\ell,n}^2$$

How many candidates  
are there for the fwooll?

# Is $w_i$ fwoll?

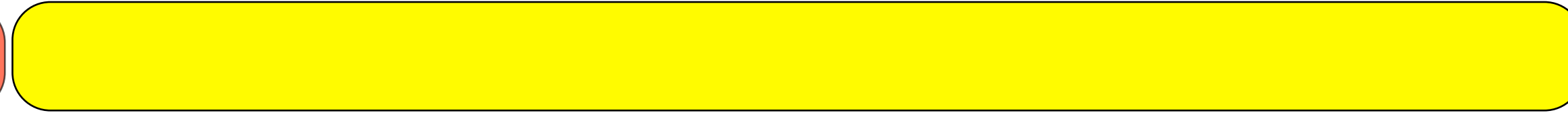
$w_1$

there is no slack (no solution even)  
because words go beyond edge!

define  $S_{1,n} = \infty$  if this happens

# Is $w_2$ fwoll?

$w_1$



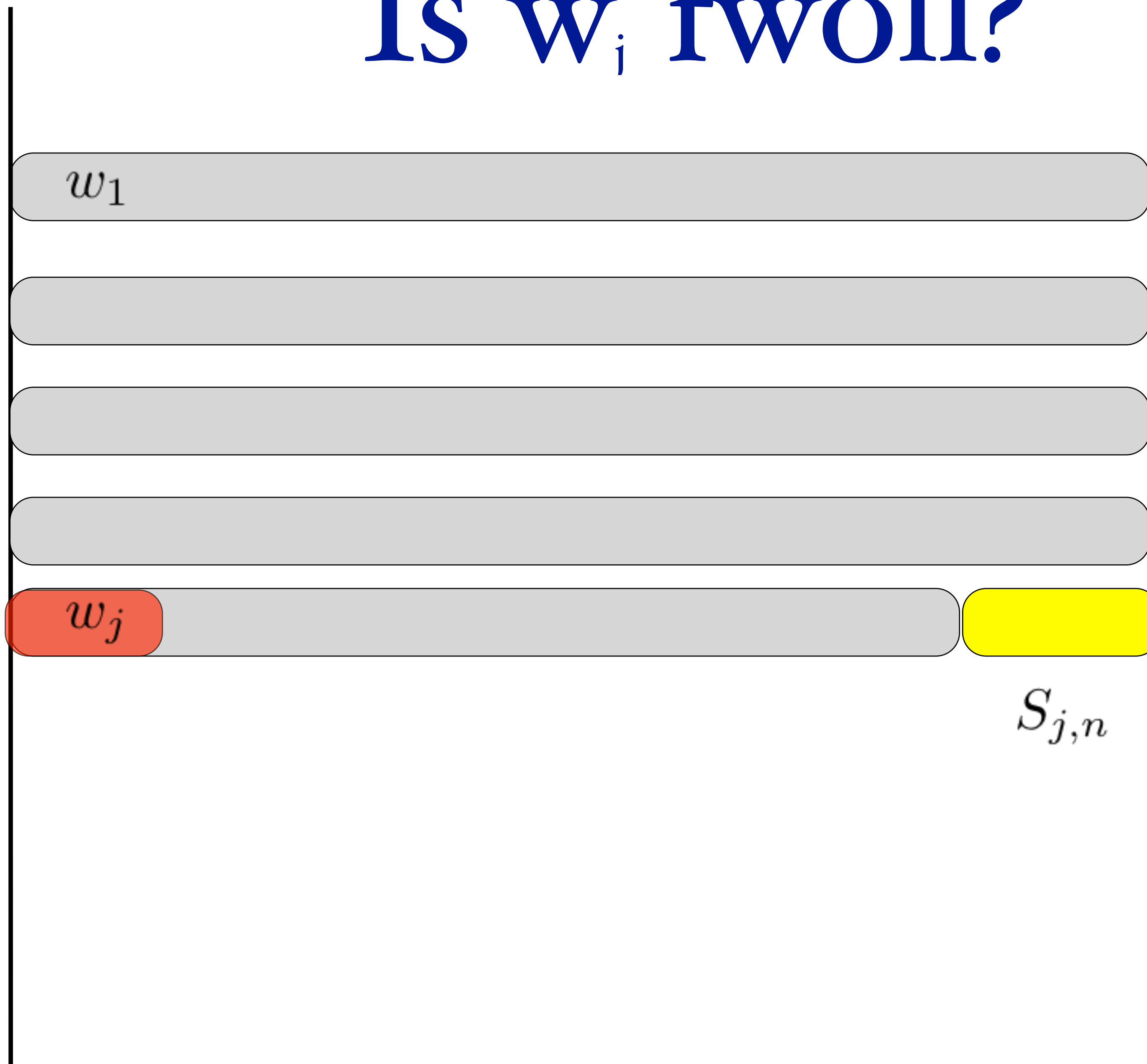
$w_2$



$$S_{2,n} = \infty$$

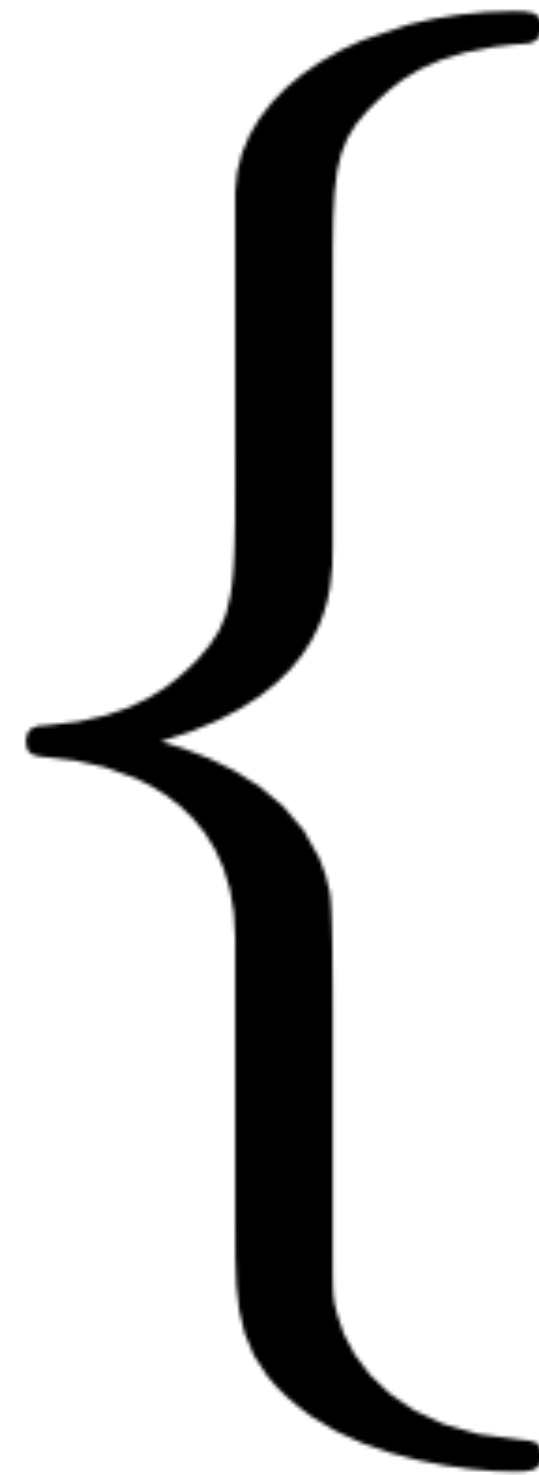


# Is $w_i$ fwoll?



# Which word is fwoll?

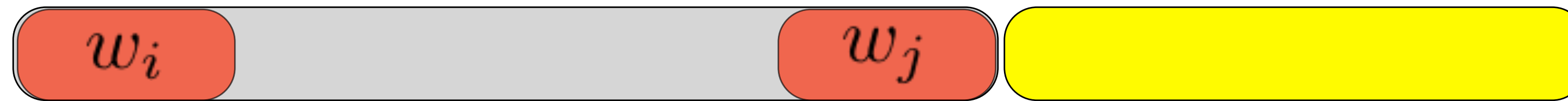
$\text{BEST}_n = \min$



# Which word is fwoll?

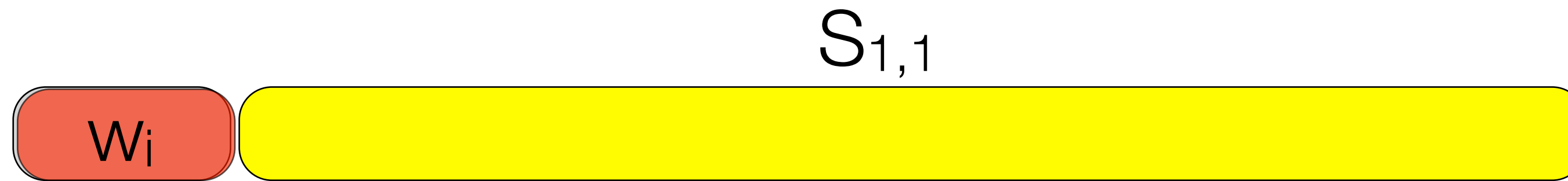
$$\text{BEST}_n = \min \left\{ \begin{array}{l} \text{BEST}_0 + S_{1,n}^2 \\ \text{BEST}_1 + S_{2,n}^2 \\ \text{BEST}_2 + S_{3,n}^2 \\ \dots \\ \text{BEST}_{\ell-1} + S_{\ell,n}^2 \\ \dots \\ \text{BEST}_{n-1} + S_{n,n}^2 \end{array} \right.$$

# How to compute $S_{i,j}$



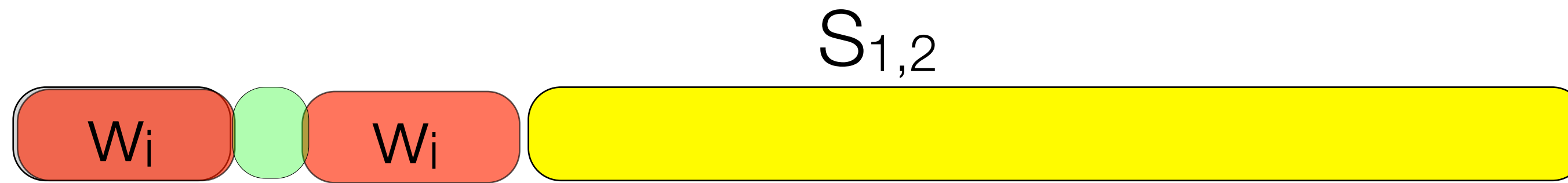
slack when line  
starts with  $w_i$   
and ends  $w_j$

# Simplest case



slack when line  
starts with  $w_i$   
and ends  $w_i$

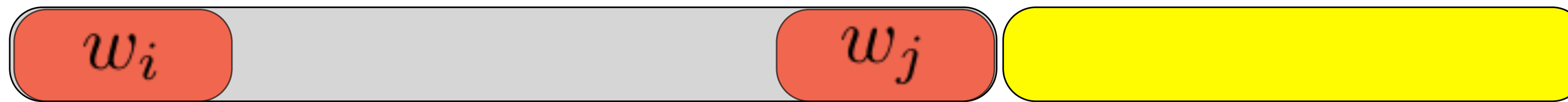
# Simplest case



slack when line  
starts with  $w_i$   
and ends  $w_2$

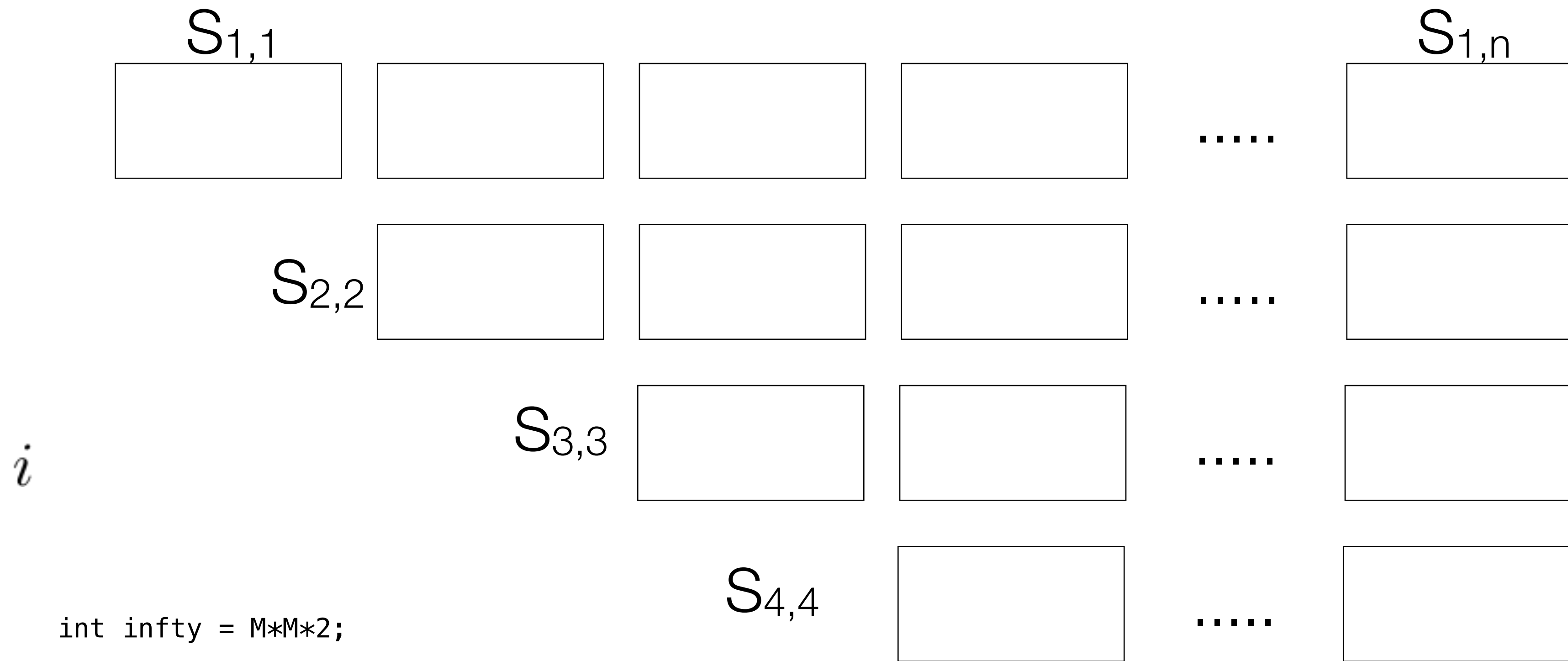
# how to compute $S_{i,j}$

$$S_{i,j}$$



slack when line  
starts with  $w_i$   
and ends  $w_j$





```
int infty = M*M*2;
```

```
// compute S_ij
```

```
int S[][] = new int[n+1][n+1];
```

```
for(int i=1; i<=n; i++) {
```

```
    S[i][i] = M - lens[i];
```

```
    for(int j=i+1; j<=n; j++) {
```

```
        S[i][j] = S[i][j-1] - lens[j] - 1;
```

```
        if (S[i][j]<0) {
```

```
            while(j<=n) { S[i][j++] = infty; }
```

```
        }
```

```
    }
```

```
}
```

# Typesetting algorithm

make table for  $S_{i,j}$

# Typesetting algorithm

make table for  $S_{i,j}$

for  $i=1$  to  $n$

$best[i] = \min\{ best[j] + s[j+1][i]^2 \}$

```
// compute best_0, ..., best_n
int best[] = new int[n+1];
int choice[] = new int[n+1];
best[0] = 0;
for(int i=1; i<=n; i++) {
    int min = infty;
    int ch = 0;
    for(int j=0; j<i; j++) {
        int t = best[j] + S[j+1][i]*S[j+1][i];
        if (t<min) { min = t; ch = j;}
    }
    best[i] = min;
    choice[i] = ch;
}
```

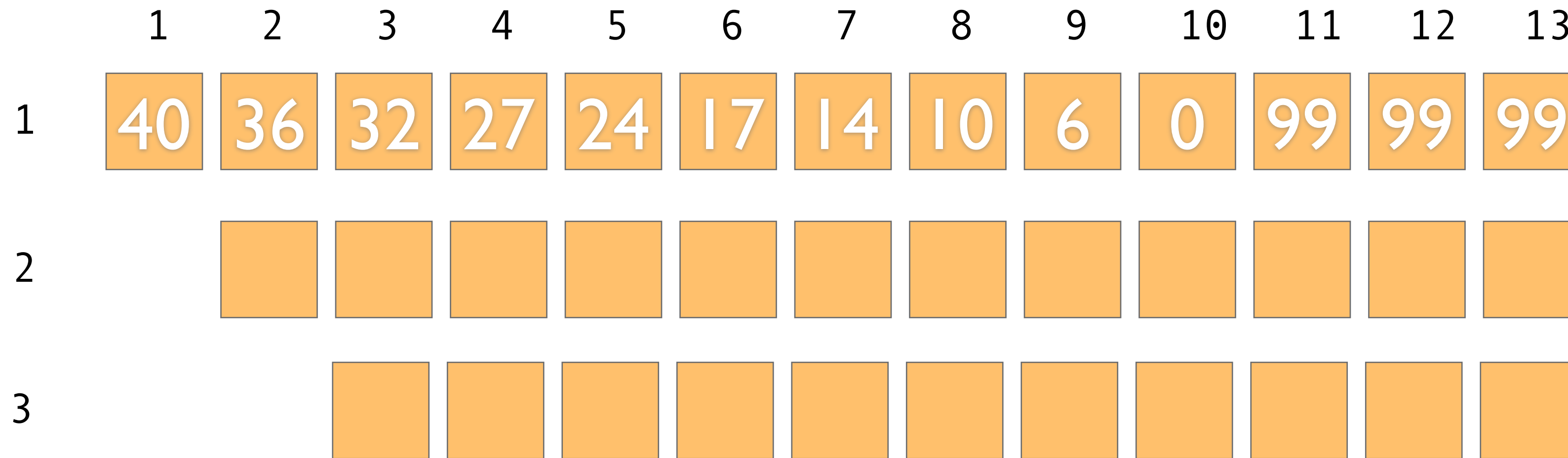
# Example

It was the best of times, it was the worst of times; it was the age of wisdom, it was the age of foolishness; it was the epoch of belief, it was the epoch of incredulity; it was the season of

2 3 3 4 2 6 2 3 3 5 2 6 2 3 3 3 2 7 2 3 3 3  
2 12 2 3 3 5 2 7 2 3 3 5 2 12 2 3 3 6 2



# First step: make $S_{i,j}$



2	3	3	4	2	6	2	3	3	5	2	6	2	3	3	3	2	7	2	3	3	3
2	12	2	3	3	5	2	7	2	3	3	5	2	12	2	3	3	6	2			

$$S_{i,i} = M - |w_i|$$

$$S_{i,j} = S_{i,j-1} - 1 - |w_j|$$

$M = 42$

# First step: make $S_{i,j}$

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	40	36	32	27	24	17	14	10	6	0	99	99	99
2		39	35	30	27	20	17	13	9	3	0	99	99
3													

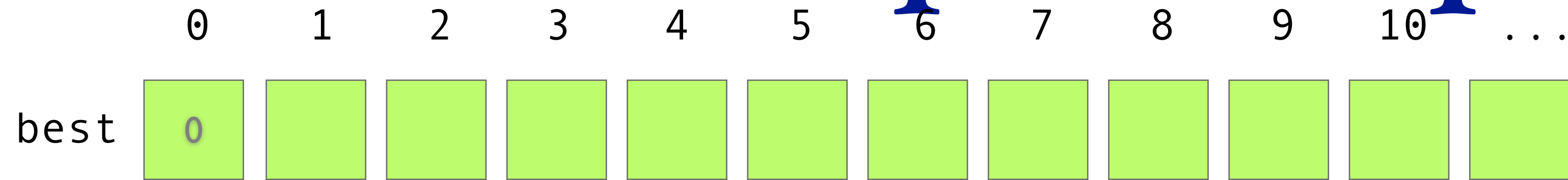
2 3 3 4 2 6 2 3 3 5 2 6 2 3 3 3 2 7 2 3 3 3  
 2 12 2 3 3 5 2 7 2 3 3 5 2 12 2 3 3 6 2

$$S_{i,i} = M - |w_i|$$

$$S_{i,j} = S_{i,j-1} - 1 - |w_j|$$



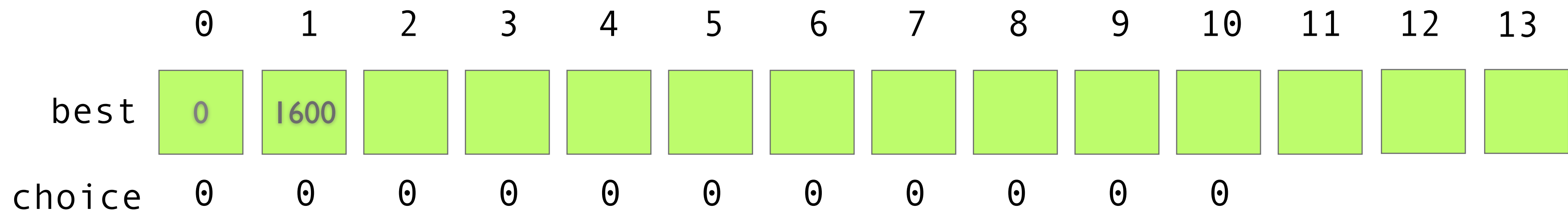
# second step: compute



$$\text{BEST}_i = \min_{j=0}^{i-1} \left\{ \text{BEST}_j + S_{j+1,i}^2 \right\}$$

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	40	36	32	27	24	17	14	10	6	0	99	99	99
2		39	35	30	27	20	17	13	9	3	0	99	99

$$\text{BEST}_i = \min_{j=0}^{i-1} \{ \text{BEST}_j + S_{j+1,i}^2 \}$$



aa



$S_{i,j}$

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	40	36	32	27	24	17	14	10	6	0	99	99	99
2		39	35	30	27	20	17	13	9	3	0	99	99

$$\text{BEST}_i = \min_{j=0}^{i-1} \{ \text{BEST}_j + S_{j+1,i}^2 \}$$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
best	0	1600	1296											
choice	0	0	0	0	0	0	0	0	0	0	0			

aa

It was \_\_\_\_\_

$S_{i,j}$

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	40	36	32	27	24	17	14	10	6	0	99	99	99
2		39	35	30	27	20	17	13	9	3	0	99	99

$$\text{BEST}_i = \min_{j=0}^{i-1} \{ \text{BEST}_j + S_{j+1,i}^2 \}$$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
best	0	1600	1296	1024										
choice	0	0	0	0	0	0	0	0	0	0	0			

aa

It was the

$S_{i,j}$

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	40	36	32	27	24	17	14	10	6	0	99	99	99
2		39	35	30	27	20	17	13	9	3	0	99	99

$$\text{BEST}_i = \min_{j=0}^{i-1} \{ \text{BEST}_j + S_{j+1,i}^2 \}$$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
best	0	1600	1296	1024	729	576	289	196	100	36	0			
choice	0	0	0	0	0	0	0	0	0	0	0			

aa

It was the best of times, it was the worst

$S_{i,j}$

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	40	36	32	27	24	17	14	10	6	0	99	99	99
2		39	35	30	27	20	17	13	9	3	0	99	99

$$\text{BEST}_i = \min_{j=0}^{i-1} \{ \text{BEST}_j + S_{j+1,i}^2 \}$$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
best	0	1600	1296	1024	729	576	289	196	100	36	0			
choice	0	0	0	0	0	0	0	0	0	0	0			

aa

It was the best of times, it was the worst  
of \_\_\_\_\_

$$\text{Best}_{11} = \min \{$$

$$\text{BEST}_i = \min_{j=0}^{i-1} \{ \text{BEST}_j + S_{j+1,i}^2 \}$$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
best	0	1600	1296	1024	729	576	289	196	100	36	0			
choice	0	0	0	0	0	0	0	0	0	0	0			

aa

It was the best of times, it was the

worst of \_\_\_\_\_

$\text{best}_9 + S_{10,11}^2$

$$\text{BEST}_i = \min_{j=0}^{i-1} \{ \text{BEST}_j + S_{j+1,i}^2 \}$$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
best	0	1600	1296	1024	729	576	289	196	100	36	0			
choice	0	0	0	0	0	0	0	0	0	0	0			

aa

It was the best of times, it was

the worst of \_\_\_\_\_

$\text{best}_8 + S_{9,11}^2$



$$\text{BEST}_i = \min_{j=0}^{i-1} \{ \text{BEST}_j + S_{j+1,i}^2 \}$$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
best	0	1600	1296	1024	729	576	289	196	100	36	0	818		
choice	0	0	0	0	0	0	0	0	0	0	0			

aa

It was the best of times,

it was the worst of \_\_\_\_\_

$$\text{best}_6 + S_{7,11}^2$$

$$\text{BEST}_{11} = \min \left\{ \begin{array}{l} \text{BEST}_{10} + S_{11,11}^2 \\ \text{BEST}_9 + S_{10,11}^2 \\ \text{BEST}_8 + S_{9,11}^2 \\ \text{BEST}_7 + S_{8,11}^2 \\ \text{BEST}_6 + S_{7,11}^2 \\ \dots \end{array} \right.$$

This break is the best one for the first 11 words.

$$\text{BEST}_i = \min_{j=0}^{i-1} \{ \text{BEST}_j + S_{j+1,i}^2 \}$$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
best	0	1600	1296	1024	729	576	289	196	100	36	0	818	545	
choice	0	0	0	0	0	0	0	0	0	0	0	6	6	

aa

It was the best of times,

it was the worst of times, it

14

$$\text{BEST}_{13} = \min \left\{ \begin{array}{l} \text{BEST}_{12} + S_{13,13}^2 \\ \text{BEST}_{11} + S_{12,13}^2 \\ \dots \\ \text{BEST}_7 + S_{8,13}^2 \\ \text{BEST}_6 + S_{7,13}^2 \end{array} \right.$$

$$\text{best}_6 + S_{7,13}^2$$

$$\text{BEST}_i = \min_{j=0}^{i-1} \{ \text{BEST}_j + S_{j+1,i}^2 \}$$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
best	0	1600	1296	1024	729	576	289	196	100	36	0	818	545	
choice	0	0	0	0	0	0	0	0	0	0	0	6	6	

aa

It was the best of times, it  
 was the worst of times, it

---

16

$$\text{best}_7 + S_{8,13}^2$$

$$\text{BEST}_{13} = \min \left\{ \begin{array}{l} \text{BEST}_{12} + S_{13,13}^2 \\ \text{BEST}_{11} + S_{12,13}^2 \\ \dots \\ \text{BEST}_7 + S_{8,13}^2 \\ \text{BEST}_6 + S_{7,13}^2 \end{array} \right.$$

0 best: 0 ch 0  
1 best: 1600 ch 0  
2 best: 1296 ch 0  
3 best: 1024 ch 0  
4 best: 729 ch 0  
5 best: 576 ch 0  
6 best: 289 ch 0  
7 best: 196 ch 0  
8 best: 100 ch 0  
9 best: 36 ch 0  
10 best: 0 ch 0  
11 best: 818 ch 6  
12 best: 545 ch 6  
13 best: 452 ch 7  
14 best: 340 ch 7  
15 best: 244 ch 8  
16 best: 164 ch 8  
17 best: 117 ch 9  
18 best: 37 ch 9  
19 best: 16 ch 10  
20 best: 0 ch 10  
21 best: 509 ch 14  
22 best: 413 ch 15  
23 best: 344 ch 15  
24 best: 133 ch 17  
25 best: 118 ch 17  
26 best: 62 ch 18

It  
It was  
It was the  
It was the best  
It was the best of  
It was the best of times,  
It was the best of times, it  
It was the best of times, it was  
It was the best of times, it was the  
It was the best of times, it was the worst  
It was the best of times, \nit was the worst of  
It was the best of times, \nit was the worst of times,  
It was the best of times, it \nwas the worst of times, it  
It was the best of times, it \nwas the worst of times, it was  
It was the best of times, it was \nthe worst of times, it was the  
It was the best of times, it was \nthe worst of times, it was the age  
It was the best of times, it was the \nworst of times, it was the age of  
It was the best of times, it was the \nworst of times, it was the age of wisdom,  
It was the best of times, it was the worst \nof times, it was the age of wisdom, it  
It was the best of times, it was the worst \nof times, it was the age of wisdom, it was  
It was the best of times, it \nwas the worst of times, it was \nthe age of wisdom, it was the  
It was the best of times, it was \nthe worst of times, it was the \nage of wisdom, it was the age  
It was the best of times, it was \nthe worst of times, it was the \nage of wisdom, it was the age of  
It was the best of times, it was the \nworst of times, it was the age of \nwisdom, it was the age of foolishness,  
It was the best of times, it was the \nworst of times, it was the age of \nwisdom, it was the age of foolishness, it  
It was the best of times, it was the \nworst of times, it was the age of wisdom, \nit was the age of foolishness, it was

d-172-25-159-219:typeset abhi\$ java typeset charly 42

0 best: 0 ch 0  
1 best: 1600 ch 0  
2 best: 1296 ch 0  
3 best: 1024 ch 0  
4 best: 729 ch 0  
5 best: 576 ch 0  
6 best: 289 ch 0  
7 best: 196 ch 0  
8 best: 100 ch 0  
9 best: 36 ch 0  
10 best: 0 ch 0  
11 best: 818 ch 6  
12 best: 545 ch 6  
13 best: 452 ch 7  
14 best: 340 ch 7  
15 best: 244 ch 8  
16 best: 164 ch 8  
17 best: 117 ch 9  
18 best: 37 ch 9  
19 best: 16 ch 10  
20 best: 0 ch 10  
21 best: 509 ch 14  
22 best: 413 ch 15  
23 best: 344 ch 15  
24 best: 133 ch 17  
25 best: 118 ch 17  
26 best: 62 ch 18  
27 best: 32 ch 19  
28 best: 4 ch 20  
29 best: 444 ch 23  
30 best: 348 ch 23  
31 best: 277 ch 24  
32 best: 197 ch 24  
33 best: 149 ch 24  
34 best: 87 ch 26  
35 best: 66 ch 26  
36 best: 446 ch 31  
37 best: 377 ch 31  
38 best: 297 ch 32  
39 best: 233 ch 32

```
// read input
```

```
try {  
BufferedReader bin = new BufferedReader(new FileReader(args[0]));  
String line = bin.readLine();  
String words[] = line.split(" ");  
int n = words.length;  
int M = Integer.parseInt(args[1]);  
int lens[] = new int[n+1];  
for(int i=1;i<=n; i++) {  
    lens[i] = words[i-1].length();  
    if (lens[i]>M) {  
        System.out.println("word too long");  
        System.exit(1);  
    }  
}  
}
```

```
int infity = M*M*2;
```

```
// compute S_ij
```

```
int S[][] = new int[n+1][n+1];  
for(int i=1;i<=n;i++) {  
    S[i][i] = M - lens[i];  
    for(int j=i+1; j<=n; j++) {  
        S[i][j] = S[i][j-1] - lens[j] - 1;  
        if (S[i][j]<0) {  
            while(j<=n) { S[i][j++] = infity; }  
        }  
    }  
}  
}
```

```

// compute best_0,...,best_n
int best[] = new int[n+1];
int choice[] = new int[n+1];
best[0] = 0;
for(int i=1;i<=n;i++) {
    int min = infity;
    int ch = 0;
    for(int j=0;j<i;j++) {
        int t = best[j] + S[j+1][i]*S[j+1][i];
        if (t<min) { min = t; ch = j;}
    }
    best[i] = min;
    choice[i] = ch;
}

```

```

// backtrack to output linebreaks
int end = n;
int start = choice[end]+1;
String lines[] = new String[n];
int cnt = 0;
while (end>0) {
    StringBuffer buf = new StringBuffer();
    for(int j=start; j<=end; j++) {
        buf.append(words[j-1] + " ");
    }
    lines[cnt++] = buf.toString();
    end = start-1;
    start = choice[end]+1;
}

```