6.897
L7.2
Mar. 5, 2003

Binary search trees (BSTs)

- can ask all the same questions as linear search
- worst case search: $\Theta(\lg n)$
- cost model: # nodes inspected, starting at root

Stochastic model: optimal BST [see CLRS] [Gilbert & Moore 1959]

- p_i = probability of searching for i, $1 \le i \le n$
- q_i = probability of searching between i & $i+1$, $0 \le i \le n$ ← NEW: support fast unsuccessful searches
- Dynamic program:
  - root[i, j] = root of optimal tree on $q_{i-1}, p_i, q_i, \ldots, p_j, q_j$.
  - cost[i, j] = cost of this tree × probability of entering tree

Base: root[i, i] = i <<tree on 1 element>>
cost[i, i] = $q_{i-1} + p_i + q_i$ <<charge 1 upon entrance>>

Step: $\text{cost}[i,j] = \min_{i \le r \le j} \Big(\underbrace{q_{i-1} + p_i + q_i + \cdots + p_j + q_j}_{\text{access root}} + \underbrace{\text{cost}[i, r-1]}_{\text{left cost}} + \underbrace{\text{cost}[r+1, j]}_{\text{right cost}} \Big)$

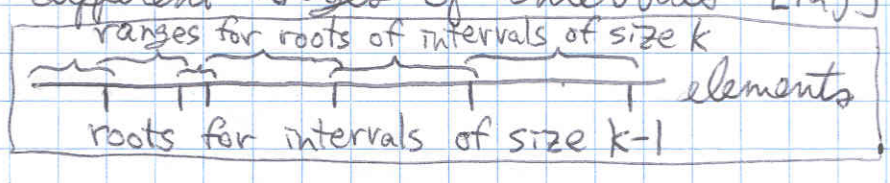
root[i, j] = argmin ...

Special case: cost[i, i-1] = 0.

- Time: $O(n^2)$ subproblems, $O(n)$ time per subproblem $\Big\} \to O(n^3)$ total time

- Speedup: root[i, j-1] ≤ root[i, j] ≤ root[i+1, j] [Knuth 1971]
  ⇒ change min to use this range instead

- Time becomes $O(n^2)$:
  - n different sizes of intervals [i, j] ("levels")



  - total #options at a level = $O(n)$.

OPEN: $o(n^2)$-time algorithm?

- Entropy $H = \sum_{i=1}^{n} p_i \lg 1/p_i + \sum_{i=0}^{n} q_i \lg 1/q_i$
- Optimal BST cost is between $H - \lg H - \lg e + 1$ and $H + 3$