$\Rightarrow$ $\boxed{O(n^2 \log L)}$ for max-perim subpolygon problem
time

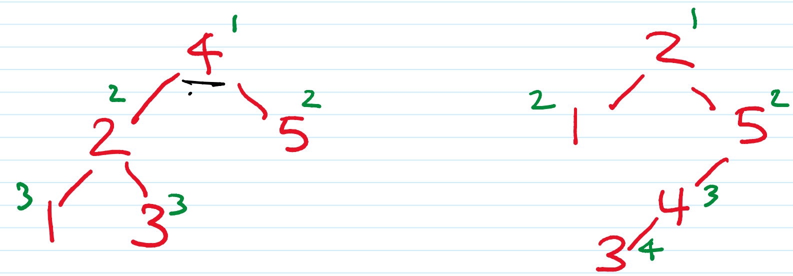
Rmk: Schieber '98 $O(n^{1+\varepsilon})$ time

Optimal Binary Search Tree

Given n elements $a_1, \ldots, a_n$
their frequencies $f_1, \ldots, f_n$,
build a binary search tree for $a_1, \ldots, a_n$
that minimizes total search cost

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

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



cost $2\cdot 1 + (10+8)2 + (4+1)\cdot 3 = \cancel{55}\ 53$

cost $10\cdot 1 + (4+2)\cdot 2 + 2\cdot 3 + 1\cdot 4 = \cancel{44}.$
32

Sort $a_1 < a_2 < \cdots < a_n$.

Define subproblems:

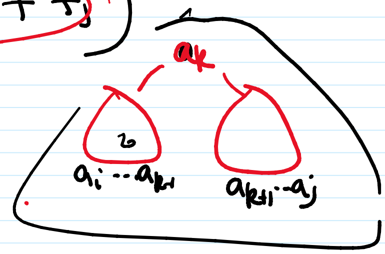
for each $1 \le i \le j \le n$,

let $D(i,j)$ = min cost for search tree for $a_i \cdots a_j$.

Want $D(1,n)$.

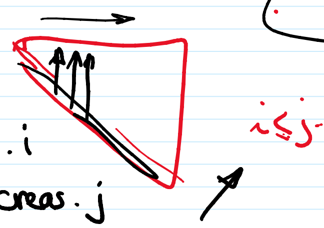
Recursive formula: (Suppose we use a_k as root)

$$D(i,j) = \min_{i \le k \le j} \left(D(i,k-1) + D(k+1,j) + \underbrace{f_i + f_{i+1} + \cdots + f_j}_{d(i,j)} \right)$$



Base case. $D(i,i-1) = 0$.

Eval order:



opt1: increas. j; for same j, decreas. i

opt2: decreas i; for same i, increas. j

opt3: increas. $j-i$

Analysis: # table entries $O(n^2)$

time per entry $O(n)$

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

improves to $\boxed{O(n^2)}$

Improvement:

Speedup Thm II (F. Yao '82 / Knuth '71)

Suppose $D(i,j) = \min_{i \le k \le j} \left(D(i,k-1) + D(k+1,j) + d(i,j) \right)$

If d is concave Monge,

then can compute D from d in $O(n^2)$ time.

Obs $d(i,j) = f_i + f_{i+1} + \cdots + f_j$

is both convex Monge & concave Monge.

Pf: for $i \le i' \le j \le j'$,

$$\underbrace{d(i,j) + d(i',j')}_{= f_i + \cdots + f_{i'-1} + 2(f_{i'} + \cdots + f_j) + f_{j+1} + \cdots + f_{j'}} = \underbrace{d(i,j') + d(i',j)}_{= \text{same thing}}$$

Subset Sum

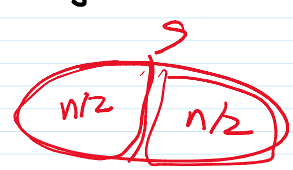
Given set S of n positive integers & target number T, decide whether $\exists$ subset $R \subseteq S$ that sums to T.

e.g. $\{14, 16, 20, 25, 40, 55, 59\}$, $T = 100$

2 variants: standard (no repeat)
multiset (allow repeats)

NP-complete (no strongly polynomial alg'm if $P \ne NP$)

trivial: $O(2^n)$ improves to $O(2^{n/2}\, n)$



DP method 1

Define subproblems: $\forall i \in \{0, .., n\}$, $t \in \{0, .., T\}$,

$C(i,t)$ = yes iff $\exists$ subset $R \subseteq \{s_1, .., s_i\}$ that sums to t

Want $C(n, T)$.

Recursive formula:

$$\boxed{C(i,t) = C(i-1,t) \vee C(i-1, t-s_i)}$$

not use s_i use s_i

Eval order: increas i

Runtime: $O(\underbrace{nT}_{\#\text{ entries}} \cdot \underbrace{1}_{\text{time per entry}}) = \boxed{O(nT)}$

multiset vers: $C(i,t) = C(i-1,t) \vee C(i, t-s_i)$

DP Method 2 (multiset vers.)

$\forall t \in \{0, .., T\}$,

Define $C(t) =$ yes iff $\exists$ $R \subseteq S$ (multiset) that sums to t

Formula:

$$\boxed{C(t) = \bigvee_{i=1}^{n} C(t-a_i)}$$

Runtime: $O(\underbrace{T}_{\#\text{ entries}} \cdot \underbrace{n}_{\text{time per entry}}) = \boxed{O(nT)}$

DP Method 3 (multiset vers. only)

$\forall t \in \{0, .., T\}$, ℓ,

define $C^{(\ell)}(t) =$ yes iff $\exists$ $R \subseteq S$ (multiset) using $\leq \ell$ elements that sums to t

Formula:

$$\boxed{C^{(\ell)}(t) = \bigvee_{t'=0}^{T} \left(C^{(\ell/2)}(t') \wedge C^{(\ell/2)}(t-t') \right)}$$

ℓ even

$\#\ \ell$ values $= O(\log T)$

$\#$ entries $O(T \log T)$

time per entry $O(T)$

$\Rightarrow O(T^2 \log T)$ worse

CONVOLUTION

can compute $C^{(\ell)}$ from $C^{(\ell/2)}$

by FFT in $O(T \log T)$ time

$\Rightarrow$ total time $\boxed{O(T \log^2 T)}$

Rmk: standard vers.?

Koilaris & Xu '17: $O(\sqrt{n}\, T \log^c T)$

Bringmann '17: $O(T \log^c T)$ rand.

Jin & Wu '19: $O(T \log^2 T)$ rand.