Question #1 : Proof of Correctness of Huffman Encoding Problem

Allen Kim, Yeshuchan (Jack), Gautam Ramasubramanian

October 19, 2016

Problem Statement

Justify the correctness of the Huffman encoding algorithm: In Huffman Encoding Algorithm (sort the symbols in order and merge the top two symbols iteratively until only one node is left), prove that for two symbols A and B with probabilities,

$$P(A) \geq P(B)$$

then in the result representation sequence according to the Huffman Encoding Procedure, the length of symbol A is no longer than that of symbol B. In other words, if L is a function the outputs the length of a symbol, then

$$L(A) \leq L(B)$$

Huffman Encoding Intro

Suppose we want to encode the following set of symbols - They could be ASCII characters, Unicode characters, etc. We will denote them as

$$V = v_1, v_2, v_3 \dots v_n$$

Without loss of generality, we will assume that the probabilities of v_i where i ranges from 1 to n are in non-decreasing order as shown below.

$$P(v_1) \leq P(v_2) \leq P(v_3) \dots \leq P(v_n)$$

In other words, v_n is the most frequent character in the file we want to encode, and v_1 is the least frequent character in the file.

The Huffman encoding algorithm works as follows. We arrange the symbols in a min priority queue based on probability.

We pop the first two symbols out and merge them into one symbol. In this case we pop v_1 and v_2 and merge them to create $v_{1,2}$. The frequency of this merged symbol is the sum of the frequencies of the individual symbols.

$$P(v_{1,2}) = P(v_1) + P(v_2)$$

We input this merged symbol in the priority queue again.

Then, we iterate the procedure over and over again, removing the first two symbols, merging them, and adding them back to the queue. We continue until there is only one symbol remaining in the queue, which is the merger of all the individual symbols.

The final symbol represents a binary tree, where the depth of a symbol represents the code length of that symbol.

1 Proof of Huffman Encoding Correctness

Proof. Given two symbols A and B with probabilities,

$$P(A) \geq P(B)$$

we want to prove that with the Huffman algorithm,

$$L(A) \leq L(B)$$

where L is a function the outputs the length of a symbol.

Let A and B be two symbols in V, where $V = v_1, v_2, v_3 \dots v_n$ is the list of symbols.
Given the optimality of prefix codes in the Huffman encoding algorithm, we know that the expected code word length, $\sum_{i=1}^{n} P(v_i)L(v_i)$, is minimal over all expected code word lengths.

Consider swapping the code word for A with B. Then, it follows that

$$\sum_{i=1}^{n} P(v_i)L(v_i) \leq \sum_{i=1}^{n} P(v_i)L'(v_i)$$

where $L = L'$ except that $L'(A) = B$ and $L'(B) = A$.
Moving all to one side,

$$0 \leq \sum_{i=1}^{n} P(v_i)L'(v_i) - \sum_{i=1}^{n} P(v_i)L(v_i)$$

All terms cancel except for the swapped terms for A and B resulting in

$$\begin{aligned}
0 &\leq [P(A)L(B) + P(B)L(A)] - [P(A)L(A) + P(B)L(B)] \\
0 &\leq P(A)L(B) + P(B)L(A) - P(A)L(A) + P(B)L(B) \\
0 &\leq \Big(P(A) - P(B)\Big)\Big(L(B) - L(A)\Big)
\end{aligned}$$

We are given that $P(A) \geq P(B) \implies P(A) - P(B) \geq 0$. Thus, this means that $L(B) - L(A) \geq 0$ to maintain the derived inequality above. It follows that $L(A) \leq L(B)$.

□