- Huffman Tree

\[ a : 5 \quad b : 10 \quad c : 3 \quad d : 6 \]

- decision to make?
  - can construct a tree by merging its leaves.

\[ \begin{align*}
    &\text{merge} \\
    &0 \quad 0 \quad 0 \quad 0 \quad 0 \\
    &a \quad b \quad c \quad d \quad a \quad b
\end{align*} \]

- tree with \( n \) leaves, \( n-1 \) merging operation create a binary tree.

- binary tree is "good" if all intermediate nodes have two children.

Claim: any "good" binary tree can be constructed by merging.

\[ \begin{align*}
    &a \quad b \\
    &c \quad d
\end{align*} \quad \Rightarrow \quad \begin{align*}
    &a \quad b \\
    &c \quad d
\end{align*} \quad \text{not good}

- how to make the greedy choice?

- need a way to compute "cost" for merging operation.

Claim: Define cost of merging \( a \) \( b \) to be sum of frequencies of \( a \) \( b \).

\[ \text{sum of costs of the } n-1 \text{ merging operation} = \text{cost of the tree} \]

\[ \text{ex. } a \quad b \quad c \quad d \quad 5 \quad 10 \]

new section

new page

New Section 1 Page 1
sum of costs of the n-1 merging operations = cost of the tree

\[ \text{Cost} = \sum \text{Freq}_i \times \text{Depth}_i \]

\[ = \sum \text{edges} \times \text{total frequency of characters below this edge} \]

\[ = \sum \text{edges} \times \text{merge cost when the lower node is created} \]

\[ + \sum \text{Freq}_i \times \text{leaves} \]

final merge cost

---

greedy algorithm: minimize immediate cost

choose two characters that have lowest frequencies

---

Proof of correctness:

Use induction: IH: ALG is optimal for all alphabets with n characters.

Base case n=2 trivial

Induction step: assume IH holds for n, now if the alphabet has n+1 characters.

ALG has a solution that merges i, j first

\[ \text{two characters with smallest frequency} \]

assume towards contradiction that there is a better solution OPT

if in OPT, i and j are not merged

look at node i and j in OPT

wlog. depth_i^{OPT} \leq depth_j^{OPT}

Case 1 i has a sibling k

swapping i, k cannot increase cost because \( \text{Freq}_i \leq \text{Freq}_k \)

Case 2 let k_1, k_2 be a pair of leaves that is merged in the sibling tree of i
Case 2: let $k_1, k_2$ be a pair of leaves that is merged in the sibling tree of $j$.

$OPT_{k_1} = OPT_{k_2} > OPT_j$

Depth $k_1 = Depth k_2 > Depth j$

Swapping $(i, k_1), (j, k_2)$ will decrease cost.

- Can always transform $OPT$ into $OPT'$ where $i, j$ are merged first.

- By induction hypothesis $ALG$ is optimal after merging $i, j$

so $ALG$ is also optimal for this alphabet of size $n+1$. □