COL351: Analysis and Design of Algorithms

Ragesh Jaiswal, CSE, IITD

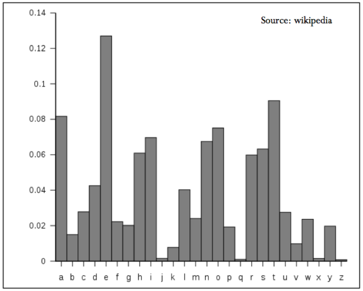
Greedy Algorithms

- A wants to send an email to B but wants to minimize the amount of communication (number of bits communicated).
- How do you encode an email into bits?
  - ASCII: 8 bits per character
  - Is this the best way to encode the email given that the goal is to minimize the amount of communication?

- A wants to send an email to B but wants to minimize the amount of communication (number of bits communicated).
- How do you encode an email into bits?
    - ASCII: 8 bits per character
    - Is this the best way to encode the email given that the goal is to minimize the amount of communication?
    - Different alphabets have different frequency of occurrence in a standard English document.

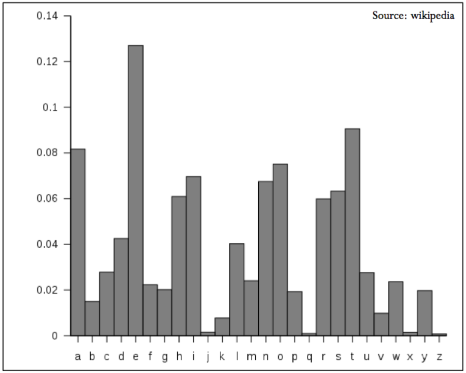
Greedy Algorithms
Huffman coding

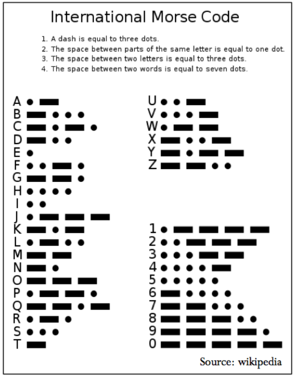
- A wants to send an email to B but wants to minimize the amount of communication (number of bits communicated).
- How do you encode an email into bits?
  - ASCII: 8 bits per character
  - Is this the best way to encode the email given that the goal is to minimize the amount of communication?
  - Different alphabets have different frequency of occurrence in a standard English document.



Greedy Algorithms
Huffman coding

- The encoding of “e” should be shorter than the encoding of “x”.
- In fact, Morse code was designed with this in mind.




International Morse Code

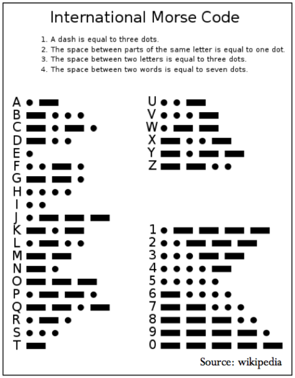
1. A dash is equal to three dots.
2. The space between parts of the same letter is equal to one dot.
3. The space between two letters is equal to three dots.
4. The space between two words is equal to seven dots.

A	• ▬	U	• • ▬
B	▬ • • •	V	• • • ▬
C	▬ • ▬ •	W	• ▬ ▬
D	▬ • •	X	▬ • • ▬
E	•	Y	▬ • ▬ ▬
F	• • ▬ •	Z	▬ ▬ • •
G	▬ ▬ •		
H	• • • •		
I	• •		
J	• ▬ ▬ ▬		
K	▬ • ▬	1	• ▬ ▬ ▬ ▬
L	• ▬ • •	2	• • ▬ ▬ ▬
M	▬ ▬	3	• • • ▬ ▬
N	▬ •	4	• • • • ▬
O	▬ ▬ ▬	5	• • • • •
P	• ▬ ▬ •	6	▬ • • • •
Q	▬ ▬ • ▬	7	▬ ▬ • • •
R	• ▬ •	8	▬ ▬ ▬ • •
S	• • •	9	▬ ▬ ▬ ▬ •
T	▬	0	▬ ▬ ▬ ▬ ▬

Source: wikipedia

Greedy Algorithms
Huffman coding

- Suppose you receive the following Morse code from your friend:
  ●●●—
- What is the message?



International Morse Code

1. A dash is equal to three dots.
2. The space between parts of the same letter is equal to one dot.
3. The space between two letters is equal to three dots.
4. The space between two words is equal to seven dots.

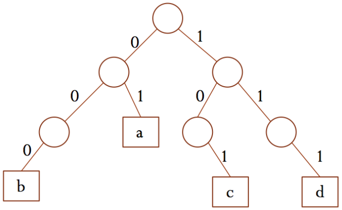
A	● ▬	U	● ● ▬
B	▬ ● ● ●	V	● ● ● ▬
C	▬ ● ▬ ●	W	● ▬ ▬
D	▬ ● ●	X	▬ ● ● ▬
E	●	Y	▬ ● ▬ ▬
F	● ● ▬ ●	Z	▬ ▬ ● ●
G	▬ ▬ ●		
H	● ● ● ●		
I	● ●		
J	● ▬ ▬ ▬		
K	▬ ● ▬	1	● ▬ ▬ ▬ ▬
L	● ▬ ● ●	2	● ● ▬ ▬ ▬
M	▬ ▬	3	● ● ● ▬ ▬
N	▬ ●	4	● ● ● ● ▬
O	▬ ▬ ▬	5	● ● ● ● ●
P	● ▬ ▬ ●	6	▬ ● ● ● ●
Q	▬ ▬ ● ▬	7	▬ ▬ ● ● ●
R	● ▬ ●	8	▬ ▬ ▬ ● ●
S	● ● ●	9	▬ ▬ ▬ ▬ ●
T	▬	0	▬ ▬ ▬ ▬ ▬

Source: wikipedia

- Prefix-free encoding: An encoding f is called prefix-free if for any pair of alphabets (a_1, a_2), $f(a_1)$ is not a prefix of $f(a_2)$.
- Morse code is certainly not prefix-free.
- Consider a binary tree with 26 leaves and associate each alphabet with a leaf in this tree.
    - Binary tree: A rooted tree where each non-leaf node has at most two children.
- Label an edge 0 if this edge connects the parent to its left child and 1 otherwise.
- $f(x) =$ The label of edges connecting the root with x.

Greedy Algorithms
Huffman coding

- Consider a binary tree with 26 leaves and associate each alphabet with a leaf in this tree.
  - Binary tree: A rooted tree where each non-leaf node has at most two children.
- Label an edge 0 if this edge connects the parent to its left child and 1 otherwise.
- $f(x) =$ The label of edges connecting the root with x.

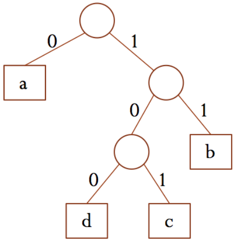


Simple example with 4 alphabets

- $f(a) = 01, f(b) = 000, f(c) = 101, f(d) = 111$.
- Is f prefix-free?

- Suppose you are given a prefix-free encoding g.
- Can you construct a binary tree with 26 leaves, associate each leaf with an alphabet, and label the edges as defined previously such that for any alphabet, the label of edges connecting the root with $x = g(x)$?
- For example: $g(a) = 0, g(b) = 11, g(c) = 101, g(d) = 100$.

- Suppose you are given a prefix-free encoding g.
- Can you construct a binary tree with 26 leaves, associate each leaf with an alphabet, and label the edges as defined previously such that for any alphabet, the label of edges connecting the root with $x = g(x)$?
- For example: $g(a) = 0, g(b) = 11, g(c) = 101, g(d) = 100$.



Simple example with 4 alphabets

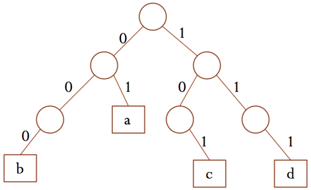
Greedy Algorithms
Huffman coding

Problem

Huffman Coding: Given alphabets $\Sigma = (a_1, ..., a_n)$ and the frequency of occurrence of alphabets $(t(a_1), ..., t(a_n))$, find a prefix-free encoding f that minimizes:

$$O_f = |f(a_1)| \cdot t(a_1) + |f(a_2)| \cdot t(a_2) + ... + |f(a_n)| \cdot t(a_n)$$

- Consider $\Sigma = (a, b, c, d)$, $t(a) = 0.6$, $t(b) = 0.2$, $t(c) = 0.1$, $t(d) = 0.1$ and consider the prefix-free encoding given by the binary tree below:
- What is the value of O_f for the prefix-free code given by the binary tree below?



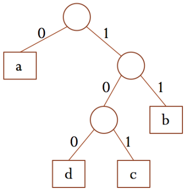
Simple example with 4 alphabets

Problem

Huffman Coding: Given alphabets $\Sigma = (a_1, ..., a_n)$ and the frequency of occurrence of alphabets $(t(a_1), ..., t(a_n))$, find a prefix-free encoding f that minimizes:

$$O_f = |f(a_1)| \cdot t(a_1) + |f(a_2)| \cdot t(a_2) + ... + |f(a_n)| \cdot t(a_n)$$

- Consider $\Sigma = (a, b, c, d)$, $t(a) = 0.6$, $t(b) = 0.2$, $t(c) = 0.1$, $t(d) = 0.1$ and consider the prefix-free encoding given by the binary tree below:
- What is the value of O_f for the prefix-free code given by the binary tree below?



Simple example with 4 alphabets

- Node depth: The *depth* of a vertex v, denoted by $d(v)$, is the length of the path from root to v.
- Every binary tree gives a prefix-free encoding and every prefix-free encoding gives a binary tree.
- We will now use these properties to rephrase the previous problem in terms of binary trees and depths of leaves.

Greedy Algorithms
Huffman coding

Problem

Huffman Coding: Given alphabets $\Sigma = (a_1, ..., a_n)$ and the frequency of occurrence of alphabets $(t(a_1), ..., t(a_n))$, find a ~~prefix-free encoding~~ ~~f~~ binary tree T with n leaves (each leaf labeled with unique alphabet) that minimizes:

$$O_f = |d(a_1)| \cdot t(a_1) + |d(a_2)| \cdot t(a_2) + ... + |d(a_n)| \cdot t(a_n),$$

where $d(a_i)$ denotes the depth of the leaf labeled a_i.

- What are the properties of the optimal tree T^*?
    - Claim 1: T^* is a complete binary tree.
        - Complete binary tree: Every non-leaf node has exactly two children.

Greedy Algorithms
Huffman coding

Problem

Huffman Coding: Given alphabets $\Sigma = (a_1, ..., a_n)$ and the frequency of occurrence of alphabets $(t(a_1), ..., t(a_n))$, find a ~~prefix-free encoding~~ ~~f~~ binary tree T with n leaves (each leaf labeled with unique alphabet) that minimizes:

$$O_f = |d(a_1)| \cdot t(a_1) + |d(a_2)| \cdot t(a_2) + ... + |d(a_n)| \cdot t(a_n),$$

where $d(a_i)$ denotes the depth of the leaf labeled a_i.

- What are the properties of optimal tree T^*?
  - Claim 1: Any T^* is a complete binary tree.
  - Claim 2: Consider two alphabets x and y with least frequencies. Then x and y have maximum depth in any optimal tree T^*. Moreover, there is an optimal tree T^* where x and y are siblings.

- Let Ω be a new symbol not present in Σ.
- Consider the following (*smaller*) problem:
  - $\Sigma' = \Sigma - \{x, y\} \cup \{\Omega\}$
  - For all $z \in \Sigma$, $t'(z) = t(z)$
  - $t(\Omega) = t(x) + t(y)$
  - Find an optimal binary tree for the new alphabet set Σ' and new frequencies t'.
- Let T' be any optimal binary tree for the above problem.
- Consider the leaf v labeled with Ω in T'.
- Consider the tree T which is the same as T' except that the node v has two children labeled as x and y.
- Claim 3: T is an optimal binary tree for the original problem.

Algorithm

```
Huffman-Tree
    - Let $v_1, ..., v_n$ be the nodes each denoting an alphabet
    - $S \leftarrow \{v_1, ..., v_n\}$
    - While $(|S| > 1)$:
        - Pick two nodes $x, y$ with least values of $t(x)$ and $t(y)$
        - Create a new node $z$ and set $t(z) \leftarrow t(x) + t(y)$
        - Set $x$ as the left child of $z$ and $y$ as the right child of $z$
        - $S \leftarrow S - \{x, y\} \cup \{z\}$
    - Return the only node in $S$ as the root node of the Binary Tree
```

- What is the running time of the above algorithm?

- An example:
  - A DNA sequence has four characters A, C, T, G and these characters appear with frequency 30%, 20%, 10%, and 40% respectively.
  - We have to encode a sequence of length 1 million in bits.
  - If we use two bits for each character, the encoding will use 2 million bits.
  - How many bits will be required if we do Huffman encoding?

End