L9 5800

feb 15/17 2022

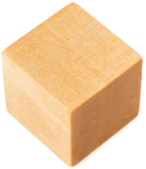
shelat

Knapsack

Sack has Capacity W

W

w_1 v_1



w_2 v_2

w_3 v_3

Each item
has a weight w_i
and a value v_i

Goal is to select
a set of items that
"**fit**" into the Knapsack
and have the **greatest** value.
Can only use each item once
(versus Logcutter)

Define a quantity that captures the optimal solution:

Best({1,…,n}, ~~C~~ W) : Max value attainable using items {1..n} with capacity W.

Consider the very first item. Is it part of the max solution?

$$\text{Best}(\{1\ldots n\}, W) = \max \begin{cases} \text{Best}(\{2\ldots n\}, W) \\ v_1 + \text{Best}(\{2\ldots n\}, W - w_1) \end{cases}$$

Define a quantity that captures the optimal solution:

Best({1,…,n}, C) : max value obtainable from items {1…n} that fit in sack of size C

Consider the very first item. Is it part of the max solution?

w_1 v_1

Define a quantity that captures the optimal solution:

Best({1,…,n}, C) : max value obtainable from items {1…n} that fit in sack of size C

Consider the very first item. Is it part of the max solution?

$$B(1 \ldots n, C) = \max \left\{ \begin{array}{ll} B(2 \ldots n, C) & \text{if not included} \\ v_1 + B(2 \ldots n-1, C - w_1) & \text{if in} \end{array} \right\}$$

Recursive structure

Either the best solution doesn't include item i

$$B(\{i \ldots n\}, W) = \max \begin{cases} Best(\{i+1, \ldots n\}, W) \\ V_i + Best(\{i+1, n\}, W - w_i) & \text{if } W - w_i \geq 0 \end{cases}$$

Or, it includes item i and the best solution for the remaining space, $W\text{-}w_i$

as long as the item still fits, i.e. $W - w_i \geq 0$

Recursive structure

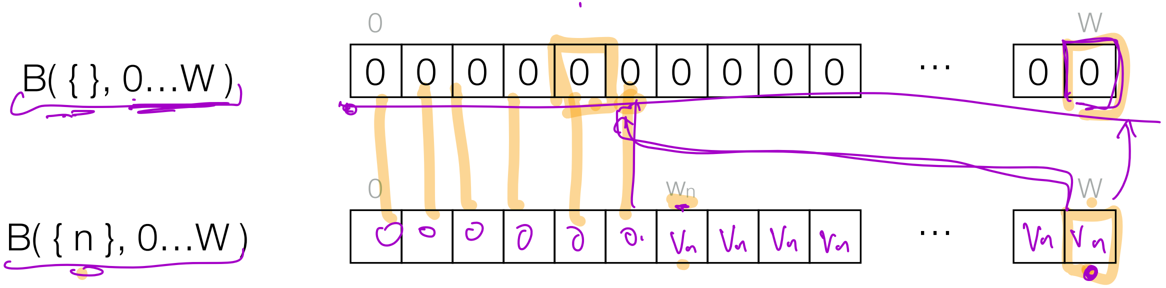
Either the best solution doesn't include item i

$$B(\{i \ldots n\}, W) = \max \begin{cases} B(\{i+1 \ldots n\}, W) \\ v_i + B(\{i+1 \ldots n\}, W - w_i) \end{cases} \quad \text{If } W - w_i \geq 0$$

Or, it includes item i and the best solution for the remaining space, $W\text{-}w_i$

Pick an order

Start from the last item



$$\max \begin{cases} Best(\{\}, c) \\ V_n + Best(\{\}, c - w_n) \end{cases}$$

$V_n + Best(\{\}, W - w_n)$

If $W - w_n > 0$

Pick an order

Start from the last item

B({ }, 0…W)

0												W
0	0	0	0	0	0	0	0	0	0	…	0	0

B({ n }, 0…W)

0						w_n						W
										…		

$$B(\{n\}, W) = \max \begin{cases} B(\{\}, W) \\ v_n + B(\{\}, W - w_n) & \text{If } W - w_n > 0 \end{cases}$$

Pick an order

Start from the last item

$B(\{\ \}, 0 \ldots W)$

0											W
0	0	0	0	0	0	0	0	0	0	…	0	0

$B(\{\ n\ \}, 0 \ldots W)$

0						w_n						W
0	0	0	0	0	0	v_n	v_n	v_n	v_n	…	v_n	v_n

$$B(\{n\}, W) = \max \begin{cases} B(\{\}, W) \\ v_n + B(\{\}, W - w_n) \end{cases} \quad \text{If } W - w_n > 0$$

Pick an order

Start from the last item

B({ }, 0…W)

0												W
0	0	0	0	0	0	0	0	0	0	…	0	0

B({ n }, 0…W)

0						w_n						W
0	0	0	0	0	0	v_n	v_n	v_n	v_n	…	v_n	v_n

B({ n-1…n }, 0…W)

						15		17				40
0						w_n		w_{n-1}				W
										…		

Pick an order

Start from the last item

B({ }, 0…W)

0												W
0	0	0	0	0	0	0	0	0	0	…	0	0

B({ n }, 0…W)

0						w_n						W
0	0	0	0	0	0	v_n	v_n	v_n	v_n	…	v_n	v_n

B({ n-1…n }, 0…W)

						15		17				40
0						w_n		w_{n-1}				W
										…		

$$B(\{i\ldots n\}, W) = \max \begin{cases} B(\{i+1\ldots n\}, W) \\ v_i + B(\{i+1\ldots n\}, W - w_i) \end{cases}$$

Example

	1	2	3	4	5	6
v_i	4	5	9	1	10	3
w_i	1	3	2	4	7	2

$$C = 15$$

$B(\{\}, \ldots)$

0					5					10					W=15
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

$B(\{6\}, \ldots)$

0	1	2			5					10					W=15
0	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3

$$B(\{i \ldots n\}, W) = \max \begin{cases} B(\{i+1 \ldots n\}, W) \\ v_i + B(\{i+1 \ldots n\}, W - w_i) \end{cases}$$

If $W - w_i > 0$

Example

	1	2	3	4	5	6
v_i	4	5	9	1	10	3
w_i	1	3	2	4	7	2

$$C = 15$$

$B(\{\}, \ldots)$

0					5					10					W=15
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

$B(\{6\}, \ldots)$

0					5					10					W=15
0	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3

$$B(\{i \ldots n\}, W) = \max \begin{cases} B(\{i+1 \ldots n\}, W) \\ v_i + B(\{i+1 \ldots n\}, W - w_i) \end{cases}$$

If $W - w_i > 0$

Example

	1	2	3	4	5	6
v_i	4	5	9	1	10	3
w_i	1	3	2	4	7	2

$$C = 15$$

	0					5					10					W=15
$B(\{6\}, \ldots)$	0	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3
$B(\{5,6\}, \ldots)$	0	0	3	3	3	3	3	10	10	13				13	13	13

$$B(\{i \ldots n\}, W) = \max \begin{cases} B(\{i+1 \ldots n\}, W) \\ v_i + B(\{i+1 \ldots n\}, W - w_i) \end{cases}$$

If $W - w_i > 0$

Example

	1	2	3	4	5	6
v_i	4	5	9	1	10	3
w_i	1	3	2	4	7	2

$$C = 15$$

$B(\{6\}, \ldots)$

0					5					10					W=15
0	0	3	3	3	3	3	3	3	3	3	3	3	3	3	3

$B(\{5,6\}, \ldots)$

0					5					10					W=15
0	0	3	3	3	3	3	10	10	13	13	13	13	13	13	13

$$B(\{i \ldots n\}, W) = \max \begin{cases} B(\{i+1 \ldots n\}, W) \\ v_i + B(\{i+1 \ldots n\}, W - w_i) \end{cases}$$

If $W - w_i > 0$

Example

(exercise for home)

	1	2	3	4	5	6
v_i	4	5	9	1	10	3
w_i	1	3	2	4	7	2

$$C = 15$$

$B(\{5,6\}, \ldots)$

0					5					10					W=15
0	0	3	3	3	3	3	10	10	13	13	13	13	13	13	13

$B(\{4,5,6\}, \ldots)$

0					5					10					W=15

$$B(\{i \ldots n\}, W) = \max \begin{cases} B(\{i+1 \ldots n\}, W) \\ v_i + B(\{i+1 \ldots n\}, W - w_i) \end{cases}$$

If $W - w_i > 0$

Knapsack($\{w_i, v_i\}_n$, W)

Initialize B({n-1}, 0…W) = 0

for i from n to 1

for j from 0 to W

$$B(i, j) = \max \begin{cases} B(i+1, j) \\ v_i + B(i+1, j - w_i) \end{cases}$$

as long as $j > w_i$
because otherwise,
this term is negative

Return B(1,W)

Knapsack($\{w_i, v_i\}_n$, W)

Initialize B({n-1}, 0…W) = 0

for i from n to 1

for j from 0 to W

B(i, j) = B(i+1, j)

if j > wi and B(i+1, j-wi)+ vi > S(i, j)

B(i,j) = B(i+1, j-wi)+ vi

Return B(1,W)

How can we determine WHICH items are selected?

Knapsack($\{w_i, v_i\}_n$, W)

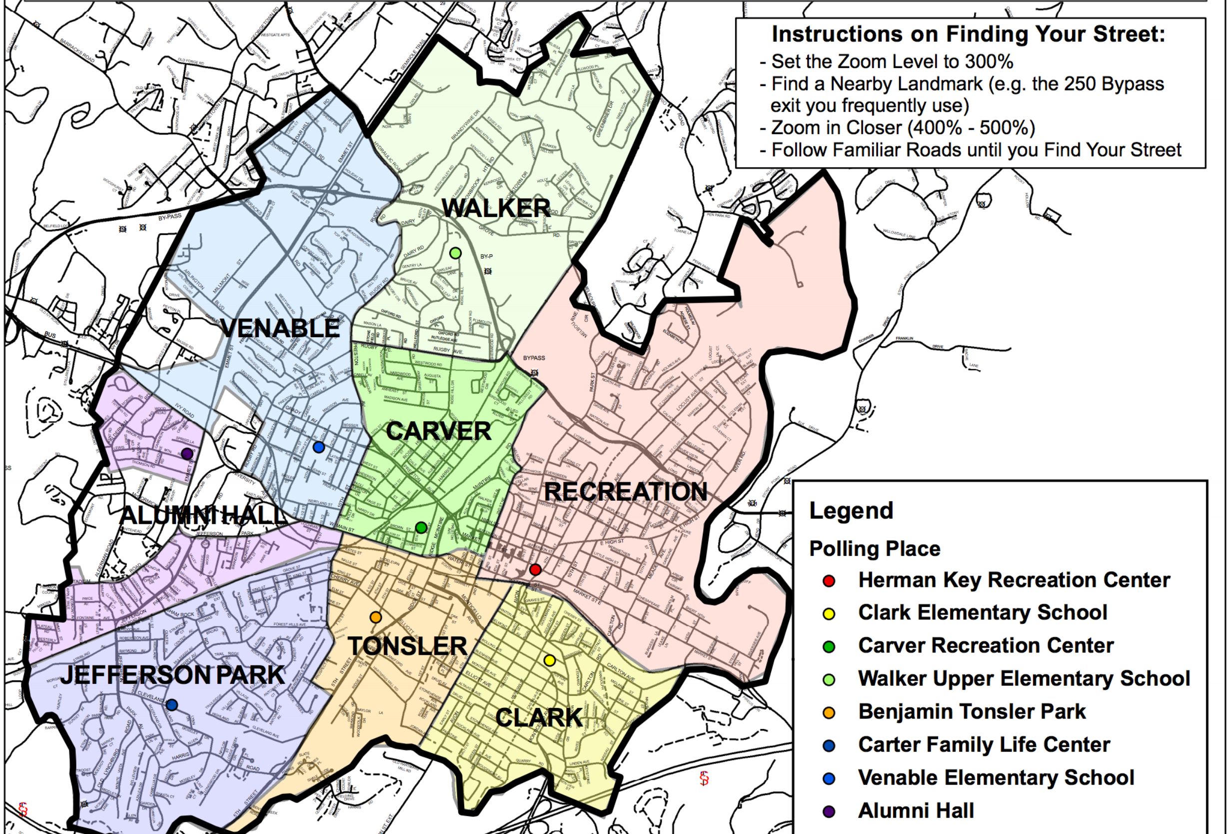
```
Initialize B( {n-1}, 0…W) = 0

for i from n to 1

        for j from 0 to W

            B( i, j ) = B( i+1, j)
            pick( i, j ) = false
            if j > wi and B( i+1, j-wi )+ vi > B( i, j)
                B(i,j) = B( i+1, j-wi )+ vi
                pick( i, j ) = true
//Backtrack to find solution
cap = W, sol = {}
for i from 1 to n

        if picked(i,cap) = true { sol = sol + {i};  cap = cap - wi; }
```

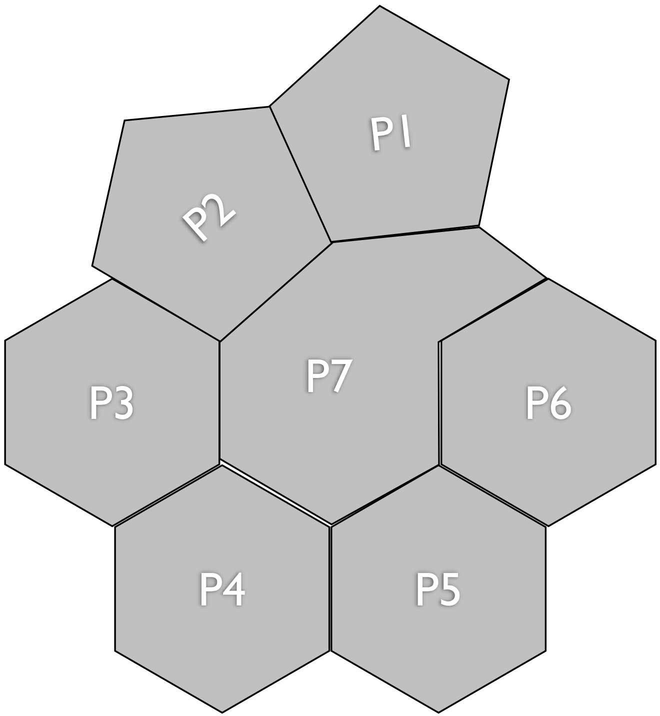
Gerrymander

Congressional District 5

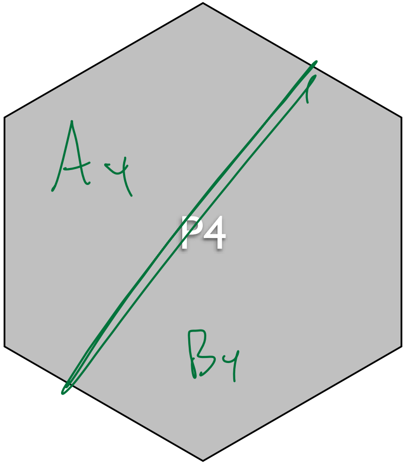
Virginia (11 Distr

Map of Charlottesville Precincts and Polling Places

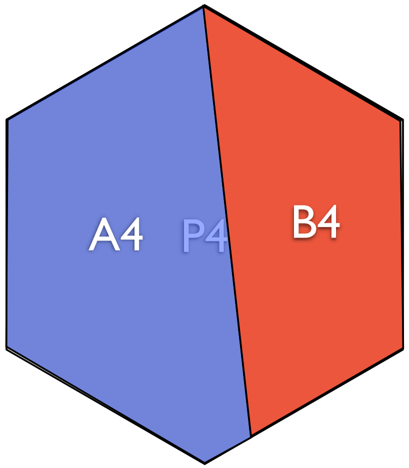




EACH precinct has a fraction of party A voters & party B voters.



Each precinct has a split between voters.



Each precinct P_i has a A_i voters for party A and B_i voters for party B.

gerrymander problem

n is even

given: precinct population M, and precincts $A_1, A_2, A_3, \ldots A_n$

$\uparrow$ voters for party A in P_i.

output: district D_1, D_2 such that

$$|D_1| = |D_2| \quad \text{(same \# of precincts)}$$

$$A(D_1) > \frac{Mn}{4} \qquad \text{majority A in } D_1$$

$$A(D_2) > \frac{Mn}{4} \qquad \text{majority A in } D_2.$$

or "failure" if not possible

gerrymander problem

given: m $\quad A_1, A_2, \ldots, A_n$ n is even

Total voters — Voters for party A in each precinct

output: D_1, D_2

such that $|D_1| = |D_2|$

$$A(D_1) > \frac{mn}{4}$$

$$A(D_2) > \frac{mn}{4}$$

or "failure" if no such solution is possible

Example

Imagine 4 precincts divided into 2 districts.

A1=65	A3=45
A2=57	A4=47

M=100

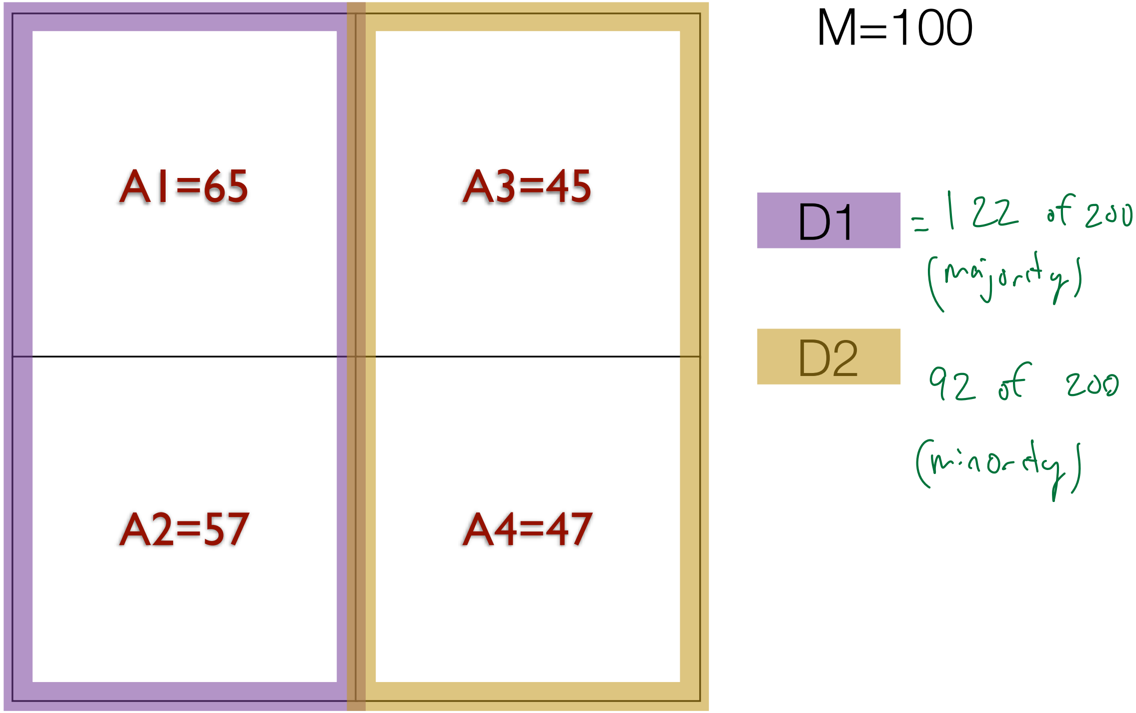
D1

D2

Example

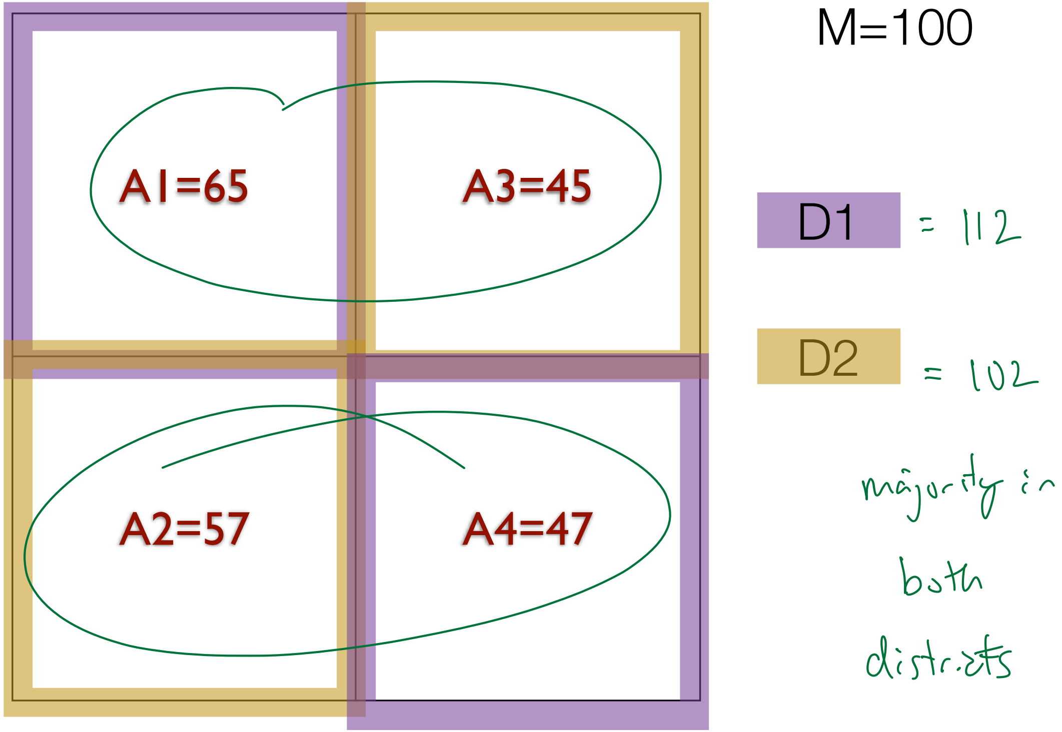
Imagine 4 precincts divided into 2 districts.

M=100



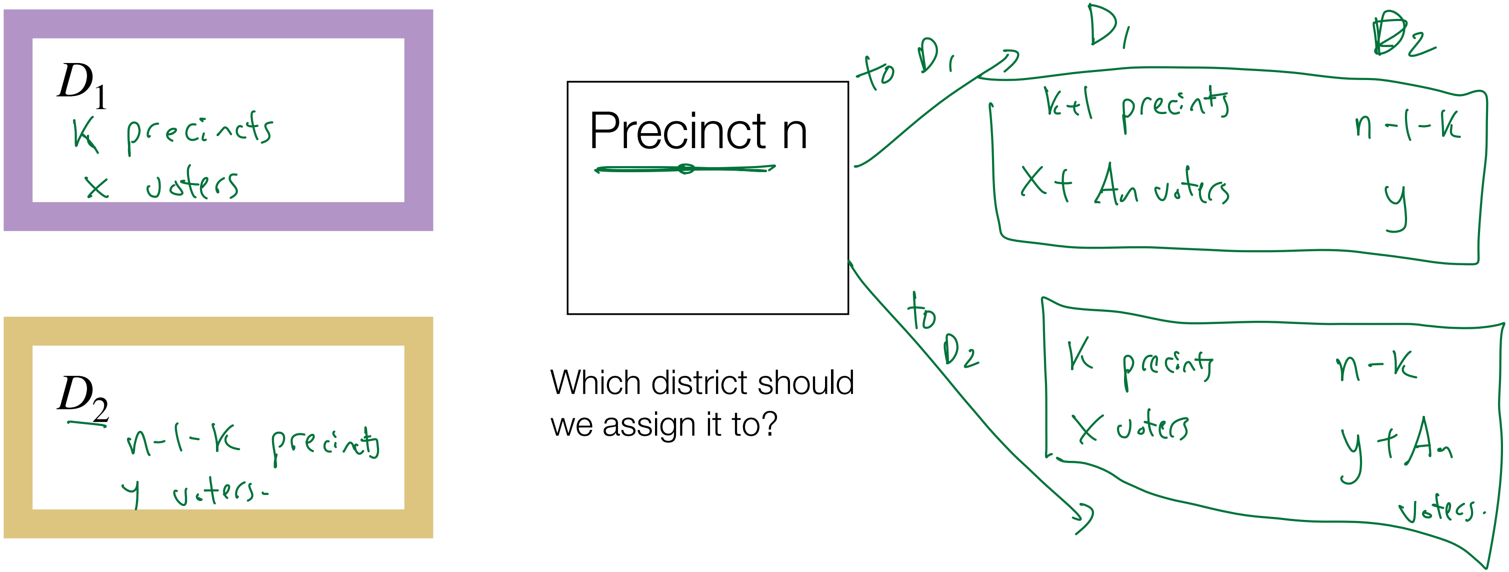
Example

Imagine 4 precincts divided into 2 districts.



Gerrymander

imagine very last precinct and how it is assigned:



Which district should we assign it to?

Gerrymander

imagine very last precinct and how it is assigned:

D_1
k precincts
x votes for A

D_2
$n - k - 1$ precincts
y votes for A

Precinct n

Which district should we assign it to?

Gerrymander

imagine very last precinct and how it is assigned:

D_1 D_2

D_1
k precincts
x votes for A

Precinct n

D_2
$n - k - 1$ precincts
y votes for A

Which district should we assign it to?

Gerrymander

imagine very last precinct and how it is assigned:

D_1
k precincts
x votes for A

D_2
$n-k-1$ precincts
y votes for A

Precinct n

Which district should we assign it to?

	D_1	D_2
Option 1	$k+1$ precincts, $x+A_n$ votes	$n-k-1$ precincts, y votes
Option 2	k precincts, x	$n-k$ precincts, $y+A_n$

Gerrymander

Boolean

$S_{j,k,x,y}$ = True if

D_1 has k precincts and x voters and

D_2 has (j-k) precincts and y voters.

Gerrymander

$$S_{j,k,x,y} = \text{TRUE if there exists an assignment of the first } j \text{ precincts such that } |D_1| = k,\ A(D_1) = x,\ A(D_2) = y$$

Gerrymander

$S_{j,k,x,y}$ = there is a split of first **j** precincts in which |D1|=**k** and **x** people in D1 vote A, **y** people in D2 vote A

How can we express this value in an equation:

$S_{j,k,x,y}$ = if we assign jth precinct to D_1 and $S_{j-1,k-1,x-A_j,y} = T$

OR

or we asign to D_2 and $S_{j-1,k,x,y-A_j} = T$

$$S_{j,k,x,y} = S_{j-1,k-1,x-A_j,y} \;\vee\; S_{j-1,k,x,y-A_j}$$

Gerrymander(P,A,m)

initialize array S[0,0,0,0]

$$S_{j,k,x,y} = S_{j-1,k-1,x-A_j,y} \;\vee\; S_{j-1,k,x,y-A_j}$$

Gerrymander(P,A,m)

```
initialize array S[0,0,0,0]
for j=1,...,n
  for k=1,...,j
   for x=0,...,jm
    for y=0,...,jm
          fill table according to equation

 search for true entry at S[n, n/2, >(mn/4)…mn, >(mn/4)…mn]
```

Problem: reduce image width

scaling: distortion

deleting column: distortion

delete the most invisible seam

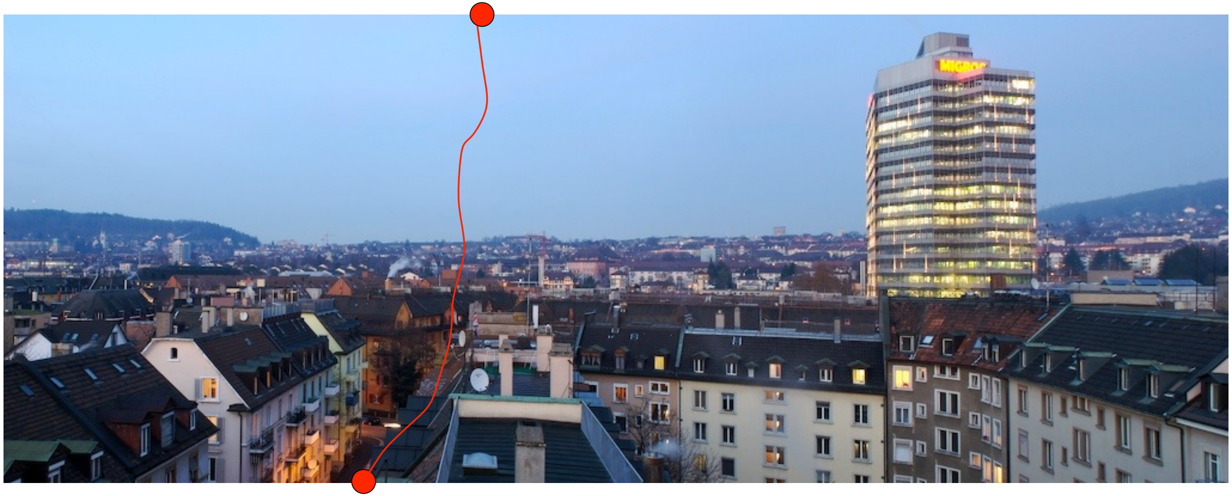
http://www.youtube.com/watch?v=qadw0BRKeMk

http://www.youtube.com/watch?v=qadw0BRKeMk

DEMO?

Demo

WHICH SEAM TO DELETE?



ENERGY OF AN IMAGE

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

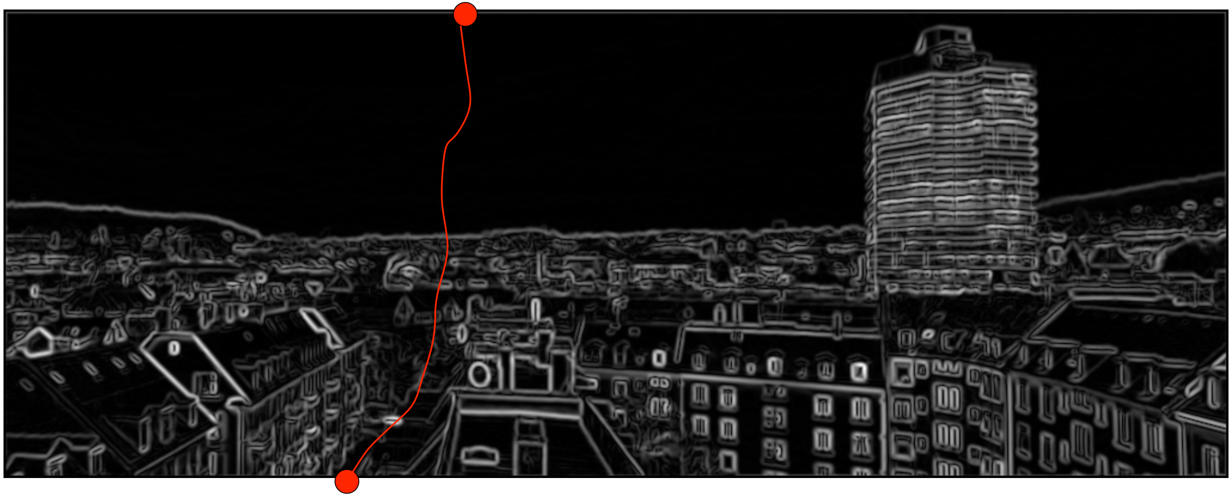
"magnitude of gradient at a pixel"

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

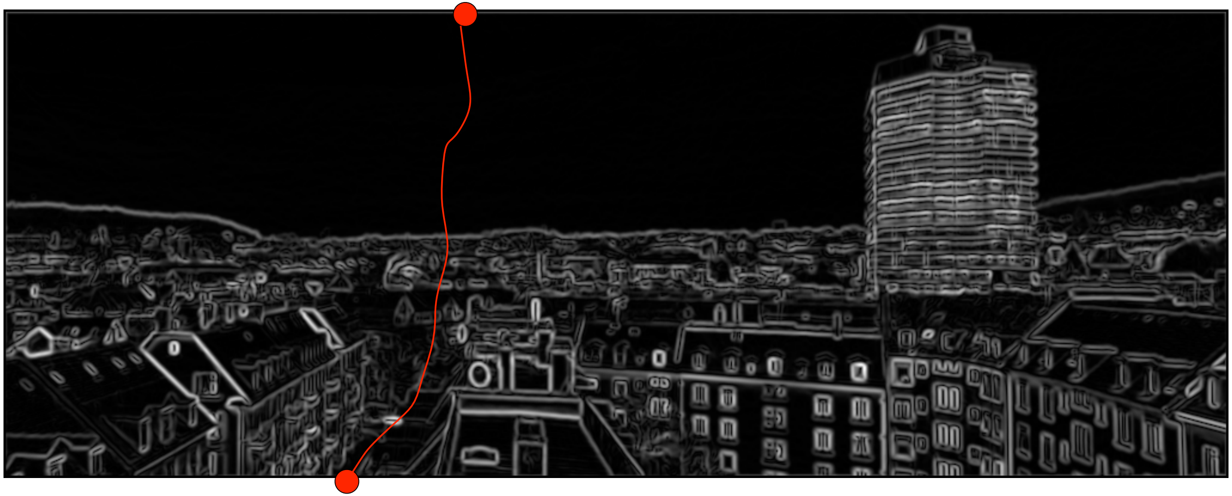
energy of sample image

thanks to Jason Lawrence for gradient software

BEST SEAM HAS LOWEST ENERGY



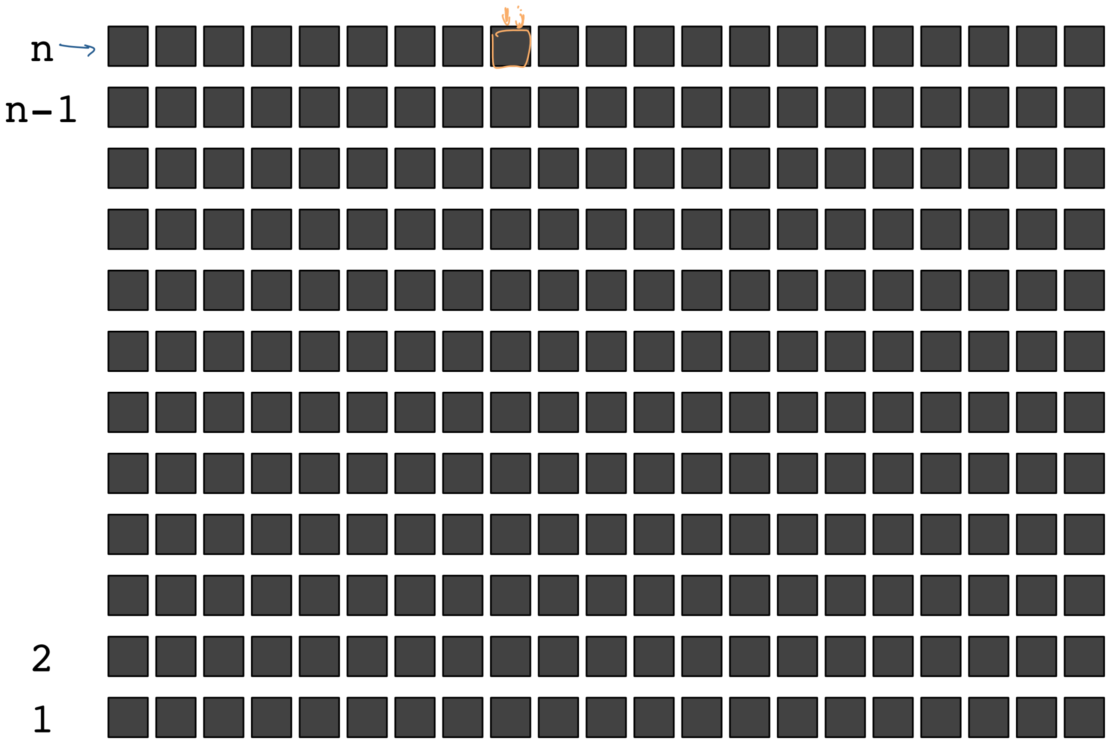
Finding lowest energy seam?



Define a variable:

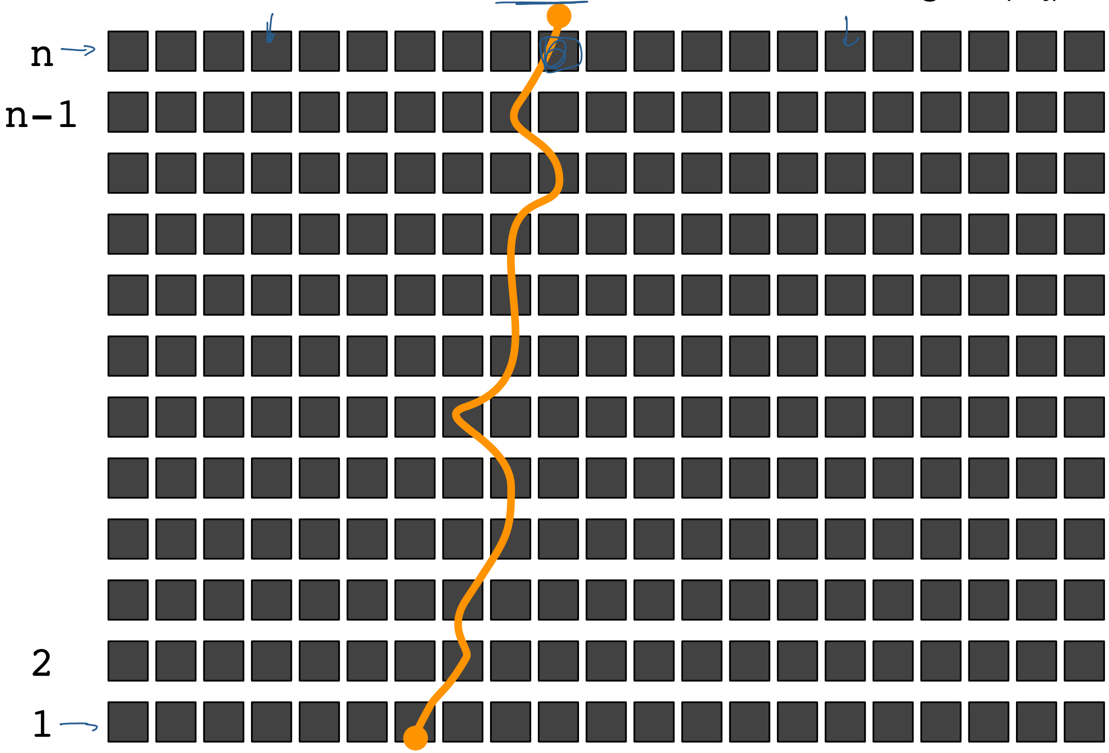
$$S_i(j)$$

definition: $S_n(j)$ Value of lowest energy seam that ends at (n,j)



definition:

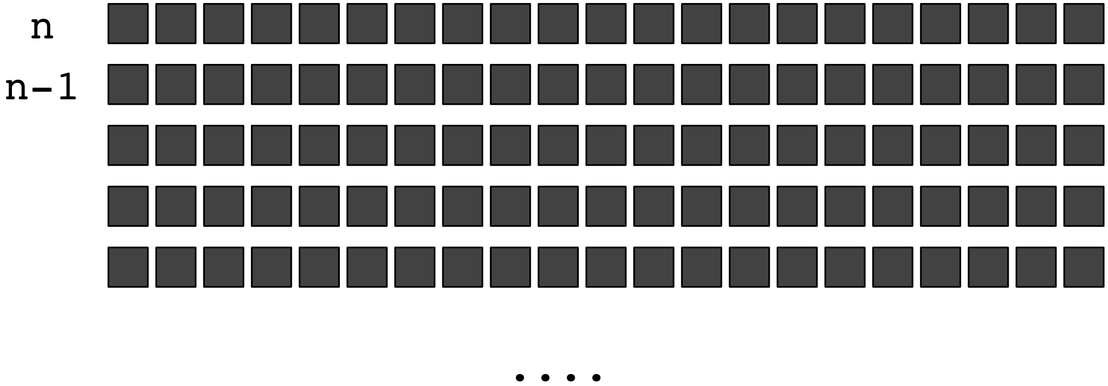
$S_n(j)$ best seam ending at (n,j)



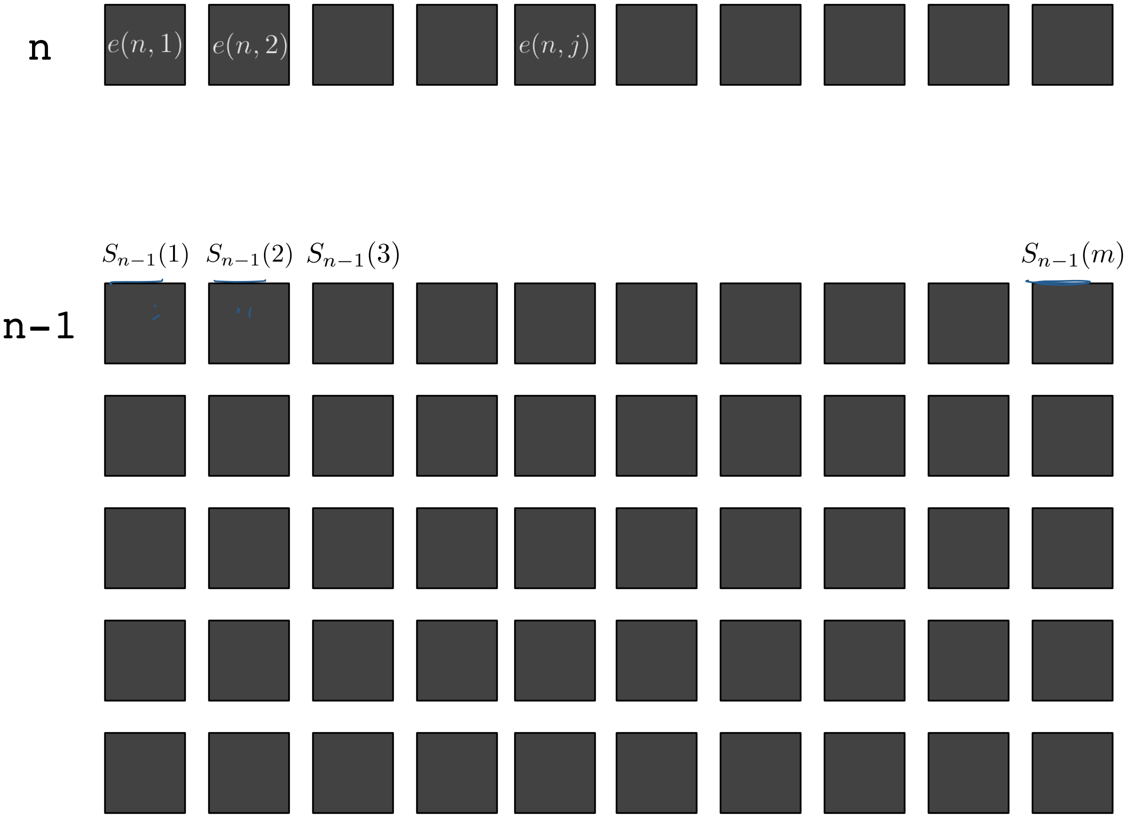
BEST SEAM TO DELETE HAS TO BE THE BEST AMONG

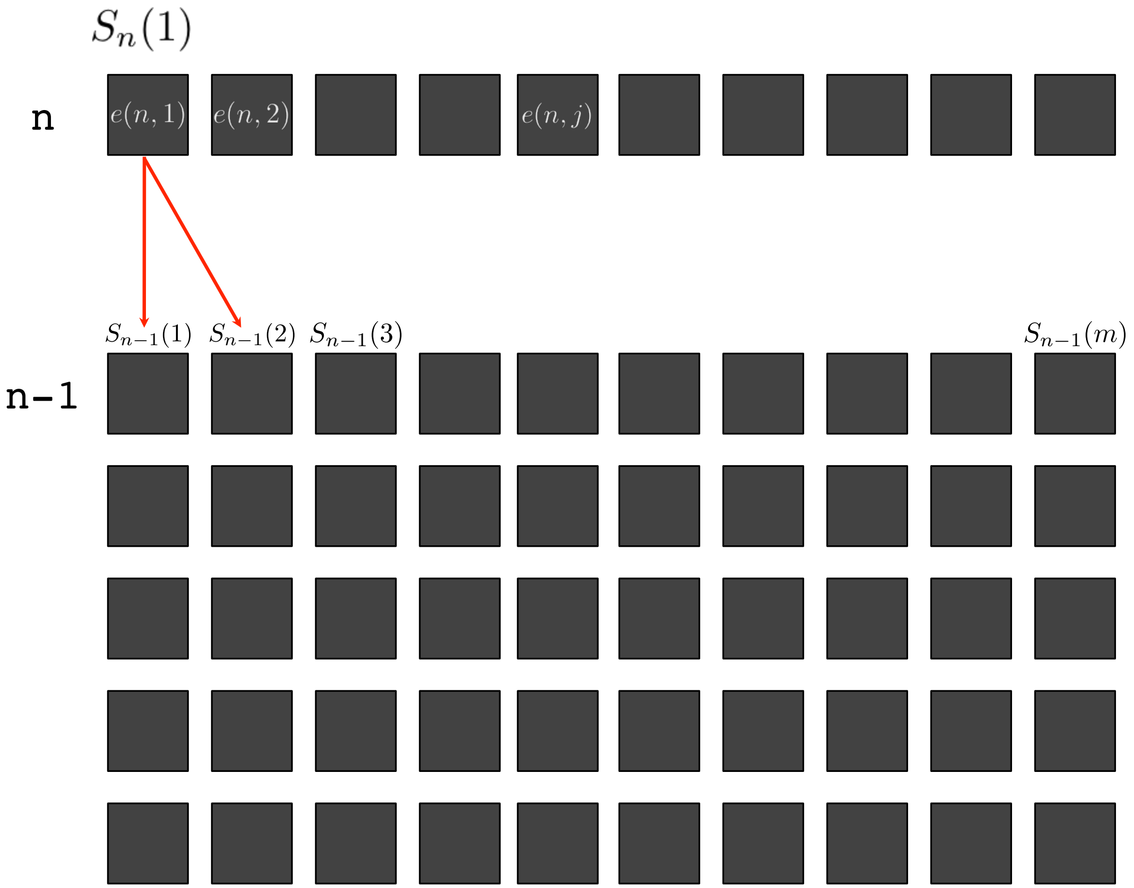
$$S_n(1), S_n(2), \ldots, S_n(m)$$

IDEA: COMPUTE + COMPARE

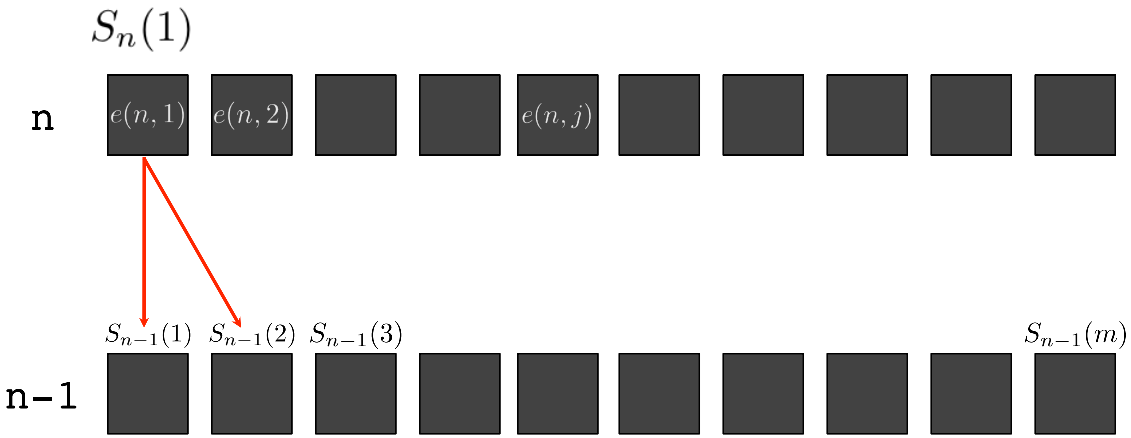


IMAGINE YOU HAVE THE
SOLUTION TO THE
FIRST N-1 ROWS

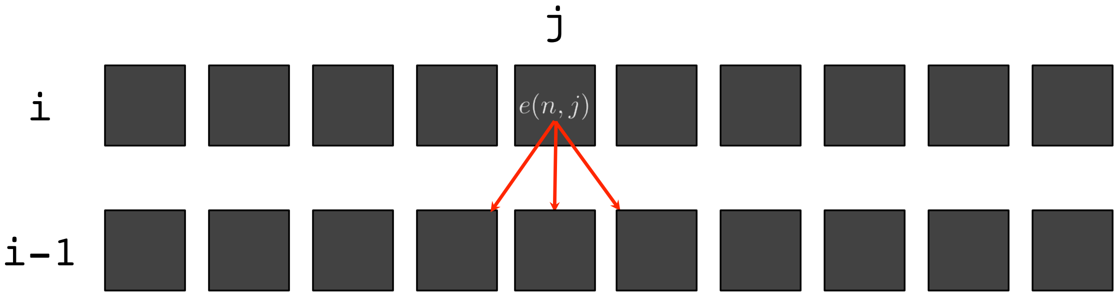

n
$e(n,1)$
$e(n,2)$
$e(n,j)$
$S_{n-1}(1)$
$S_{n-1}(2)$
$S_{n-1}(3)$
$S_{n-1}(m)$
n-1



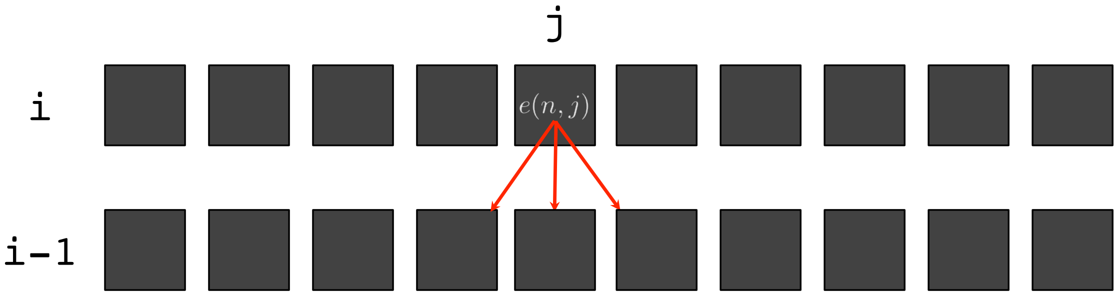
$S_n(1)$
n
$e(n,1)$
$e(n,2)$
$e(n,j)$
$S_{n-1}(1)$
$S_{n-1}(2)$
$S_{n-1}(3)$
$S_{n-1}(m)$
n-1



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



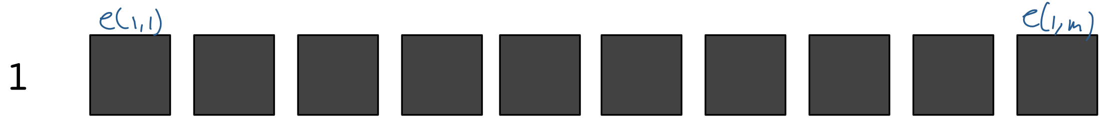
$$S_i(j) =$$



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

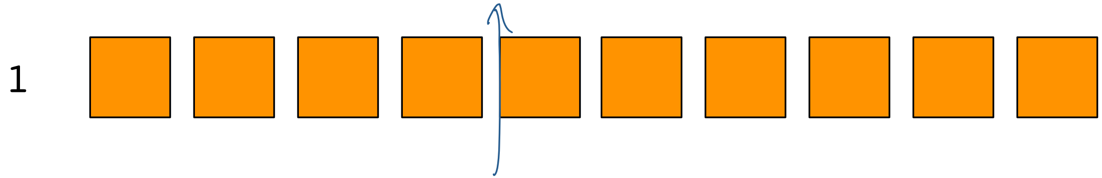
ALGORITHM

start at bottom of picture



ALGORITHM

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

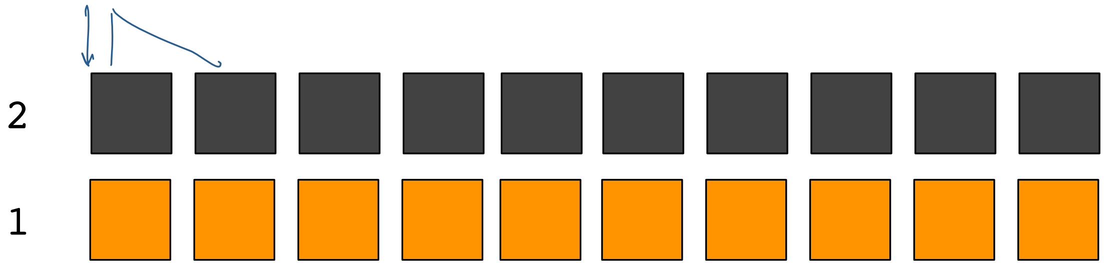


ALGORITHM

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

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

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

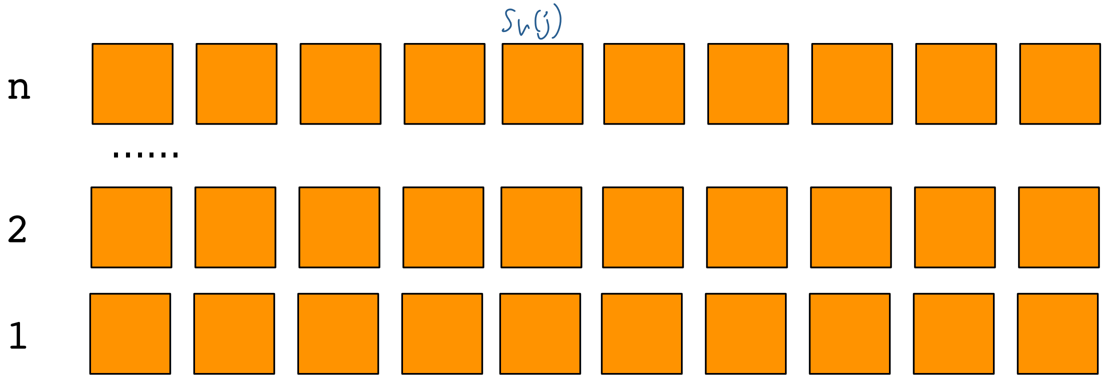


ALGORITHM

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

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

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



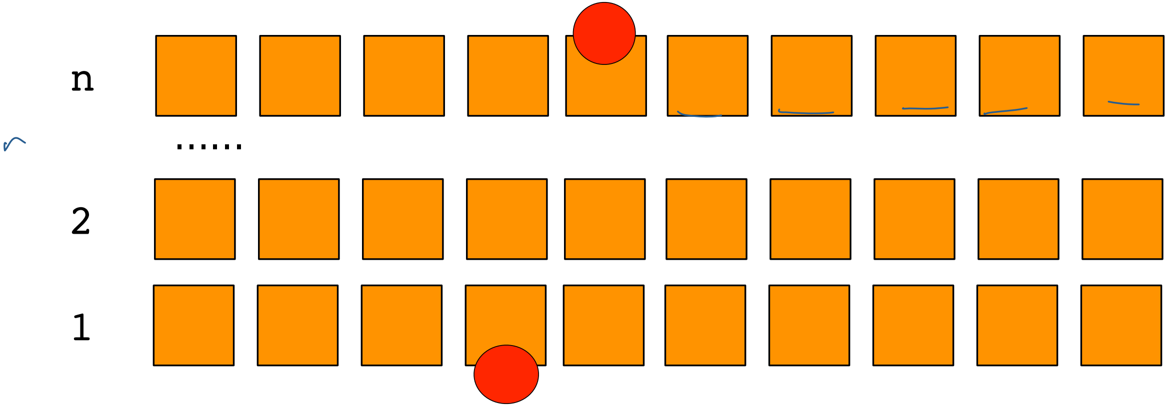
ALGORITHM

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.