Optimal Binary Search Tree
(with dummy keys)

Optimal binary Search Trees

Given: search probabilities p_i for each key k_i $(i = 1, \ldots, n)$ and q_i of each interval d_i $(i = 0, \ldots, n)$ between search keys of a binary search tree. $\sum_{i=1}^{n} p_i + \sum_{i=0}^{n} q_i = 1$.

Optimal binary Search Trees

Given: search probabilities p_i for each key k_i $(i = 1, \dots, n)$ and q_i of each interval d_i $(i = 0, \dots, n)$ between search keys of a binary search tree. $\sum_{i=1}^{n} p_i + \sum_{i=0}^{n} q_i = 1$.
Wanted: optimal search tree T with key depths $\mathrm{depth}(\cdot)$, that minimizes the expected search costs

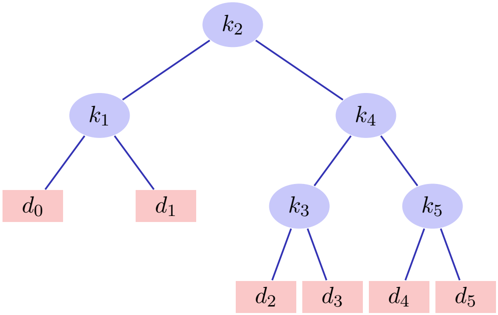
$$\begin{aligned} C(T) &= \sum_{i=1}^{n} p_i \cdot (\mathrm{depth}(k_i) + 1) + \sum_{i=0}^{n} q_i \cdot (\mathrm{depth}(d_i) + 1) \\ &= 1 + \sum_{i=1}^{n} p_i \cdot \mathrm{depth}(k_i) + \sum_{i=0}^{n} q_i \cdot \mathrm{depth}(d_i) \end{aligned}$$

Example

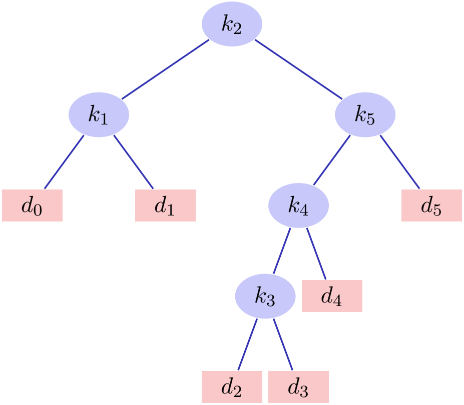
Expected Frequencies

i	0	1	2	3	4	5
p_i		0.15	0.10	0.05	0.10	0.20
q_i	0.05	0.10	0.05	0.05	0.05	0.10

Example



Search tree with expected costs 2.8



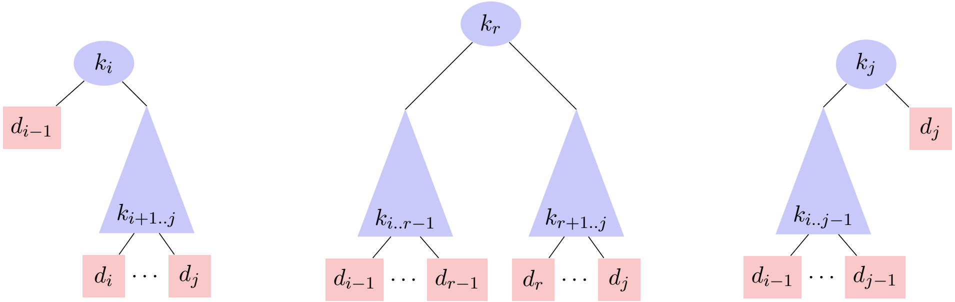
Search tree with expected costs 2.75

Structure of a optimal binary search tree

- Subtree with keys $k_i, \dots, k_j$ and intervals $d_{i-1}, \dots, d_j$ must be optimal for the respective sub-problem.[39]
- Consider all subtrees with roots k_r and optimal subtrees for keys $k_i, \dots, k_{r-1}$ and $k_{r+1}, \dots, k_j$

[39]The usual argument: if it was not optimal, it could be replaced by a better solution improving the overal solution.

Sub-trees for Searching



empty left subtree

non-empty left and right subtrees

empty right subtree

Expected Search Costs

Let $\text{depth}_T(k)$ be the depth of a node k in the sub-tree T. Let k be the root of subtrees T_r and T_{L_r} and T_{R_r} be the left and right sub-tree of T_r. Then

$$\text{depth}_T(k_i) = \text{depth}_{T_{L_r}}(k_i) + 1,\ (i < r)$$
$$\text{depth}_T(k_i) = \text{depth}_{T_{R_r}}(k_i) + 1,\ (i > r)$$

Expected Search Costs

Let $e[i,j]$ be the costs of an optimal search tree with nodes $k_i, \ldots, k_j$.
Base case $e[i, i-1]$, expected costs d_{i-1}
Let $w(i,j) = \sum_{l=i}^{j} p_l + \sum_{l=i-1}^{j} q_l$.
If k_r is the root of an optimal search tree with keys $k_i, \ldots, k_j$, then

$$e[i,j] = p_r + (e[i, r-1] + w(i, r-1)) + (e[r+1, j] + w(r+1, j))$$

with $w(i,j) = w(i, r-1) + p_r + w(r+1, j)$:

$$e[i,j] = e[i, r-1] + e[r+1, j] + w(i,j).$$

Dynamic Programming

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

Computation

Tables $e[1\dots n+1, 0\dots n], w[1\dots n+1, 0\dots m], r[1\dots n, 1\dots n]$ Initially

- $e[i, i-1] \leftarrow q_{i-1}$, $w[i, i-1] \leftarrow q_{i-1}$ for all $1 \leq i \leq n+1$.

We compute

$$
\begin{aligned}
w[i,j] &= w[i, j-1] + p_j + q_j \\
e[i,j] &= \min_{i \leq r \leq j} \{e[i, r-1] + e[r+1, j] + w[i,j]\} \\
r[i,j] &= \arg\min_{i \leq r \leq j} \{e[i, r-1] + e[r+1, j] + w[i,j]\}
\end{aligned}
$$

for intervals $[i,j]$ with increasing lengths $l = 1, \dots, n$, each for $i = 1, \dots, n-l+1$. Result in $e[1,n]$, reconstruction via r. Runtime $\Theta(n^3)$.

Example

i	0	1	2	3	4	5
p_i		0.15	0.10	0.05	0.10	0.20
q_i	0.05	0.10	0.05	0.05	0.05	0.10

w

j						
0	0.05					
1	0.30	0.10				
2	0.45	0.25	0.05			
3	0.55	0.35	0.15	0.05		
4	0.70	0.50	0.30	0.20	0.05	
5	1.00	0.80	0.60	0.50	0.35	0.10
	1	2	3	4	5	6 i

e

j						
0	0.05					
1	0.45	0.10				
2	0.90	0.40	0.05			
3	1.25	0.70	0.25	0.05		
4	1.75	1.20	0.60	0.30	0.05	
5	2.75	2.00	1.30	0.90	0.50	0.10
	1	2	3	4	5	6 i

r

j					
1	1				
2	1	2			
3	2	2	3		
4	2	2	4	4	
5	2	4	5	5	5
	1	2	3	4	5 i