



Algorithms: Design
and Analysis, Part II

Huffman Codes

Correctness Proof


Correctness of Huffman's Algorithm

Theorem: [Huffman 52] Huffman's algorithm computes a binary tree (with leaves \leftrightarrow symbols of Σ) that minimizes the average encoding length

$$L(T) = \sum_{i \in \Sigma} p_i \cdot [\text{depth of leaf } i \text{ in } T]$$

Proof: by induction on $n = |\Sigma|$. (Can assume $n \geq 2$.)

Base case: when $n=2$, algorithm outputs the optimal tree
(need 1 bit per symbol)



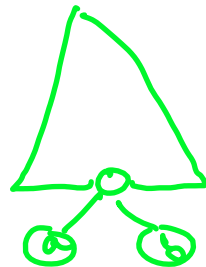
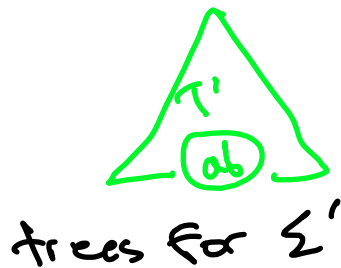
Inductive step: Fix input with $n = |\Sigma| > 2$.

By inductive hypothesis: algorithm solves smaller subproblem (for Σ') optimally.

Inductive Step

Let $\Sigma' = \Sigma$ with a, b replaced by meta-symbol ab .
 symbols with smallest frequencies Define $P_{ab} = p_a + p_b$.

Recall: exact correspondence between



trees for Σ
 that have a, b
 as siblings

X_{ab}

Important: for every such pair T' and T , $L(T) - L(T')$ is (after cancellation)

$$\begin{aligned}
 & p_a \cdot [\text{depth of } a \text{ in } T] \\
 & + p_b \cdot [\text{depth of } b \text{ in } T] \\
 & - (p_a + p_b) \cdot [\text{depth of } ab \text{ in } T']
 \end{aligned}$$

$\rightarrow = p_a + p_b$
 \rightarrow each is one more than
 \leftarrow than

$$\begin{aligned}
 & = p_a(d+1) + p_b(d+1) - (p_a + p_b) \cdot d \\
 & = \boxed{p_a + p_b} - \text{independent of } T, T'!
 \end{aligned}$$

Proof of Theorem

Inductive hypothesis: Huffman's algorithm computes a tree \hat{T}' that minimizes $L(T')$ for Σ' .

Upshot of last slide: corresponding tree \hat{T} minimizes $L(T)$ for Σ over all trees in X_{ab} . i.e., where a & b are siblings

Key lemma: [completes proof of theorem] there is an optimal tree (for Σ) in X_{ab} . [i.e., a & b were "safe" to merge]

Intuition: can make an optimal tree better by pushing a & b as deep as possible (since a & b have smallest frequencies).

Proof of Key Lemma

By exchange argument. Let T^* be any tree that minimizes $L(T)$ for Σ .

Let x, y be siblings at the deepest level of T^* .

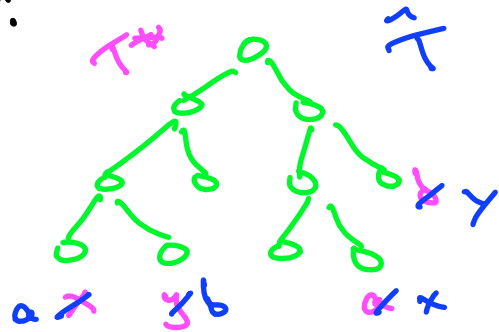
The exchange: obtain \hat{T} from T^* by:

- swapping labels $a \leftrightarrow x, b \leftrightarrow y$

Note: $\hat{T} \in X_{ab}$ (by choice of x, y).

To finish: will show that $L(\hat{T}) \leq L(T^*)$ [$\Rightarrow \hat{T}$ also optimal, completes proof]
 ≥ 0 by choice of x, y

Reason: $L(T^*) - L(\hat{T}) = (p_x - p_a) \cdot [\text{depth of } x \text{ in } T^* - \text{depth of } a \text{ in } \hat{T}] + (p_y - p_b) \cdot [\text{depth of } y \text{ in } T^* - \text{depth of } b \text{ in } \hat{T}]$
 ≥ 0 since a, b have smallest frequencies!
 ≥ 0 . QED!



Notes on Running Time

Naive implementation: $O(n^2)$ time, where $n = |\Sigma|$.

Speed ups: - use a heap! [to perform repeated minimum computations]

- use keys = frequencies
 - after extracting the two smallest-frequency symbols, re-Insert the new meta-symbol [new key = sum of the 2 old ones]
- \Rightarrow iterative, $O(n \log n)$ implementation

Even faster: (non-trivial exercise) Sorting + $O(n)$ additional work.

- manage (meta-) symbols using two queues