Topic 4: Monotonicity

DP Review, Optimization

Review: Longest Increasing Subsequence (LIS)

4 3 1 5 2 3 6

Review: Longest Increasing Subsequence (LIS)

4 3 1 5 2 3 6

naive DP: $\Theta(n^2)$

$dp[i]$: the longest increasing subsequence ending at $a[i]$.

$$dp[i] = \max_{j<i \land a[j]<a[i]} dp[j] + 1$$

Review: Longest Increasing Subsequence (LIS)

4 3 1 5 2 3 6

naive DP: $\Theta(n^2)$

dp[i]: the longest increasing subsequence ending at a[i].

$$dp[i] = \boxed{\max_{j<i \wedge a[j]<a[i]} dp[j]} + 1$$

Range Query!

Imagine an array where

$val[a[j]] = dp[j]$

Then $dp[i] = \max_{1 \le k < a[i]} (val[k]) + 1$ Comp. $\Theta(n \log n)$

Review: Longest Increasing Subsequence (LIS)

4 3 1 5 2 3 6

naive DP: $O(n^2)$

alternative $O(n\log n)$ solution

Redefine $dp[i]$: the smallest last value s.t. the length of the LIS is i.

a:	4	3	1	5
i:	0	1	2	3
dp[i]:	0	1 (1)	5 (4 3) (1 5)	∞

Observe that $dp[i]$ is monotonically increasing

Hence, for some new $a[j]$

$\exists k$ s.t. $dp[k] < a[j] \le dp[k+1]$

Claim: we only need to set $dp[k+1] = a[j]$.

DP Optimization Techniques

By techinques (Fenwick tree)

By observation (different DP state)

By implementation ? (make code really fast ?)

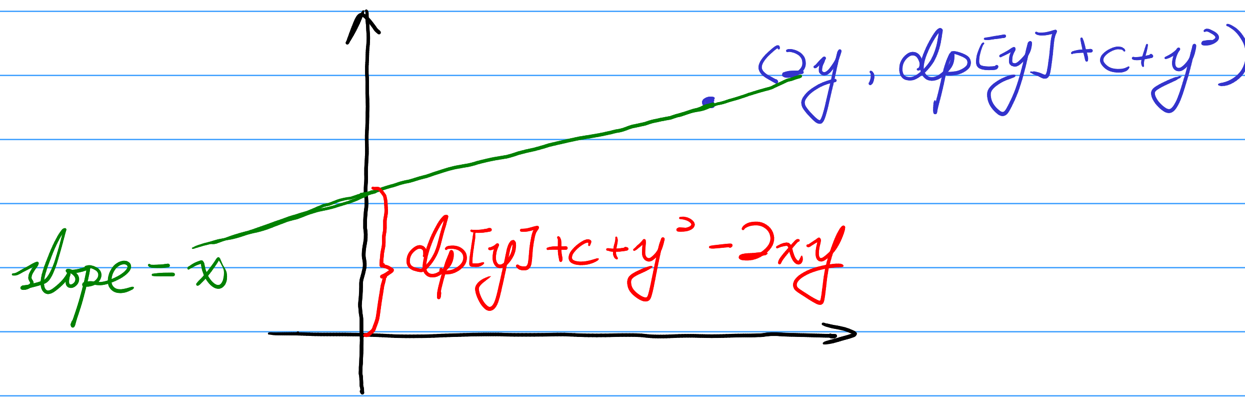
Covered Walkway (Slope Trick)

Covered Walkway (Slope Trick)

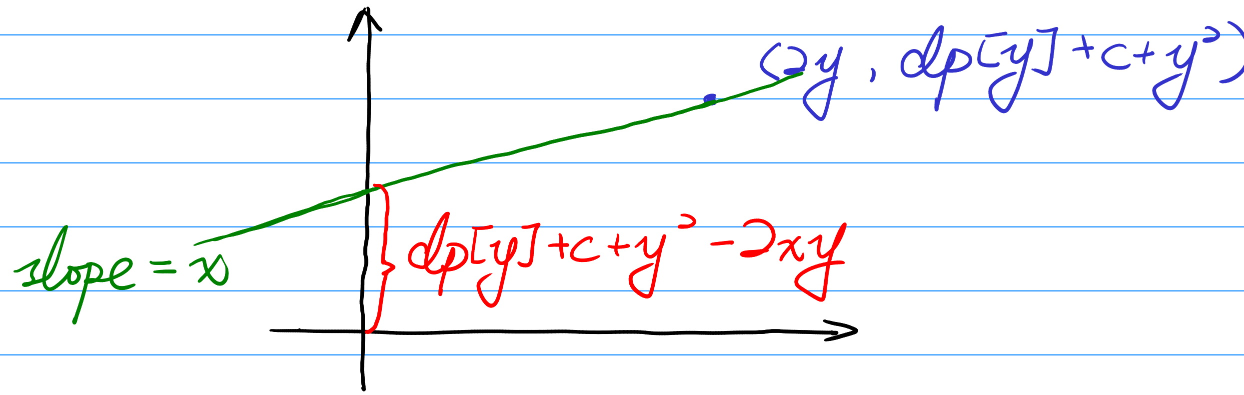
$$dp[x] = \min_{y<x} dp[y] + c + (x-y)^2$$ — $O(n^2)$

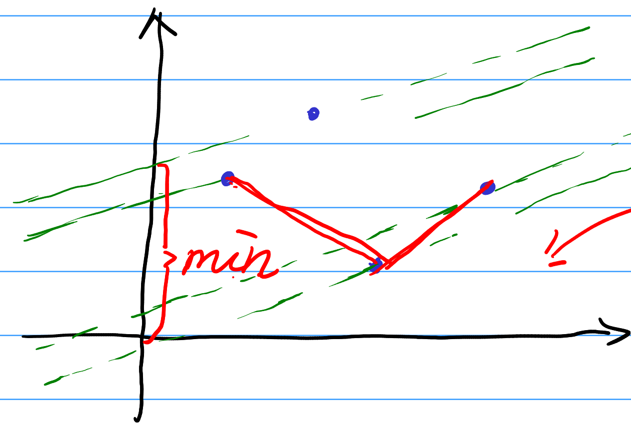
$$= \min(\underbrace{dp[y]+c+y^2}_{\text{constant w.r.t. } x} \underbrace{-2xy}_{\text{linear}}) + x^2$$ — not relevant

Consider it as a geo problem



Covered Walkway (Slope Trick)





The potential points form a lower convex hull.

Covered Walkway (Slope Trick)

$$dp[x] = \min_{y<x} dp[y] + c + (x-y)^2$$

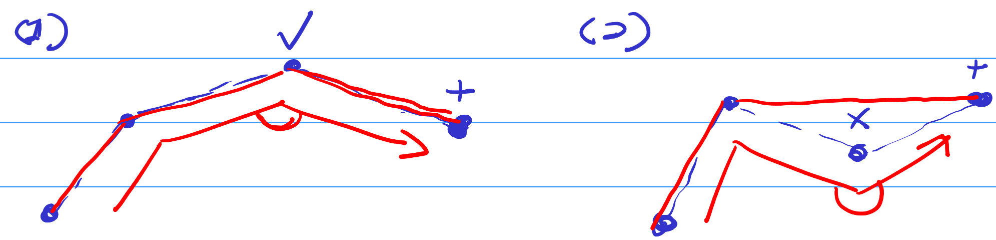
$$= \min(\underbrace{dp[y]+c+y^2}_{\text{constant w.r.t. } x} - \underbrace{2xy}_{\text{linear}}) + x^2 \text{ — not relevant}$$

Generic Solution: Dynamic convex hull +
Binary search for queries
(very annoying)

For most probs: $(\underset{\text{increasing}}{2y}\uparrow,\ dp[y]+c+y^2)$

Graham Scan

(1) ✓ + (2) + ×



Keep popping until the points turn right

Covered Walkway (Slope Trick)

$$dp[x] = \min_{y<x} dp[y] + c + (x-y)^2$$

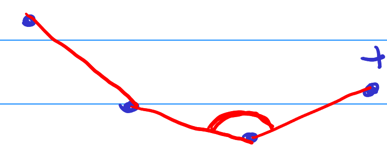
$$= \min\big(\underbrace{dp[y]+c+y^2}_{\text{constant w.r.t. } x} - \underbrace{2xy}_{\text{linear}}\big) + x^2$$

— not relevant

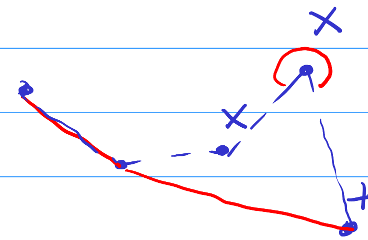
For most probs : $(2y\uparrow,\ dp[y]+c+y^2)$

increasing

Graham Scan (1)



(2)



Keep popping until the points turn left

Queries : Slope $= x \uparrow$ increasing

The point we pick keeps moving to the right (two-pointer-ish)

The Elder

The Bakery (Decision Monotonicity)

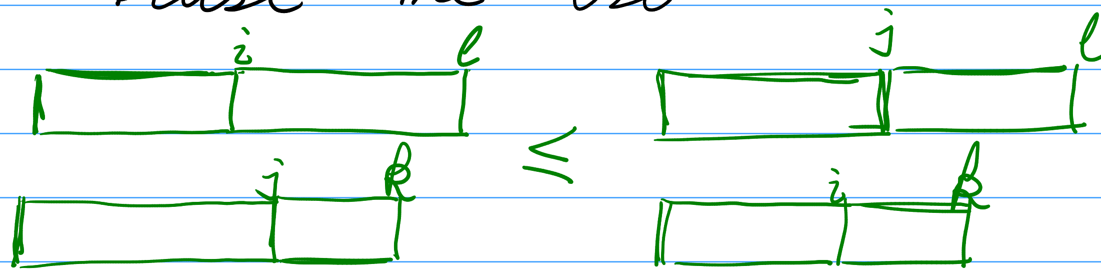
The Bakery (Decision Monotonicity)

$dp[i][j] = \max_k \left(dp[i-1][k] + unique(k+1, j)\right)$ — $O(k \cdot n^2)$

For every i, j, we have decision $k(i,j)$

Claim: $k(i,j) \leq k(i,j+1)$

Proof: Trust me bro



Quadrangle Inequality: $\forall\, i<j<k<l$

$$val[i][l] + val[j][k] \leq val[i][k] + val[j][l]$$

If $val[j][k] \geq val[i][k]$

then $val[i][l] \leq val[j][l]$

The Bakery (Decision Monotonicity)

$$dp[i][j] = \max_k \left(dp[i-1][k] + \text{unique}(k+1, j)\right).$$

For every i, j, we have decision $k(i,j)$

Claim: $k(i,j) \le k(i, j+1)$

Compute(i, l, r, kl, kr) { ← Computes $dp[i][l..r]$ with $k \in [kl, kr]$

$m = (l+r)/2$

Find $k(i,m)$ in $O(n)$

Compute$(i, l, m, kl, k(i,m))$

Compute$(i, m+1, r, k(i,m), kr)$

}

Convince yourself this runs in $O(k \cdot n \log n)$

Monkey Party (Knuth Optimization)

Monkey Party

$$dp[i][j] = \min_{i \le k < j}(dp[i][k] + dp[k+1][j] + sum(i,j))$$

$O(n^3)$

Claim: $k(i, j-1) \le k(i,j) \le k(i+1, j)$

Proof: Trust me Bro

Quadrangle inequality

$$val[i][l] + val[j][k] \ge val[i][k] + val[j][l]$$

equal in this case

for len in 1 .. n

for i in 1 ..

$dp[i][i+len] = \min(\ldots)$

$k(i, i+len-1) \le k \le k(i+1, i+len)$

Convince yourself that this is $O(n^2)$

Covered Walkway w/ monotonicity

$dp[i] \leftarrow dp[j] + val(j, i)$

Claim: $j(i) \leq j(i+1)$

Proof: $val(i, l) + val(j, k) \geq val(i, k) + val(j, l)$

Binary search is on when $j(i)$ goes from k to $k+1$?

Not correct...

$\underbrace{\qquad}_{j(i)=1}\underbrace{\qquad}_{j(i)=2}$

$j(i)$ \ i	100	101	102
1	5	6	7
2	1	10	5
3	1	5	100

Decision monotonicity does not imply binary searchable!

However more complicated D&C techs exist (see LBN technique on my reference)

Summary

DP Optimization

by technique (Fenwick, Segment tree...)

by observation (Switch up states)

convex hull (linear)

monotonicity

1. $dp[i][j] \leftarrow dp[i-1][k] + val(k, j)$ 2D1D

D&C

2. $dp[i][j] \leftarrow dp[i][k] + dp[k][j] + val(i, j)$ 2D2D

Knuth Optimization

3. $dp[i] \leftarrow dp[j] + val(j, i)$ — 1D1D

Covered Walkway (LBN Technique) (See reference)