"Hw 11" out later today

Practice Final next Tue here

Dec 17 7-10pm in this room

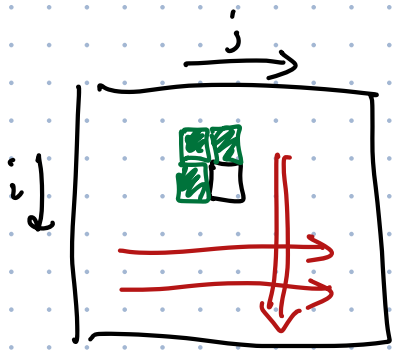
Conflict — Dec 16 8am — please fill out reg form by Fri

Edit distance

DIST
~~DISTANCES~~
DISTANCES

Edit(i,j) = edit distance between A[1..i] and B[1..j]

$$Edit(i,j) = \begin{cases} i & \text{if } j = 0 \\ j & \text{if } i = 0 \\ \min \begin{cases} Edit(i,j-1)+1 & \leftarrow \text{ins} \\ Edit(i-1,j)+1 & \leftarrow \text{del} \\ Edit(i-1,j-1)+[A[i] \neq B[j]] & \leftarrow \text{rep} \end{cases} & \text{otherwise} \end{cases}$$



$O(n^2)$ time

Given the table we can compute the optimal edit sequence

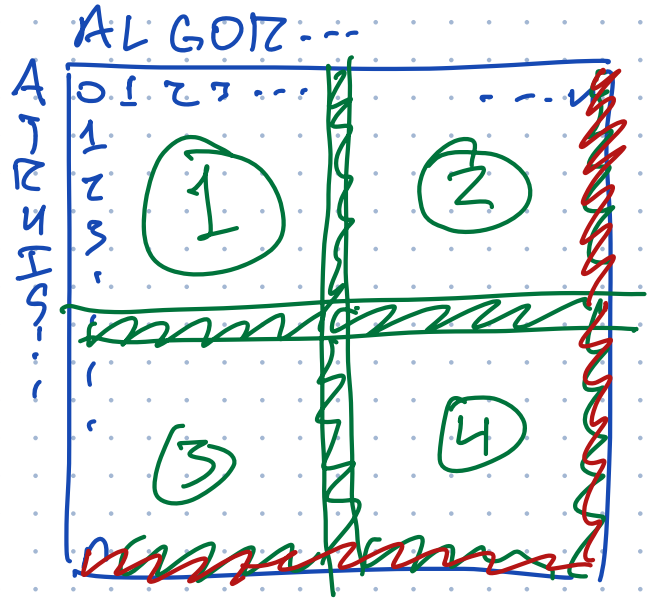
$O(n^2)$ time
$O(n^2)$ space

		A	L	G	O	R	I	T	H	M
	0	1	2	3	4	5	6	7	8	9
A	1	0	1	2	3	4	5	6	7	8
L	2	1	0	1	2	3	4	5	6	7
T	3	2	1	1	2	3	4	4	5	6
R	4	3	2	2	2	2	3	4	5	6
U	5	4	3	3	3	3	3	4	5	6
I	6	5	4	4	4	4	3	4	5	6
S	7	6	5	5	5	5	4	4	5	6
T	8	7	6	6	6	6	5	4	5	6
I	9	8	7	7	7	7	6	5	5	6
C	10	9	8	8	8	8	7	6	6	6

IF we only want distance, we only need to maintain one row

$O(n^2)$ time
$O(n)$ space

Porque no los dos?



Given first row
first col
Compute last row
last col

Chowdhury
Ramachandran
2006ish

$T(n) = 4T(\frac{n}{2}) + O(n) = O(n^2)$

$S(n) = S(\frac{n}{2}) + O(n) = O(n)$

Edit		A	L	G	O	R	I	T	H
	0	1	2	3	4	5	6	7	8
A	1								
L	2								
T	3								
R	4								
U	5								
I	6								
S	7								
T	8								

Edit		A	L	G	O	R	I	T	H
	0	1	2	3	4	5	6	7	8
A	1				3				
L	2				2				
T	3				2				
R	4	3	2	2	2				
U	5								
I	6								
S	7								
T	8								

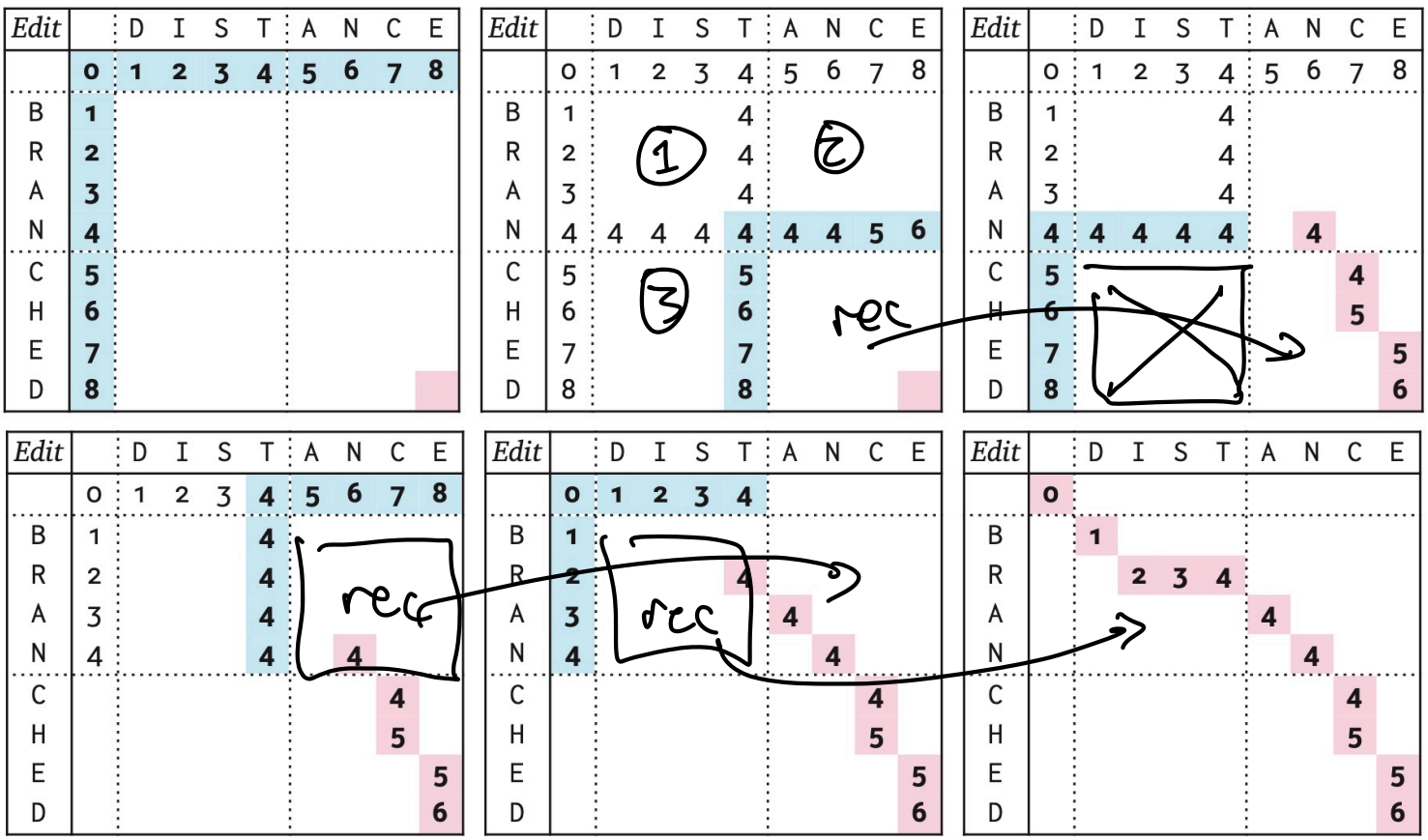
Edit		A	L	G	O	R	I	T	H
	0	1	2	3	4	5	6	7	8
A	1				3				7
L	2				2				6
T	3				2				5
R	4	3	2	2	2	2	3	4	5
U	5								
I	6								
S	7								
T	8								

Edit		A	L	G	O	R	I	T	H
	0	1	2	3	4	5	6	7	8
A	1				3				7
L	2				2				6
T	3				2				5
R	4	3	2	2	2	2	3	4	5
U	5				3				5
I	6				4				5
S	7				5				5
T	8	7	6	6	6	6	5	4	5

Edit		A	L	G	O	R	I	T	H
	0	1	2	3	4	5	6	7	8
A	1				3				7
L	2				2				6
T	3				2				5
R	4	3	2	2	2	2	3	4	5
U	5				3				5
I	6				4				5
S	7				5				5
T	8	7	6	6	6	6	5	4	5

Edit		A	L	G	O	R	I	T	H
									8
A									7
L									6
T									5
R									5
U									5
I									5
S									5
T	8	7	6	6	6	6	5	4	5

```
EditBoundary(i, j, w, T[0..w], L[0..w]):
  if w = 1
      B[0] ← L[1];  R[0] ← T[1]
      compute B[1] using the edit distance recurrence
      R[1] ← B[1]
      return R[0..1], B[0..1]
  else
      T_11 ← T[0..w/2];  T_12 ← T[w/2..w]
      L_11 ← L[0..w/2];  L_21 ← L[w/2..w]
      T_21, L_12 ← EditBoundary(i,       j,       w/2, T_11, L_11)
      T_22, R_12 ← EditBoundary(i,       j + w/2, w/2, T_12, L_12)
      B_21, L_22 ← EditBoundary(i + w/2, j,       w/2, T_21, L_21)
      B_22, R_22 ← EditBoundary(i + w/2, j + w/2, w/2, T_22, L_22)
      return B_21 · B_22, R_12 · R_22      ⟨⟨Concatenation⟩⟩
```

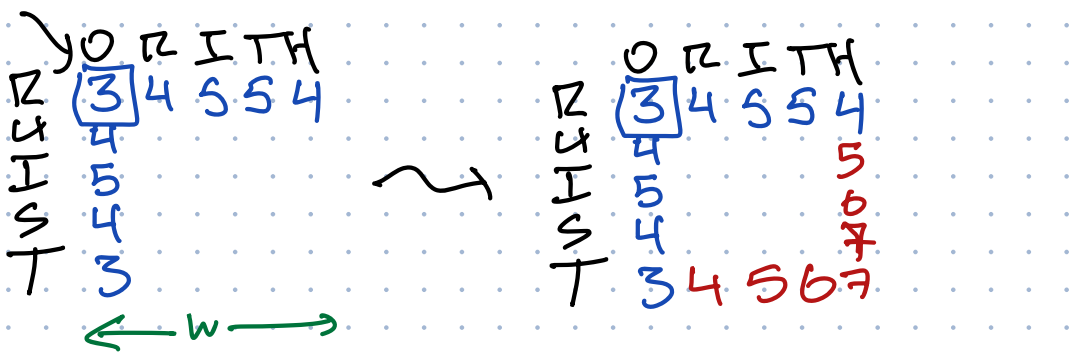


$$T(n) = O(n^2) + 3T\left(\frac{1}{2}\right) = O(n^2)$$

① ② ③

$$S(n) = O(n) + S\left(\frac{n}{2}\right) = O(n)$$

```
EditSequence(i, j, w, T[0..w], L[0..w], ti, tj):
  if w = 1
    ⟨⟨Base case: Brute force⟩⟩
    compute operation Z and indices si, sj using the edit distance recurrence
    return Z, si, sj
  else if (ti ≤ i or ti ≥ i + w or tj ≤ j or tj ≥ j + w or (ti < i + w and tj < j + w))
    ⟨⟨Trivial case: target indices are not on the outer block boundary⟩⟩
    return ε, ti, tj
  else
    ⟨⟨Forward phase: Set up inputs for recursive subproblems⟩⟩
    T11 ← T[0..w/2];  T12 ← T[w/2..w];
    L11 ← L[0..w/2];  L21 ← L[w/2..w]
    T21, L12 ← EditBoundary(i,       j,       w/2, T11, L11)
    T22, R12 ← EditBoundary(i,       j + w/2, w/2, T12, L12)
    B21, L22 ← EditBoundary(i + w/2, j,       w/2, T21, L21)
    ⟨⟨Backward phase: Recursively backtrack along the optimal edit path.⟩⟩
    ⟨⟨At most three of these recursive calls actually do anything.⟩⟩
    ⟨⟨Each nontrivial recursive call updates ti and tj.⟩⟩
    Z22, ti, tj ← EditSequence(i + w/2, j + w/2, w/2, T22, L22, ti, tj)
    Z21, ti, tj ← EditSequence(i + w/2, j,       w/2, T21, L21, ti, tj)
    Z12, ti, tj ← EditSequence(i,       j + w/2, w/2, T12, L12, ti, tj)
    Z11, ti, tj ← EditSequence(i,       j,       w/2, T11, L11, ti, tj)
    return Z11 · Z12 · Z21 · Z22, ti, tj
```



characters in alphabet = $s = O(1)$

$w = \frac{1}{2}\log_s n$

Specify subproblem: $O(\sqrt{n})$

Two strings: $\lceil 2 \cdot s^w \rceil \rightarrow O(w \log s) = O(w)$ bits

One number: $O(\log w)$ bits $= 1$ word

differences: $O(w)$ bits $\rightarrow O(\sqrt{n})$ possibilities

diff subprobs $= o(n) = O(\sqrt{n})$

Precompute ALL $O(\sqrt{n})$ block outputs in $O(\sqrt{n}\log^2 n)$ time
$\ll O(n)$

Short-circuit CK algo at $w \times w$

$$\Rightarrow \text{Time} = O(n^2/\log^2 n)$$

$$\text{Space} = O(n/\log n) + \text{output}$$