CS F364: Design & Analysis of Algorithm

14 Optimal Binary Search Trees

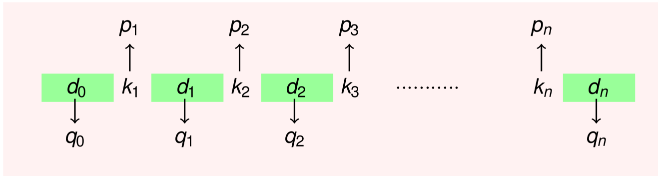
Dr. Kamlesh Tiwari
Assistant Professor, Department of CSIS,
BITS Pilani, Pilani Campus, Rajasthan-333031 INDIA

Feb 17, 2021 ONLINE (Campus @ BITS-Pilani Jan-May 2021)

http://ktiwari.in/algo

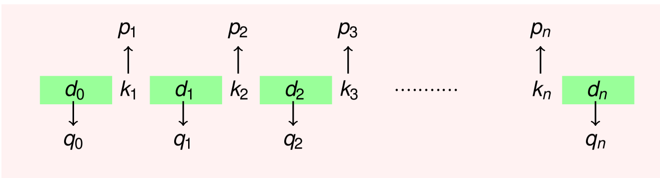
Searching

Consider a subsequence $K = < k_1, k_2, ..., k_n >$ of n distinct keys (in sorted order). Let p_i be the probability of searching k_i



- We wish to construct a binary search tree (BST) with minimum expected search cost

Introduction



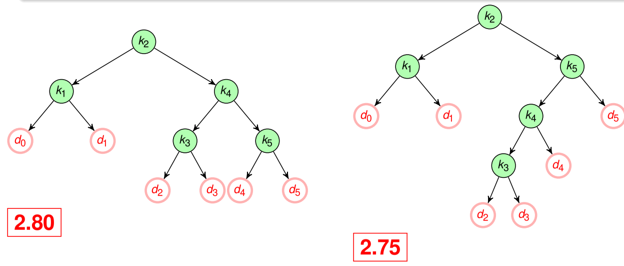
$$\sum_{i=1}^{n} p_i + \sum_{i=0}^{n} q_i = 1$$

Expected Search Cost in tree T

$$E[T] = \sum_{i=1}^{n}(depth_T(k_i) + 1) \times p_i + \sum_{i=0}^{n}(depth_T(d_i) + 1) \times q_i$$

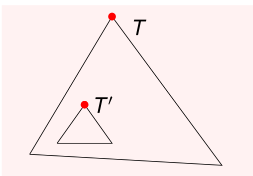
Example: Expected Search Cost

Let $< p_1, p_2, p_3, p_4, p_5 > = < 0.15, 0.10, 0.05, 0.10, 0.20 >$ and $< q_0, q_1, q_2, q_3, q_4, q_5 > = < 0.05, 0.10, 0.05, 0.05, 0.05, 0.10 >$



Optimal Substructure

- Any non-leaf sub-tree of BST would must contains keys in continuous range $k_i, ...k_j$ for some $1 \leq i \leq j \leq n$
- Subtree T' of an optimal BST T must be optimal



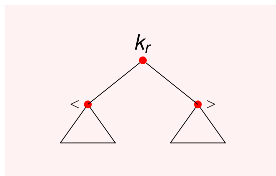
Contradiction: If T'' is optimal then put T'' in T at the place of T'

- Bruit-force would take $\Omega(4^n/n^{3/2})$ time

Problem formulation

Let k_r be at root

- $T_<$ has keys $k_0, k_1, ..., k_{r-1}$
- $T_>$ has keys $k_{r+1}, k_{r+2}, ..., k_n$
- Let $E[T_<]$ is expected search cost of $T_<$ and
  $w[T_<] = \sum_{i=1}^{r-1} p_i + \sum_{i=0}^{r-1} q_i$,
  $w[T_>] = \sum_{i=r+1}^{n} p_i + \sum_{i=r}^{n} q_i$



Expected search cost of the tree is

$$\begin{aligned} E[T] &= E[T_<] + w[T_<] + E[T_>] + w[T_>] + p_r \\ &= E[T_<] + E[T_>] + 1 \end{aligned}$$

Overlapping subproblems?

Using Dynamic Programming

- $e[i,j]$ expected search cost for optimal BST for keys $k_i, ..., k_j$
- $w[i,j] = \sum_{v=i}^{j} p_v + \sum_{v=i-1}^{j} q_v$

If k_r is root then

$$\begin{aligned} e[i,j] &= p_r + e[i,r-1] + w[i,r-1] + e[r+1,j] + w[r+1,j] \\ &= e[i,r-1] + e[r+1,j] + w[i,j] \end{aligned}$$

We have to choose r that maximizes $e[i,j]$

$$e[i,j] = \begin{cases} q_{i-1} & \text{if } j = i-1 \\ \min_{i \le r \le j}\{e[i,r-1] + e[r+1,j] + w[i,j]\} & \text{otherwise} \end{cases}$$

The algorithm

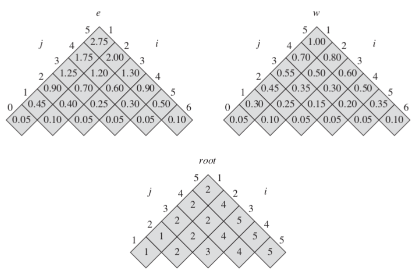
Algorithm 1: Optimal-BST(p, q, n)

```
for i = 1 to n+1 do
    e[i, i-1] = w[i, i-1] = q_{i-1}
for l = 1 to n do
    e[i, i-1] = w[i, i-1] = q_{i-1} for i = 1 to n-l+1 do
        j = i + l - 1
        e[i, j] = ∞
        w[i, j] = w[i, j-1] + p_j + q_j
        for i = 1 to n+1 do
            if e[i, r-1] + e[r+1, j] + w[i, j] < e[i, j] then
                e[i, j] = e[i, r-1] + e[r+1, j] + w[i, j]
                root[i, j] = r
return e and root
```

Complexity: time $O(n^3)$, space $O(n^2)$

Example: Expected Search Cost

Let $< p_1, p_2, p_3, p_4, p_5 > = < 0.15, 0.10, 0.05, 0.10, 0.20 >$ and $< q_0, q_1, q_2, q_3, q_4, q_5 > = < 0.05, 0.10, 0.05, 0.05, 0.05, 0.10 >$



Thank You!

Thank you very much for your attention! (Reference[1])

Queries ?

[1] Book - *Introduction to Algorithm*, By THOMAS H. CORMEN, CHARLES E. LEISERSON, RONALD L. RIVEST, CLIFFORD STEIN