Observations

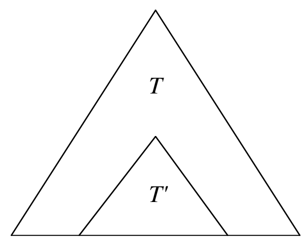
- Optimal BST might not have smallest height.
- Optimal BST might not have highest-probability key at root.

Build by exhaustive checking?

- Construct each n-node BST.
- For each, put in keys.
- Then compute expected search cost.
- But there are $\Omega(4^n/n^{3/2})$ different BSTs with n nodes.

Step 1: The structure of an optimal binary search tree

Consider any subtree of a BST. It contains keys in a contiguous range $k_i, \ldots, k_j$ for some $1 \le i \le j \le n$.

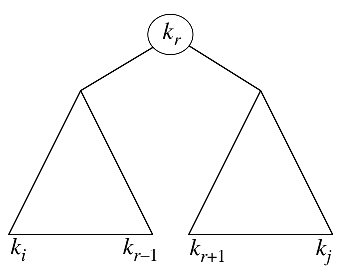


If T is an optimal BST and T contains subtree T' with keys $k_i, \ldots, k_j$, then T' must be an optimal BST for keys $k_i, \ldots, k_j$.

Proof Cut and paste. ■

Use optimal substructure to construct an optimal solution to the problem from optimal solutions to subproblems:

- Given keys $k_i, \ldots, k_j$ (the problem).
- One of them, k_r, where $i \le r \le j$, must be the root.
- Left subtree of k_r contains $k_i, \ldots, k_{r-1}$.
- Right subtree of k_r contains $k_{r+1}, \ldots, k_j$.



- If
    - you examine all candidate roots k_r, for $i \le r \le j$, and
    - you determine all optimal BSTs containing $k_i, \ldots, k_{r-1}$ and containing $k_{r+1}, \ldots, k_j$,

  then you're guaranteed to find an optimal BST for $k_i, \ldots, k_j$.