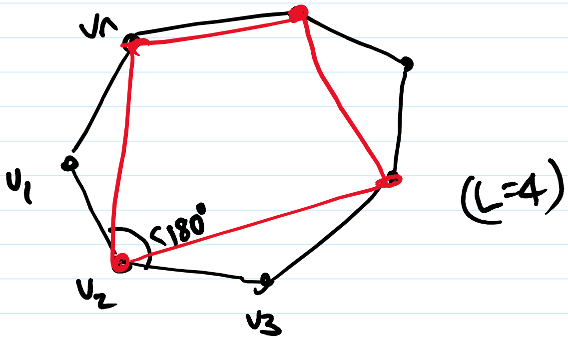
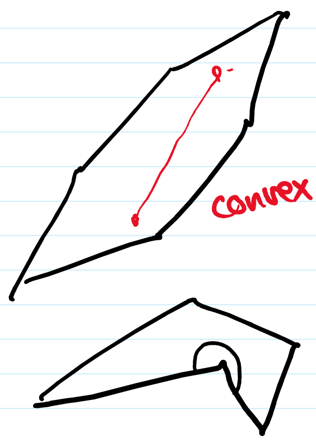
DYNAMIC PROG.

Max-Perimeter L-gon

Given convex polygon $v_1 v_2 \cdots v_n$ and $L \le n$,
find sub-polygon with L vertices
maximizing perimeter



(L=4)



not convex

DP Method 1

Define subproblems:

for each $1 \le i < j \le n$, $\ell \le L$,
let $D^{(\ell)}(i,j)$ = max dist of ℓ-link path from v_i to v_j

"hops"

$O(n^2 L)$ sub problems

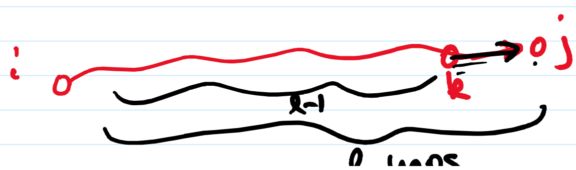
Want

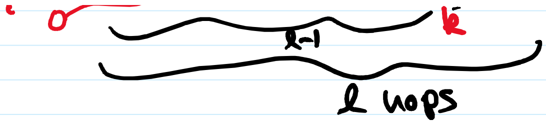
$$\max_{1 \le i < j \le n} \left(D^{(L-1)}[i,j] + d(j,i) \right)$$

Euclidean dist from v_j to v_i

Recursive formula:

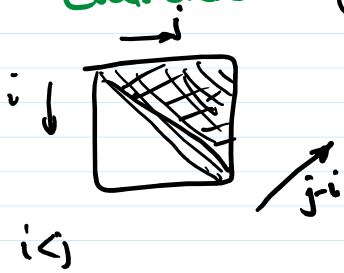
$$D^{(\ell)}(i,j) = \max_{k \in \{i+1,\dots,j-1\}} \left(D^{(\ell-1)}(i,k) + d(k,j) \right)$$





Base case: $D^{(1)}(i,j) = \underline{d(i,j)}$ $\quad (1 \le i < j \le n)$

Eval. order: increasing order of ℓ
(increasing order of $j-i$)



$\Rightarrow$ $O(n^2 L)$ table entries

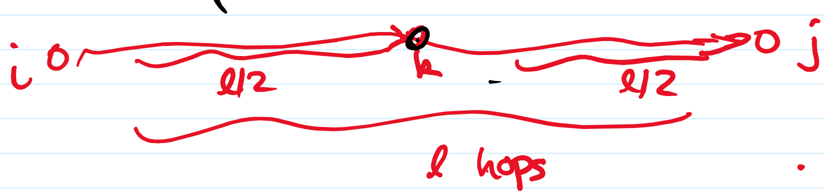
time per entry $O(n)$

$\Rightarrow$ $\boxed{O(n^3 L)}$ total time

$\le O(n^4)$

DP Method 2:

$$D^{(\ell)}(i,j) = \begin{cases} \max\limits_{i<k<j} \left(D^{(\ell/2)}(i,k) + D^{(\ell/2)}(k,j) \right) & \text{if } \ell \text{ even} \\ \max\limits_{i<k<j} \left(D^{(\ell-1)}(i,k) + d(k,j) \right) & \text{if } \ell \text{ odd} \end{cases}$$



only need $O(\log L)$ values of ℓ

$D^{(16)}$
$\uparrow$
$D^{(8)}$
$\uparrow$
$D^{(4)}$
$\uparrow$
$D^{(2)}$

$\Rightarrow$ $\boxed{O(n^3 \log L)}$ time

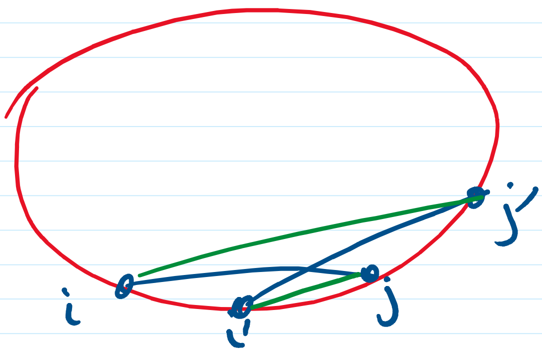
("repeated squaring")

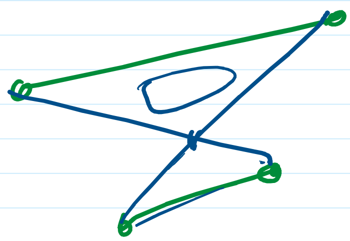
Improved DP Method 2:

Improved DP Method 2:

Def Function $d(\cdot,\cdot)$ satisfies convex Monge property
if $\forall\, i \le i' \le j \le j'$,

$$d(i,j) + d(i',j') \geqslant d(i,j') + d(i',j)$$





follows from triangle inequality (twice)

Lemma (~~F. Yao '82~~)

Let $d(\cdot,\cdot)$ is convex Monge.
Let $D(i,j) = \max\limits_{i \le k \le j} \left(d(i,k) + d(k,j) \right)$

$K(i,j) = k$ that attains this max

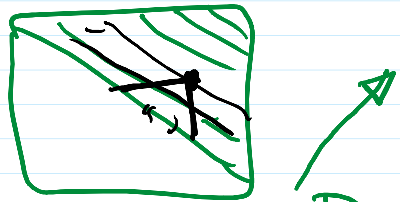
Then (a) K is monotone increas. in each row & in each column

(b) D is also convex Monge.

Thm (F. Yao '82)

If d convex Monge,
can compute D in $O(n^2)$ time.

Pf: Fix Δ.



D $\quad j-i=\Delta \quad j=i+\Delta$

Given $D(i, i+\Delta)$ for all i, $K(i, i+\Delta)$.

Want to compute $\underline{D(i, i+\Delta+1)}$.

Observe $\underline{K(i,i+\Delta)} \le \underline{K(i, i+\Delta+1)} \le \underline{K(i+1, i+\Delta+1)}$ by (a).

$$D(i, i+\Delta+1) = \max_{\underline{K(i,i+\Delta)} \le k \le \underline{K(i+1,i+\Delta+1)}} \left(d(i,k) + d(k, i+\Delta+1)\right)$$

time per diagonal (i.e. per Δ)

$$O\left(\sum_{i=1}^{n-\Delta} \underbrace{\left(K(i+1, i+\Delta+1) - K(i, i+\Delta) \pm 1\right)}_{f(i+1) \; - \; f(i)}\right)$$

$$= f(2) - f(1) + f(3) - f(2) + f(4) - f(3)$$

$$= O(n)$$

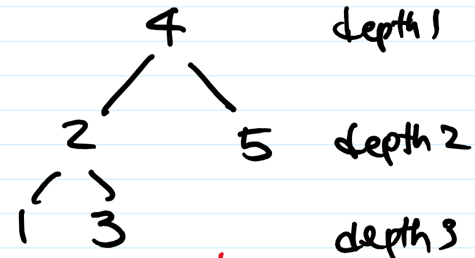
$\Rightarrow$ total time over all Δ $\boxed{O(n^2)}$. $\square$

Total time of whole alg'm $\boxed{O(n^2 \log L).}$

Rmk: Schieber '98 $O\left(n\, c^{\sqrt{\log L \log\log n}}\right)$

$$\le O(n^{1+\varepsilon}) \text{ for any const } \varepsilon > 0.$$

Optimal Binary Search Tree



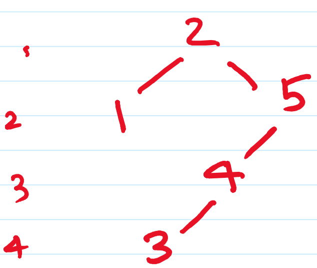
Given n values $a_1, \ldots, a_n$,
& their "frequencies" $f_1, \ldots, f_n$,

build a binary search tree for $a_1, a_2, \ldots, a_n$
that minimizes total search cost

$$\sum_{i=1}^{n} f_i \cdot \text{depth}(a_i)$$

e.g. $a = 1, 2, 3, 4, 5$
$f = 4, 10, 1, 2, 8$

$$\text{cost} = 4\cdot3 + 10\cdot2 + 1\cdot3 + 2\cdot1 + 8\cdot2 = 55$$



$$\text{cost} = 4\cdot2 + 10\cdot1 + 1\cdot4 + 2\cdot3 + 8\cdot2 = 44$$