Correction: HW3 Oct 4

L9

4102

Sep 24 2013

abhi shelat

Dynamic programming: log cutter, matrix chains, typesetting

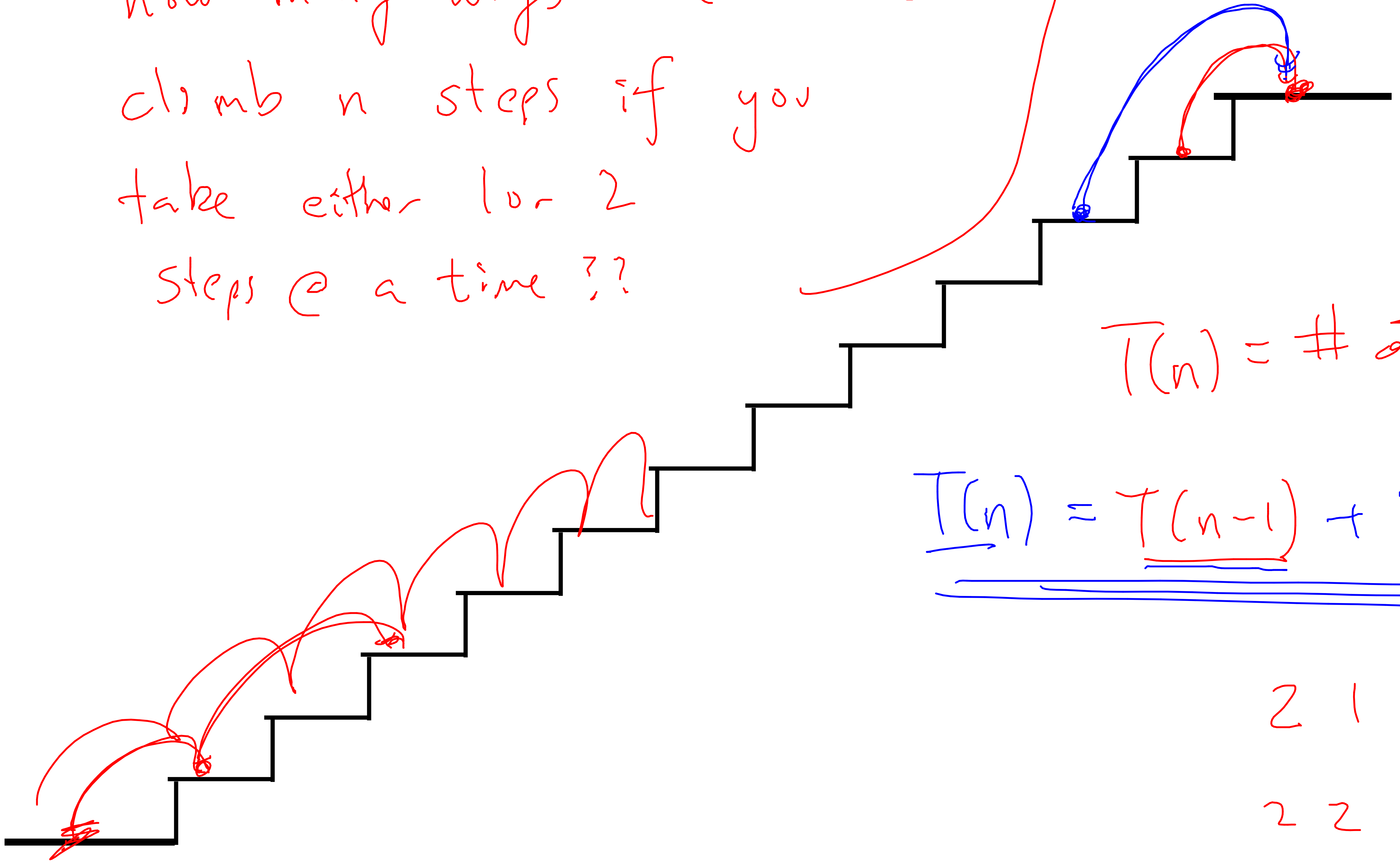
What are the inputs and outputs of the FFT algorithm?

Describe the algorithm in a few sentences.

Do you remember any applications of the FFT?

Name:

how many ways are there to climb n steps if you take either 1 or 2 steps @ a time ??



$n = 98$

$T(n)$ = # of ways to climb n steps.

$$T(n) = T(n-1) + T(n-2)$$

Recurrence.

Fibonacci

2 1 1 1 1 1

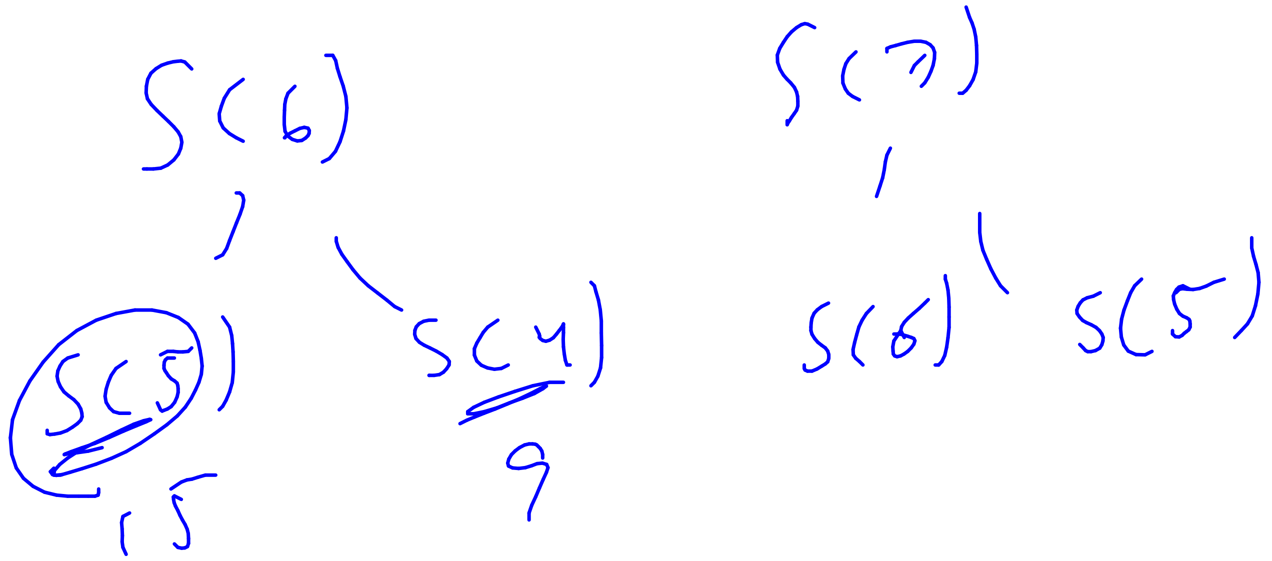
2 2 1 1 1 1

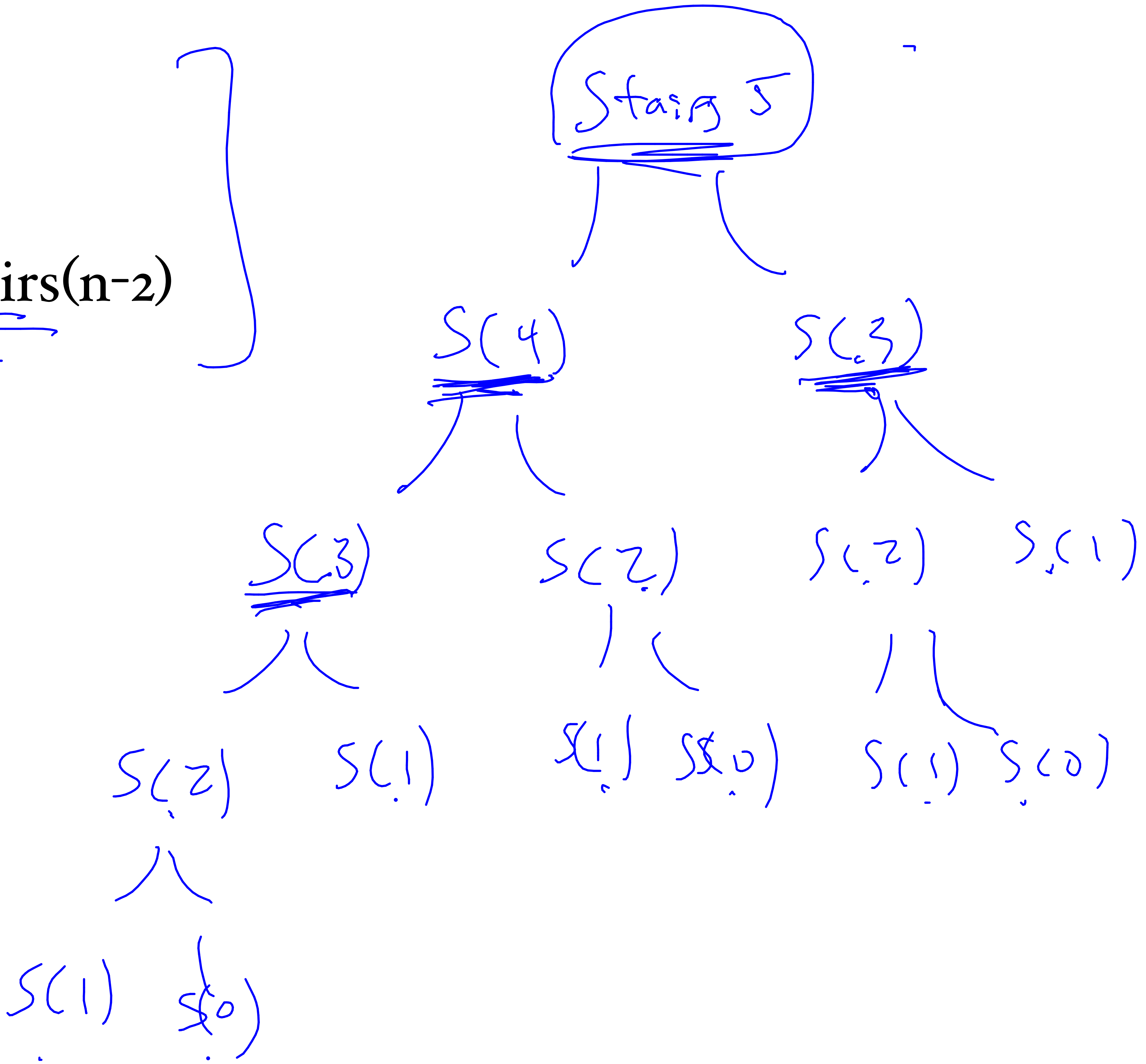
1 2 1 1 1

$\sim \phi^n$

golden ratio

```
Stairs(n)
    if n<=1 return 1

    return Stairs(n-1) + Stairs(n-2)
```





15 recursive calls.

$S(n) \sim S(n-1) + S(n-2) \rightarrow$ Fibonacci number $\sim O(\phi^n)$

Stairs(n) if n<=1 return 1
ret Stairs(n-1) + Stairs(n-2)

Stairs(5)

Stairs(4)

Stairs(3)

```
Stairs(n)  if n<=1 return 1
           ret Stairs(n-1) + Stairs(n-2)
```

Stairs(5)

Stairs(4) Stairs(3)

Stairs(3) Stairs(2) Stairs(2) Stairs(1)

Stairs(n) if n<=1 return 1
ret Stairs(n-1) + Stairs(n-2)

Stairs(5)

Stairs(4) Stairs(3)

Stairs(3) Stairs(2) Stairs(2) Stairs(1)

Stairs(2) Stairs(1) Stairs(1) Stairs(0) Stairs(1) Stairs(0) initialize memory M

Stairs(n)

Base case as before

if M contains n, return M[n]

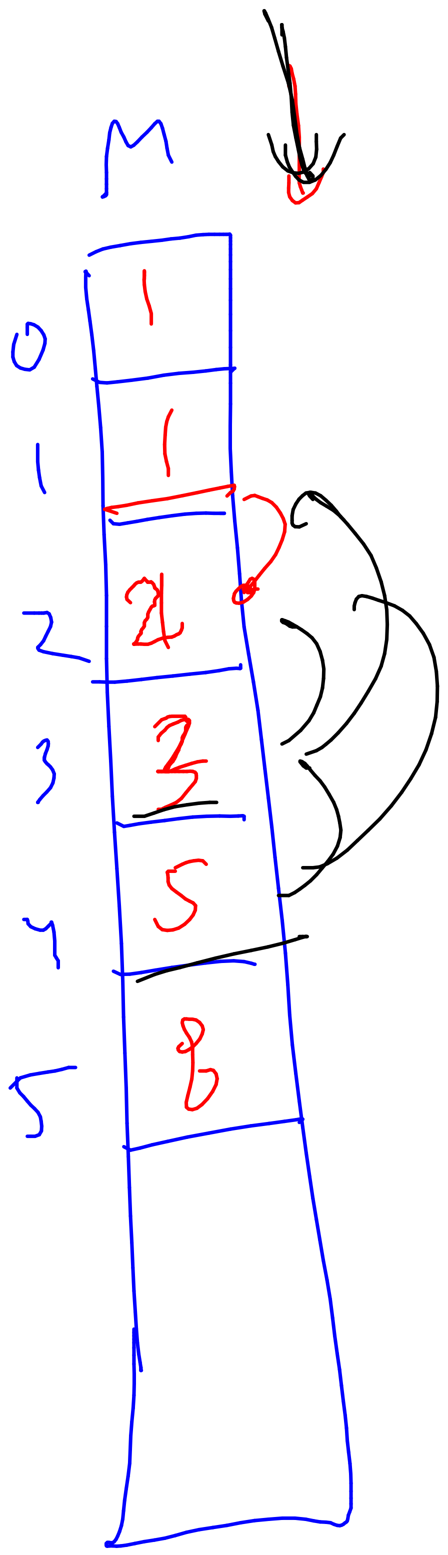
else answer = Stairs(n-1) + Stairs(n-2)

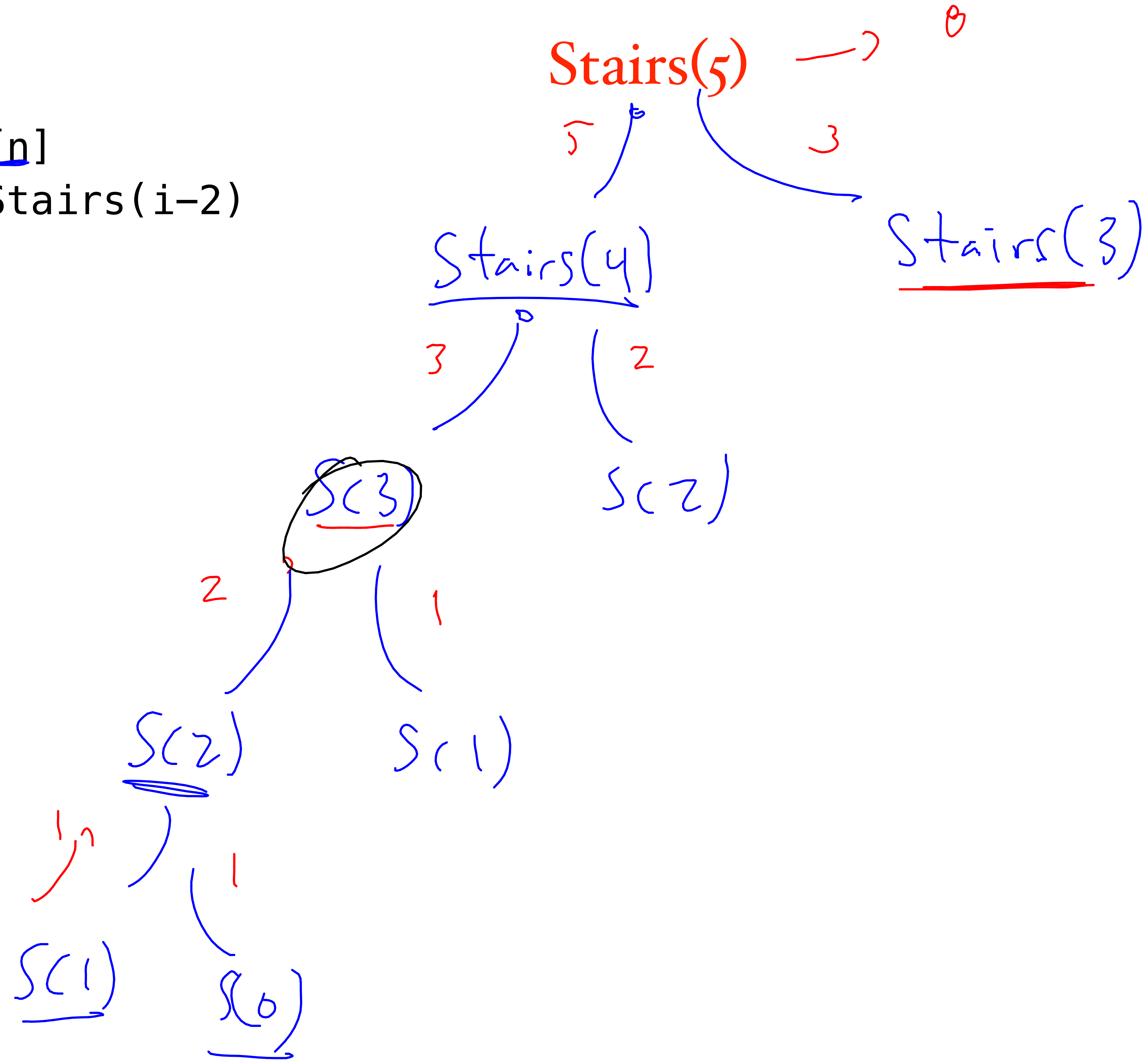
M[n] = answer

return answer

```
    initialize memory M

Stairs(n)
   if n<=1 then return n
   if n is in M, return M[n]
   answer = Stairs(i-1)+ Stairs(i-2)
   M[n] = answer
   return answer
```

```
Stairs(n)
    if n<=1 then return 1
    if n is in M, return M[n]
    answer = Stairs(i-1)+ Stairs(i-2)
    M[n] = answer
    return answer
```





Stairs(n)

```
stair[0]=1
stair[1]=1
for(i=2, to n)
    stairs[i]=stairs[i-1] + stairs[i-2]
return stairs[n]
```

Stairs(n)

```
stair[0]=1
stair[1]=1
for i=2 to n
  stair[i] = stair[i-1]+stair[i-2]
return stair[i]
```

Dynamic Programming

two big ideas

(1) recursive substructure.

$$T(n) = T(n-1) + T(n-2)$$

(2) memoization

keep track of intermediate results,

solve the intermediate problems in a specific order to maximize efficiency

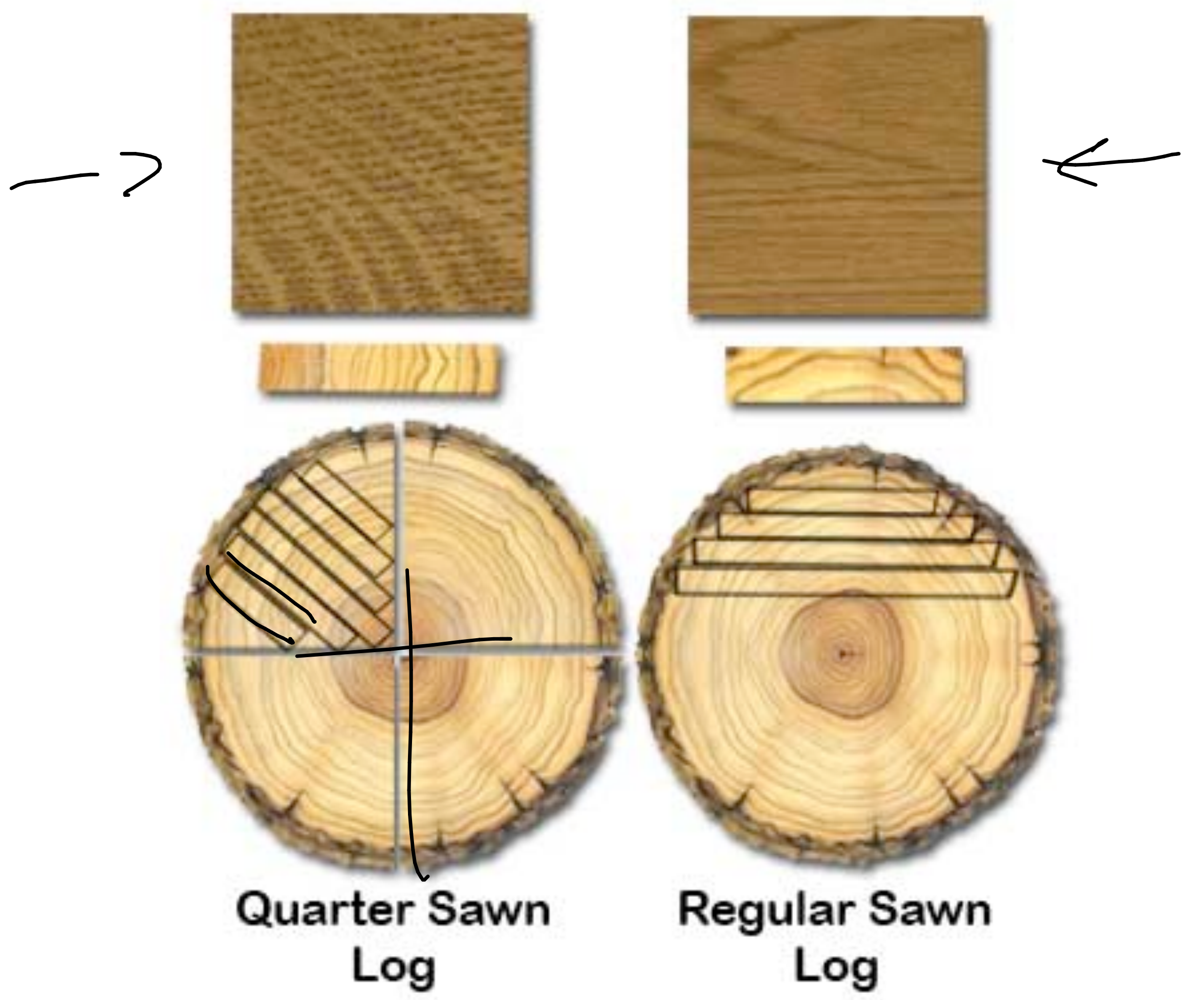
two big ideas

recursive structure

+

memoizing

wood cutting



http://www.amishhandcraftedheirlooms.com/quarter-sawn-oak.htm

http://snlm.files.wordpress.com/2008/08/bill-wakefield-and-carl-fie.gif

Spot price for lumber

1” 2” 3” 4” 5” 6” 7” 8”

p_1 p_2 p_3 p_4 . . . p_8

$p_i \rightarrow$ spot price for an i''-wide slab of lumber

$n = 5$

p_1 p_4

p_2 p_3

p_5

$n = 200''$

Log cutter dilemna

input to the problem: $n, (p_1, \ldots, p_n)$

"n" wide log

spot prices for slabs of width "i"

goal: MAXIMIZE profits!!

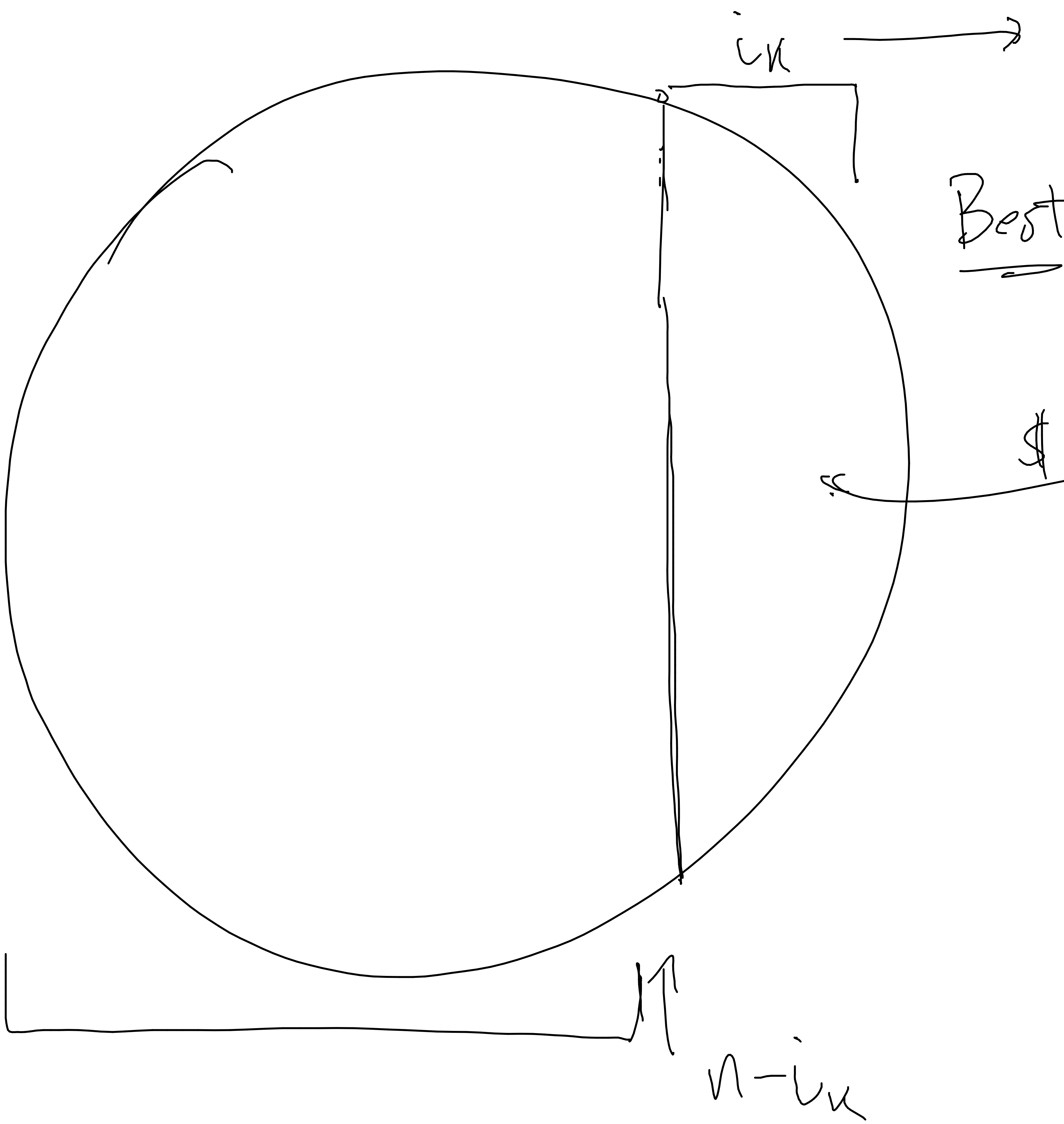
find a set of cuts $i_1\ i_2\ i_3 \ldots i_k$

$$\sum_{j=0}^{k} i_j \leq n$$

$$\max \sum_{j=0}^{k} p_{i_j}$$

Observation

i_k → best move in the optimal solution



$$Best_n = P_{i_k} + Best_{n-i_k}$$

Solution equation

$B_n = P_{ik} + B_{n-ik}$

how many possible values of ik are there ??

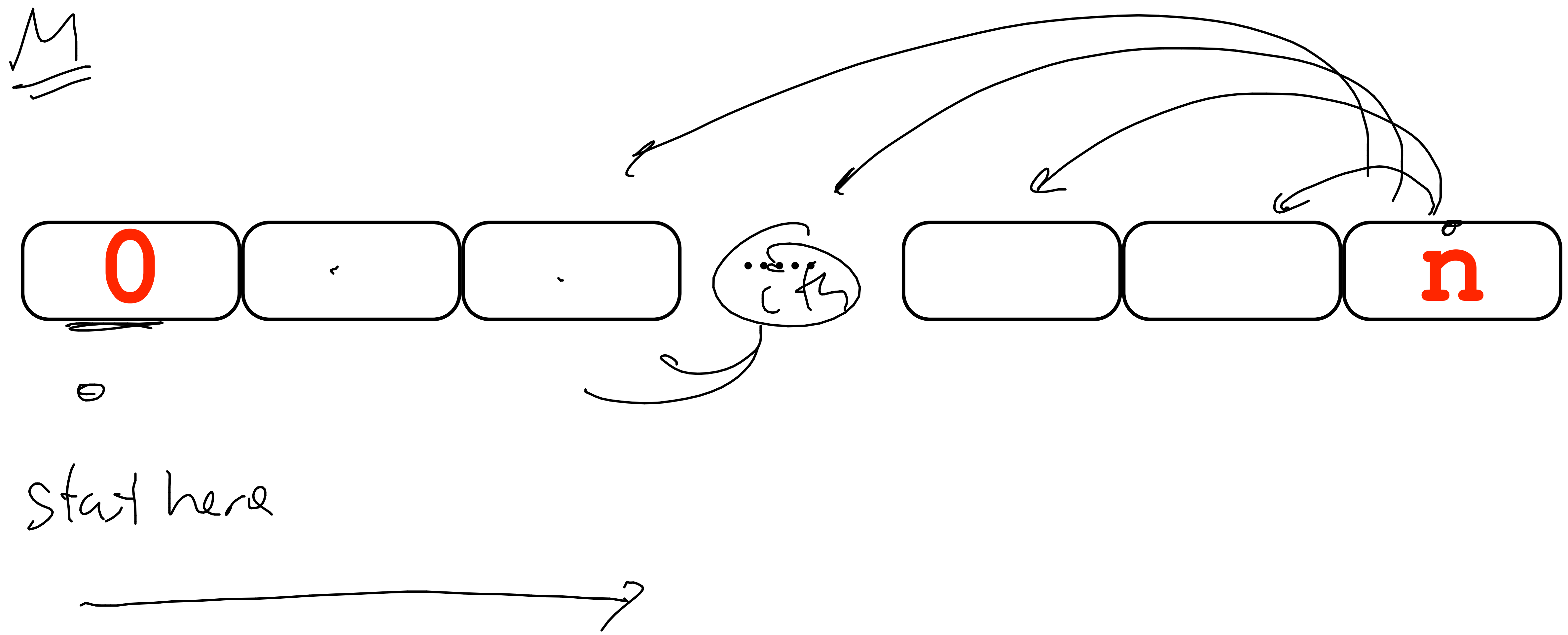
n

$$B_n = \max_{i=1}^{n} \{ P_i + B_{n-i} \}$$

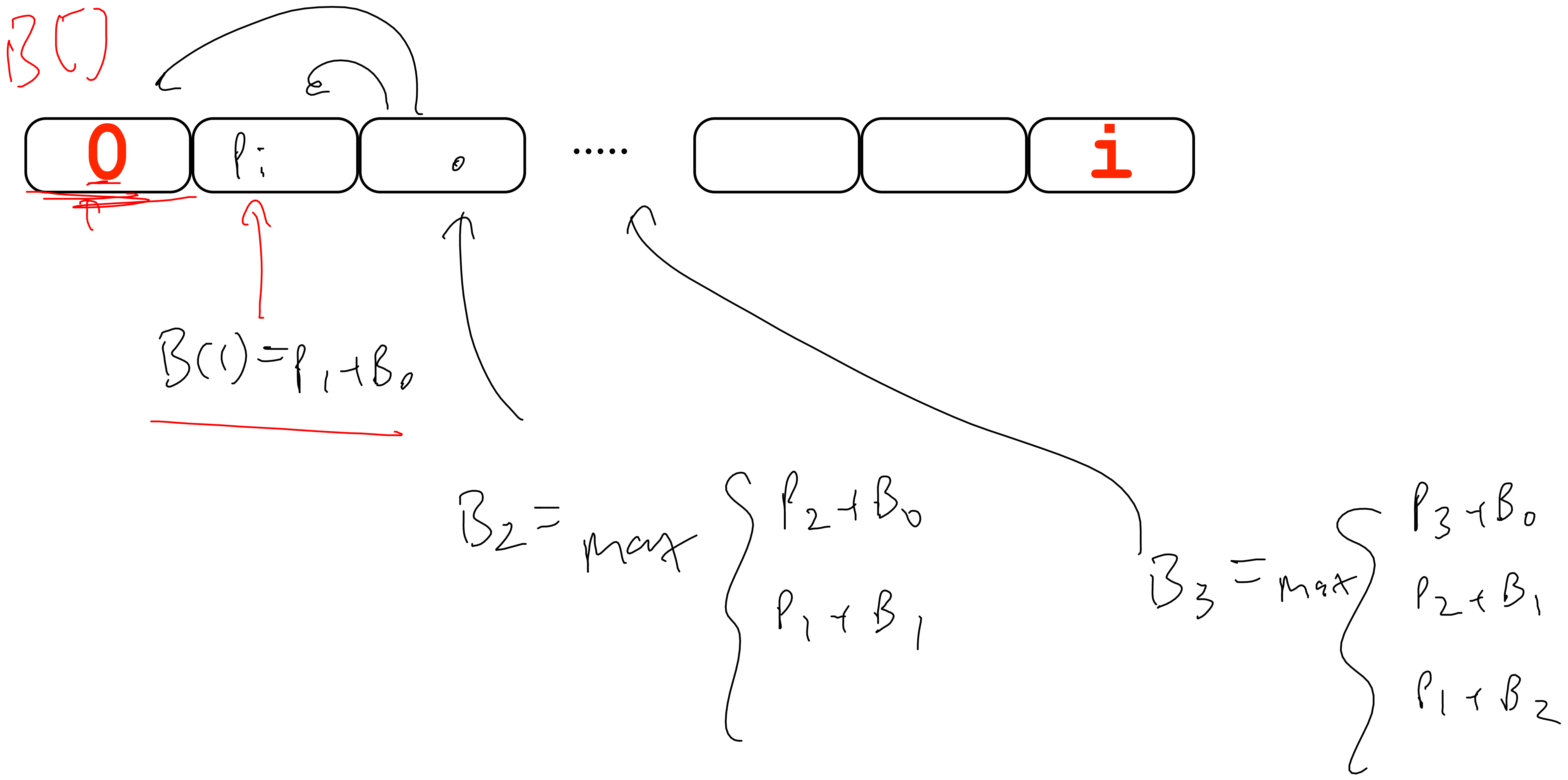
$$Best_{200} = \max \begin{cases} P_1 + B_{199} \\ P_2 + B_{198} \\ P_3 + B_{197} \\ \vdots \\ P_{200} + B_0 \end{cases}$$

$\begin{cases} 1 + B_{198} \\ 2 + B_{197} \end{cases}$

Approach



Approach



BestLogs($n, (p_1, \ldots, p_n)$)

```
    if n<=0 return 0
```

→ for $i = 1$ to n

$$B[i] = \max_{j=1}^{i} \{ P_j + B[i-j] \}$$

→

```
B[i] = -∞
for j=1 to i
    t = P_j + B[i-j]
    if t > B[i]
        B[i] = t
```

Running time: $1 + 2 + 3 + \ldots + (n-1) \sim \Theta(n^2)$

BestLogs($n, (p_1, \ldots, p_n)$)

```
if n<=0 return 0
for i=1 to n
    Best[i] = max_{k=1...i} {p_k + Best[i-k]}
    Choice[i] = k*;
return Best[n]
```

$$\text{Best}[i] = \max_{k=1\ldots i} \{p_k + \text{Best}[i-k]\}$$

Choice[i] = k* ← the particular value of k that resulted in the max at this step

(work an example)

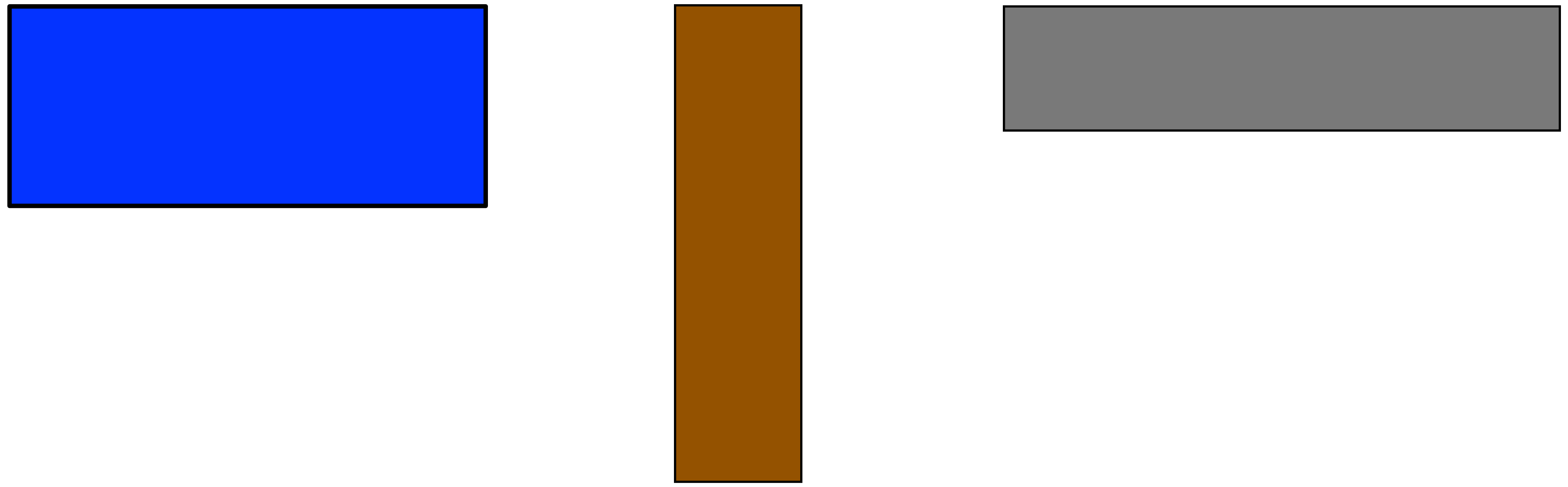
The actual cuts?

BestLogs($n, (p_1, \ldots, p_n)$)

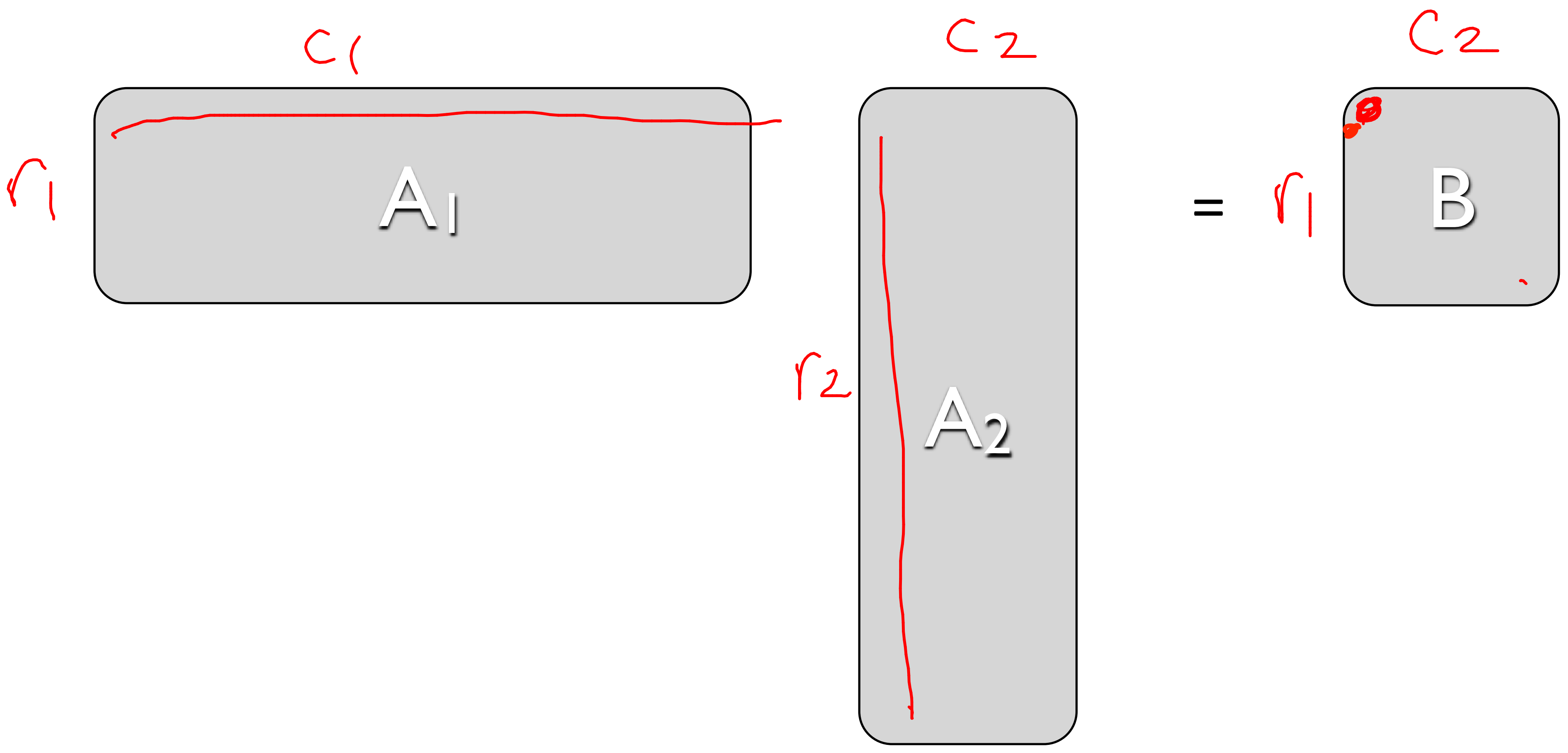
```
if n<=0 return 0
for i=1 to n
 Best[i] = max_{k=1...i} {p_k + Best[i-k]}

return Best[n]
```

$$\text{Best}[i] = \max_{k=1\ldots i}\{p_k + \text{Best}[i-k]\}$$

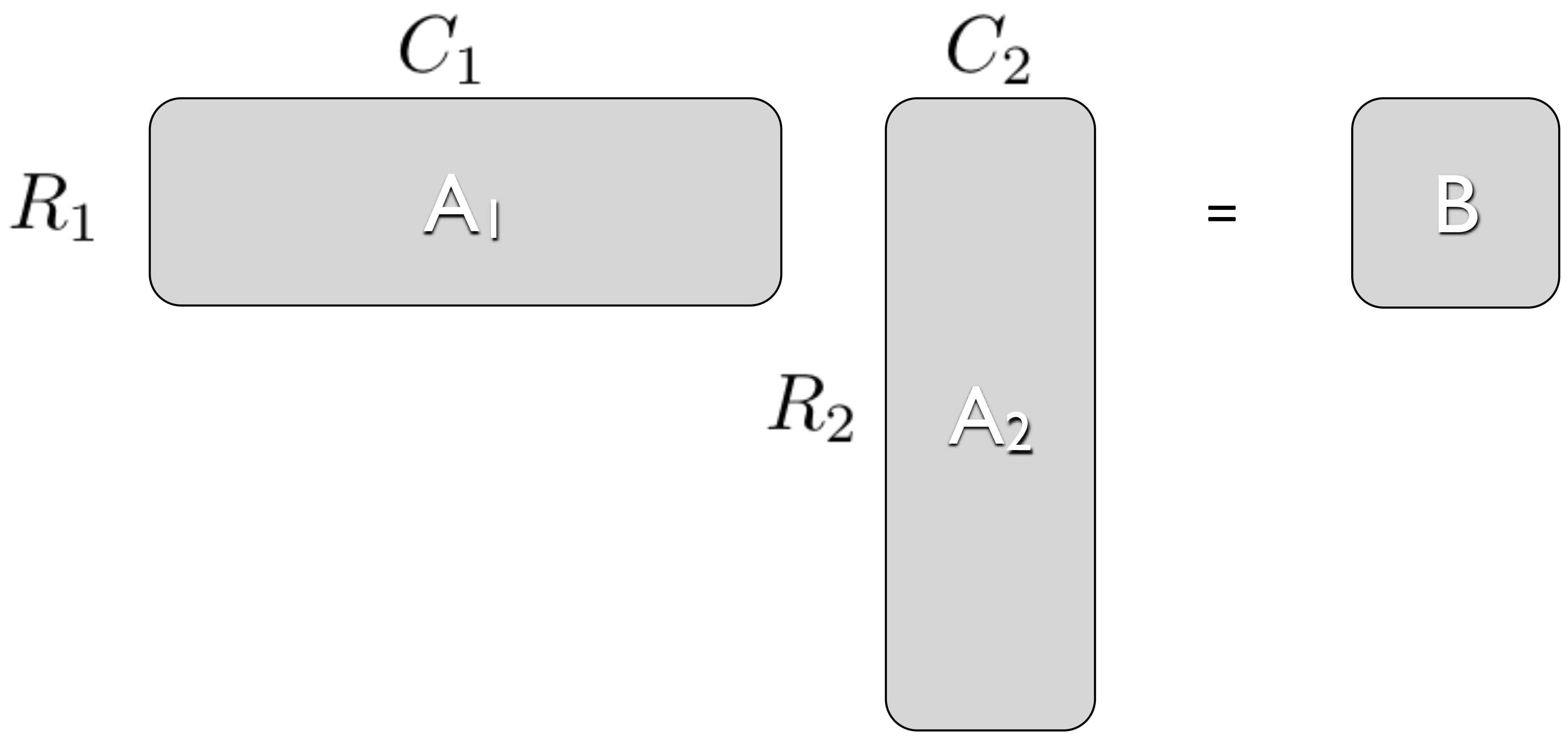
Matrix



$c_1 = r_2$



$c_1 \cdot r_1 \cdot c_2$ = # of operations


C_1
C_2
R_1
A1
=
B
R_2
A2

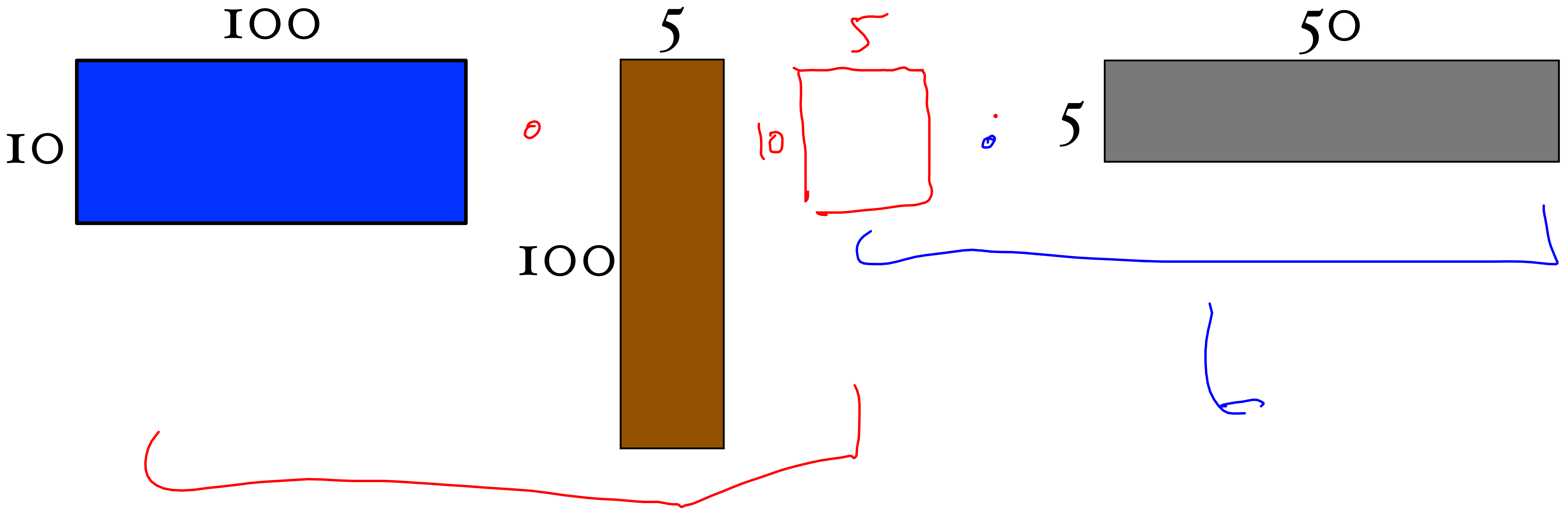
$$(A_1 \cdot A_2) \cdot A_3$$

$$A_1 \cdot (A_2 \cdot A_3)$$

$$A_1 \cdot A_2 \cdot A_3$$

$$(A_1 \cdot A_2) \cdot A_3 \qquad A_1 \cdot (A_2 \cdot A_3)$$

$$(A_1 \cdot A_2) \cdot A_3$$



$10 \cdot 100 \cdot 5$

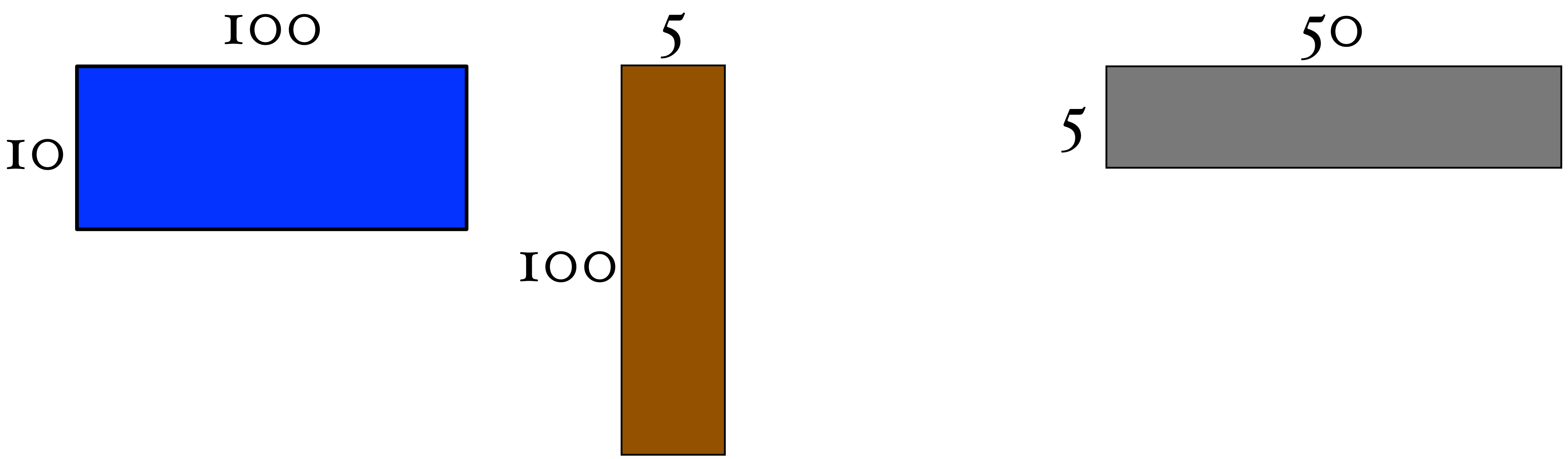
$10 \cdot 5 \cdot 50$

5000

2500

$= 7500$

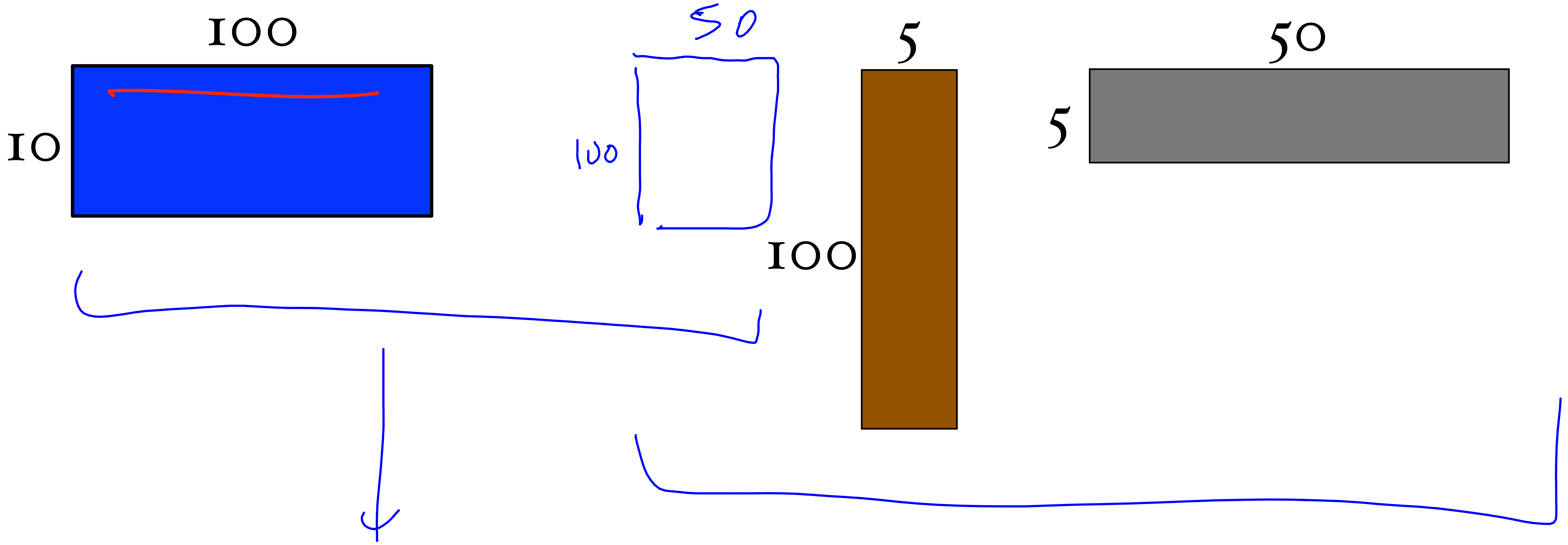
$$(A_1 \cdot A_2) \cdot A_3$$



$$10 \cdot 100 \cdot 5 + 10 \cdot 5 \cdot 50$$

operations

$$A_1 \cdot A_2 \cdot A_3$$



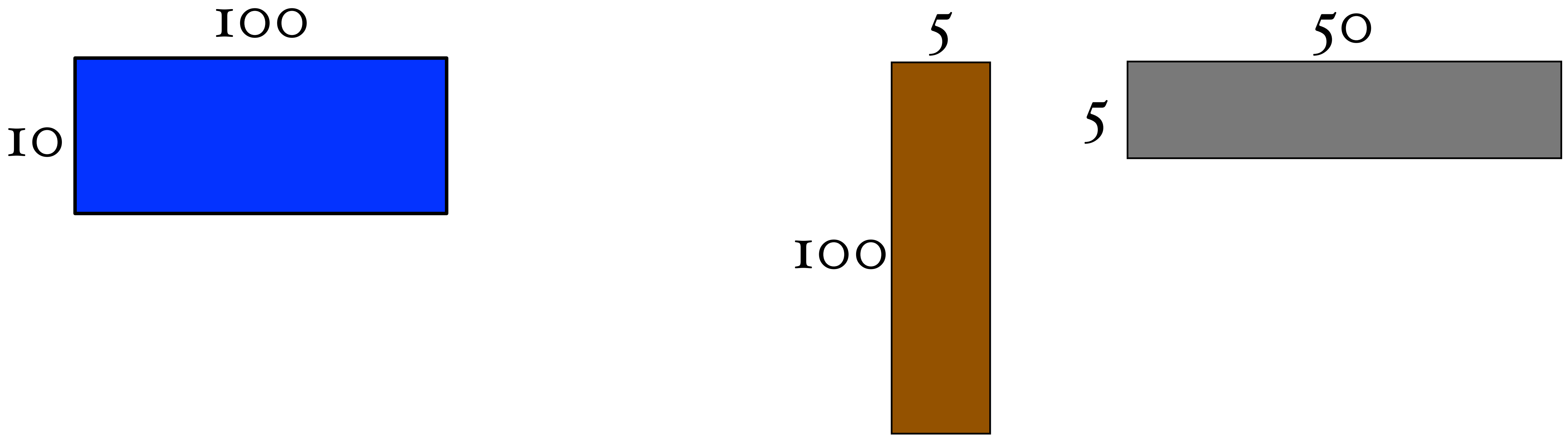
$10 \cdot 100 \cdot 50 =$

$50{,}000$

$100 \cdot 5 \cdot 50 = 25{,}000$!!

$\Rightarrow 75{,}000$!!

$$A_1 \cdot A_2 \cdot A_3$$



$$100 \cdot 5 \cdot 50 + 10 \cdot 100 \cdot 50$$

operations

order matters

(for efficiency)

how many ways to compute?

$$A_1 A_2 A_3 \ldots A_n$$

how many ways to compute?

$$A_1 A_2 A_3 \ldots A_n$$

how do we solve it?

identify smaller instances of the problem

devise method to combine solutions

small # of different subproblems

  solved them in the right order

optimal way to compute

$$A_1 A_2 A_3 A_4 \ldots A_n$$

optimal way to compute

$$A_1 A_2 A_3 A_4 \ldots A_n$$

$$B_{1,n} = B_{1,\ell} + B_{\ell+1,n} + r_1 c_\ell c_n$$

optimal way to compute

$$A_1 A_2 A_3 A_4 \ldots A_n$$

B[1,n]

optimal way to compute

$$A_1 A_2 A_3 A_4 \cdots A_n$$

B[1,n]

B[1,1]	B[1,2]	...	B[1,n-2]	B[1,n-1]
B[2,n]	B[3,n]	...	B[n-1,n]	B[n,n]
$R_1 C_1 C_n$	$R_1 C_2 C_n$		$R_1 C_{n-2} C_n$	$R_1 C_{n-1} C_n$

$$B(i,i) = 1$$

$$B(1,n) = \min \left\{ \begin{array}{l} \\ \\ \\ \end{array} \right.$$

$$B(i,i) = 1$$

$$B(1,n) = \min \begin{cases} B(1,1) + B(2,n) + r_1 c_1 c_n \\ B(1,2) + B(3,n) + r_1 c_2 c_n \\ \vdots \\ B(1,n-1) + B(n,n) + r_1 c_{n-1} c_n \end{cases}$$

$$B(i,j) =$$

$$\begin{cases} 0 \text{ if } i = j \\ \min_k\{B(i,k) + B(k+1,j) + r_i c_k c_j \end{cases}$$

how did we solve it?

identified smaller instances of the problem

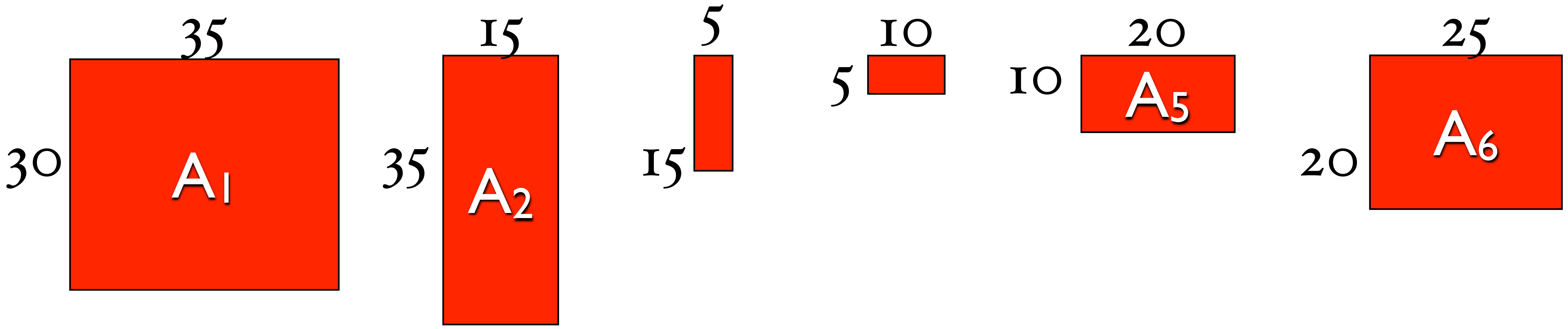
devised method to combine solutions

small # of different subproblems

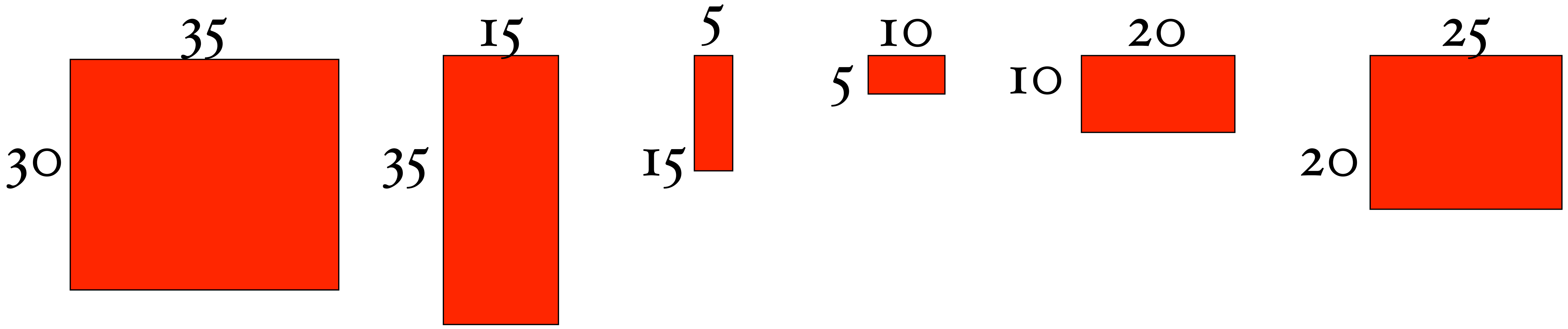
    solved them in the right order

$$B(i,j) = \begin{cases} 0 \text{ if } i = j \\ \min_k \{B(i,k) + B(k+1,j) + r_i c_k c_j \end{cases}$$

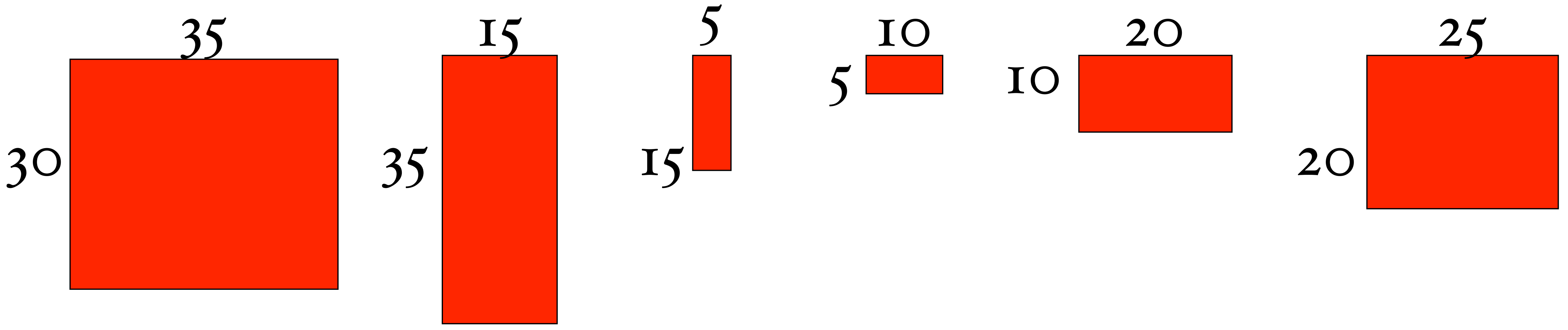
which order to solve?



$$B(1,2) =$$

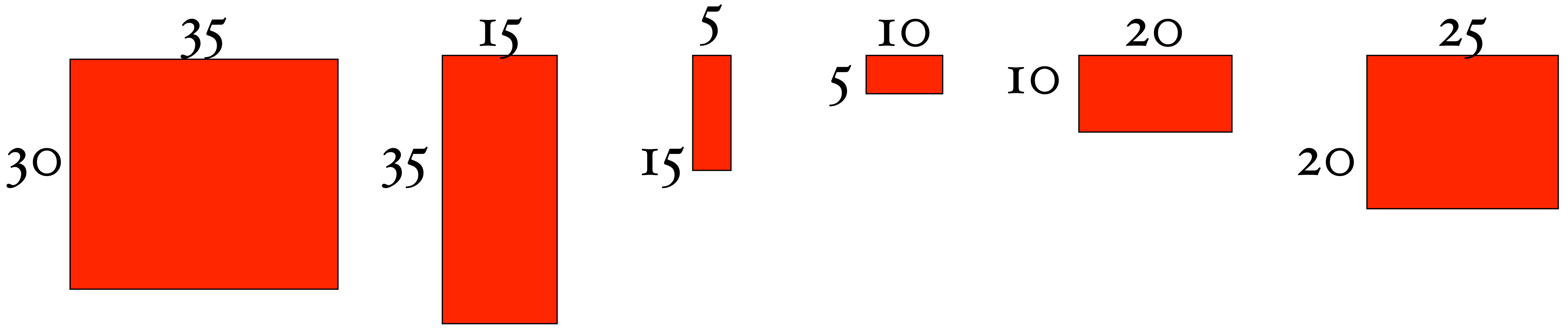


6						0
5					0	
4				0		
3			0			
2		0				
1	0					
	1	2	3	4	5	6



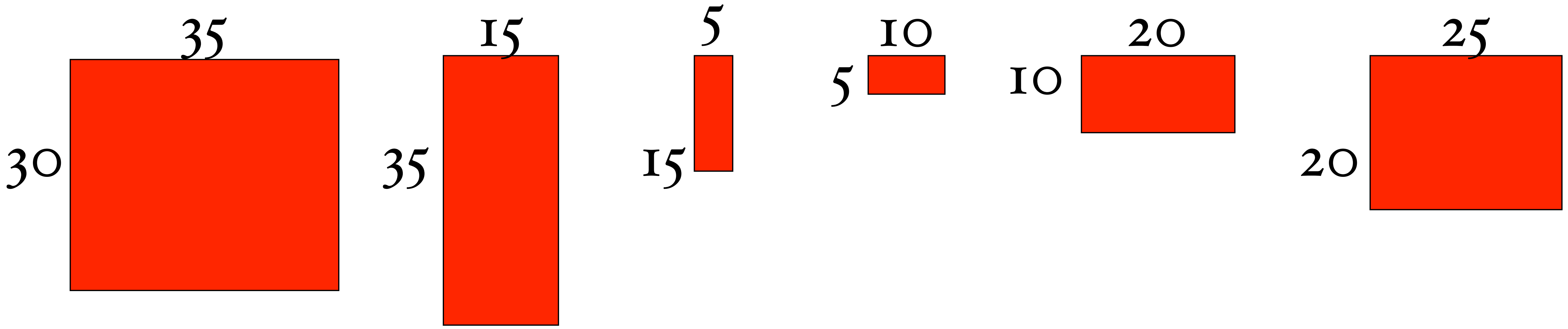
	1	2	3	4	5	6
6						0
5					0	
4				0		
3			0			
2		0				
1	0					

$$B(i,j) = \begin{cases} 0 \text{ if } i = j \\ \min_k \{B(i,k) + B(k+1,j) + r_i c_k c_j \end{cases}$$



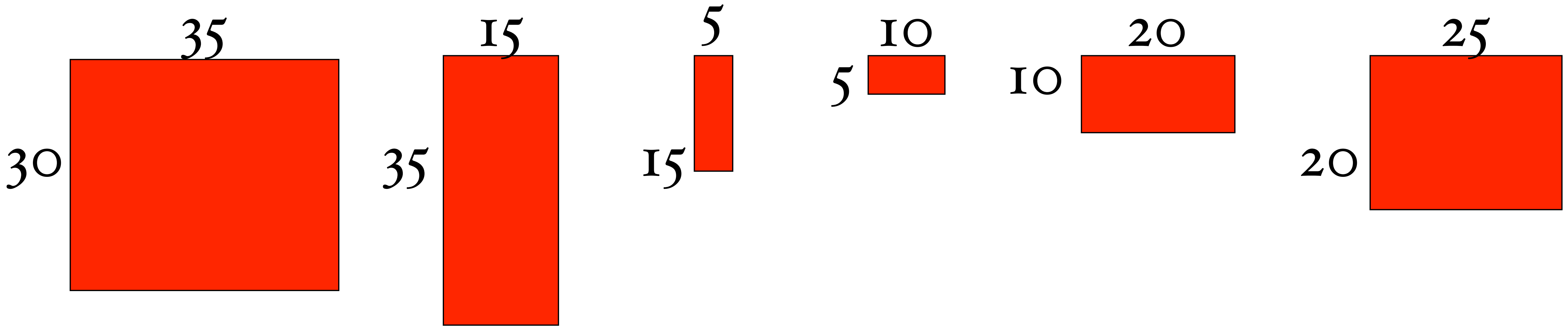
6					10*20*25 = 5000	0
5				5*10*20 = 1000	0	
4			15*5*10 = 750	0		
3		35*15*5 = 2625	0			
2	30*35*15 = 15750	0				
1	0					
	1	2	3	4	5	6

$$B(i,j) = \begin{cases} 0 \text{ if } i = j \\ \min_k\{B(i,k) + B(k+1,j) + r_i c_k c_j \end{cases}$$

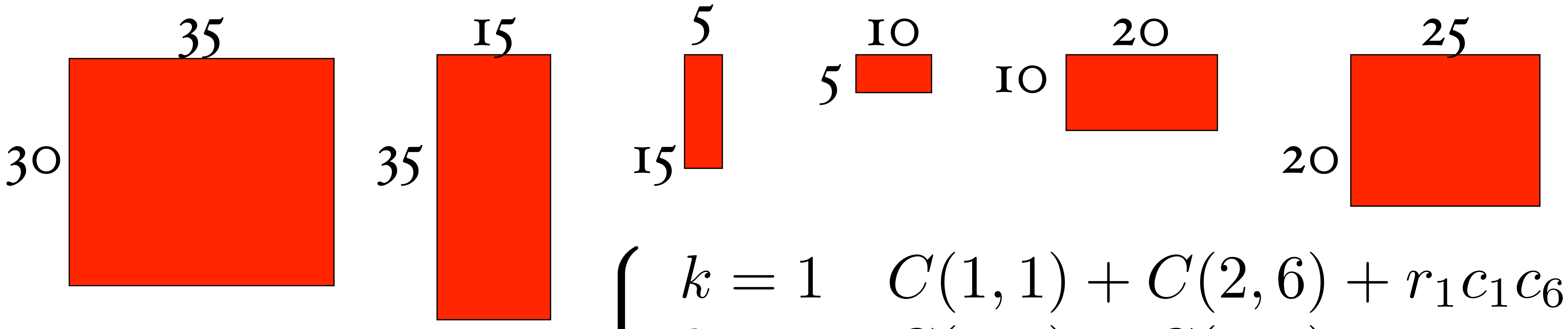


3		35*15*5 = 2625	0			
2	30*35*15 = 15750	0				
1	0					
	1	2	3	4	5	6

$$B(i,j) = \begin{cases} 0 \text{ if } i = j \\ \min_k \{B(i,k) + B(k+1,j) + r_i c_k c_j \end{cases}$$

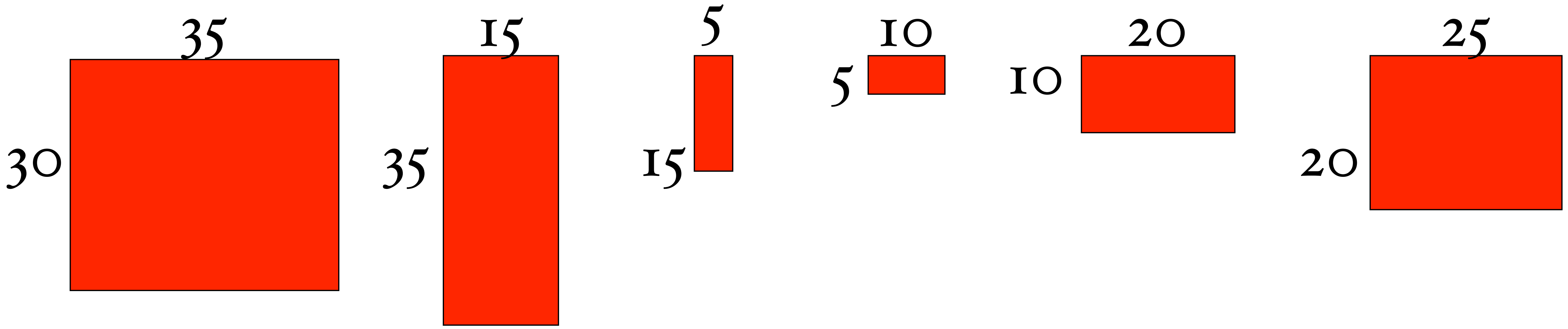


6	$C(1,6) =$					0
5					0	
4				0		
3			0			
2		0				
1	0					
	1	2	3	4	5	6



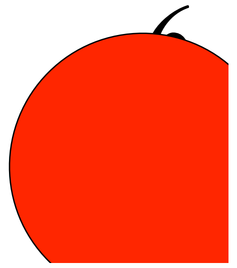
$$C(1,6) = \min \begin{cases} k=1 & C(1,1)+C(2,6)+r_1c_1c_6 \\ k=2 & C(1,2)+C(3,6)+r_1c_2c_6 \\ k=3 & C(1,3)+C(4,6)+r_1c_3c_6 \\ k=4 & C(1,4)+C(5,6)+r_1c_4c_6 \\ k=5 & C(1,5)+C(6,6)+r_1c_5c_6 \end{cases}$$

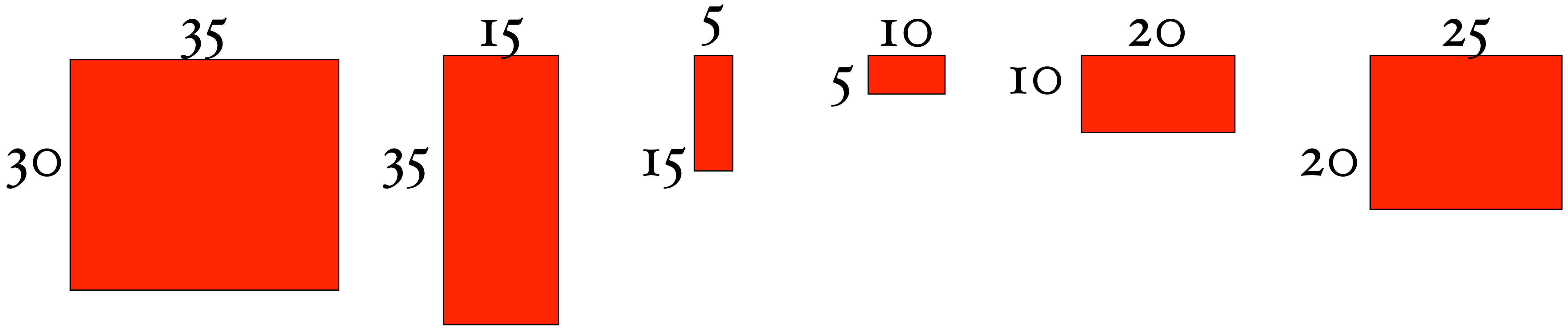
6						0
5					0	
4				0		
3			0			
2		0				
1	0					
	1	2	3	4	5	6



6		10500	5375	3500	10*20*25 = 5000	0
5	11875	7125	2500	5*10*20 = 1000	0	
4	9375	4375	15*5*10 = 750	0		
3	7875	35*15*5 = 2625	0			
2	30*35*15 = 15750	0				
1	0					
	1	2	3	4	5	

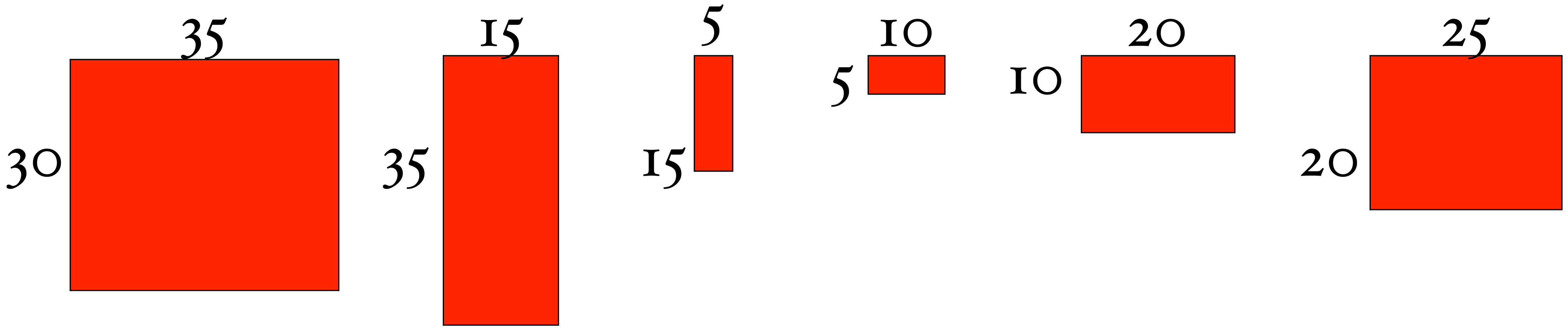
$$B(i,j) = \begin{cases} 0 \text{ if } i = j \\ \min_k \{B(i,k) + B(k+1,j) + r_i c_k c_j \end{cases}$$





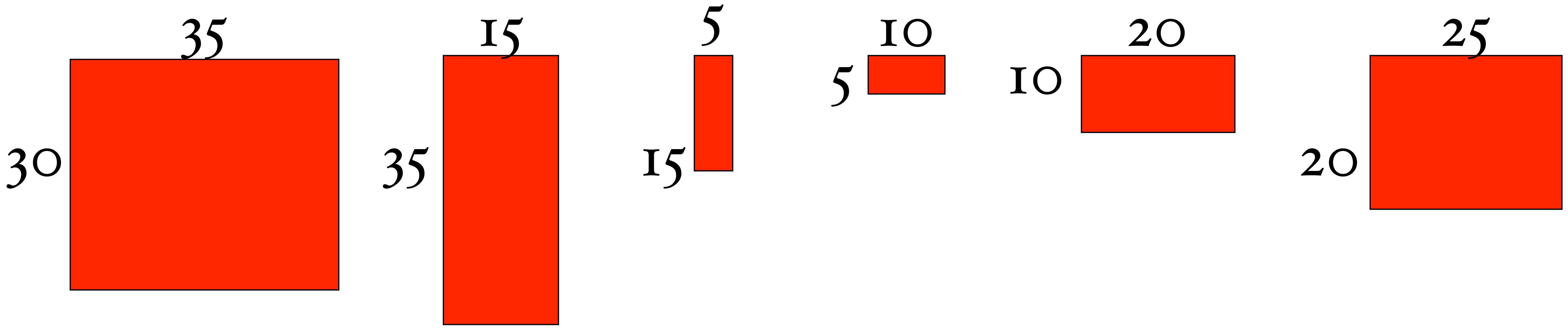
6

$$C(1,6) = \min \begin{cases} k=1 & C(1,1) + C(2,6) + r_1 c_1 c_6 \\ k=2 & C(1,2) + C(3,6) + r_1 c_2 c_6 \\ k=3 & C(1,3) + C(4,6) + r_1 c_3 c_6 \\ k=4 & C(1,4) + C(5,6) + r_1 c_4 c_6 \\ k=5 & C(1,5) + C(6,6) + r_1 c_5 c_6 \end{cases}$$



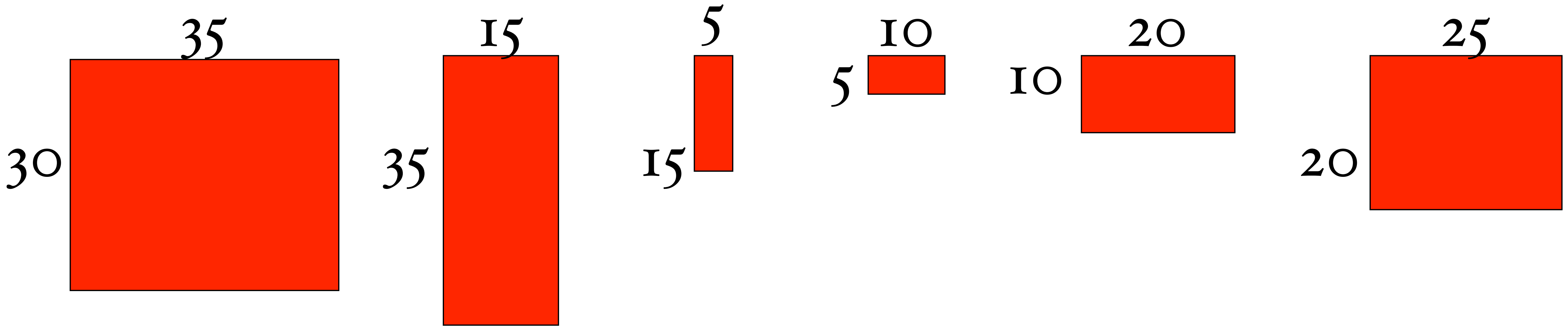
6 

$$C(1,6) = \min \begin{cases} k = 1 & 0 + 10500 + 30 \cdot 35 \cdot 25 \\ k = 2 & 15750 + 5375 + 30 \cdot 15 \cdot 25 \\ k = 3 & 7875 + 3500 + 30 \cdot 5 \cdot 25 \\ k = 4 & 9375 + 5000 + 30 \cdot 10 \cdot 25 \\ k = 5 & 11875 + 0 + 30 \cdot 20 \cdot 25 \end{cases}$$

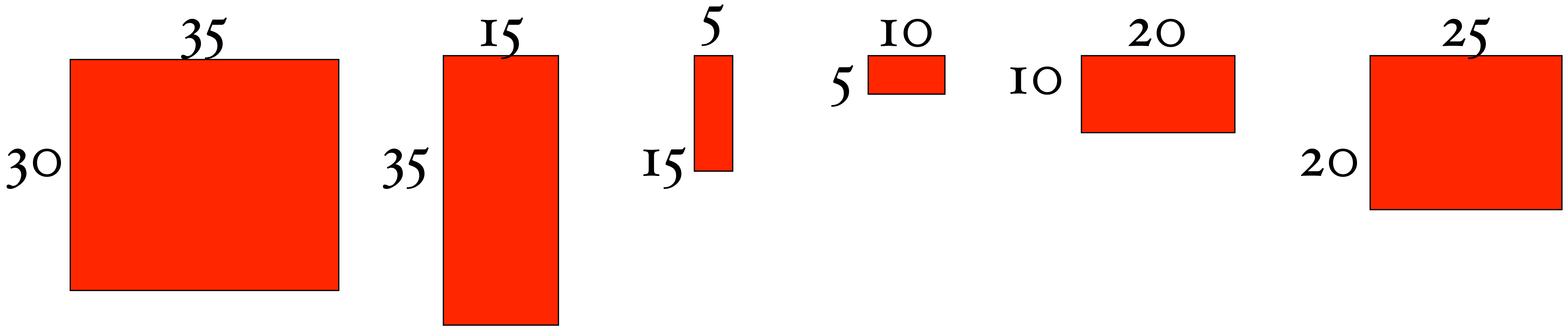


6

$$C(1,6) = \min \begin{cases} k=1 & 0 + 10500 + 26250 \\ k=2 & 15750 + 5375 + 11250 \\ k=3 & 7875 + 3500 + 3750 \\ k=4 & 9375 + 5000 + 7500 \\ k=5 & 11875 + 0 + 15000 \end{cases}$$



6	15125 3	10500	5375	3500 ★	10*20*25 = 5000	0
5	11875	7125	2500	5*10*20 = 1000	0	
4	9375	4375	15*5*10 = 750	0		
3	7875 ★	35*15*5 = 2625	0			
2	30*35*15 = 15750	0				
1	0					
	1	2	3	4	5	6



6	15125 3	10500	5375	3500	10*20*25 = 5000	0
5	11875	7125	2500	5*10*20 = 1000	0	
4	9375	4375	15*5*10 = 750	0		
3	7875	35*15*5 = 2625	0			
2	30*35*15 = 15750	0				
1	0					
	1	2	3	4	5	6

matrix-chain-mult(p)

initialize array m{x,y} to zero

matrix-chain-mult(p)

initialize array m[x,y] to zero

starting at diagonal, working towards upper-left

compute m[i,j] according to

$$\begin{cases} 0 \text{ if } i = j \\ \min_k \{B(i,k) + B(k+1,j) + r_i c_k c_j \end{cases}$$

running time?

initialize array m[x,y] to zero

starting at diagonal, working towards upper-left

compute m[i,j] according to

$$\begin{cases} 0 \text{ if } i = j \\ \min_k \{ B(i,k) + B(k+1,j) + r_i c_k c_j \end{cases}$$

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.

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.

It was the best of times, it was the worst	0	0
of times, it was the age of wisdom, it was	0	0
the age of foolishness, it was the epoch	2	4
of belief, it was the epoch of	12	144
incredulity, it was the season of Light,	2	4
it was the season of Darkness, it was the	1	1
spring of hope, it was the winter of	6	36
despair, we had everything before us, we	2	4
had nothing before us, we were all going	2	4
direct to heaven, we were all going direct	0	0
the other way - in short, the period was		
so far like the present period, that some of its		197
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	6	36
worst of times, it was the age of wisdom,	1	1
it was the age of foolishness, it was the	1	1
epoch of belief, it was the epoch of	6	36
incredulity, it was the season of Light,	2	4
it was the season of Darkness, it was the	1	1
spring of hope, it was the winter of	6	36
despair, we had everything before us, we	2	4
had nothing before us, we were all going	2	4
direct to heaven, we were all going direct	0	0
the other way - in short, the period was		
so far like the present period, that some		123
of its noisiest authorities insisted on		
its being received, for good or for evil,		
in the superlative degree of comparison		
only.		

Typesetting problem

input:

output:

such that

Typesetting problem

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

output: $L = (w_1, \ldots, w_{\ell_1}), (w_{\ell_1+1}, \ldots, w_{\ell_2}), \ldots, (w_{\ell_x+1,\ldots,w_n})$

such that

Typesetting problem

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

output: $L = (w_1, \ldots, w_{\ell_1}), (w_{\ell_1+1}, \ldots, w_{\ell_2}), \ldots, (w_{\ell_x+1,\ldots,w_n})$

$$c_i = \left(\sum_{j=\ell_i+1}^{\ell_{i+1}} |w_j| \right) + (\ell_{i+1} - \ell_i - 1)$$

such that $c_i \leq M \quad \forall i$

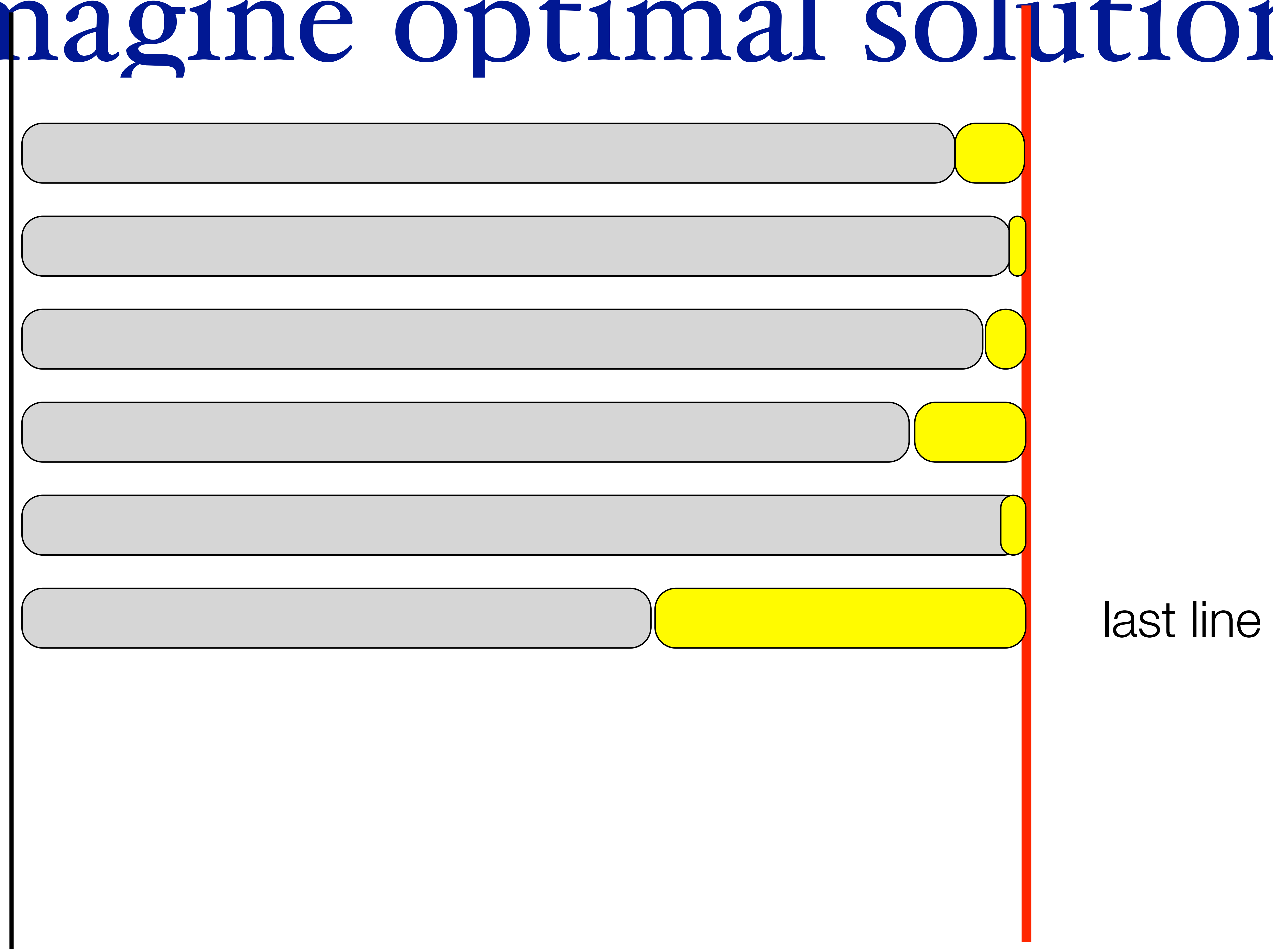
$$\min \sum (M - c_i)^2$$

how to solve

define the right variable:

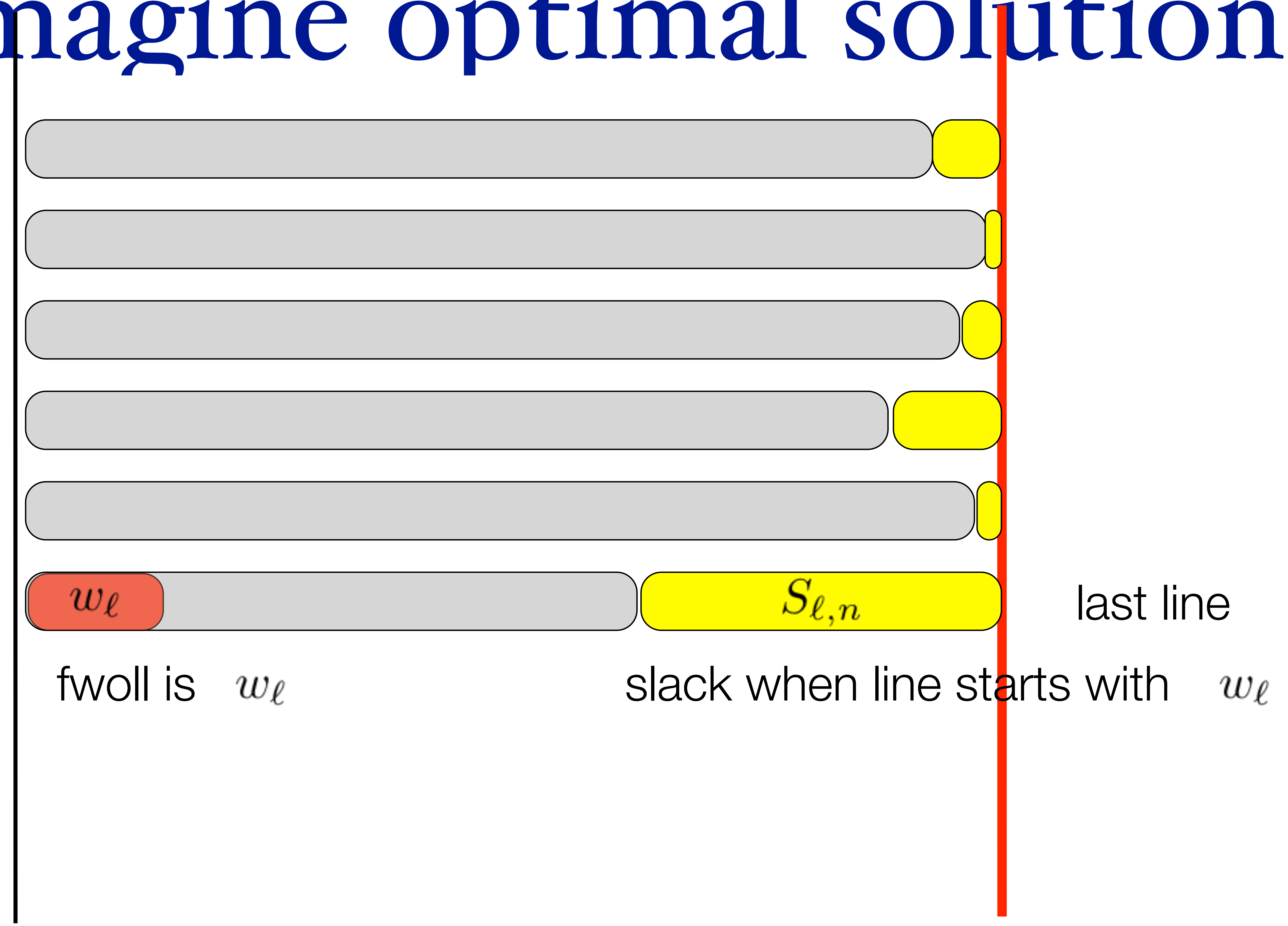
imagine optimal solution

imagine optimal solution

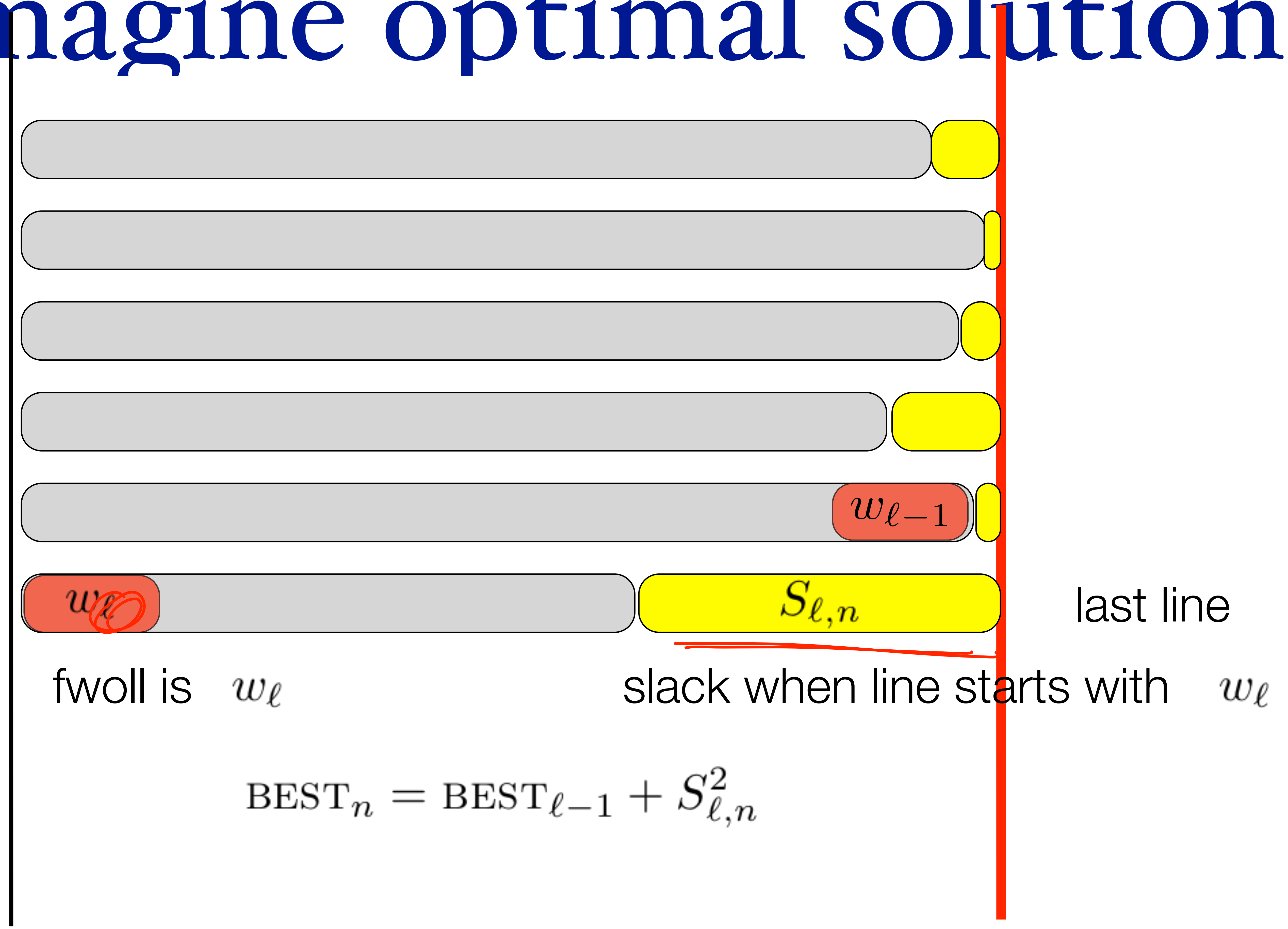


some word has to be the
first-word-of-last-line
(fwoll)

imagine optimal solution



imagine optimal solution



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

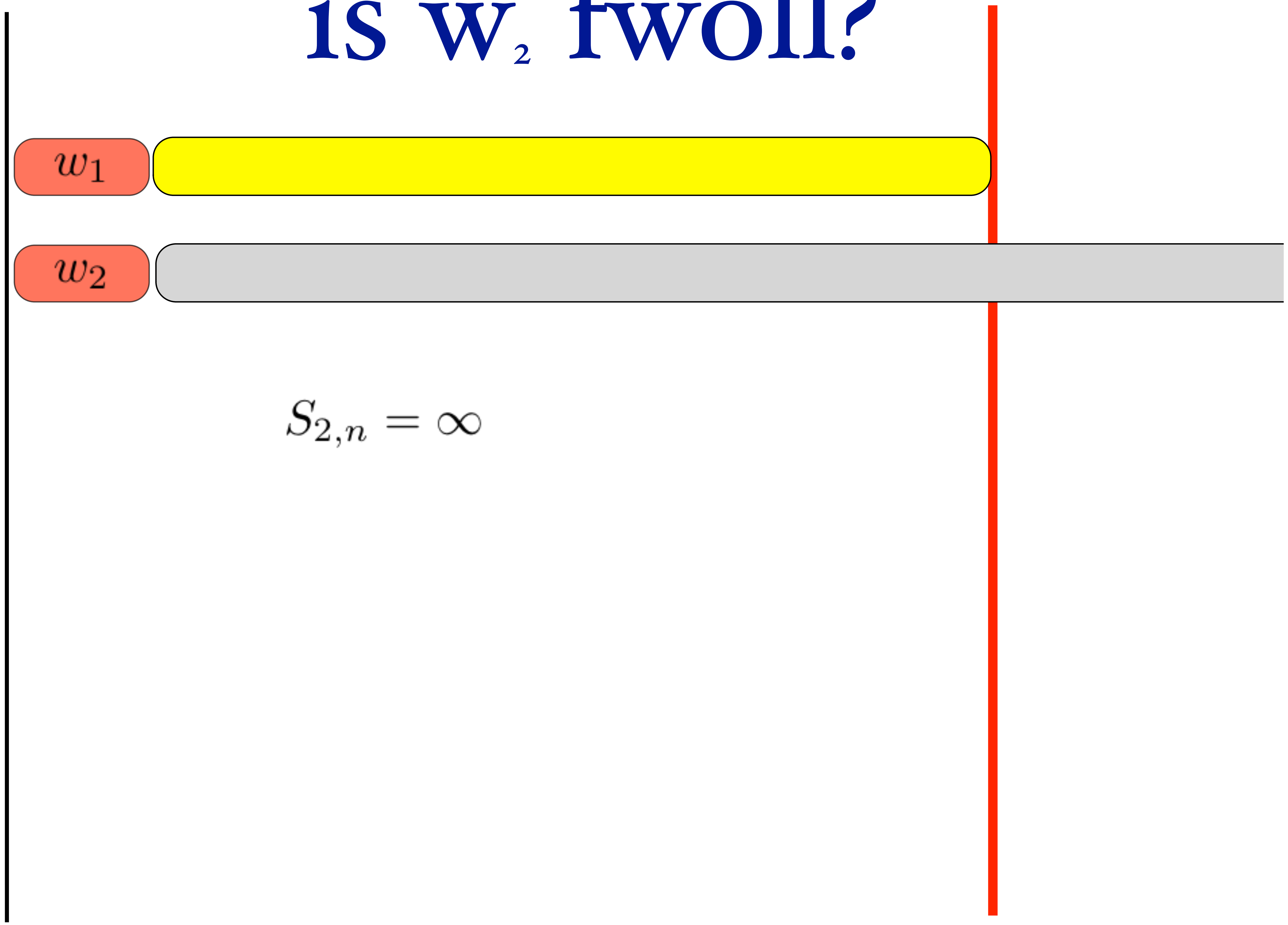
how many candidates
are there for the fwoll?

is w_1 fwoll?

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

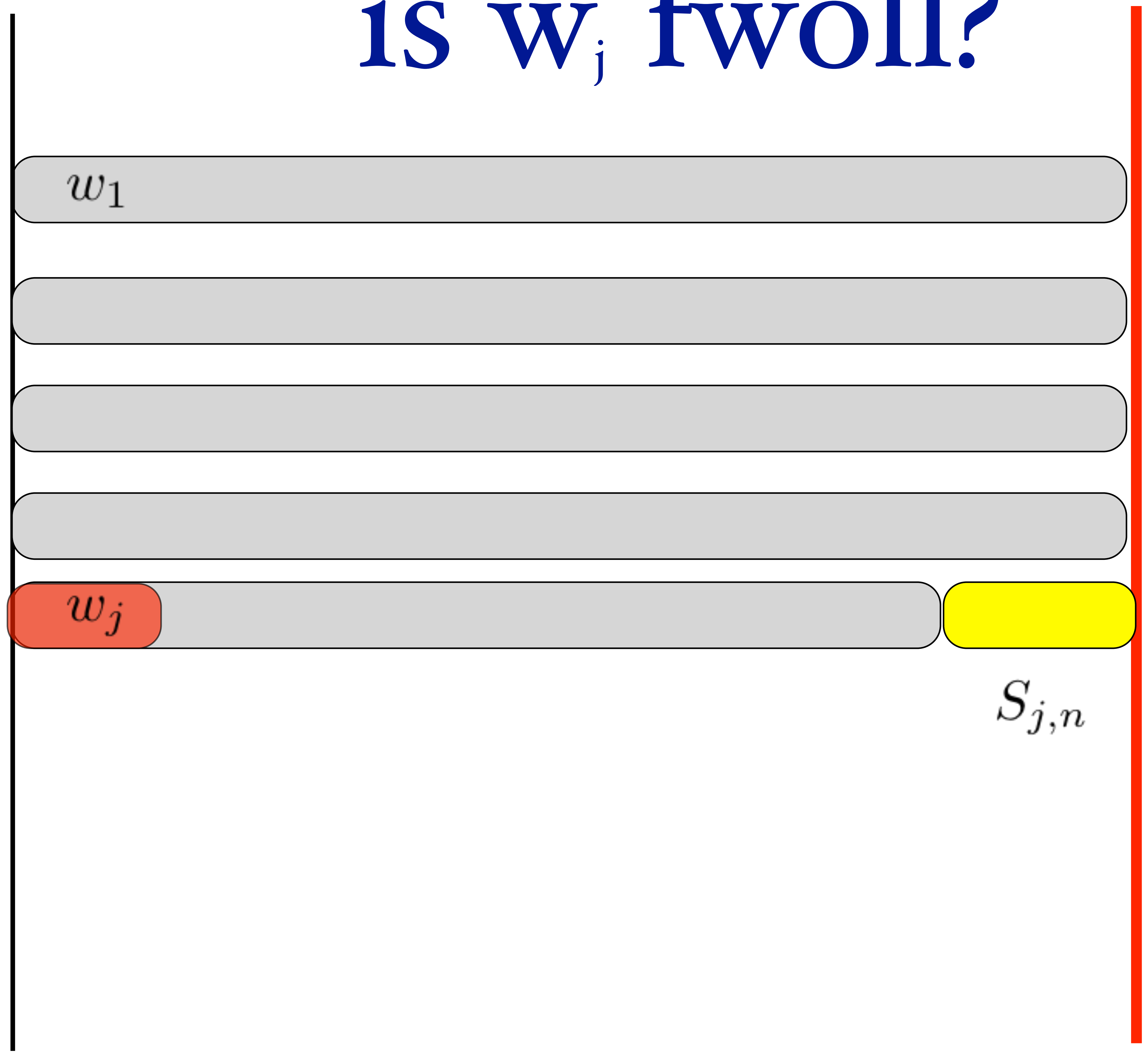
define $S_{1,n} = \infty$ if this happens

is w_2 fwoll?

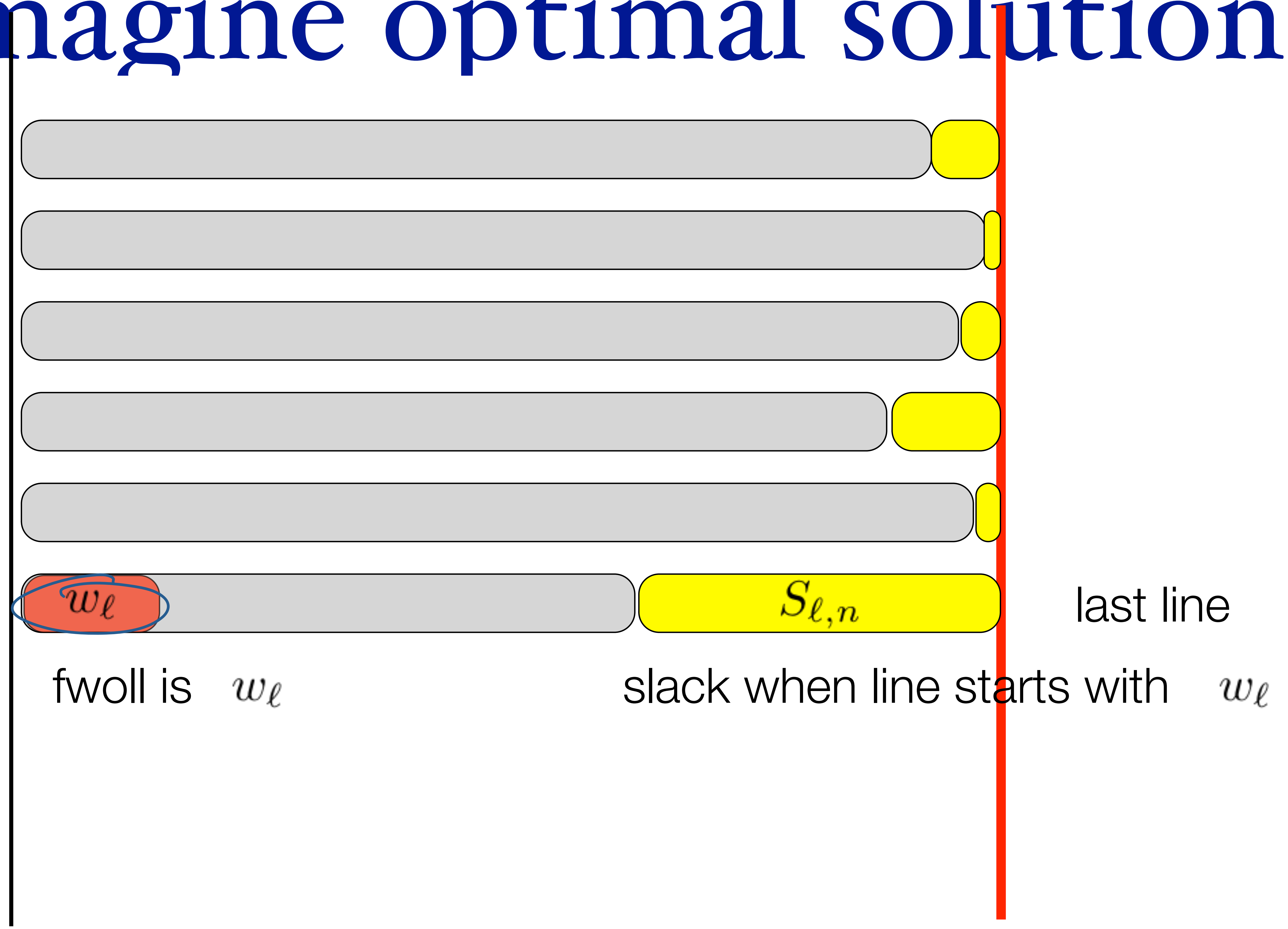


$$S_{2,n} = \infty$$

is w_j fwoll?



imagine optimal solution



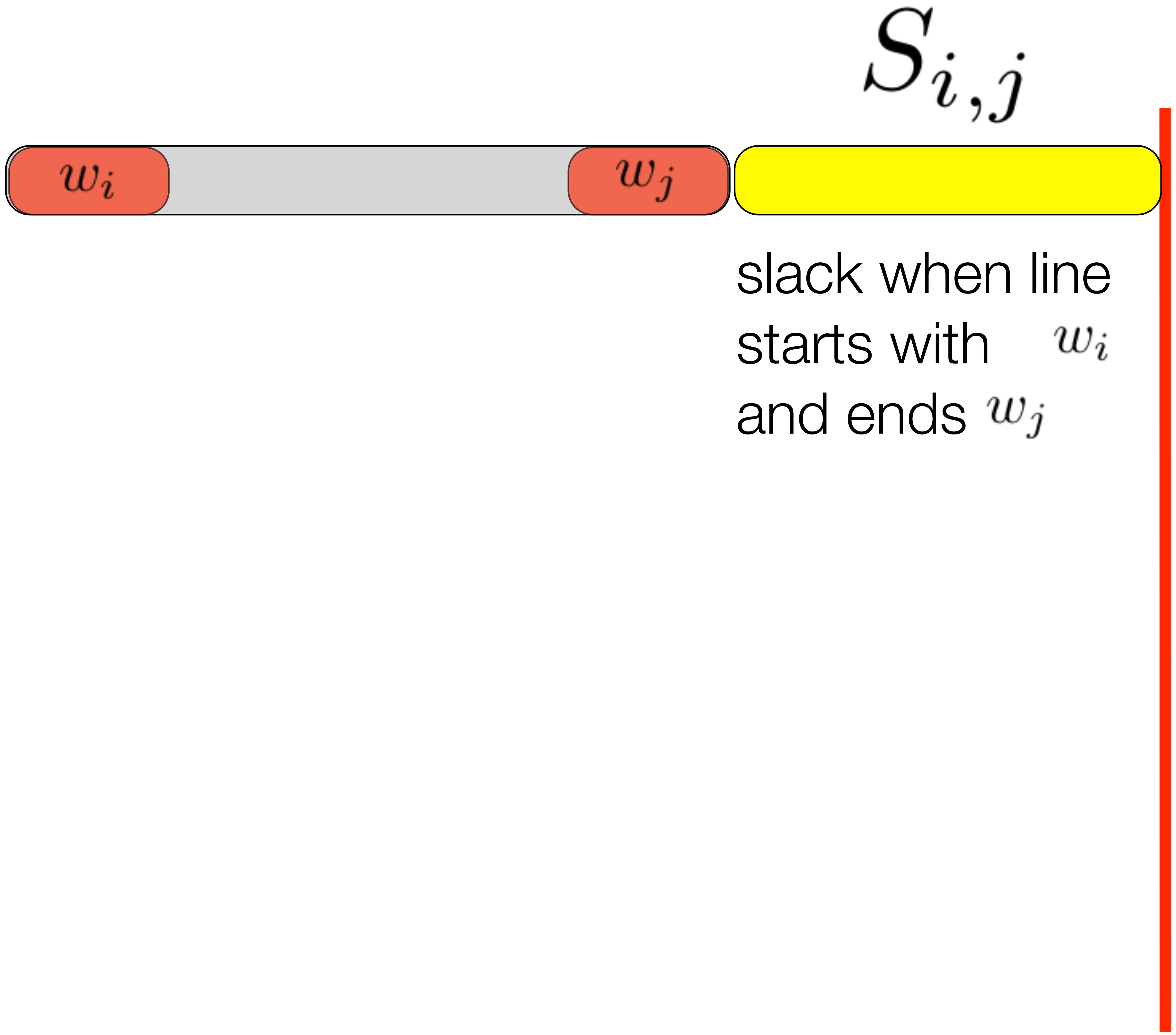
which word is fwoll?

$$\mathrm{BEST}_n = \min \left\{ \right.$$

which word is fwoll?

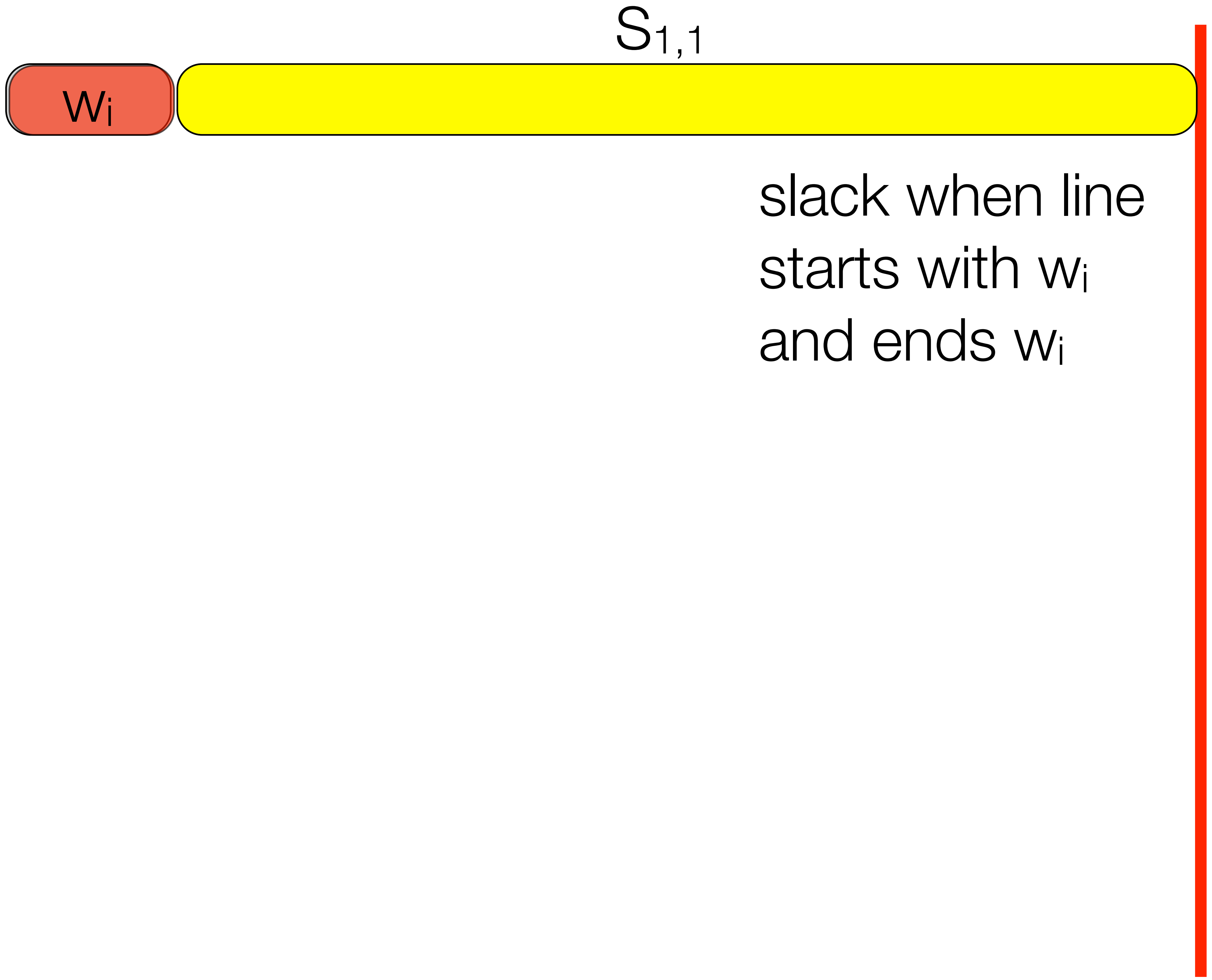
$$\text{BEST}_n = \min \begin{cases} \text{BEST}_0 + S_{1,n}^2 \\ \text{BEST}_1 + S_{2,n}^2 \\ \text{BEST}_2 + S_{3,n}^2 \\ \quad\cdots \\ \text{BEST}_{\ell-1} + S_{\ell,n}^2 \\ \\ \quad\cdots \\ \text{BEST}_{n-1} + S_{n,n}^2 \end{cases}$$

how to compute $S_{i,j}$

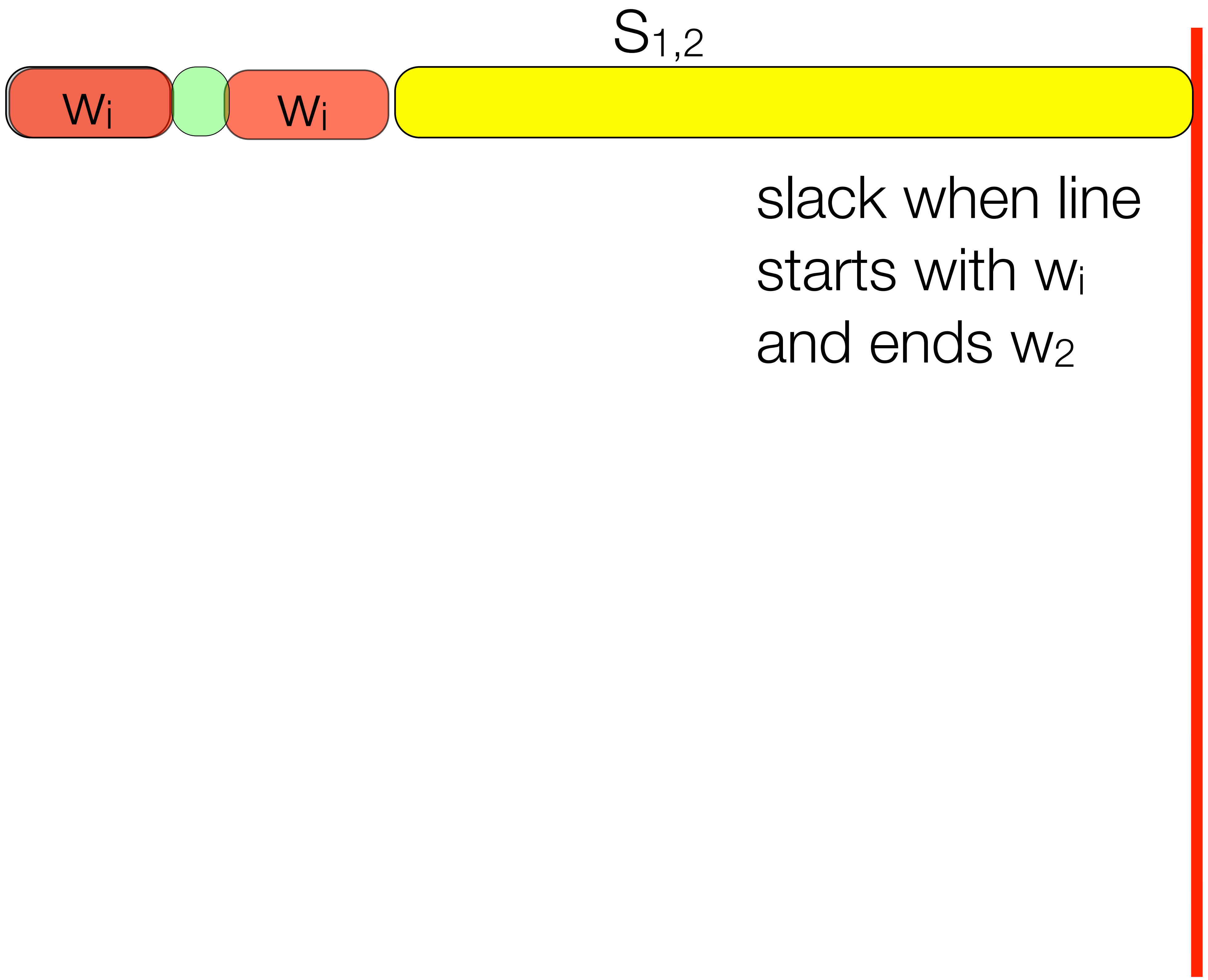


slack when line starts with w_i and ends w_j

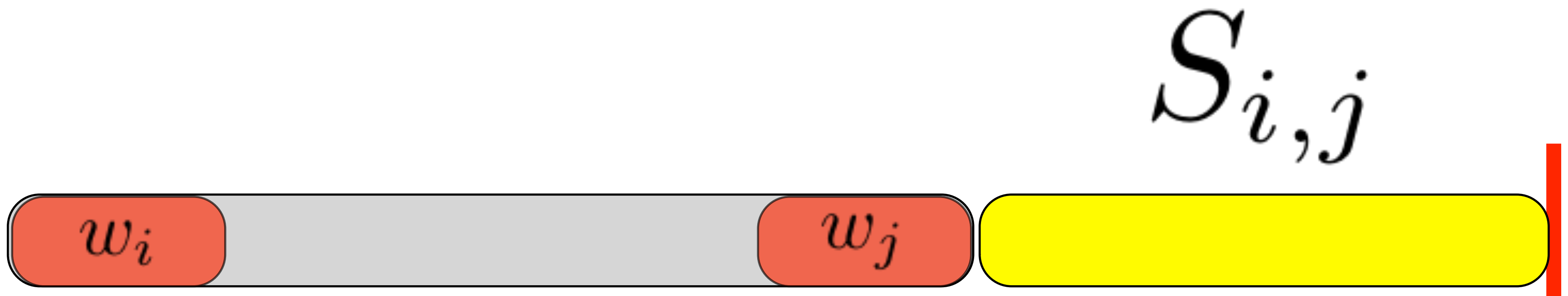
Simplest case



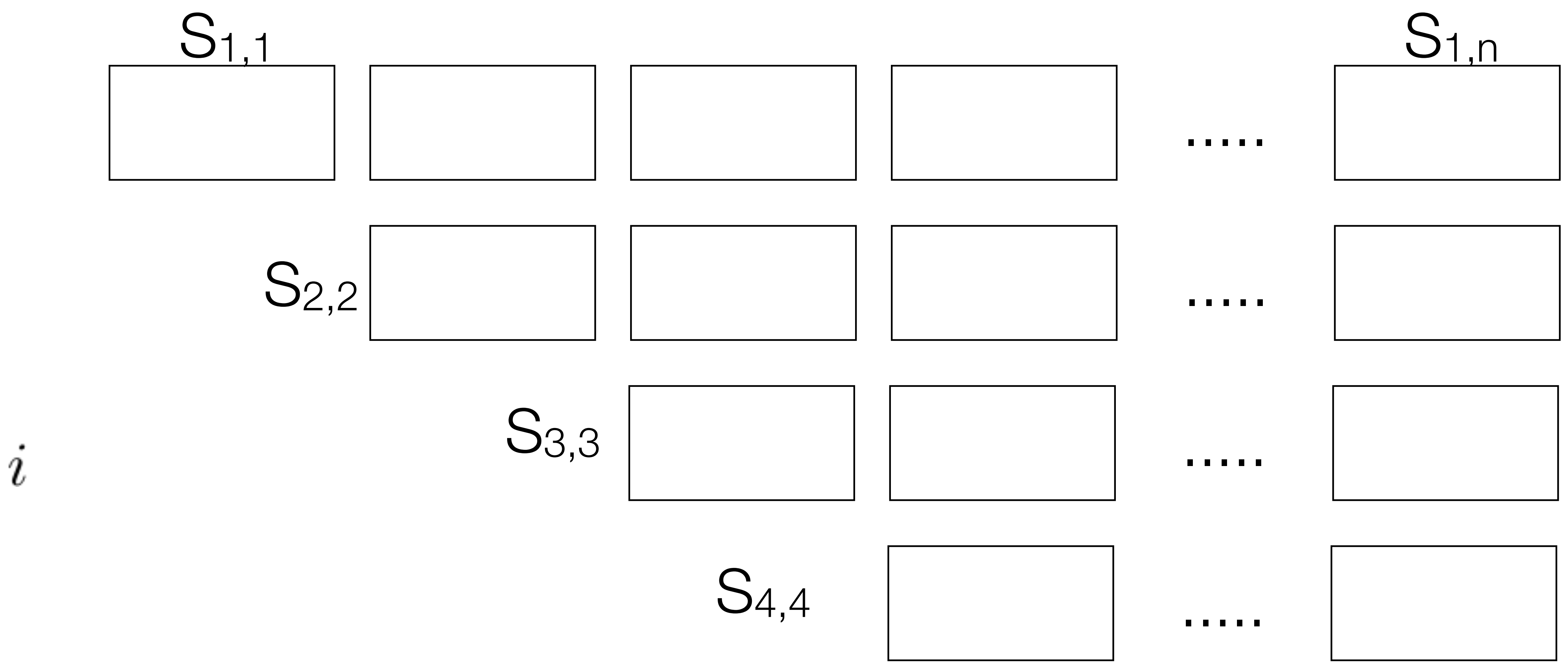
Simplest case



how to compute $S_{i,j}$



slack when line starts with w_i and ends w_j


S1,1
S1,n
.....
S2,2
.....
S3,3
.....
i
S4,4
.....

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 o
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 $S_{i,j}$

	1	2	3	4	5	6	7	8	9	10	11	12	...
1	?												

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

$M = 42$

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

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

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													

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

$M = 42$

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

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

	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

second step: compute

	0	1	2	3	4	5	6	7	8	9	10	...
best	0											

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

	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

second step: compute

	0	1	2	3	4	5	6	7	8	9	10	...
best	0	1600										

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

	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

second step: compute

	0	1	2	3	4	5	6	7	8	9	10	...
best	0	1600	1296									

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

	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

Running time

make table for $S_{i,j}$

for `i=1` to `n`

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

PROBLEM: REDUCE IMAGE

scaling: distortion

deleting column: distortion

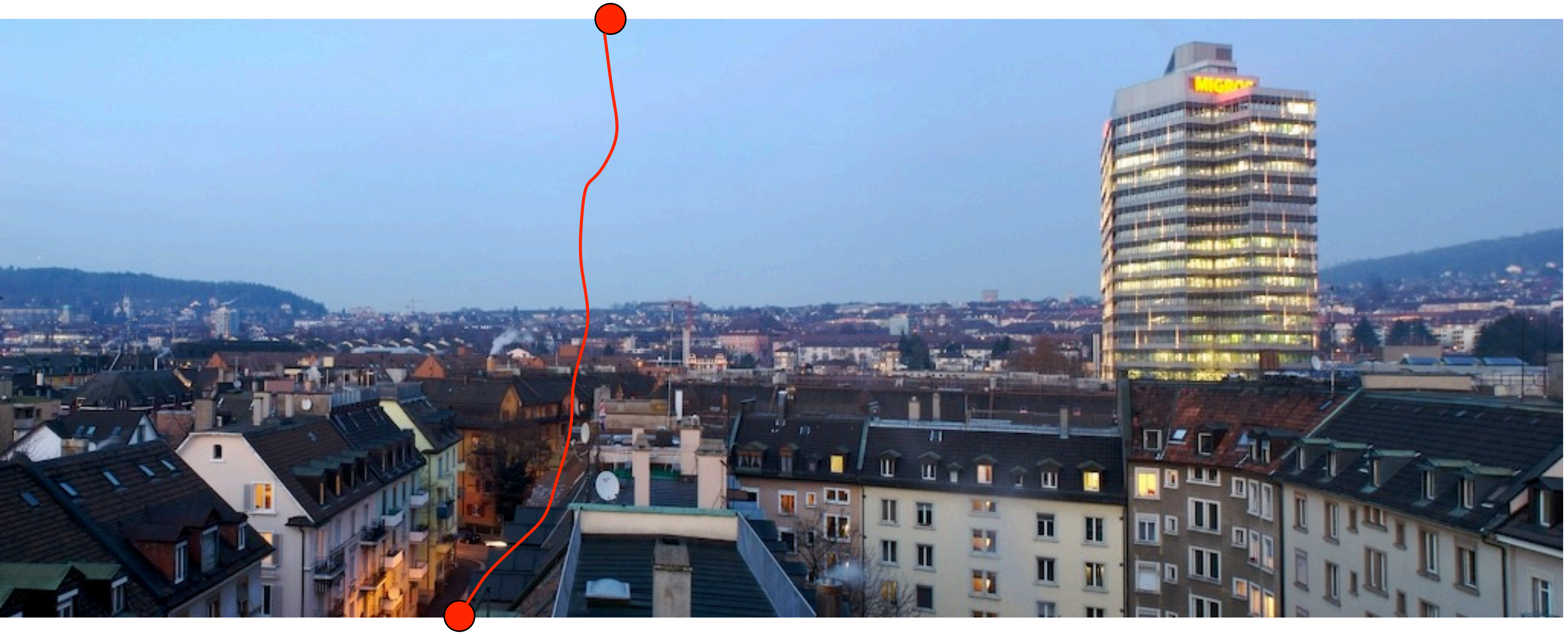
delete the most invisible seam


Retarget col se...
Image Resizing
Shai Avidan
Mitsubishi Electric Research Lab
Ariel Shamir
The interdisciplinary Center & MERL

DEMO?

http://rsizr.com/

WHICH SEAM TO DELETE?

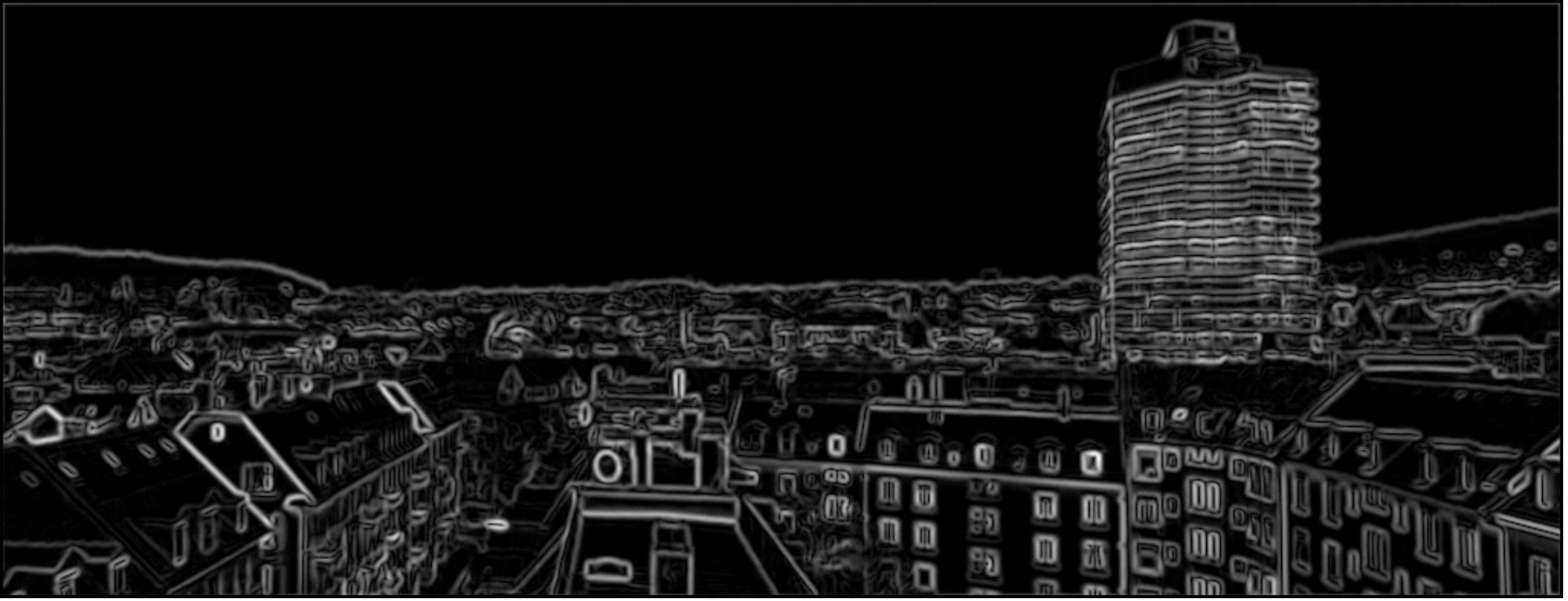


ENERGY OF AN IMAGE

$$e(\mathbf{I}) = \left|\frac{\partial}{\partial x}\mathbf{I}\right| + \left|\frac{\partial}{\partial y}\mathbf{I}\right|$$

“magnitude of gradient at a pixel”

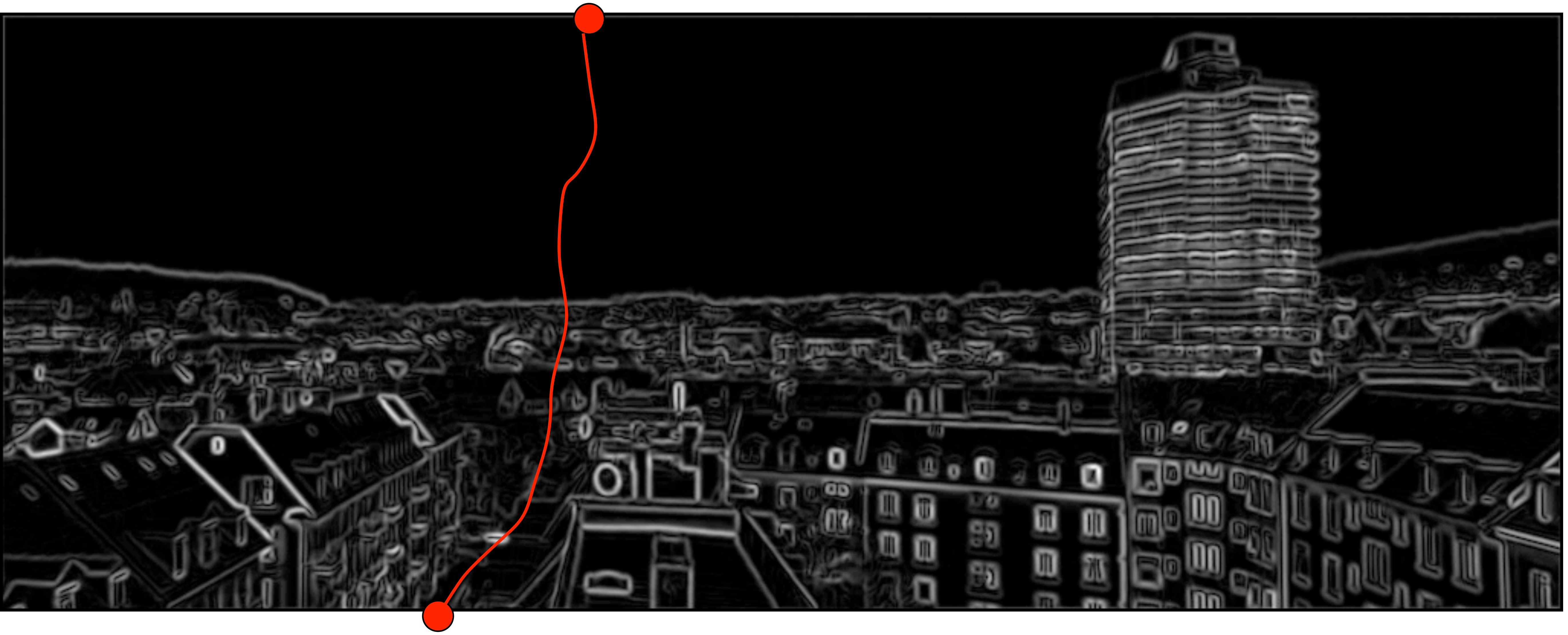
$$\frac{\partial}{\partial x} I_{x,y} = I_{x-1,y} - I_{x+1,y}$$



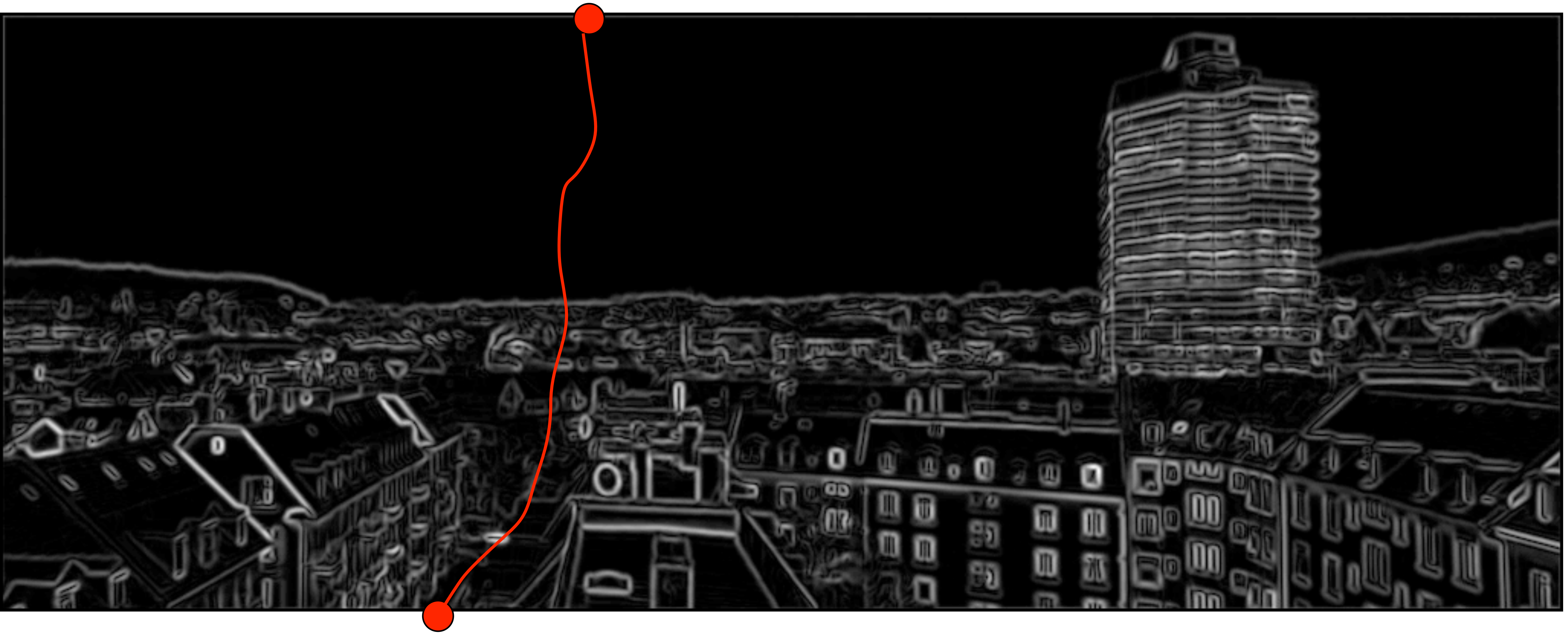
energy of sample image

thanks to Jason Lawrence for gradient software

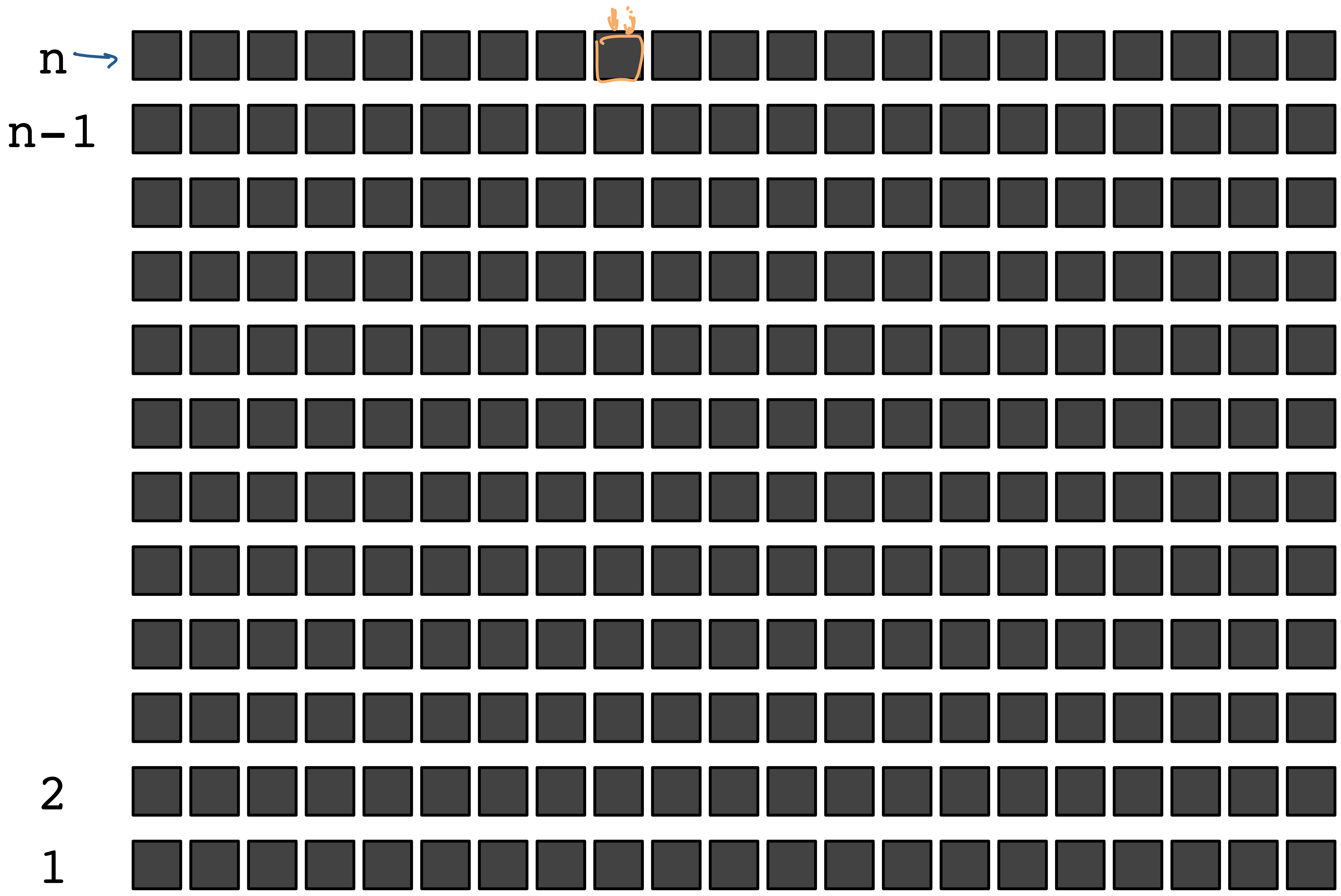
BEST SEAM HAS LOWEST ENERGY



FINDING LOWEST ENERGY SEAM?

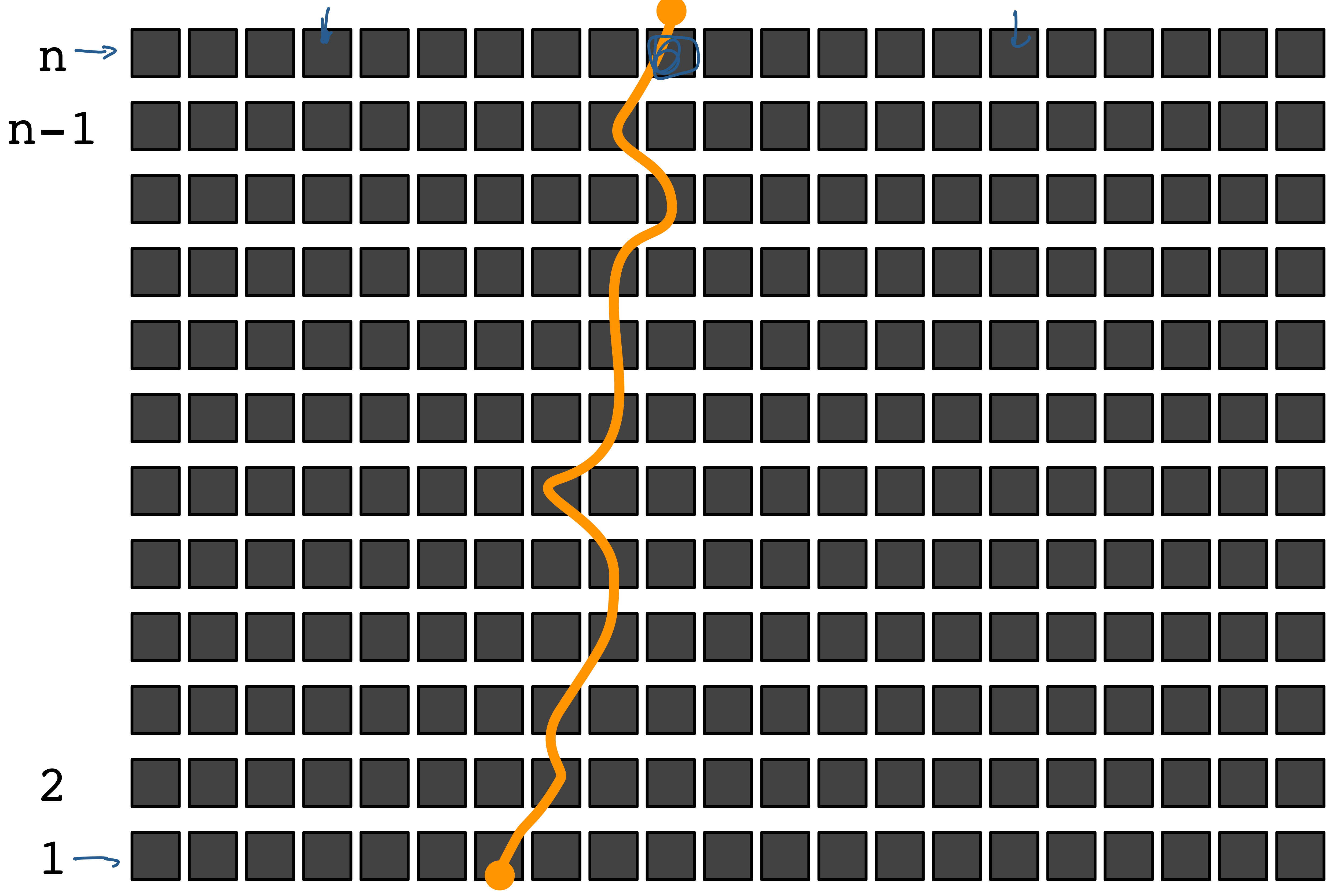


definition: $S_n(j)$



definition:

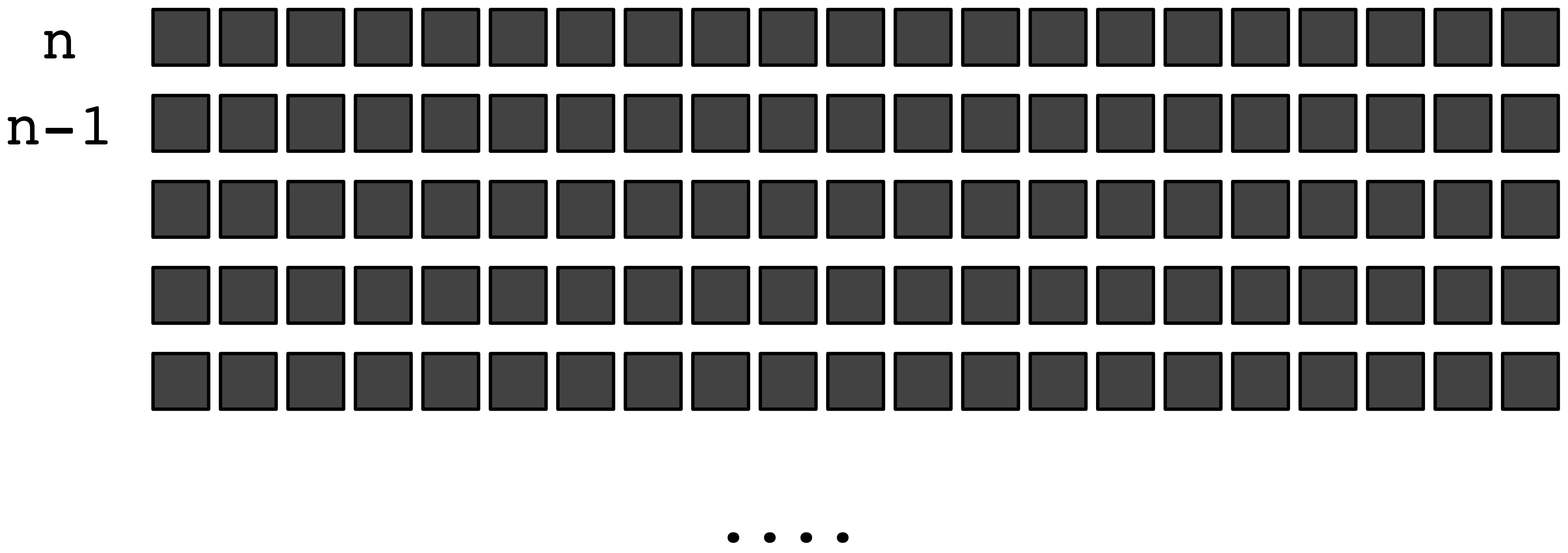
$S_n(j)$ best seam ending at (n,j)



BEST SEAM TO DELETE HAS TO BE THE BEST AMONG

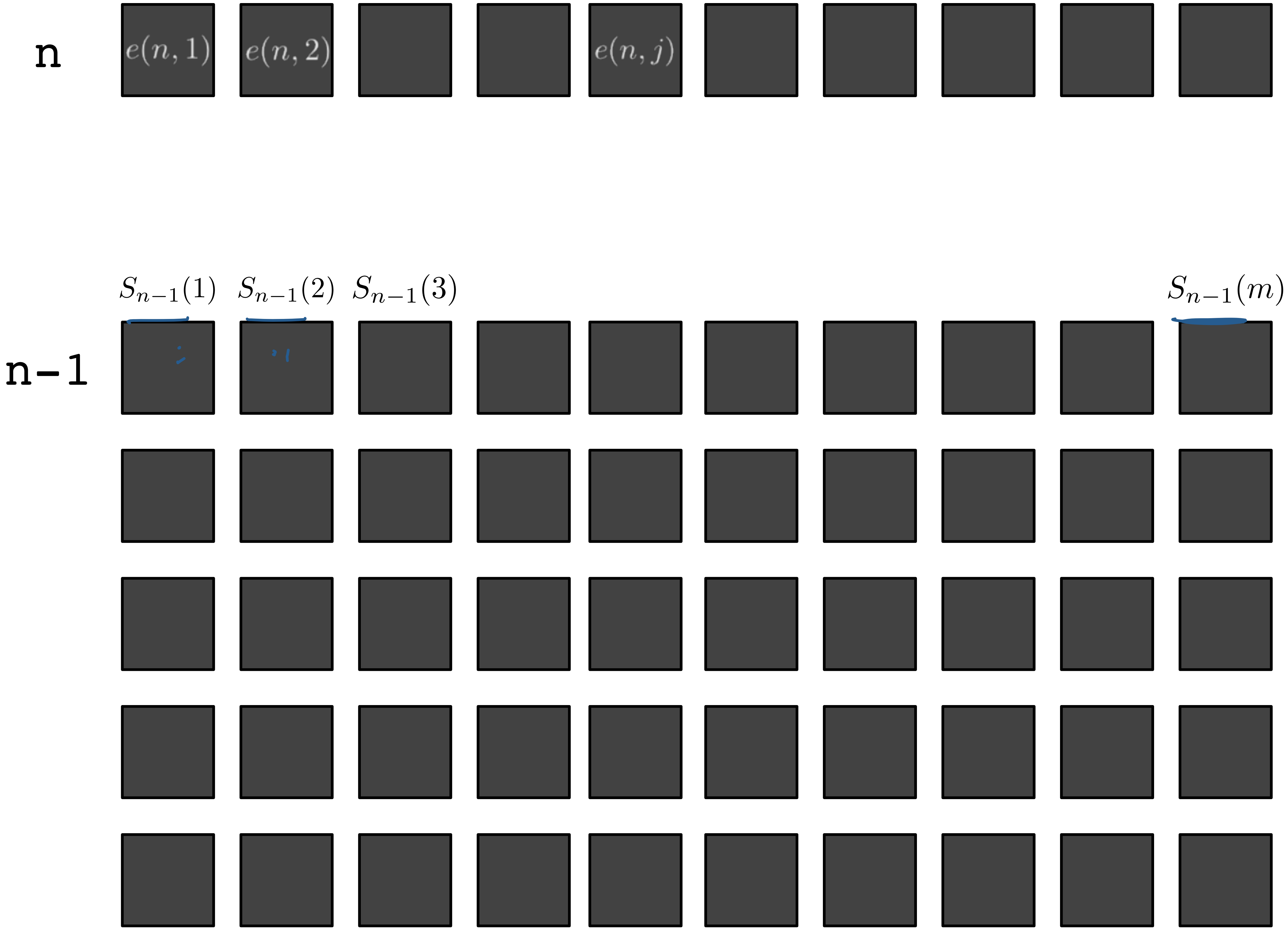
$$S_n(1), S_n(2), \ldots, S_n(m)$$

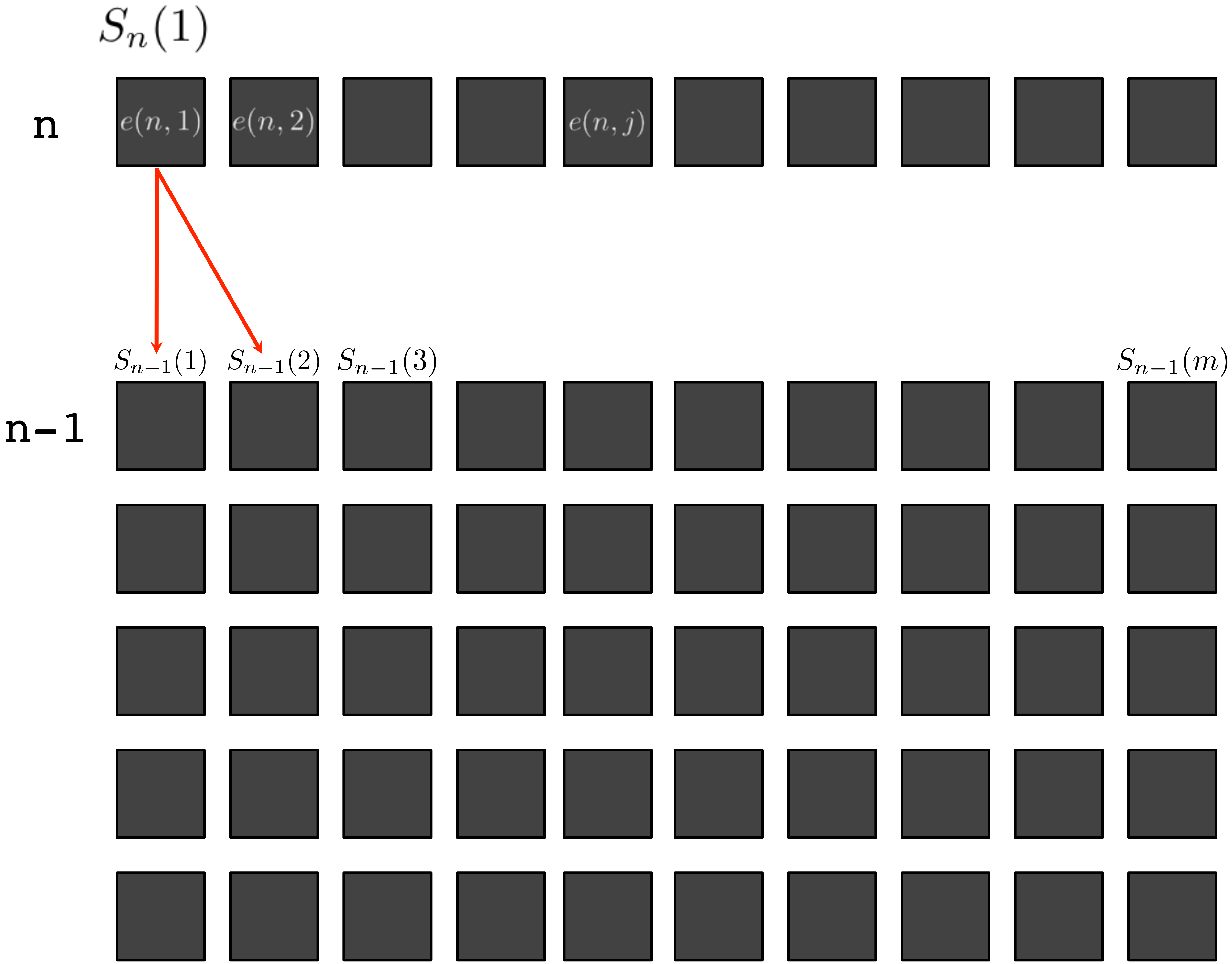
IDEA: COMPUTE + COMPARE



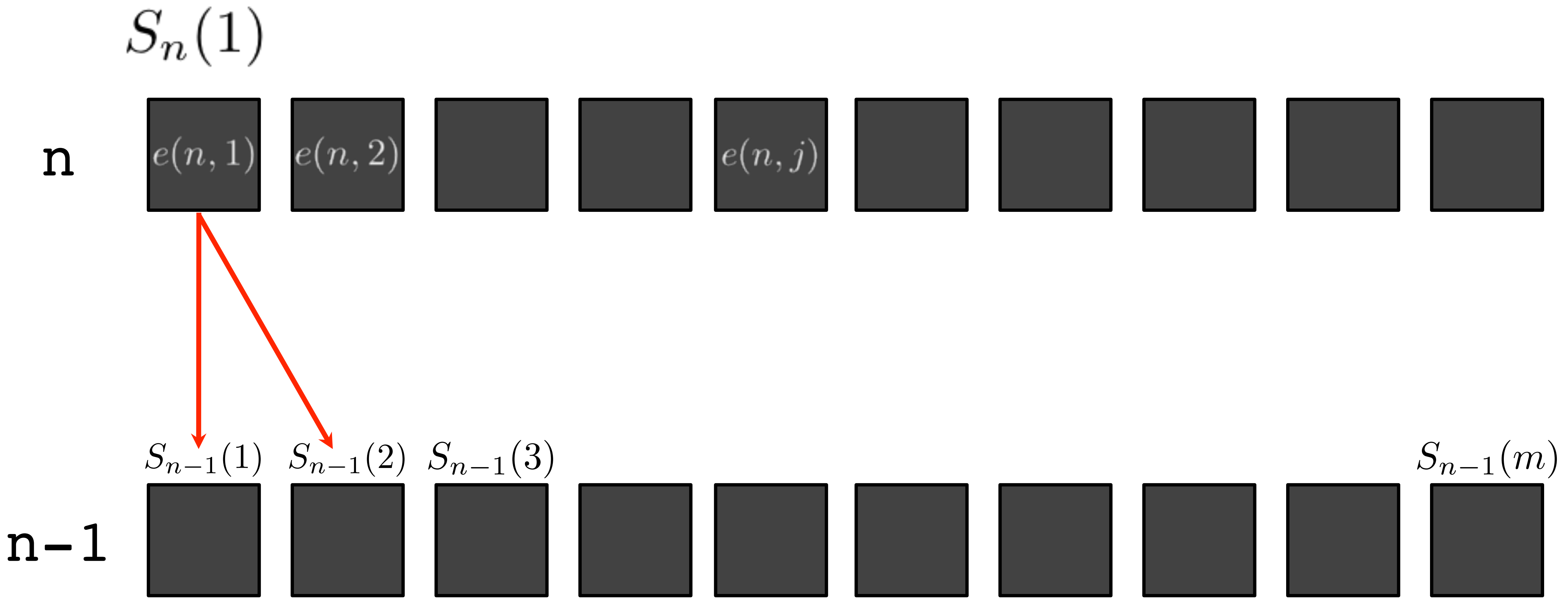
SMALLER PROBLEM APPROACH

IMAGINE YOU HAVE THE SOLUTION TO THE FIRST $n-1$ ROWS

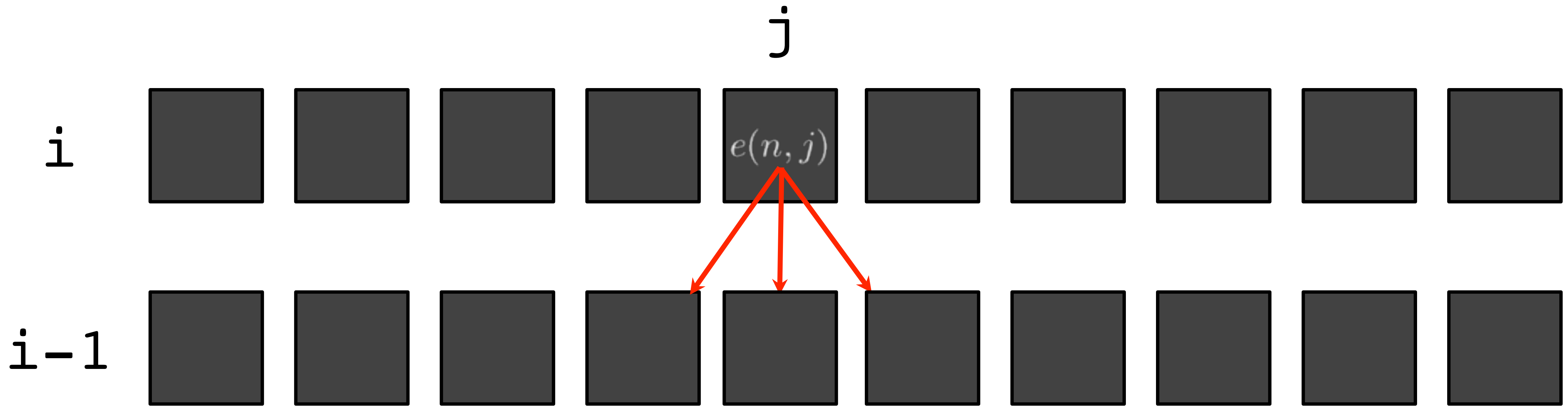

n
e(n, 1)
e(n, 2)
e(n, j)
n-1
S_{n-1}(1)
S_{n-1}(2)
S_{n-1}(3)
S_{n-1}(m)



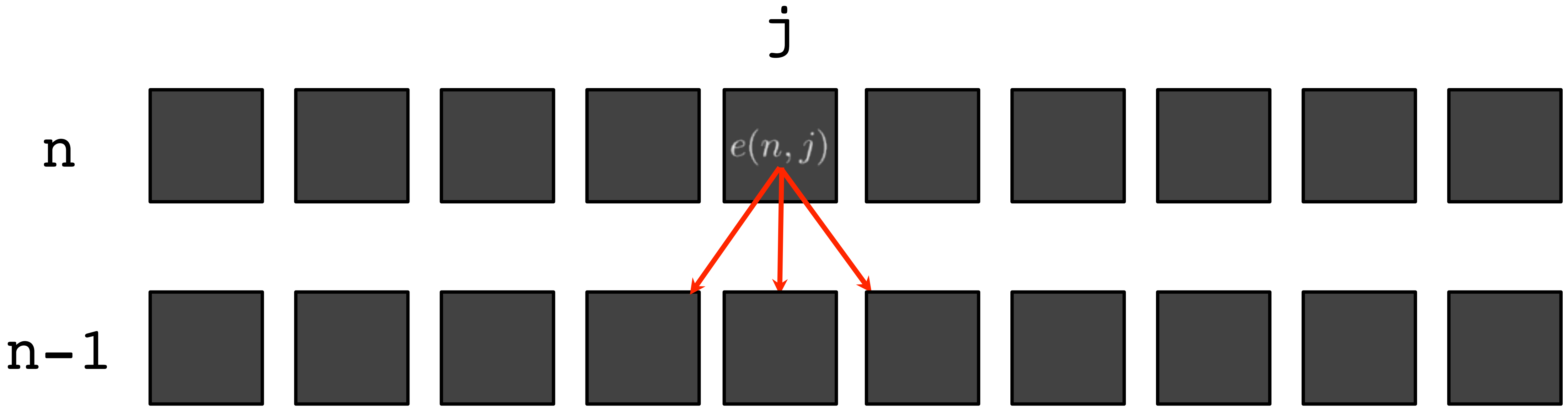
$S_n(1)$
n
$e(n,1)$
$e(n,2)$
$e(n,j)$
$S_{n-1}(1)$
$S_{n-1}(2)$
$S_{n-1}(3)$
$S_{n-1}(m)$
n-1



$$S_n(1) = e(n,1) + \min\{S_{n-1}(1), S_{n-1}(2)\}$$



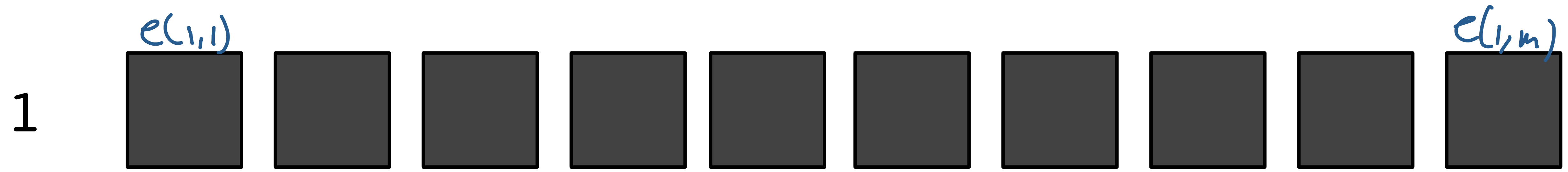
$$S_i(j) =$$



$$S_i(j) = e(i,j) + \min \begin{cases} S_{i-1}(j-1) \\ S_{i-1}(j) \\ S_{i-1}(j+1) \end{cases}$$

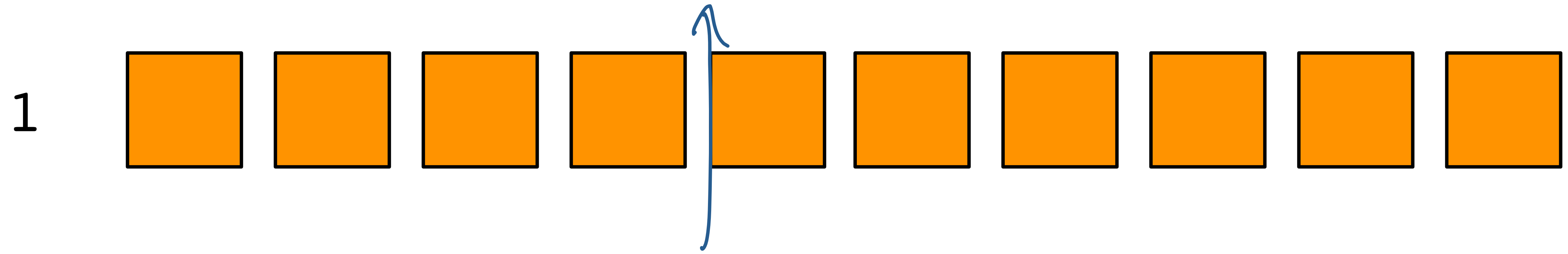
ALGORITHM

start at bottom of picture



ALGORITHM

start at bottom of picture. initialize $S_1(i) = e(1, i)$

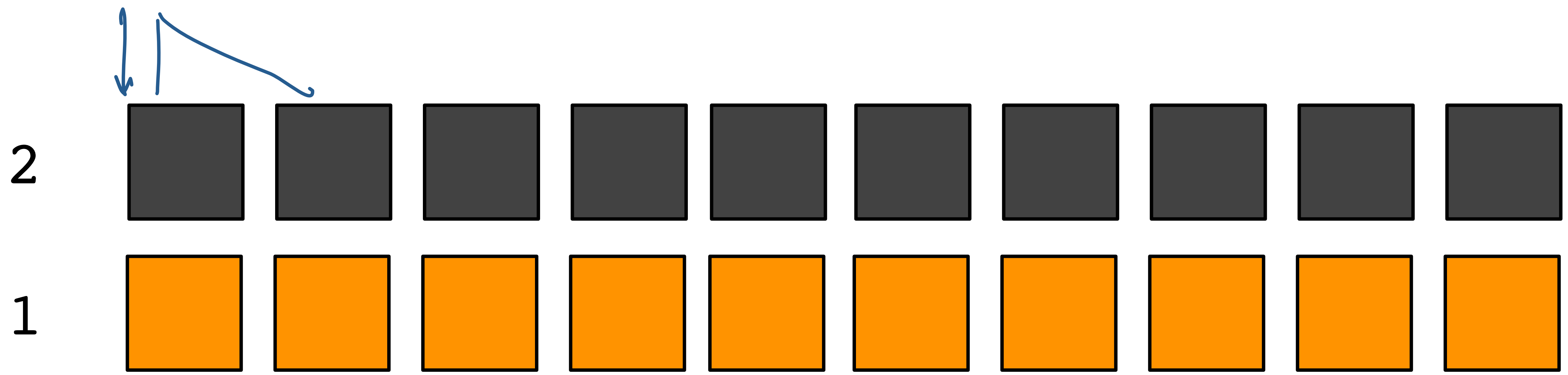


ALGORITHM

start at bottom of picture. initialize $S_1(i) = e(1, i)$

for `i=2,n` use formula to compute $S_{i+1}(\cdot)$

$$S_i(j) = e(i,j) + \min \begin{cases} S_{i-1}(j-1) \\ S_{i-1}(j) \\ S_{i-1}(j+1) \end{cases}$$

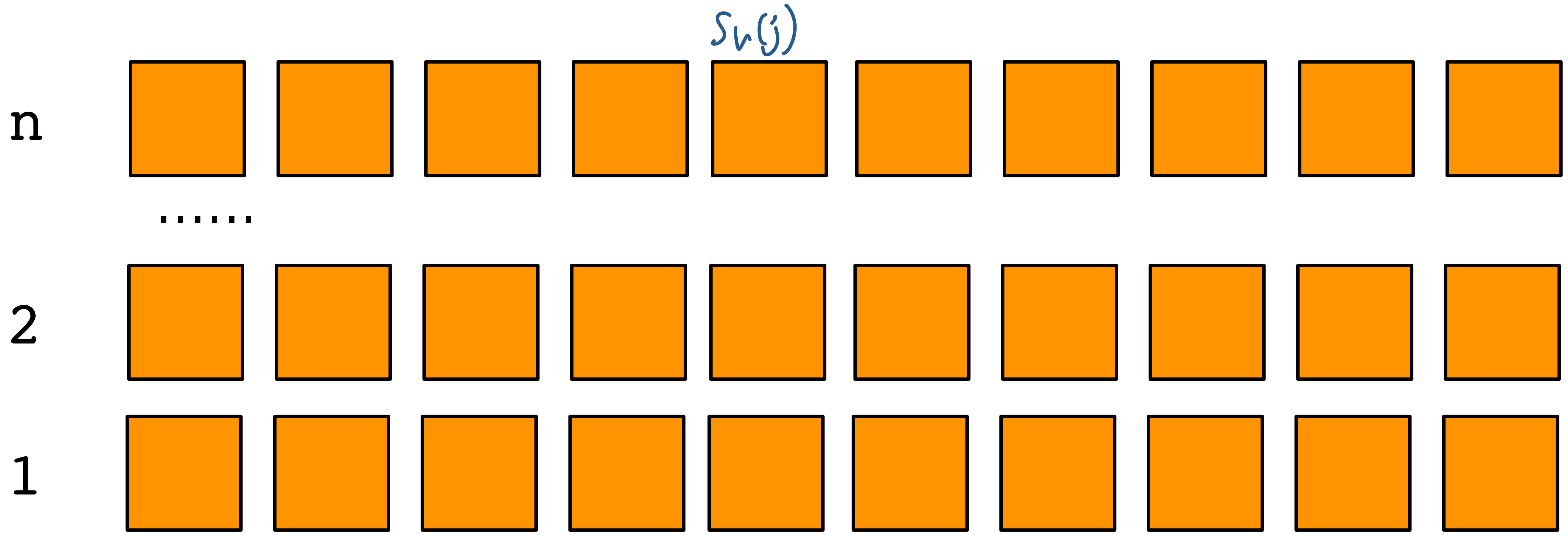


ALGORITHM

start at bottom of picture. initialize $S_1(i) = e(1, i)$

for `i=2,n` use formula to compute $S_{i+1}(\cdot)$

$$S_i(j) = e(i,j) + \min \begin{cases} S_{i-1}(j-1) \\ S_{i-1}(j) \\ S_{i-1}(j+1) \end{cases}$$



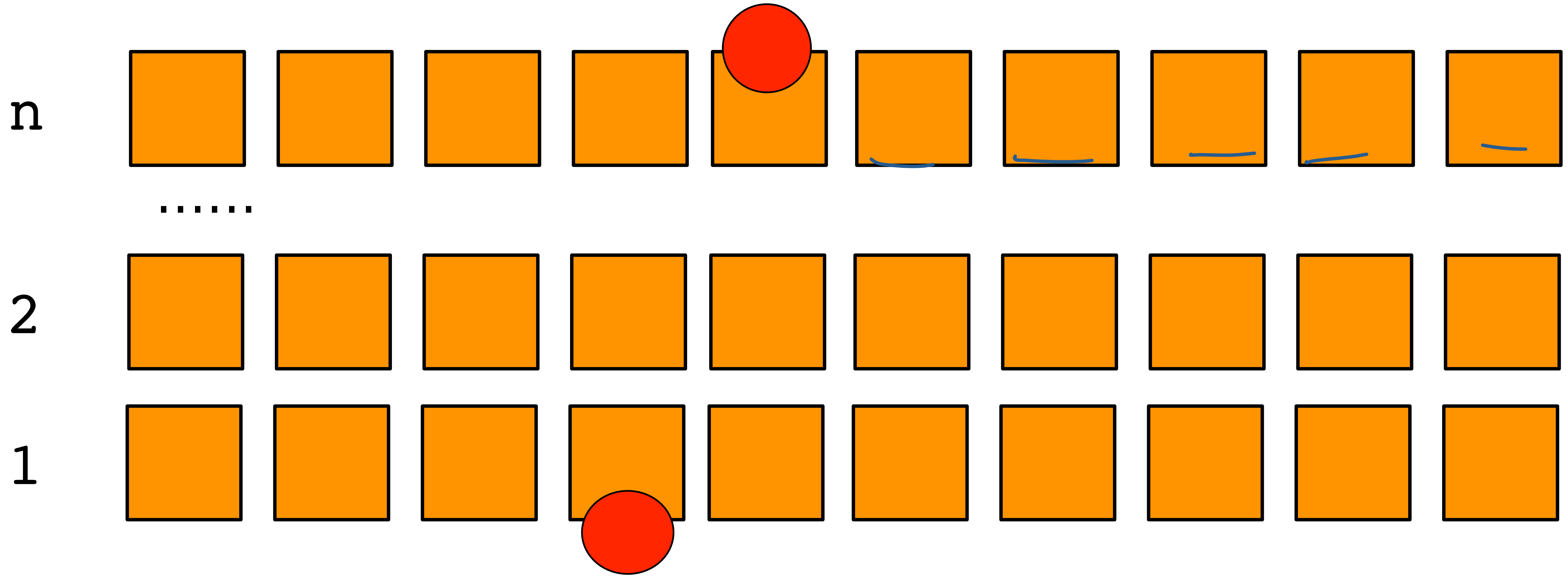
ALGORITHM

start at bottom of picture.   initialize $S_1(i) = e(1, i)$

for `i=2,n` use formula to compute $S_{i+1}(\cdot)$

$$S_i(j) = e(i,j) + \min \begin{cases} S_{i-1}(j-1) \\ S_{i-1}(j) \\ S_{i-1}(j+1) \end{cases}$$

pick best among top row, backtrack.



RUNNING TIME

start at bottom of picture. initialize $S_1(i) = e(1, i)$

for `i=2,n` use formula to compute $S_{i+1}(\cdot)$

$$S_i(j) = e(i,j) + \min \begin{cases} S_{i-1}(j-1) \\ S_{i-1}(j) \\ S_{i-1}(j+1) \end{cases}$$

pick best among top row, backtrack.