Huffman Codes

A Greedy Algorithm

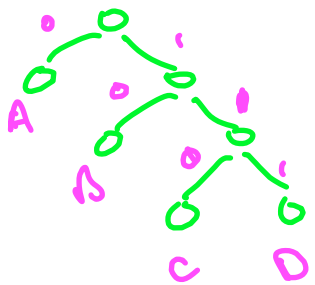
Algorithms: Design and Analysis, Part II

Problem Definition

Input: probability p_i for each character $i \in \Sigma$.

Output: binary tree (with leaves $\leftrightarrow$ symbols of Σ) minimizing the average encoding length:

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

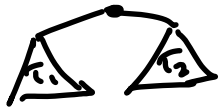


Building a Tree

Question: what's a principled approach for building a tree with leaves $\leftrightarrow$ symbols of Σ?

Natural but suboptimal idea: top-down / divide & conquer.

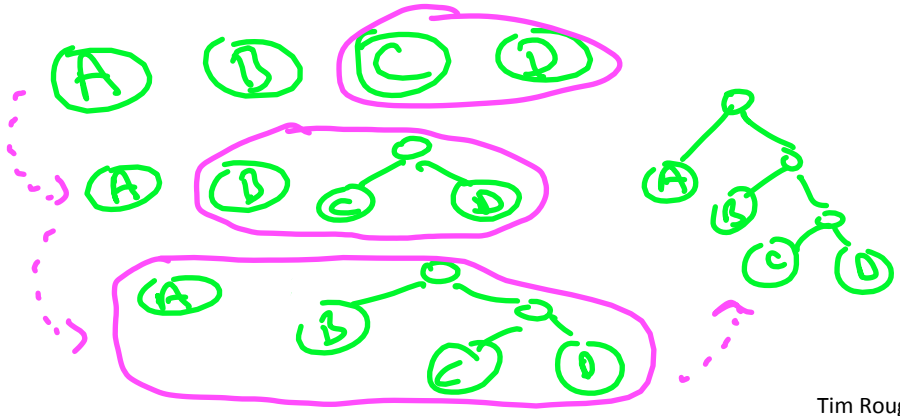
- partition Σ into Σ_1, Σ_2 each with $\approx 50\%$ of total frequency
- recursively compute T_1 for Σ_1, T_2 for Σ_2, return



Huffman's (optimal) idea:

build tree bottom-up

using successive mergers.

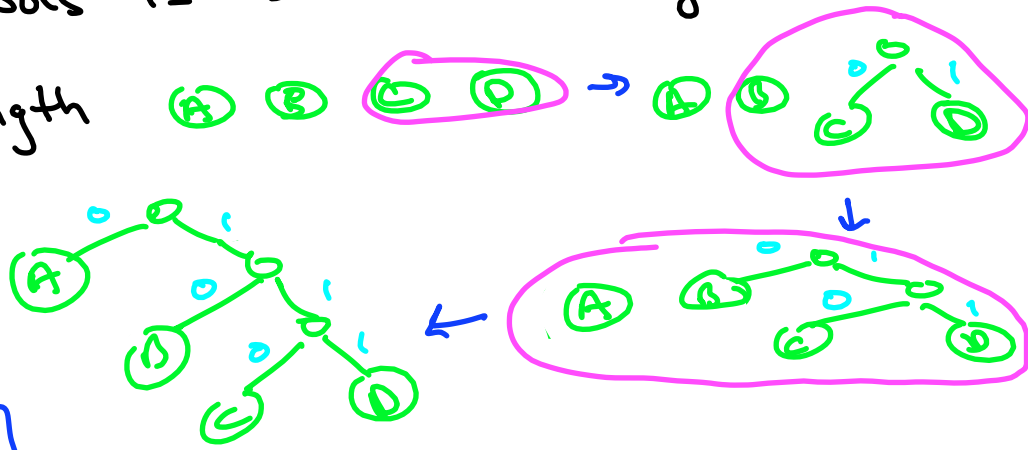


A Greedy Approach

Question: which pair of symbols is "safe" to merge?

Observation: final encoding length of $i \in \Sigma$ = # of mergers its subtree endures.

[each merger increases encoding length of participating symbols by 1]



Greedy heuristic: in first iteration, merge the two symbols with the smallest frequencies.

How to Recurse?

Suppose: 1st iteration of algorithm merges symbols a & b.

Idea: replace the symbols a, b by a new "meta-symbol" ab.

Question: what should be the frequency p_{ab} of this meta-symbol?

(A) $\max\{p_a, p_b\}$

(B) $\min\{p_a, p_b\}$

(C) $p_a + p_b$

(D) $p_a \cdot p_b$

since ab is a proxy for "a or b" (intuitively)

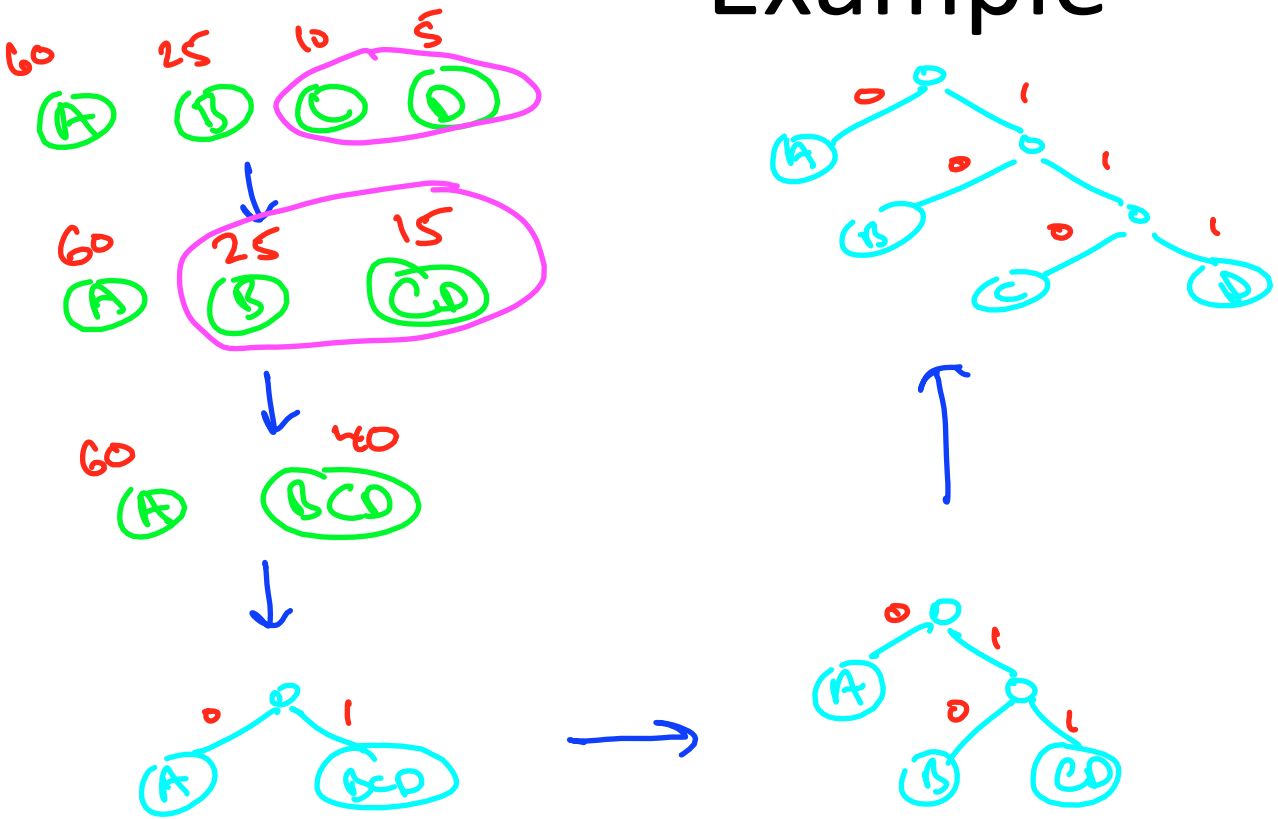
Example

$p_A = 60\%$

$p_B = 25\%$

$p_C = 10\%$

$p_D = 5\%$



Huffman's Algorithm

(given frequencies p_i as input)

If $|\Sigma| = 2$ return 

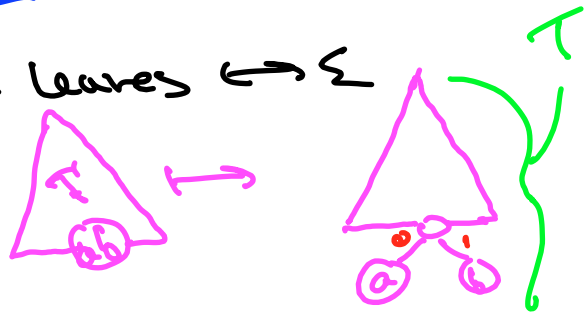
Let $a, b \in \Sigma$ have the smallest frequencies.

Let $\Sigma' = \Sigma$ with a, b replaced by new symbol ab.

Define $p_{ab} = p_a + p_b$.

Recursively compute T' (for the alphabet Σ')

Extend T' (with leaves $\leftrightarrow \Sigma'$) to a tree T with leaves $\leftrightarrow \Sigma$
by splitting leaf ab into two leaves a, b.



Return T