Question #1 : Proof of Correctness of Huffman Encoding Problem

Allen Kim, Shuchan (Jack) Ye, Gautam Ramasubramanian

October 20, 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,

$$P(A) \leq P(B)$$

and assuming that if the probabilites are equal, B is in front of A in the sorted order, 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.

Proof

Proof. The proof of the Huffman Encoding uses induction.

We will first consider the base case. Suppose we have only 2 symbols, arranged in a priority queue as such

$$Q = [B, A]$$

The left-most element is the front of the queue, and the right-most element is in the back. In this case, B and A get merged together to get AB, which has a Huffman Length of 0. Therefore,

$$L(A) = L(B) = L(AB) + 1$$

This means that $L(B) \geq L(A)$, which is the inequality we seek. Similarly for 3 symbols in the priority queue, we have

$$Q = [C, B, A]$$

or

$$Q = [B, C, A]$$

which results in

$$L(B) = L(C) = L(A) + 1$$

Therefore, $L(B) \geq L(A)$.

Now, we go to the inductive case. Suppose A and B were part of a set of n symbols. Suppose that the theorem is proven for $n-1$ symbols. In the start of the Huffman Encoding procedure, the symbols with the two lowest probabilities are merged together. Therefore, there are three cases.

1. B and A were the symbols of lowest probabilities were merged together. In that case, $L(B) = L(A)$, which means that $L(B) \geq L(A)$.

2. Neither B nor A were the symbols that were merged. Let us call the symbols that did get merged C and D. C and D merged to get combined symbol CD. Then, the problem reduced to having $n-1$ symbols such that $P(B) < P(A)$. Therefore, we know that $L(B) \geq L(A)$.

3. B and another symbol were combined - let us call that other symbol C. B and C are merged to get BC, which is put back in the queue in increasing probability order, such that anything below has probability less than or equal to that of BC and anything above has probability strictly greater than that of BC. There are two subcases.

   (a) BC is placed in front of A in the queue, which implies that $P(BC) < P(A)$. Now, we have two symbols in a set of $n-1$ symbols such that $P(BC) \leq P(A)$. This means we reduced the problem, and our inductive assumption tells us that $L(BC) \geq L(A)$. Since $L(BC) = L(B) - 1$, then $L(B) \geq L(A)$.

   (b) BC is placed behind A. This means that $P(BC) \geq P(A)$. In this case, we keep proceeding with the Huffman Encoding for k more merges until we come to the point when A is about to be merged. There are two cases here.

      i. A merges with BC. This means that $L(A) = L(BC) = L(B) - 1$, so $L(B) \geq L(A)$.

      ii. A merges with some other symbol D. Since, probabilities of A and D are both greater than or equal to those of B and C, the combined symbol AD is put behind BC because $P(AD) \geq P(BC)$. This means that $P(A) - 1 \geq P(B) - 1$, which means that $P(A) \geq P(B)$. □