Algorithms: Design and Analysis, Part II

Huffman Codes

Problem Definition

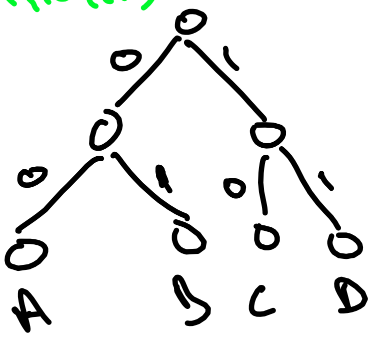
Codes as Trees

Goal: best binary prefix-free encoding for a given set of character frequencies.

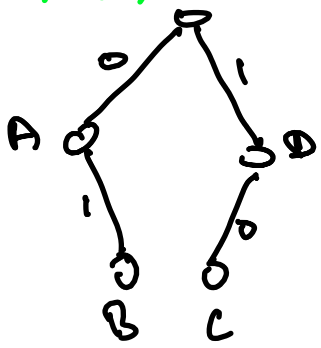
Useful fact: binary codes $\longleftrightarrow$ binary trees

Examples: ($\Sigma = \{A, B, C, D\}$)

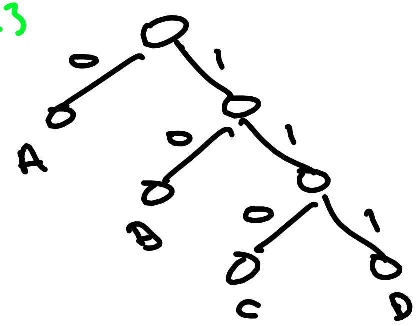
$\{00, 01, 10, 11\}$



$\{0, 01, 10, 1\}$



$\{0, 10, 110, 111\}$

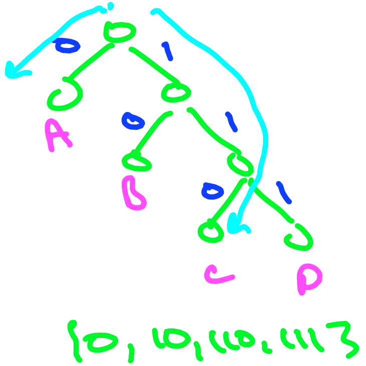


Prefix-Free Codes as Trees

In general: left child edges $\longleftrightarrow$ "0", right child edges $\longleftrightarrow$ "1"

- for each $i \in \Sigma$, exactly one node labeled "i"
- encoding of $i \in \Sigma$ $\longleftrightarrow$ bits along path from root to the node "i"
- prefix-free $\longleftrightarrow$ labelled nodes = the leaves

[since prefixes $\longleftrightarrow$ one node an ancestor of another]



To decode: repeatedly follow path from root until you hit a leaf. [ex: 0110111 $\longleftrightarrow$ ACD]
(unambiguous since only leaves are labelled)

Note: encoding length of $i \in \Sigma$ = depth of i in tree.

Tim Roughgarden

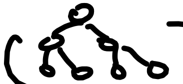
Problem Definition

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

Notation: if T = tree with leaves $\longleftrightarrow$ symbols of Σ,

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

average encoding length

$L($  $) = 2$

while

Example: if $p_A = 60\%$, $p_B = 25\%$, $p_C = 10\%$, $p_D = 5\%$, then $L($  $) = 1.55$

Output: a binary tree T minimizing the average encoding length $L(\cdot)$.

Tim Roughgarden