Optimal BST:

Input: $f[1..n]$ frequencies

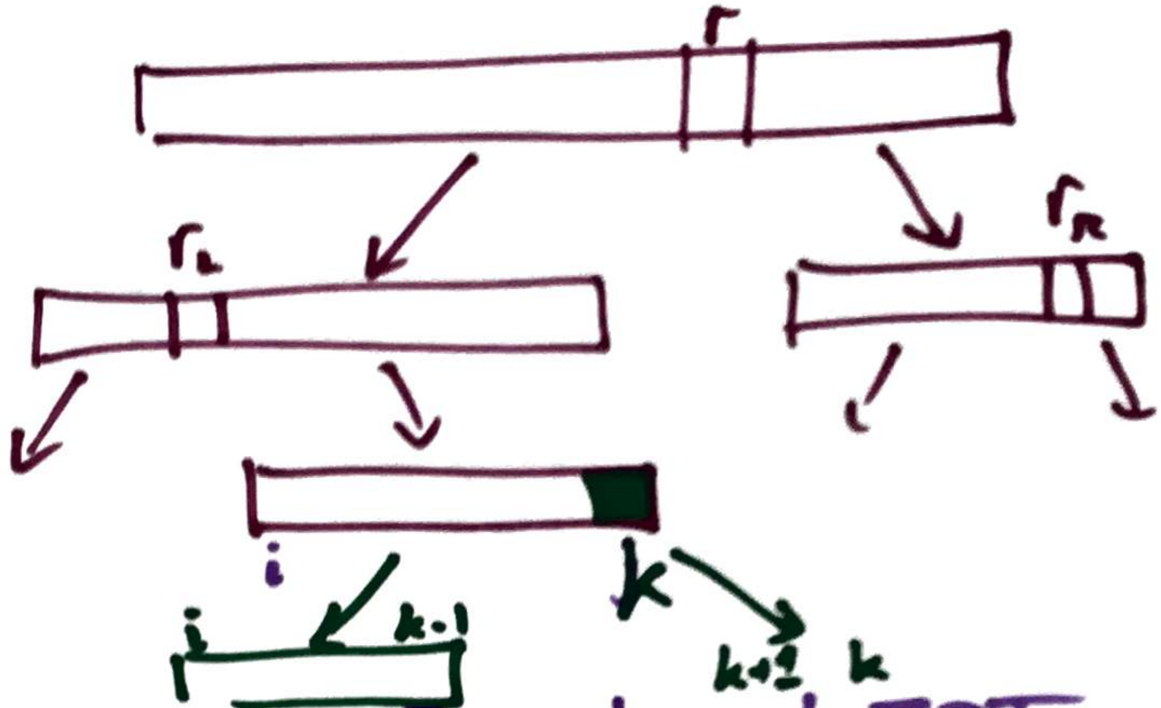
Output: total cost of all searches in the best BST for these frequencies

$$\mathrm{Cost}(\overset{?}{T}, f[1..n]) = \sum_{i=1}^{n} f[i] \cdot \#\mathrm{ancestors}(i)$$

$$= \sum_{i=1}^{n} f[i] + \mathrm{Cost}(T.\mathrm{left}, f[1..r-1])$$

$$+ \mathrm{Cost}(T.\mathrm{right}, f[r+1..n])$$

What is the root?

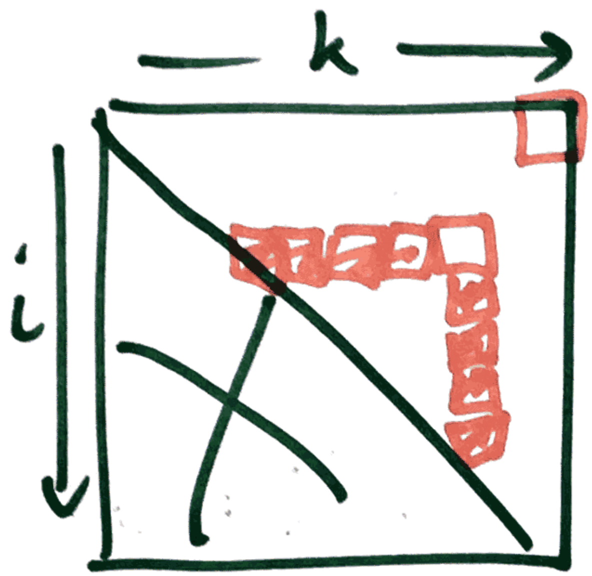


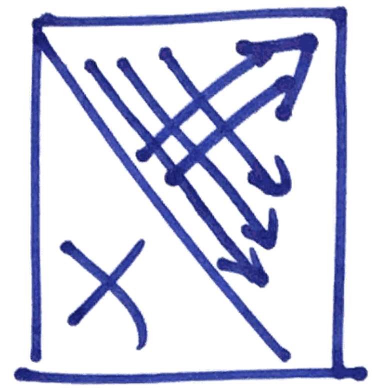
OptCost$(i, \cancel{j}\,k)$:= cost of optimal BST for frequencies $f[i..\cancel{j}\,k]$

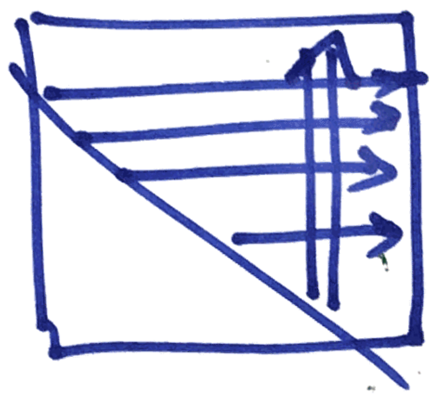
We need OptCost$(1, n)$

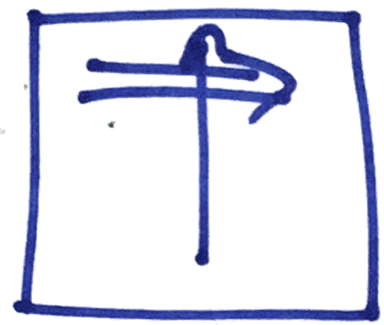
$$\text{OptCost}(i,j) = \begin{cases} \\ \displaystyle\sum_{k=i}^{j} F[k] \end{cases}$$

$$
\mathrm{OptCost}(i,k) = \begin{cases} 0 & \text{if } i > k \\ \displaystyle\sum_{j=i}^{k} f(j) + \mathop{\substack{\min \\ \cancel{\max}}}_{i \le j \le k} \left\{ \begin{array}{c} \mathrm{OptCost}(i, j-1) \\ + \\ \mathrm{OptCost}(j+1, k) \end{array} \right\} & \text{otherwise} \end{cases}
$$









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

Which element is the root of the optimal binary tree?

$OptCost(i,k) :=$ the total cost of the optimal search tree for the frequency interval $f[i..k]$.

$$OptCost(i,k) = \begin{cases} 0 & \text{if } i > k \\ \displaystyle\sum_{j=i}^{k} f[i] + \min_{i \le r \le k} \big\{OptCost(i,r-1) \;+\; OptCost(r+1,k)\big\} & \text{otherwise} \end{cases}$$

$$F(i,k) := \sum_{j=i}^{k} f[j]$$

$$F(i,k) = \begin{cases} f[i] & \text{if } i = k \\ F(i,k-1) + f[k] & \text{otherwise} \end{cases}$$

```
InitF(f[1..n]):
  for i ← 1 to n
      F[i, i−1] ← 0
      for k ← i to n
          F[i, k] ← F[i, k−1] + f[k]
```

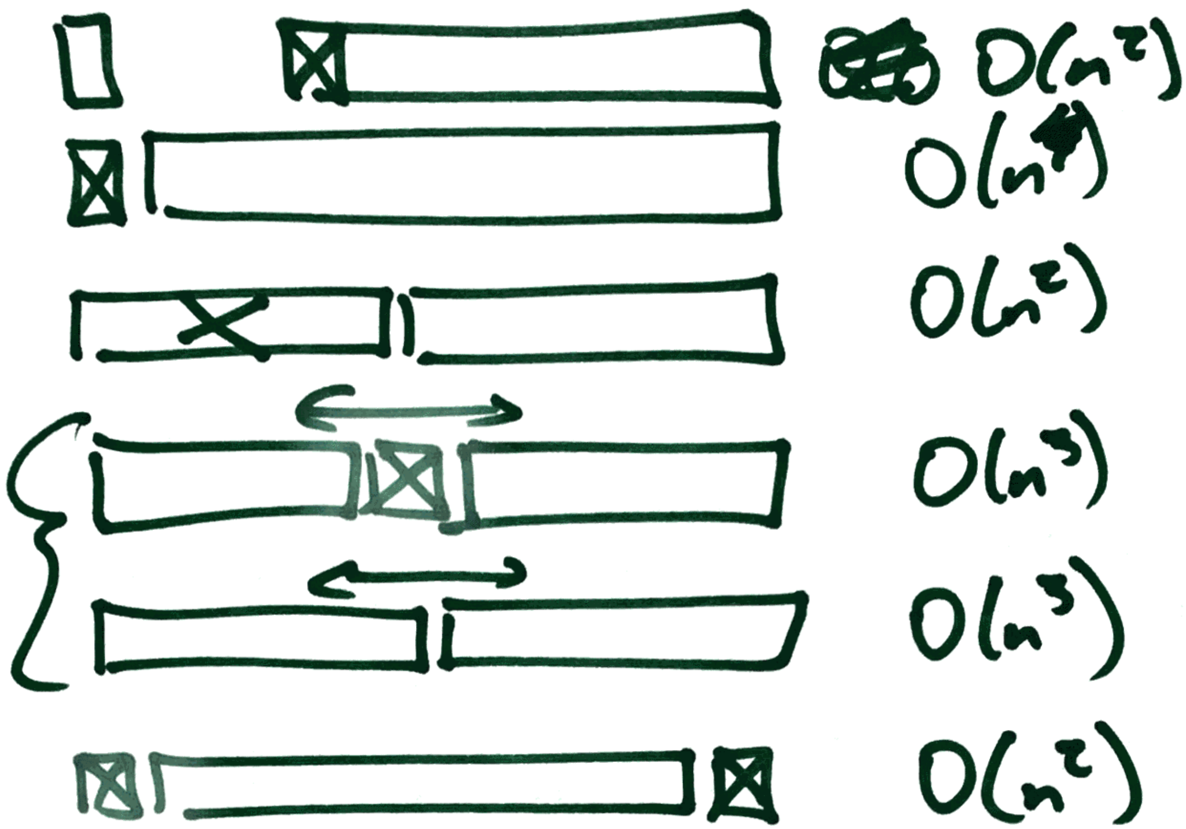
$$OptCost(i,k) = \begin{cases} 0 & \text{if } i > k \\ F(i,k) + \min\limits_{i \le r \le k} \left\{ OptCost(i, r-1) + OptCost(r+1, k) \right\} & \text{otherwise} \end{cases}$$

```
ComputeOptCost(i, k):
    OptCost[i, k] ← ∞
    for r ← i to k
        tmp ← OptCost[i, r − 1] + OptCost[r + 1, k]
        if OptCost[i, k] > tmp
            OptCost[i, k] ← tmp
    OptCost[i, k] ← OptCost[i, k] + F[i, k]
```

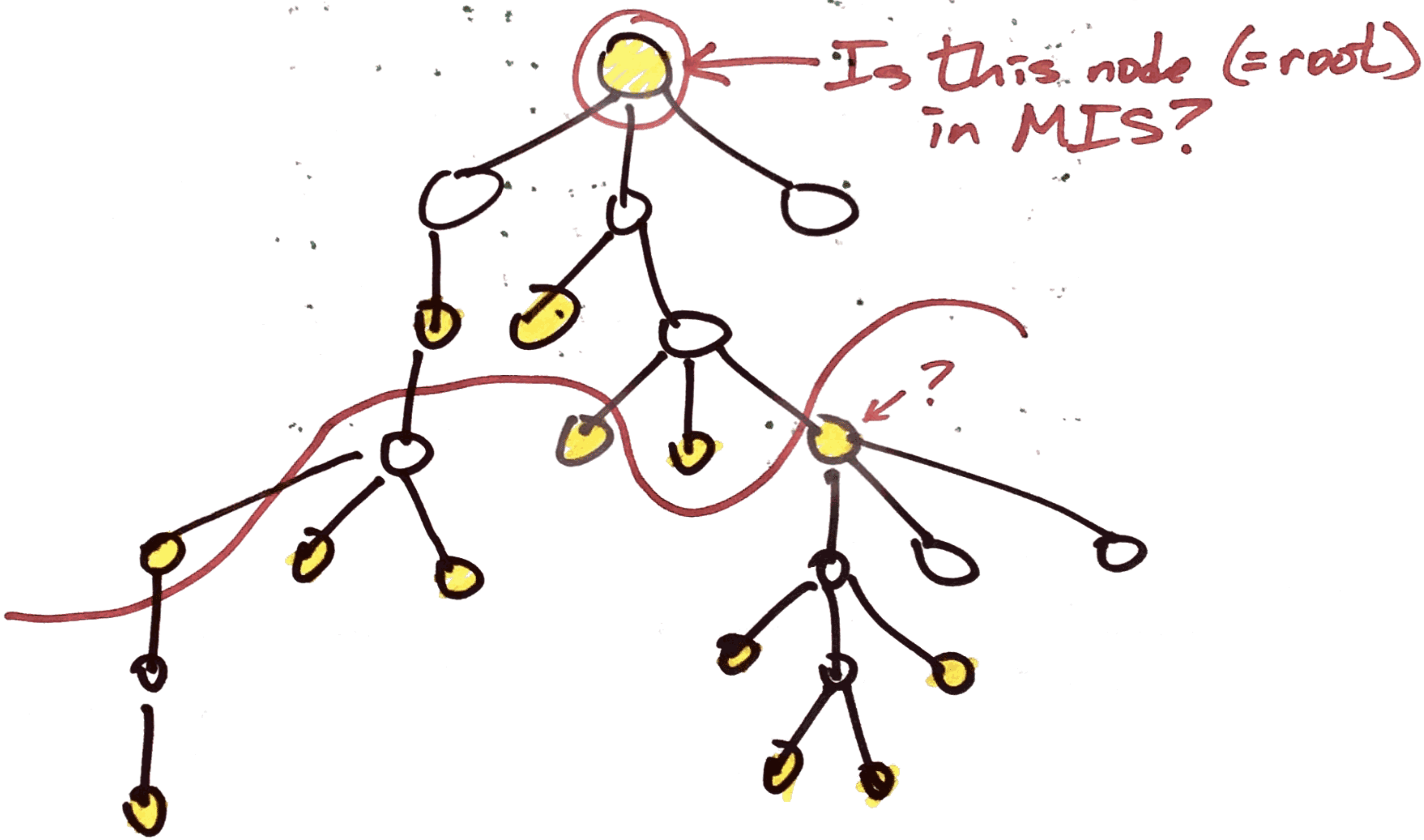
$O(n)$

```
OptimalBST(f[1..n]):
    InitF(f[1..n])
    for i ← 1 to n + 1
        OptCost[i, i − 1] ← 0
    for d ← 0 to n − 1
        for i ← 1 to n − d
            ComputeOptCost(i, i + d)      ← O(n)
    return OptCost[1, n]
```

$O(n^3)$


O(n^2)
O(n^2)
O(n^3)
O(n^3)
O(n^2)

Maximum Independent Set in Trees



MIS(v) = size of max ind. set of the subtree rooted at v.

MIS yes(v) = size of max ind set
of subtree @ v
<u>including</u> v

MIS no(v) = ~~~~~~~~~~~~ <u>excluding</u> v.

MIS(v, p) = size of max ind set
of subtree @ v
excluding v if p = True

we want ~~MIS(root, T)~~, $\boxed{\text{MIS(root, F)}}$

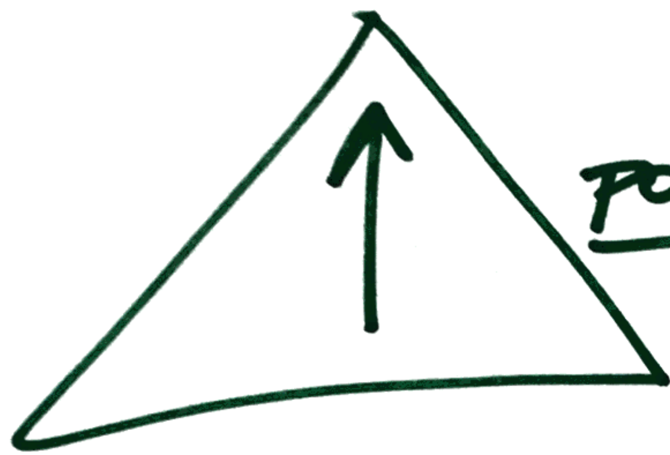
$$
MIS(v,p) = \begin{cases} \sum_{w \downarrow v} MIS(w,F) & \text{if } p = \text{TRUE} \\ \max \begin{cases} 1 + \sum_{w \downarrow v} MIS(w,T) \\ \sum_{w \downarrow v} MIS(w,F) \end{cases} & \text{if } p = \text{FALSE} \end{cases}
$$

$w \downarrow v$ means "w is a child of v"

$$\sum \emptyset = 0$$

Add two fields to every node record

$$v.\mathrm{MIST} = \mathrm{MIS}(v, \mathrm{TRUE})$$
$$v.\mathrm{MISF} = \mathrm{MIS}(v, \mathrm{FALSE})$$



<u>postorder</u>

```
MIS(T):
    MIS2(root(T), FALSE)

MIS2(v, p) =
    for all children w of v
        MIS2(v, p)

    if p
        sum ← 0
        for all children w of v
            sum ← sum + w.MISF

        return sum
    else
        ⋮
```