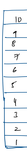
Dp lab 2

Sunday, 26 July 2021 11:06 AM

Egg Drop

* flrs = 10, Eggs = 1



Egg/break

Egg/doesn't break

Idea: BS * 1

$f_s = 1$, $f_e = 10$

$m = 5$

{At 5 th break} ; {At 5 th doesn't break}

$\left[\frac{1-10}{2}\right]$ // $\boxed{\text{Sec} = 0}$ $[6-10]$

Idea 2: Start dropping egg from flour 1

→ If it break at i^{th} day

→ ans = $i-1$

// The drop you need after which your egg break your ans = 10 drops

Flor 1 → doesn't break : 1

→ break : 0

Flor 10 → doesn't break : 10

→ break : 9

Idea:

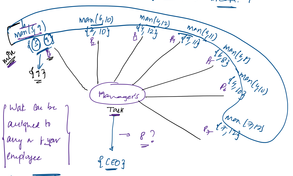
Drop egg at even floors

2 4 6 8 ...

At 6 break

[Ans can be [4, 5]]

Can we do it?



{What can be accepted to carry a 1-year employee}

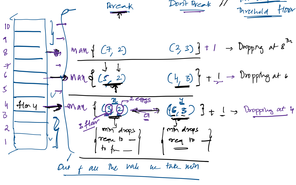
// Exit Worst time:

N flrs K Eggs : → Threshold flor *

→ Min drops you need to find Threshold flor. Can be anywhere

N = 10 flors, K = 3 Eggs // Min drops required to find Threshold flor?

Break — Dont Break // Min drops to get Threshold flor



Our f are the value we take, MAX

$dp[N][K]$ = {Min drop required to get Threshold flor with K eggs}

Space: $dp[N][K]$

{N flors & K eggs 1 based Index}

N=10, K=3

10 drops

N=10, K=3

= [K=1, N=10]

Base Case: {(K==1) return N; (N==1) return 1; (K==0) return 0}

ans = INT_MAX

// We can drop our egg from 1 to N

for(int i = 1; i <= N; i++){

// We are dropping at i^{th} flor

// Break: ↑ ⇒ [i-1, N], how many flors

dp[i-1][K-1] // a drop

// Dont: ↑ ⇒ [i+1, N], how many flors

dp[N-i][K] // a drop

ans = min(ans, max(dp[i-1][K-1] + 1, dp[N-i][K] + 1)) // dp[N, K]

}

dp[N][K] = ans;

Table Edge:

1) [N-1][K+1]

TC:

1) $C(N)(K) \to N$

* $C(N)(K)$ } TLE

→ $NK \to \log N$ → 1 → Index

→ $NK \to 2$

→ $N\log K \to 3$ → Todo

SC: $O(N)(K)$

Find ans: $dp[N][K]$

$dp[N][K] = \min_{i=1}^{N} \left\{ \max\left(dp[i-1][K-1], dp[N-i][K]\right) \right\}$

N-i

⇒ if mover, N-1 different

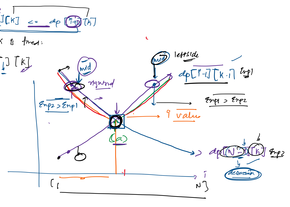
$\left(dp[\boxed{\text{mid}}][K] \le \le dp[N][K]\right)$

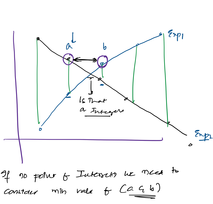
mid drops req to get N=100

N drops req to get 101

1) $dp[\ ][K] <= dp[\ ][K]$

2) If K = 1 then:





If no value is indicated we need to consider min value of (a & b)

min{ For value a: ____ ; For value b: ____ }

$dp[N][K]$ = Same Binary Search

lo = 1, hi = N, ans = —

while (lo < hi){

int mid = (lo+hi)/2;

// If we throw a egg from mid

int cnt1 = dp[mid-1][K-1]

int cnt2 = dp[N-mid][K]

if(cnt1 > cnt2){

hi = mid-1; ans=mid;

}

else if (cnt1 == cnt2){

lo = mid; ans=mid;

}

else{

lo = mid; break;

}

}

ans = lo

→ Off Knapsack

Constraints:

$1 \le N \le 100$

$1 \le K \le 10^2$

$1 \le F \le \frac{10^9}{2}$

$1 \le W \le 10^9$

$N, \{N\}$

{ → max profit we can get throw flors N times

& Array output & }

TC: $\{O(K \times N^2)\}$ → TLE

→ $\frac{N^2}{2}$ → 10^4

Time: $10^2 \times 10^4 = 10^6$

// Change parameters &

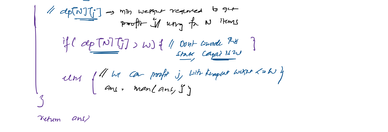
dp table

TC: $100 \times 100 \times 100 > 20$

$\left\{ \begin{array}{l} dp[N][K] \\ \frac{10^9}{2} \text{ Boards} \end{array} \right\}$ → $\frac{10^9}{2} \times 10^2 \times 10^2 = 10^{13}$

$dp[N][K]$ = {Min weight required to get with K mills from 1-N Items}

N^2 Items / Find it a item to /



Base Condit:

dp[0][0] = 0, maxP = 2000

for(int i = 1; i < maxP, i++){

dp[0][i] = -1 // INT_MAX // not possible

}

for(int i = 1; i < N; i++){

dp[i][0] = 0;

for(int j = 1; j < maxP; j++){

dp[i][j] = dp[i-1][j];

if(j - profit[i] >= 0){ dp[i][j] = min(dp[i][j], dp[i-1][j-profit[i]] + weight[i])

}

}

}

// Final ans:

int ans = INT_MIN

for int j = 1; j < maxP; j++){

// dp[N][j] → min weight required to get profit j using for N items

if(dp[N][j] > W){ // Cant carry it

break; // Cause W is

else { // We can profit j with weight W <= W

ans = max(ans, j)

}

}

return ans;

Balanced

1) 7 days

2) 1D dp problems

3) Grid dp problems

4) String dp problems

→ LCS

→ Pattern matching

5) Knapsack dp problems

6) More Dp problems

7) 2D dp → 3 dim

8) Trees → Dp

Strategy

1) Find States

2) Find Base

3) Find Transitions