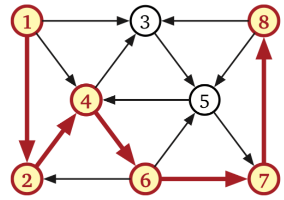
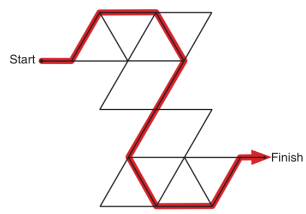
Midterm 2 topics

recursion
+ div + conq
− backtracking
+ dynamic programming

graphs
+ traversal/reachability
+ dags/top sort
+ strong components
+ shortest paths

1. Verify shortest path tree
   - given pred but no dist
   - given dist but no pred
2. Swedish hackers
3. $2 \times 3 + 0 \times 6 \times 1 + 4 \times 2$
4. Elmo
5. 

6. 

Given $G=(V,E)$ directed, weighted edges $\ell(e)$

given pred(v) for every vertex except one (s)

Verify that preds describe a shortest path tree rooted at s.

- pred edges are edges in G.

  $pred(v) \to v \in E$ for all v.

  ~~For all $v \neq s$, traverse list of edges leaving pred(v), fail if v absent~~

- pred edges define a tree!

  Is every vertex reachable from s thru pred edges?

  $O(V)$ WFS

- shortest path distances consistent in T and G
  - Compute distances in T

$$dist_T(v) = \begin{cases} 0 & \text{if } v = s \\ dist_T(pred(v)) + \ell(pred(v) \to v) & \text{o/w} \end{cases}$$

preorder traversal of T $O(V)$

— Check all edges in G if any tense, fail.
$O(E)$

→ for all vertices u
  for all edges u→v
   if pred(v) = u
    mark v

$O(E)$

if any vertex unmarked, fail $+\,O(V)$

$O(\cancel{V}+E)$ time

Elmo game 5 6 2 7 1 6 -5
Left or right

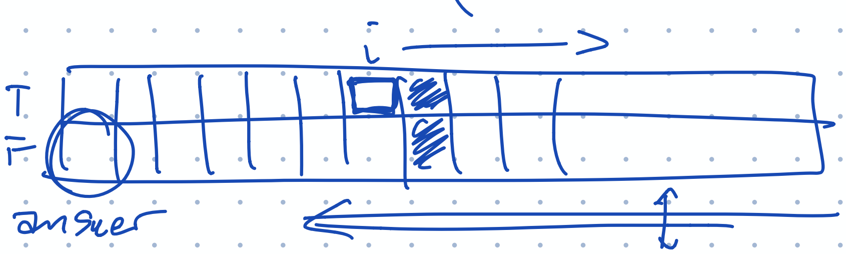
☆Elmo game 2 5 6 -2 7 -1 6 -5 →

keep end turn
or pass and go again

BestScore(i, me) = max score I can get from cards i..n
 if I go first if me = True
 Elmo if me = FALSE

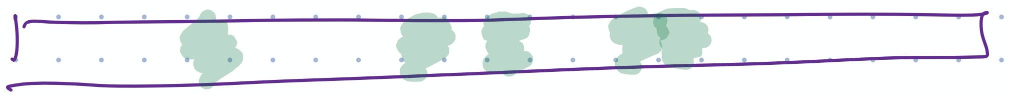
We need BestScore(1, False) Input: C[1..n] of card values

$$\text{BestScore}(i,me) = \begin{cases} 0 & \text{if } i>n \\ \max\begin{Bmatrix} C[i] + \text{BestScore}(i+1, F) \\ \text{BestScore}(i+1, T)\end{Bmatrix} & \text{if } me = \text{True} \\ \min\begin{Bmatrix} C[i] + \text{BestScore}(i+1, F) \\ \text{BestScore}(i+1, T)\end{Bmatrix} & \text{if } me = \text{FALSE} \end{cases}$$



$O(n)$ time

Swedish häckers

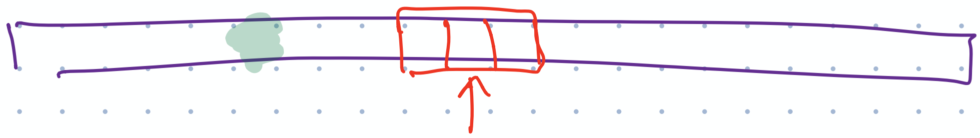


sorted
k elements corrupted

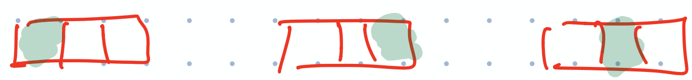
Given target value x

Find x if x is not corrupted.

k=1



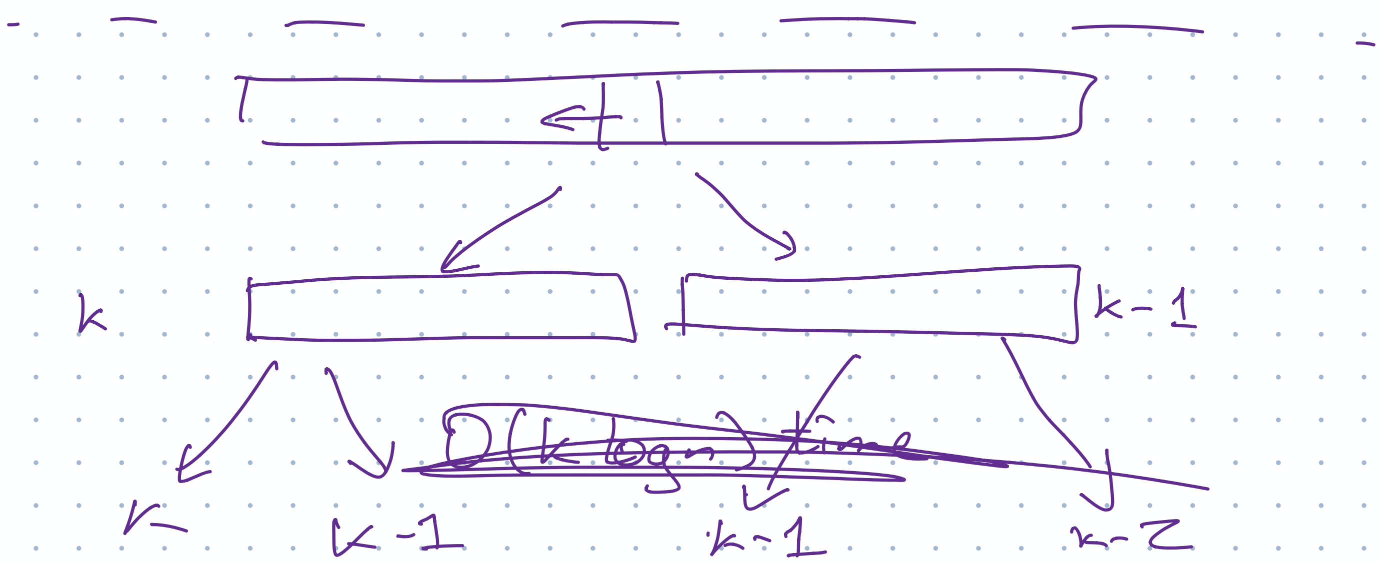
- compare all 3 to x
  follow majority opinion
- compare median of 3 with x



$O(\log n)$

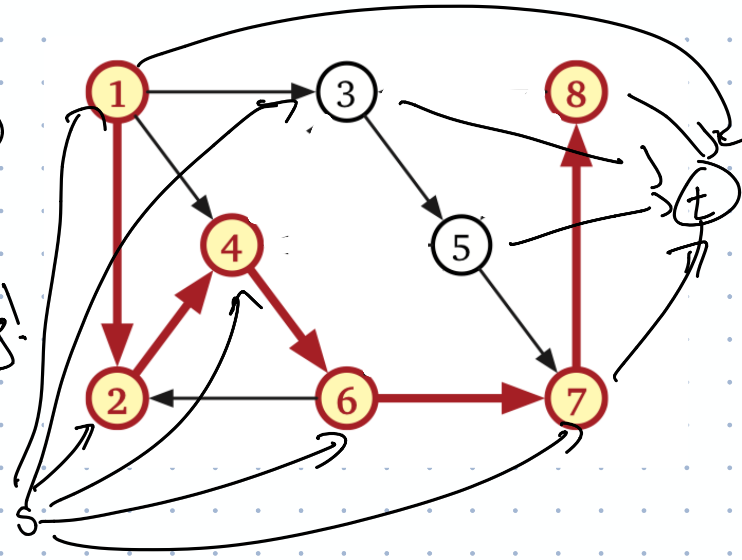
k>1 Use a window of size $2k+1$

$O(k \log(n/k))$



$O(2^k \log n)$ time?

Longest increasing path



Remove any decreasing edges O(E)

Remaining subgraph is a dag!

add s→v to every v
v→t from every v

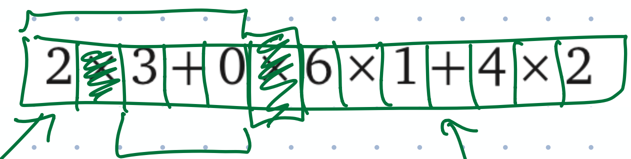
Run the longest path algo from class — known s to known t

O(V+E') time
= O(V+E)

Given an expression like this: $(2 \times 3) + ((0 \times 6) \times 1) + (4 \times 2) = 14$

Find min value we can get by inserting parens

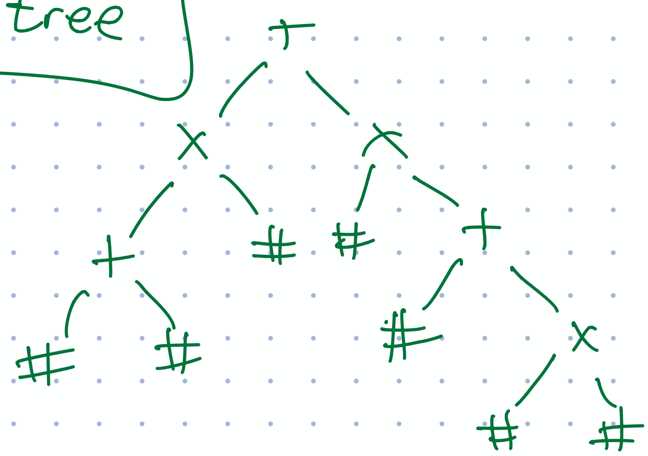
Assume input X[0..2n]



even positions: numbers

Odd positions: + or ×

Binary tree



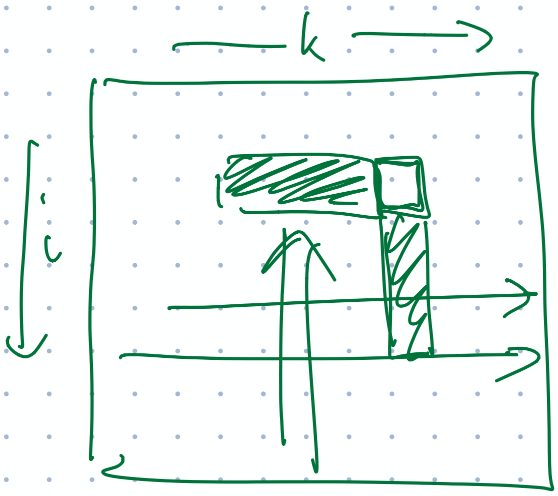
((# + #) × #) +
(# × (# + (# × #)))

MinValue(i,k) = smallest value we can get from $X[2i..2k]$ by adding parens

We need MinValue[0, n)

~~∞ if $i > k$~~

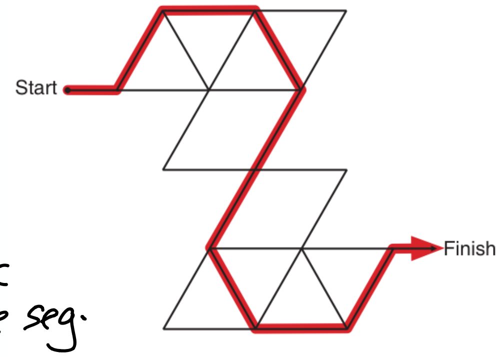
$$\text{MinValue}(i,k) = \begin{cases} X[2i] & \text{if } i = k \\ \min\limits_{i \le j < k} \begin{cases} \text{MinValue}(i,j) + \text{MinValue}(j+1,k) & \text{if } X[2j+1] = + \\ \text{MinValue}(i,j) * \text{MinValue}(j+1,k) & \text{if } X[2j+1] = * \end{cases} \end{cases}$$



$O(n^3)$ time

```
for i ← n down to 1
    for k ← i to n
        [ ]
```

Obtuse angle maze



Given undir graph
$G=(V,E)$
coords for every vertex
every edge is straight line seg.
Two verts start and finish

Find shortest walk in G from s to f
that only makes obtuse turns.

Build a new graph $G'=(V',E')$

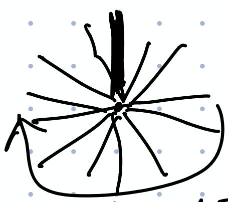
$V' = \{(u,v) \mid uv \text{ is an edge}\} \cup \{s'\}$ (prev, current)
$V' = 2E + 1 = O(E)$

$E' = \{(u,v)\to(v,w) \mid uvw \text{ is obtuse}\}$ ← $O(1)$ time
$\cup \{s' \to (s,v) \mid sv \text{ is edge}\}$

We need shortest path in G' from s' to anyvertex $(\_, \text{finish})$

BFS at s'
time $= O(V'+E')$
$= O(VE)$

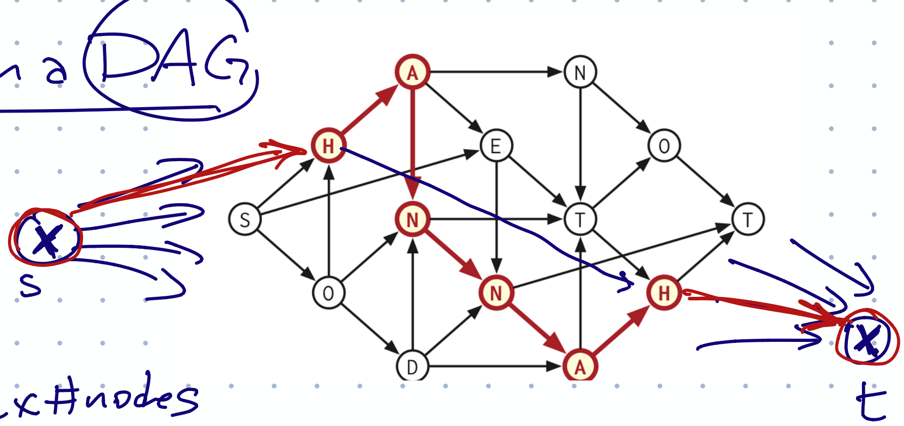
Crudely:
$E' \le \sum_u \deg(u)^2 = O(VE)$
$\le \sum_u V\cdot\deg(u) = V\cdot\sum_u \deg(u)$
$= O(V\cdot E)$



deg 12

obtuse $= 7\cdot 12$

Longest Palindrome in a DAG



① Top sort G

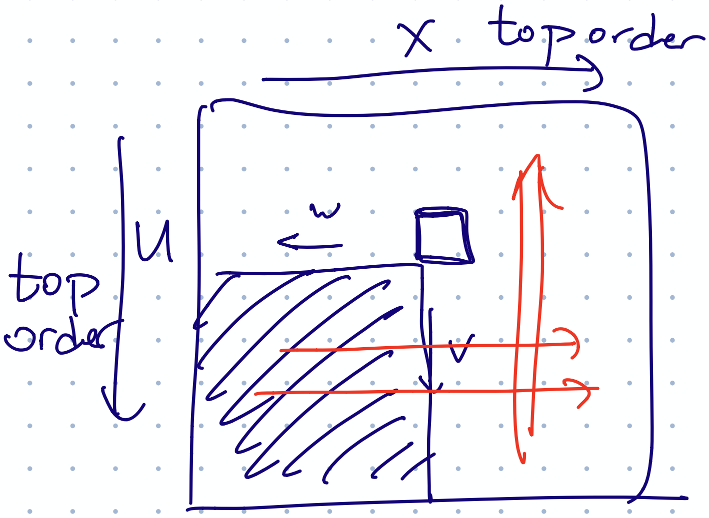
LPP(u, x) = length of longest pal. path starts at u, ends at x
(length = #nodes; max #nodes)

Add source s, sink t: s → v, v → t

Now we need LPP(s, t) − 2

$$LPP(u,x) = \begin{cases} -\infty & \text{if } label(u) \neq label(x) \\ 1 & \text{if } u = x \\ \max\{LPP(v,w) + 2 \mid u \to v \in E,\ w \to x \in E\} & \text{o/w} \end{cases}$$

(also 2 if u → x is edge)



for all verts u in rev top order
 for all verts x in top order
 [ ]

~~O(V⁴)~~ O(E²) time = O(V⁴)