COL 106 Lecture 14

Topic: Huffman Coding : Analysis

Announcement: Assignment 1 scores published

Recap:

Huffman Coding Problem

Input : Finite set $S = \{a_1, \ldots, a_n\}$ of "letters"
A +ve real number p_i for each a_i.

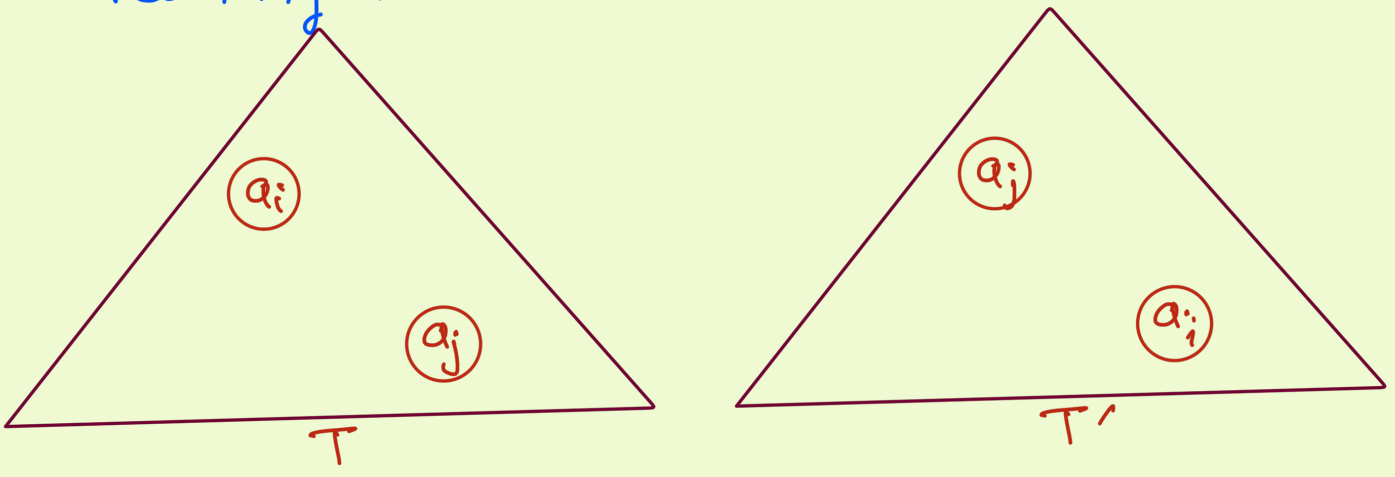
Output: A binary tree T with leaves $a_1, \ldots, a_n$ that minimizes

$$cost(T) = \sum_{i=1}^{n} p_i \cdot depth(a_i).$$

Claim 1: If $p_i < p_j$ then $depth_T(a_i) \geq depth_T(a_j)$ in every optimum binary tree T

Proof: By contradiction. Suppose $p_i < p_j$ and $depth_T(a_i) < depth_T(a_j)$ in the optimum binary tree, say T.

Exchange the locations of a_i and a_j. Call the resulting tree T'.



$$cost(T) - cost(T') = p_i\, depth_T(a_i) + p_j\, depth_T(a_j) - p_i\, depth_{T'}(a_i) - p_j\, depth_{T'}(a_j)$$

$depth_{T'}(a_i) = depth_T(a_j)$, $depth_{T'}(a_j) = depth_T(a_i)$

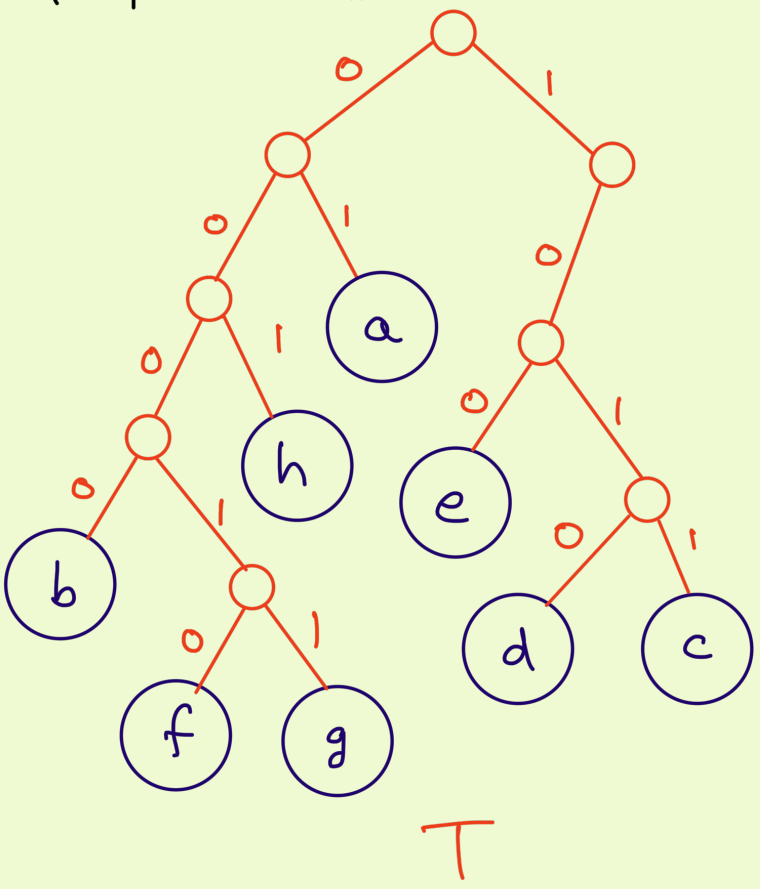
$$= p_i\,(depth_T(a_i) - depth_T(a_j)) - p_j\,(depth_T(a_i) - depth_T(a_j))$$

$$= (p_i - p_j)\,(depth_T(a_i) - depth_T(a_j)) > 0$$

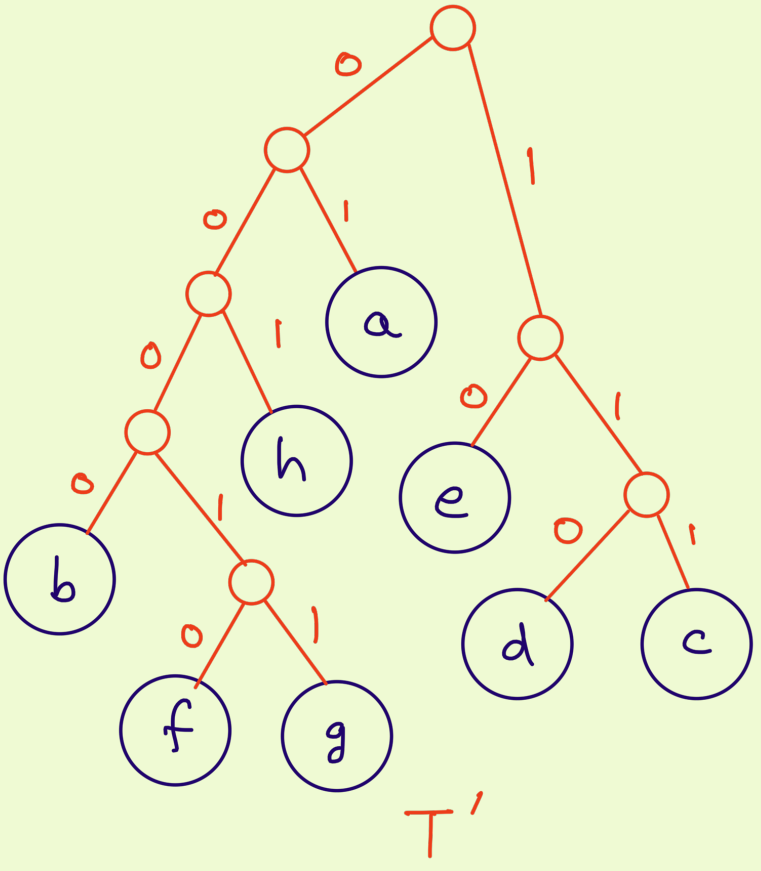
$(p_i - p_j) < 0$, $(depth_T(a_i) - depth_T(a_j)) < 0$

$\therefore$ T' is better than T $\rightarrow$ contradiction to the assumption that T is an optimal tree.

$S = \{a, b, c, d, e, f, g, h\}$. Can T be the optimal tree for some $p_a, p_b, \ldots, p_h > 0$?



T



T'

Claim 2: An optimal tree can never contain a node having exactly 1 child.

Proof: Bypass such a node to get a better tree.

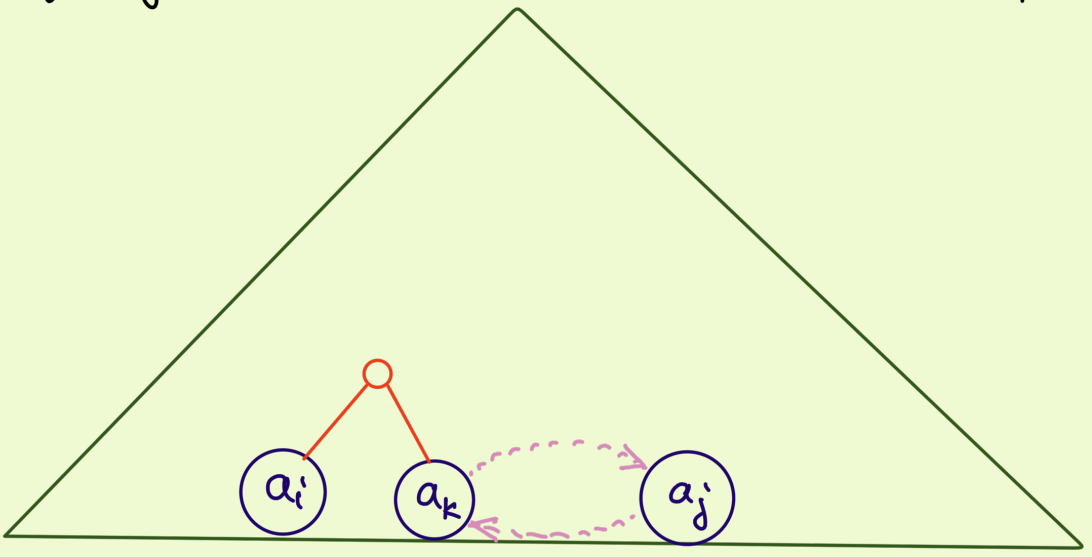
Let a_i : least frequent letter $p_i \leq p_j \ \forall j$

Assume $|S| \geq 2$

Let a_j : second least freq. letter.

Suppose a_k is the sibling of a_i and $k \neq j$

Exchanging a_j and a_k gives us another optimal tree.



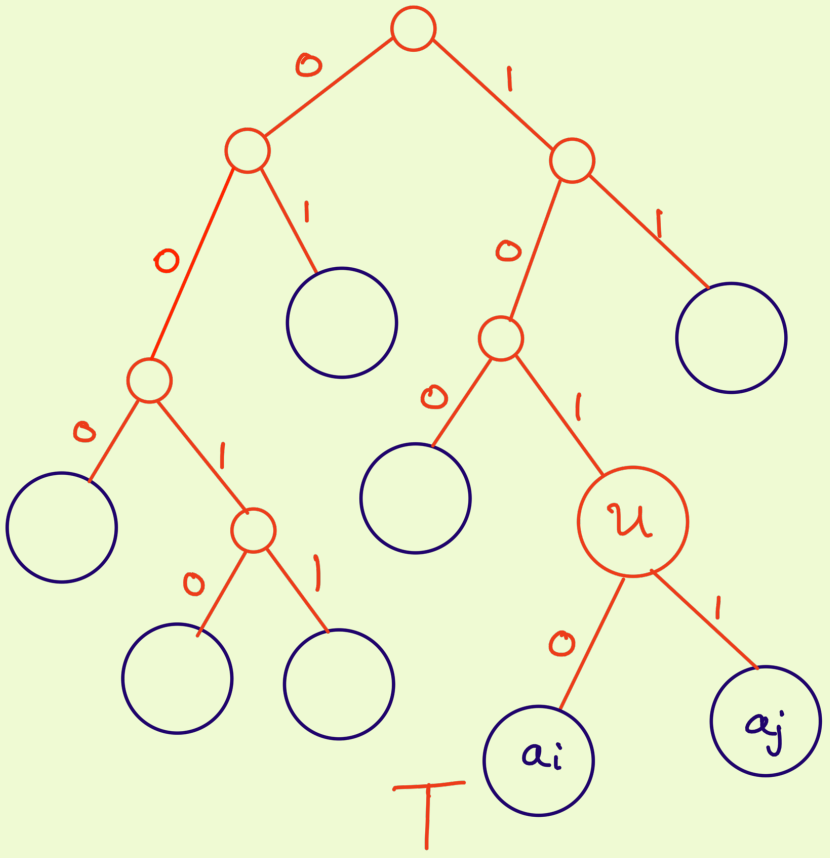
Claim: There exists an optimal tree in which the two least frequent letters are siblings. (Assuming $n > 1$)

Proof: ≥ 2 leaves in the deepest level of an optimal tree. (due to claim 1)

$\therefore$ Two least frequent letters, say a_i, a_j in the bottom most level) (due to claim 2)

a_i has a sibling, say a_k (due to claim 1)

If $k \neq j$, exchange a_k and a_j to obtain another optimum tree.



$$\text{cost}(T) = \sum_k p_k \cdot \text{depth}_T(a_k)$$

$$= \sum_{k \neq i,j} p_k \, \text{depth}_T(a_k) + p_i \, \text{depth}_T(a_i) + p_j \, \text{depth}_T(a_j)$$

$$= \sum_{k \neq i,j} p_k \, \text{depth}_T(a_k) + (p_i + p_j)(1 + \text{depth}(u))$$

$$= \sum_{k \neq i,j} p_k \, \text{depth}_T(a_k) + (p_i + p_j)\,\text{depth}(u) + p_i + p_j$$

minimize — const.

$S = \{a_1, \ldots a_i \ldots a_j, a_n\}$ $\quad p_1 \ldots p_i \ldots p_j, \ldots, p_n$

$\longrightarrow$ $S' = (S \setminus \{a_i, a_j\}) \cup \{u\}$

$p'_k = p_k$ if $k \neq i, j$

$p'_u = p_i + p_j$

Recursive Algorithm:

1. Let a_i, a_j be two least freq. letters.
2. Construct a new instance with
   $S' = (S \setminus \{a_i, a_j\}) \cup \{u\}$, $p'_k = p_k$ for $k \neq i, j$
   $p'_u = p_i + p_j$
3. Recursively find the best tree, say T' for the new instance. (T' has u as a leaf).
4. Attach a_i, a_j as children of u, and return this tree.