Chapter 5, Dynamic Programming:

- Introduction and sample problems (last week Friday)
- Principles of DP (Monday)
- DP algorithms, solutions to sample problems (Wednesday)
- Optimal BSTs (**Today**)
- Finish up optimal BSTs, review for test 2 (next week Monday)
- **Test 2**, next week Wed Apr 6, *not* covering DP

Today (and Monday):

- Optimal BST definition
- The Optimal-BST-building problem
- The dynamic programming solution

Why this problem?

- It connects dynamic programming with the quest for a better map.
- Its hardness is in the right places (building the table—hard; reconstructing solution—trivial).
- It is a representative of a bigger concept: What if we had more information—how would that change the problem.

Game plan:

- Understand the problem itself
- Understand the recursive characterization
- Understand the table-building algorithm

The **optimal binary search tree** problem:

- Assume we know all the keys $k_0, k_1, \ldots k_{n-1}$ ahead of time.
- Assume further that we know the probabilities $p_0, p_1, \ldots p_{n-1}$ of each key's lookup.
- Assume even further that we know the "miss probabilities" $q_0, q_1, \ldots q_n$ where q_i is the probability that an *extraneous key* falling between k_{i-1} and k_i will be looked up.
- We want to build a tree to minimize the *expected cost* of a look up, which is the *total weighted depth* of the tree:

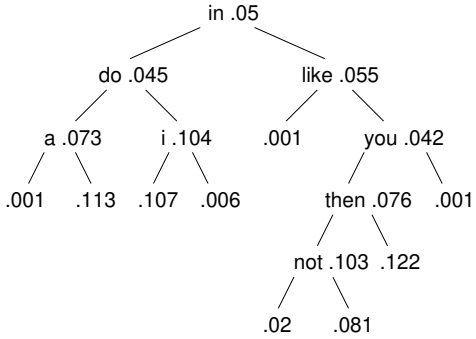
$$\sum_{i=0}^{n-1} p_i \ depth(k_i) + \sum_{i=0}^{n} q_i \ depth(m_i)$$

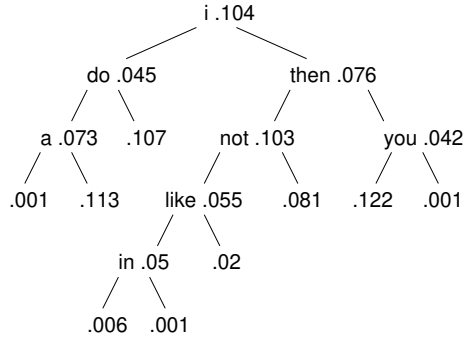
  where $depth(x)$ is the number of nodes to be inspected on the route from the root to node x, k_i stands for the node containing key k_i [notational abuse], and m_i is the dummy node between keys k_{i-1} and and k_i.
- Note that the rules of probability require $\sum_{i=0}^{n-1} p_i + \sum_{i=0}^{n} q_i = 1$

i	84	eat	24	ham	10	fox	7	rain	4
not	83	will	21	there	9	on	7	see	4
them	61	sam	19	train	9	tree	6	try	4
a	59	with	19	anywhere	8	say	5	boat	3
like	44	am	16	house	8	so	5	that	3
in	40	could	14	mouse	8	be	4	are	2
do	36	here	11	or	8	goat	4	good	2
you	34	the	11	box	7	let	4	thank	2
would	26	eggs	10	car	7	may	4	they	2
and	24	green	10	dark	7	me	4	if	1

Key or miss event	combined frequency
{ }	0
a	59
{ am and anywhere are be boat box car could dark }	92
do	36
{ eat eggs fox goat good green ham here house }	86
i	84
{ if let }	5
in	40
{ }	0
like	44
{ may me mouse }	16
not	83
{ on or rain same say see so thank that the }	65
then	61
{ there they train tree try will with would }	99
you	34
{ }	0

	0	1	2	3	4	5	6	7
k_i	a	do	i	in	like	not	then	you
p_i	.073	.045	.104	.05	.055	.103	.076	.042
q_i	.001	.113	.107	.006	.001	.02	.081	.122	.001





Total weighted depth for a given tree (expected lookup cost):

$$\underbrace{\sum_{i=0}^{n-1} p_i depth(k_i)}_{\text{keys}} + \underbrace{\sum_{i=0}^{n} q_i \ depth(m_i)}_{\text{misses}}$$

Let $depth_{k_a}(k_i)$ be the depth of the node with k_i in the subtree rooted at node with k_1. For example, if k_r is the root of the entire tree and k_a is a child of the root, then

$$depth_{k_r}(k_i) = depth_{k_a}(k_i) + 1$$

Then we can rewrite the total weighted depth as

$$\underbrace{\sum_{i=0}^{r-1} p_i \ depth_{k_r}(k_i) + \sum_{i=0}^{r} q_i \ depth_{k_r}(m_i)}_{\text{left subtree total weighted depth (absolute)}} + p_r + \underbrace{\sum_{i=r+1}^{n-1} p_i \ depth_{k_r}(k_i) + \sum_{i=r+1}^{n} q_i \ depth_{k_r}(m_i)}_{\text{right subtree total weighted depth (absolute)}}$$

Again, let k_r be the root of the entire tree and k_a and k_b be the root's children. Then

$$\underbrace{\sum_{i=0}^{r-1} p_i(depth_{k_a}(k_i)+1) + \sum_{i=0}^{r} q_i(depth_{k_a}(m_i)+1)}_{\text{left subtree total weighted depth (absolute)}} + p_r + \underbrace{\sum_{i=r+1}^{n-1} p_i(depth_{k_b}(k_i)+1) + \sum_{i=r+1}^{n} q_i(depth_{k_r}(m_i)+1)}_{\text{right subtree total weighted depth (absolute)}}$$

Convert to "relative depth":

$$\underbrace{\sum_{i=0}^{n-1} p_i + \sum_{i=0}^{n} q_i}_{\text{total probability}} + \underbrace{\sum_{i=0}^{r-1} p_i\ depth_{k_a}(k_i) + \sum_{i=0}^{r} q_i\ depth_{k_a}(m_i)}_{\text{left subtree total weighted depth (relative)}} + \underbrace{\sum_{i=r+1}^{n-1} p_i\ depth_{k_b}(k_i) + \sum_{i=r+1}^{n} q_i\ depth_{k_r}(m_i)}_{\text{right subtree total weighted depth (relative)}}$$

Let $TWD(k)$ be the total weighted depth of the tree rooted at k (relative to k) and $TP(k)$ be the total probability of the tree rooted at k. Then

$$TWD(k_r) = TP(k_r) + TWD(k_a) + TWD(k_b)$$

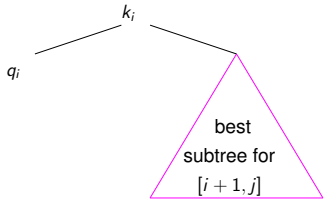
Let $P[i][j]$ be the total probabilities of the keys and misses in the range $[i, j]$:

$$P[i][j] = \sum_{k=i}^{j} p_k + \sum_{k=i}^{j+1} q_k$$

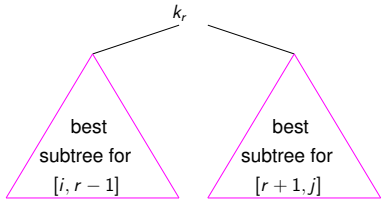
Let $C[i][j]$ be the least total weighted depth of any BST composed from keys in the range $[i, j]$. The recursive characterization is

$$C[i][j] = \begin{cases} 2q_i + p_i + 2q_{i+1} & \text{if } i = j \\ P[i][j] + \min \left\{ \begin{array}{l} q_i + C[i+1][j] \\ C[i][r-1] + C[r+1][j] \text{ for } r \in (i, j) \\ C[i][j-1] + q_{j+1} \end{array} \right\} & \text{if } i < j \end{cases}$$

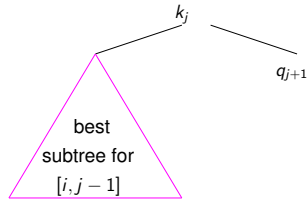
$$C[i][j] = \begin{cases} 2q_i + p_i + 2q_{i+1} & \text{if } i = j \\ P[i][j] + \min \left\{ \begin{array}{l} q_i + C[i+1][j] \\ C[i][r-1] + C[r+1][j] \text{ for } \ r \in (i,j) \\ C[i][j-1] + q_{j+1} \end{array} \right\} & \text{if } i < j \end{cases}$$



$q_i + C[i+1][j]$

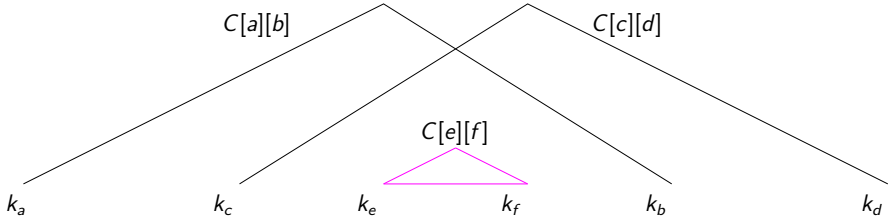


$C[i][r-1] + C[r+1][j]$



$C[i][j-1] + q_{j+1}$

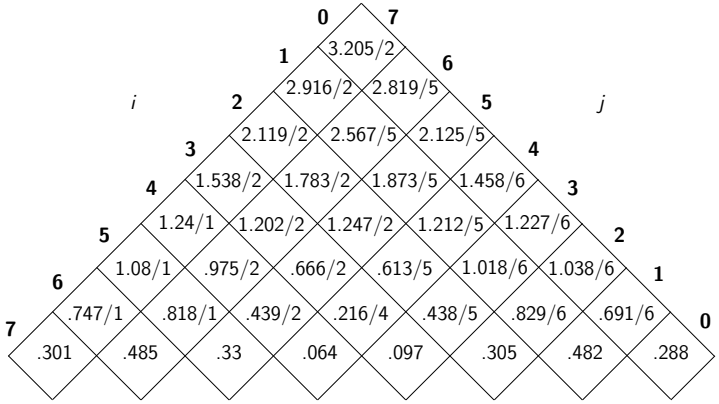
$$C[i][j] = \begin{cases} 2q_i + p_i + 2q_{i+1} & \text{if } i = j \\ P[i][j] + \min \left\{ \begin{array}{l} q_i + C[i+1][j] \\ C[i][r-1] + C[r+1][j] \text{ for } r \in (i,j) \\ C[i][j-1] + q_{j+1} \end{array} \right\} & \text{if } i < j \end{cases}$$



$$C[i][j] = \begin{cases} 2q_i + p_i + 2q_{i+1} & \text{if } i = j \\ P[i][j] + \min \left\{ \begin{array}{l} q_i + C[i+1][j] \\ C[i][r-1] + C[r+1][j] \text{ for } \ r \in (i,j) \\ C[i][j-1] + q_{j+1} \end{array} \right\} & \text{if } i < j \end{cases}$$

$$P[i][j] = \begin{cases} q_i + p_i + q_{i+1} & \text{if } i = j \\ \left\{ \begin{array}{ll} & q_i + p_i + P[i+1][j] \\ \text{or} & P[i][r-1] + p_r + P[r+1][j] \text{ for } \ r \in (i,j) \\ \text{or} & P[i][j-1] + p_j + q_{j+1} \end{array} \right\} & \text{if } i < j \end{cases}$$

	0	1	2	3	4	5	6	7
k_i	a	do	i	in	like	not	then	you
p_i	.073	.045	.104	.05	.055	.103	.076	.042
q_i	.001	.113	.107	.006	.001	.02	.081	.122	.001



Coming up:

Catch up on projects...

Due **Fri, Apr 1** *(end of day)*
Do Project 6.1.b as a practice problem
Take quiz (on Section 6.4)

Due **Mon, Apr 4** *(end of day—though a skim is recommmended for Apr 1)*
Read Section 6.5
(No quiz on Section 6.5)

(See Schoology for practice problems for Test 2)

Do **Optimal BST** *project (suggested by Friday, April 9)*