Problem :

You need to do following operations : search, max-min, range etc. } (1)

Search : probabilities of Searching keywords are different } (2)

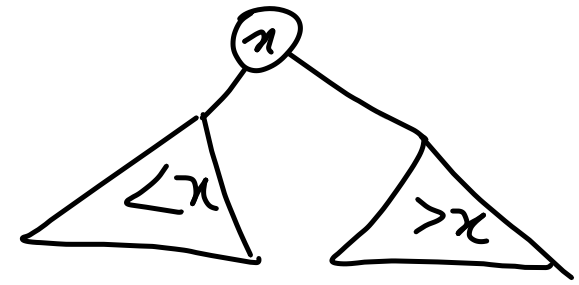
- Sorted Array
- Balanced Binary Search Tree } (1)

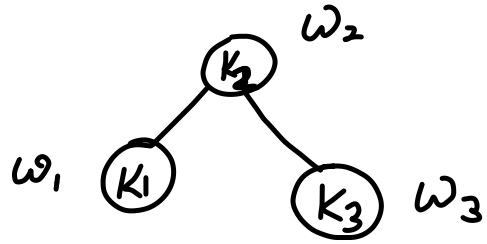
$k_1 < k_2 < k_3 < \cdots < k_n$

$\downarrow$ $\downarrow$ $\downarrow$ $\downarrow$

w_1 w_2 w_n

Optimal Binary Search Tree





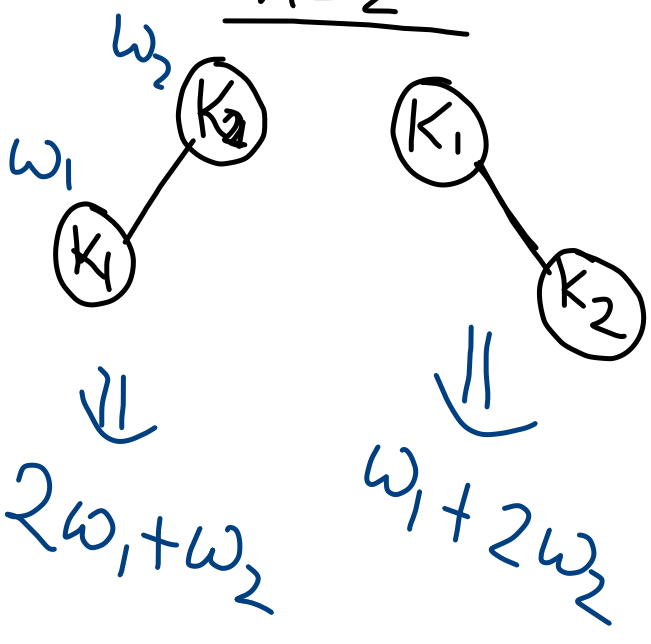
$K_1 < K_2 < K_3 < \cdots < K_n$

$\downarrow \omega_1 \quad \downarrow \omega_2 \quad \downarrow \omega_3 \qquad \downarrow \omega_n$

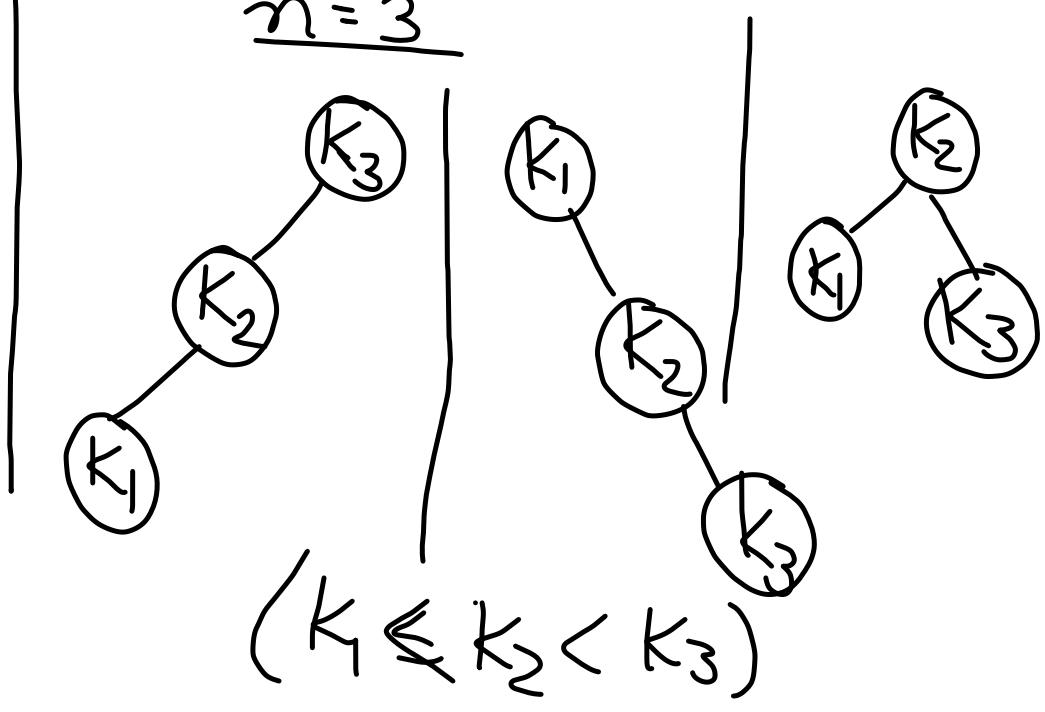
Minimize $\sum_{i=1}^{n} \omega_i \left(\text{depth}(K_i) + 1\right)$

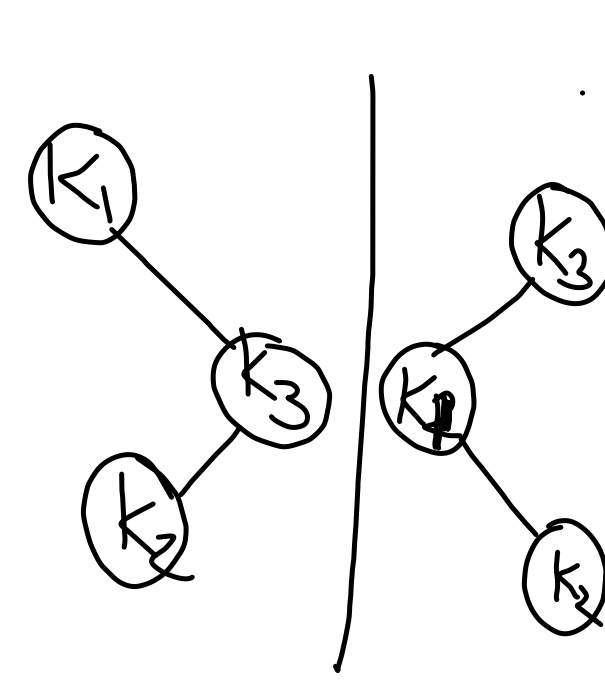
T

$n = 2$



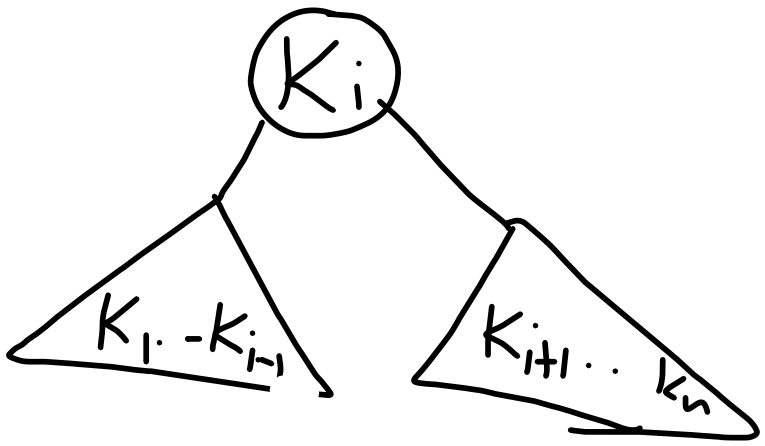
$n = 3$





$n \rightarrow$ nodes. How many BSTs?

$K_1 < \cdots < K_n$



$$T(n) = \sum_{i=1}^{n} T(i-1) \cdot T(n-i)$$

$\Downarrow$

Catalan no.

(Exponential)

$\Leftarrow$

Brute force not possible

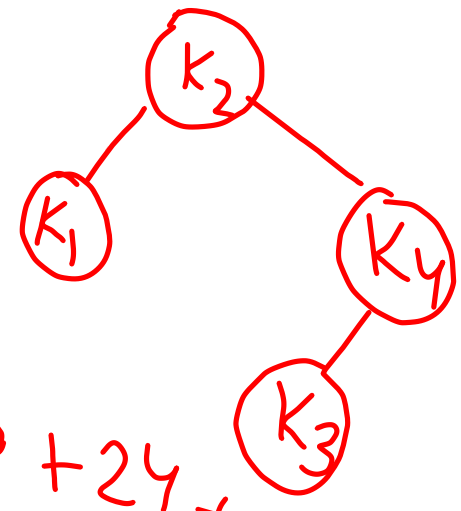
Possible Solution:

(Greedy)

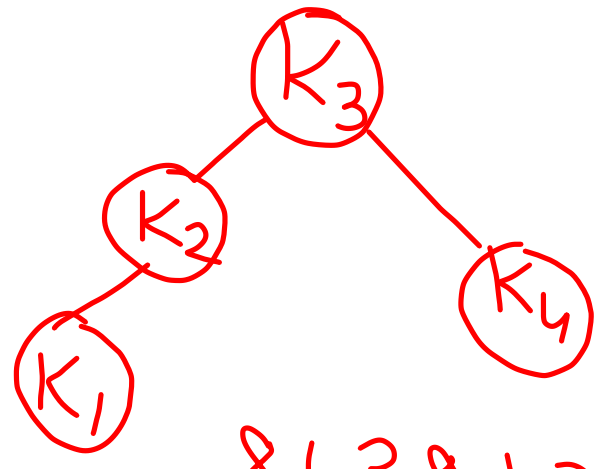
- Choose the max weight node & put it in the root
- Recursively use the same algo.

K_1	K_2	K_3	K_4
1	10	8	9

Contradiction



10+18+24+2
= 54



8+38+3
= 49

Dynamic Prog

- Deciding the root node
  - Guess (Possible choices n)

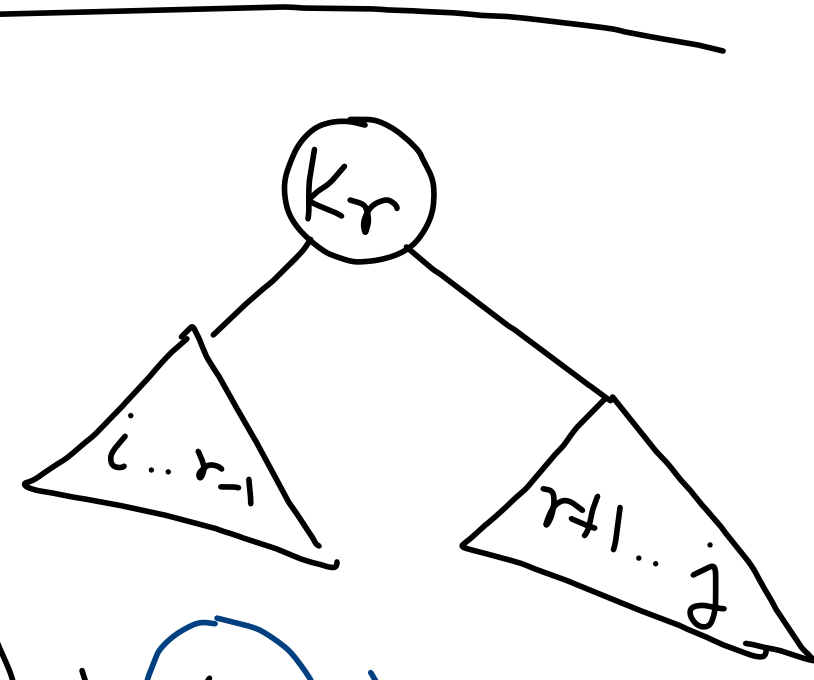
$k_i \dots k_j$ (BST)

$$e(i,j) = \begin{cases} w_i, & i = j \\ \min_{r=i+1}^{j-1} \left[e(i,r-1) + e(r+1..j) + w_r \right] & \end{cases}$$

$w_r \times$ $w_{i..j} \Rightarrow \sum_{k=i}^{j} w_k$

$k_1 \dots k_n$

Time $\Rightarrow \Theta(n^3)$



Coin Game

x_1 . . . . . x_{50}

100 | 55 | 4 | 60

R - 60
T - 100
R - 55
T - 4

R - 100
T - 60
R - 55

1st player can ensure 155

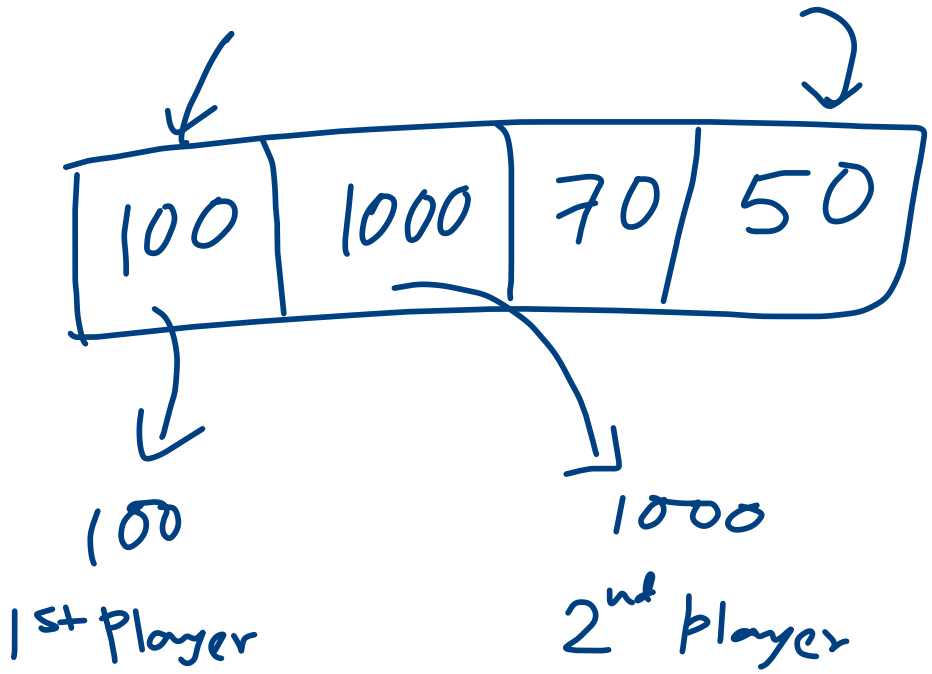
(First player)

Rajdeep's Strategy

$X_{odd}(x_1 + x_3 + \cdots + x_{49})$
$X_{even}(x_2 + x_4 + \cdots + x_{50})$

$X_{odd} > X_{even}$

Always take the odd coin.



How to maximize the amount of money for 1st player? (P1)

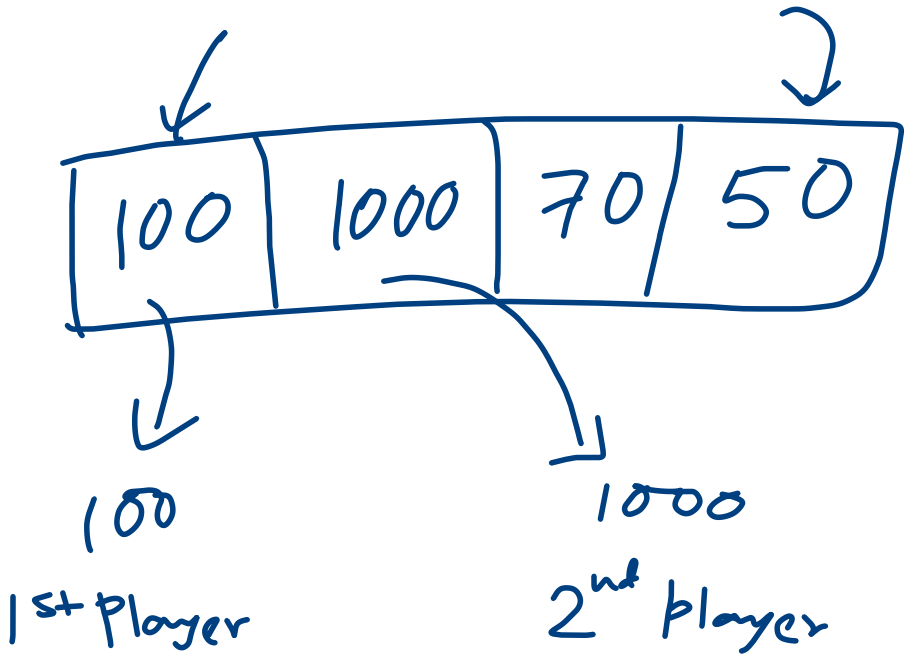
$X(i,j)$ → max value P1 can definitely win if it is his/her turn & the coins left are $x_i, \ldots, x_j$

⇓

- $X(i,i) \rightarrow x_i$
- $X(i,i+1) \rightarrow \max\{x_i, x_{i+1}\}$

$$X(i,j) = \max\begin{cases} X(i+1,j) \\ X(i,j-1) \end{cases} \begin{cases} +x_i & \text{i-choose} \\ +x_j & \text{j-choose} \end{cases}$$

- Opponent's move
- You cannot control



How to maximize the amount of money for 1st player? (P1)

$X(i,j)$ → max value P1 can definitely win if it is his/her turn & the coins left are $x_i, \ldots, x_j$

— $X(i,i) \rightarrow x_i$

— $X(i,i+1) \rightarrow \max\{x_i, x_{i+1}\}$

$$X(i,j) = \max\{\underbrace{X^*_{i+1,j} + x_i}_{i\text{-choose}}, \underbrace{X^*_{i,j-1} + x_j}_{j\text{-choose}}\}$$

$X(i+1,j)$

$X(i,j-1)$

- Opponent's move
- You cannot control

Complexity = $\Theta(n^2)$

Subproblem $X(i+1, j)$ → opponent choose

$$\min\{X(i+2, j),\ X(i+1, j-1)\} \longrightarrow X^*_{i+1, j}$$

(opponent picks $i+1$)

(opponent picks j)

Subproblem $X(i, j-1)$

$$\min\{X(i+1, j-1),\ X(i, j-2)\} \longrightarrow X^*_{i, j-1}$$