First mini-project:
Binary prefix codes and Huffman's codes

João Luís Sobrinho

I. Introduction

A *binary code* is a representation of set of symbols that assigns a distinct bit string—a code word—to every symbol in the set. A *binary prefix code* is a binary code where no bit string of the code is a prefix of another. With a binary prefix code, we can unambiguously decode any string of bits without marks separating the code words. A binary prefix code can be constructed from any two-tree with the symbols positioned at the leaves of the tree. Associate a 0 to the left child and a 1 to the right child of every node with children. The code word of a symbol is the sequence of bits read along the unique path from the root to the leaf holding the symbol. Figure 1 shows a prefix code and the two-tree from which is was generated.

Suppose that we are given a set of n symbols together with their relative frequencies of occurrence. Let f_i and s_i denote the relative frequency and code word length, respectively, of symbol a_i, for $1 \leq i \leq n$. The symbols are encoded with a binary prefix code. The average length of the code is given by

$$\sum_{i=1}^{n} f_i s_i. \tag{1}$$

An optimal binary prefix code is a prefix code of minimum average length. David Huffman, in 1951, invented a method to derive such codes. In his honor, optimal binary prefix codes bear his name. Today, Huffman's codes are

Code

symbol	code word
a	00
b	010
c	011
d	10
e	110
f	111

Two-tree representation

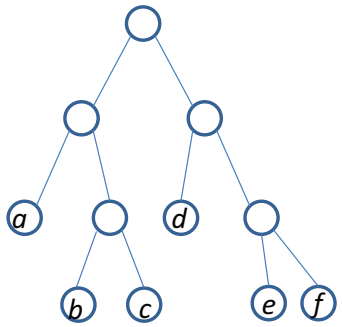

Fig. 1. A prefix code for the set of symbols $\{a, b, c, d, e\}$.

Code

symbol	freq.	code word
a	0,22	011
b	0,03	0000
c	0,14	001
d	0,14	010
e	0,41	1
f	0,06	0001

Two-tree representation

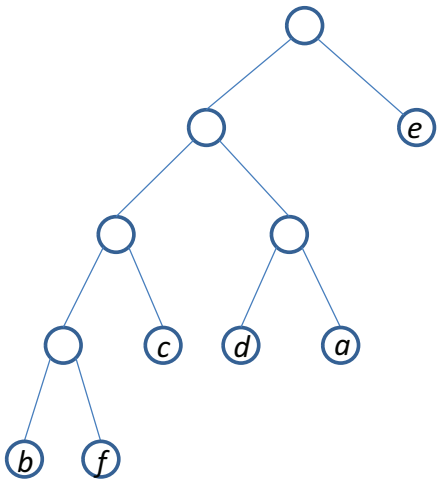

Fig. 2. A Huffman's code for the set of symbols $\{a, b, c, d, e\}$.

widely used to compress data; they are used, for example, in compressing MP3 files. Here is a brief description of Huffman's method in prose. The method builds a two-tree with the symbols positioned at the leaves. It starts with a forest, where each symbol and its associated frequency is the root of a trivial tree with just one node. At each iteration, the algorithm finds the two trees whose roots have the smallest frequency over all trees in the current forest. A new node is created having the previous two roots as children, so that a new tree is formed from the created node and its two children. The frequency of the new node is the sum of the frequencies of its two children. The algorithm terminates after $n - 1$ iterations. Figure 2 shows a Huffman's code and the final two-tree of Huffman's method for the same set of symbols as in Figure 1. The average length of the Huffman's code is $2,27$ whereas, for the same relative frequencies, the average length of the prefix code of Figure 1 is $2,64$.

II. Your assignment

What you have to do.

- Implement the procedure *GenerateCode(Root, Code)* that receives as input a two-tree *Root* with the symbols at the leaves and produces as output the corresponding binary prefix code *Code*. Your code should be efficient.
- Implement the procedure *Decode(Root, InString, OutString)* that receives as input a two-tree *Root* representing a binary prefix code and a bit string *InString* and produces as output the decoded string of symbols *OutString*.
- Write program *PrefixCode* that the reads a sequence of symbols from a file (symbols are separated by spaces or returns) and prints to screen any binary prefix code of your choice for the set of symbols. Then, the program reads binary strings from the keyboard and prints the decoded string of symbols to the screen.
- Implement procedure *HuffmanCode(Symbols, Freq, Code)* that receives as input an array *Symbols* of symbols and an array *Freq* with their relative frequencies, and produces as output a Huffman's code *Code* for the set of symbols. Your code should be efficient.

- Write a program *Huffman* that reads a sequence of symbols and their relative frequencies from a file (a symbol is separated from its frequency by a space or returns; a frequency is separated from the next symbol by spaces or returns) and prints a Huffman's code for the set of symbols.

What do you have to deliver, how, and when.

- You have to deliver your code and a report of no more than three pages containing a text explanation of your algorithms, their pseudo-codes, and a short discussion.
- The code and the report should be sent in a .zip file to my email address with subject p1.<group number>.zip where <group number> is your group number. You will also have to deliver a printed version of the report.
- The deadline for the .zip file is October 14, 2016, 23:59. The printed report should be delivered in the class on October 17.

How I will evaluate your assignment.

- The first thing I will evaluate is the clarity of your report and of your pseudo-code. Your writing is your way of communicating your ideas and results to others so that your work can be well understood and publicized. Remember that a pseudo-code is a high-level description of an algorithm. Choose descriptive names for variables and procedures. You can include comments in your pseudo-code.
- The second thing I will evaluate is the correctness of your code. Be sure to test your code before delivery.
- The third thing I will evaluate is the efficiency of your algorithms, but mostly in asymptotic terms.
- I will have a discussion with you about your report and will test your code at the end of the semestre, jointly with the other assignments, but I may give you some feedback before the second assignment.