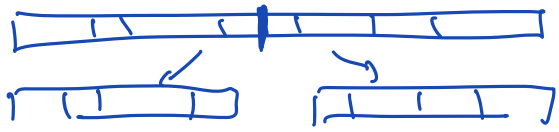
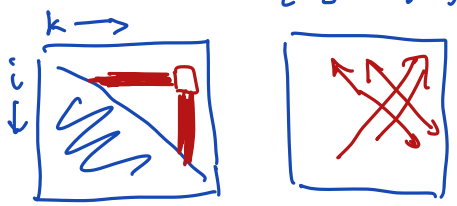
HW2 (and in general): It's enough for full credit to compute the cost of optimal structure.

HW0 regrade requests → due in two weeks

Woodcutter's problem

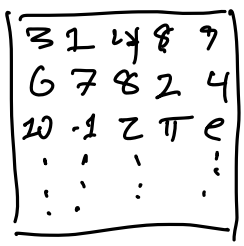


$$\text{MinCost}(i,k) = \underbrace{\text{Length}(i,k)}_{x[k]-x[i]} + \min_{i<j<k} \left(\text{MinCost}(i,j) + \text{MinCost}(j,k)\right)$$



$O(n^3)$ time $\Rightarrow$ $O(n^2)$ time

Common subproblem: Find the smallest entry in each row of an $m \times n$ array



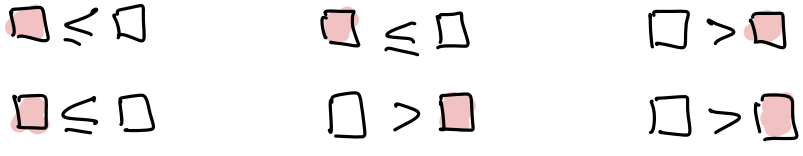
Brute force: $O(mn)$ time is optimal!

A 2d array is monotone if min elements in earlier rows above/left of min elements in later rows

12	21	38	76	27
74	14	14	29	60
21	8	25	10	71
68	45	29	15	76
97	8	12	2	6

Totally monotone if every submatrix/minor is monotone

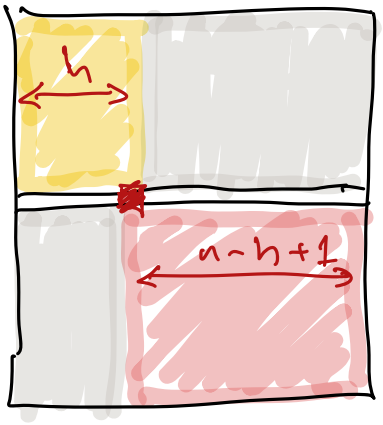
12	21	38	76	89
47	14	14	29	60
21	8	20	10	71
68	16	29	15	76
97	8	12	2	6



Row minima in monotone array m=#rows n=#cols

FILTER: $O(m + n \log m)$ time

Top Down



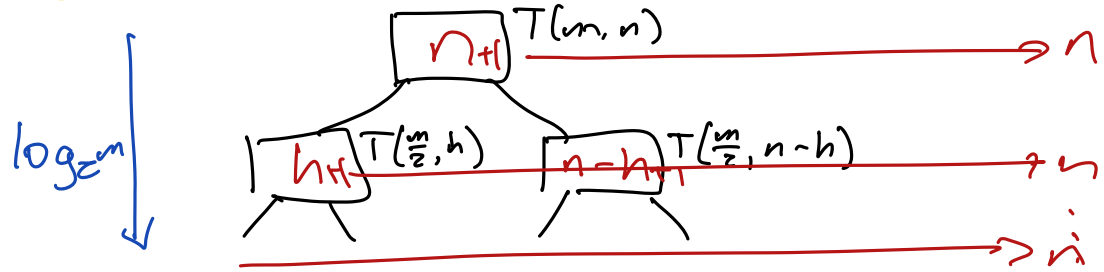
① Find min entry in middle row

$M[\frac{m}{2}, h]$

② Recurse in $M[1..\frac{m}{2}-1, 1..h]$

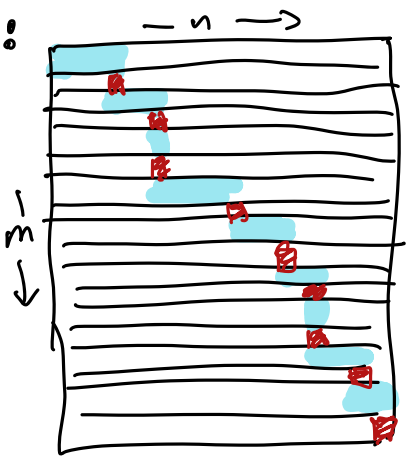
③ Recurse in $M[\frac{m}{2}+1..m, h..n]$

$$T(m,n) = O(n) + T(\tfrac{m}{2}, h) + T(\tfrac{m}{2}, n-h)$$



$O(n \log m + m)$

Bottom-up!



Recursively!

① Find min elements in every even row

② Search in each odd row only between mins above and below
↳ $O(m+n)$

$$T(m,n) = T(\tfrac{m}{2}, n) + O(m+n)$$

$$= O(m + n \log m)$$

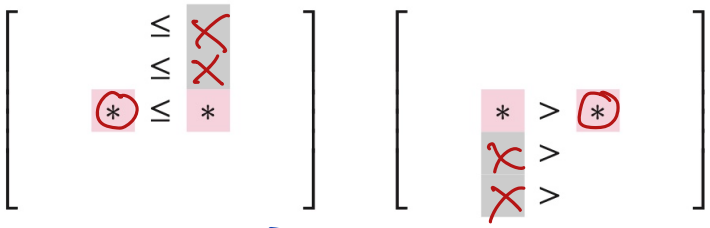
Totally monotone

every 2x2 minor is monotone

SMAWK

Shor
Moran
Aggarwal
Wilber
→ Klawe

One comparison kills a bunch of entries in one column

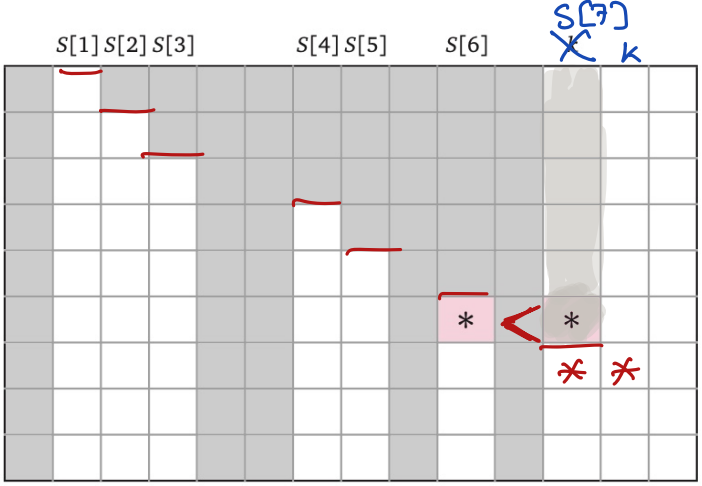


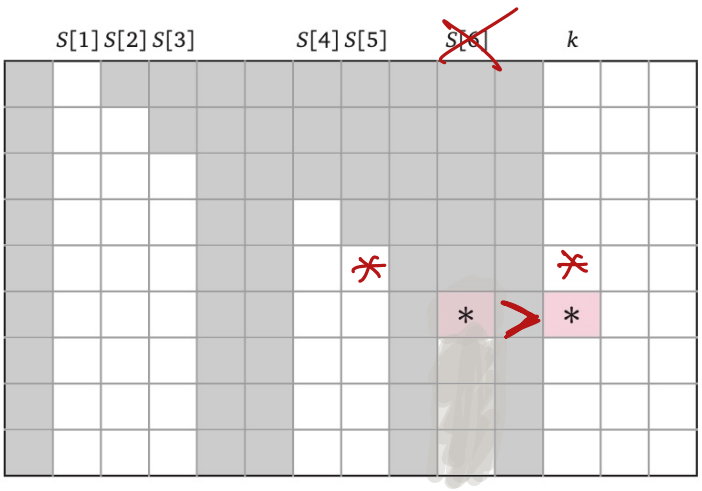
n>m wide array → O(n) → m×m array

```
REDUCE(M[1..m, 1..n]):
  t ← 1
  S[t] ← 1
  for k ← 1 to n
      while t > 0 and M[t,S[t]] ≥ M[t,k]
          t ← t − 1        ⟨⟨pop⟩⟩
      if t < m
          t ← t + 1
          S[t] ← k         ⟨⟨push k⟩⟩
  return S[1..t]
```

Figure D.10. The SMAWK algorithm to REDUCE wide arrays

- S[1..t] is a stack of column indices sorted increasing order
- For all 1≤j≤t, top j-1 entries in column S[j] are dead
- If j<k and j not in S, column j is dead.





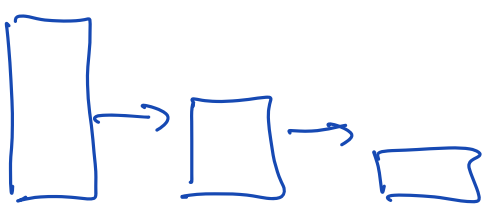
$\Rightarrow \leq 2n$ comparisons

Find row minima in a totally monotone array

(1) if $m < O(1)$ brute force $O(n)$ time

(2) if $m < n$ REDUCE to $m \times m$ subarray and recurse

(3) if $m \geq n$, FILTER to $\frac{m}{2} \times n$ minor and recurse and then scan



$$T(m,n) = \begin{cases} O(n) & \text{if } m = O(1) \\ O(n) + T(n,m) & \text{if } m < n \\ O(m) + T\left(\frac{m}{2}, n\right) & \text{if } m \geq n \end{cases}$$

$$\boxed{O(n+m)}$$