Parity Codes

Paulo E. D. Pinto*
Fábio Protti†
Jayme L. Szwarcfiter‡

Abstract

Motivated by a problem posed by Hamming in 1980, we define even codes. They are Huffman type prefix codes with the additional property of being able to detect the occurrence of an odd number of 1 bit errors in the message. We characterize optimal even codes and describe a simple method for constructing the optimal codes. Further, we compare optimal even codes with Huffman codes for equal frequencies. We show that the maximum encoding in an optimal even code is at most two bits larger than the maximum encoding in a Huffman tree. Moreover, it is always possible to choose an optimal even code such that this difference drops to 1 bit. Finally, we compare average sizes. We show that the average size of an encoding in a optimal even tree is at least 1/3 and at most 1/2 of a bit larger than that of a Huffman tree. These values represent the overhead in the encoding sizes for having the ability to detect an odd number of errors in the message.

1 Introduction

Huffman codes [4] form one of the most traditional methods of coding. One of the important aspects of these codes is the possibility of handling encodings of variable sizes. A great number of extensions an variations of the classical Huffman codes have been described throught the time. For instance, Faller [1], Gallager [2], Knuth [6] and Milidiú, Laber and Pessoa [9] adressed adaptative methods for the construction of Huffman trees. Huffman trees with minimum height were described by Schwartz [13].The consctruction of Huffman type trees with length constraints was considered by Turpin and Moffat [12], Larmore and Hirschberg [8] and Milidiú and Laber [10, 11]. On the other hand, Hamming formulated

*Universidade Estadual do Rio de Janeiro, Instituto de Matemática e Estatistica, RJ, Brasil. E-mail: pauloedp@ime.uerj.br

†Universidade Federal do Rio de Janeiro, Instituto de Matemática and NCE, Caixa Postal 2324, 20001-970, Rio de Janeiro, RJ, Brasil. Partially supported by the Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq, and Fundação de Amparo à Pesquisa do Estado do Rio de Janeiro - FAPERJ, Brasil. E-mail: fabiop@nce.ufrj.br

‡Universidade Federal do Rio de Janeiro, Instituto de Matemática, NCE and COPPE, Caixa Postal 2324, 20001-970, Rio de Janeiro, RJ, Brasil. Partially supported by the Conselho Nacional de Desenvolvimento Científico e Tecnológico - CNPq, and Fundação de Amparo à Pesquisa do Estado do Rio de Janeiro - FAPERJ, Brasil. E-mail: jayme@nce.ufrj.br

methods for the construction of error detecting codes [3]. Further, Hamming [3] posed the problem of describing a method that would combine advantages of Huffman codes with the noise protection of Hamming codes. The idea is to define a prefix code in which the encoding would contain redundancies that would allow the detection of certain kinds of errors. This is equivalent to forbid some encodings which, when present in the reception would signal an error. Such a code is a Hamming-Huffman code and its representing binary tree, a Hamming-Huffman tree. In a Huffman tree, all leaves correspond to encodings. In a Hamming-Huffman tree, there are encoding leaves and error leaves. Hitting an error leaf in the decoding process indicates the existence of an error. The problem posed by Hamming is to detect the occurrence of an error of one bit, as ilustrated in the following example given by Hamming [3], p.76. Table 1 shows the symbols and their corresponding encoding. Figure 1 depicts the corresponding Hamming-Huffman tree. Error leaves are represented by black nodes. An error of one bit in the above encodings would lead to an error leaf.

Symbol	Encoding
a	000
b	0110
c	1010
d	1100
e	1111

Table 1: Example of a Hamming-Huffman Code.

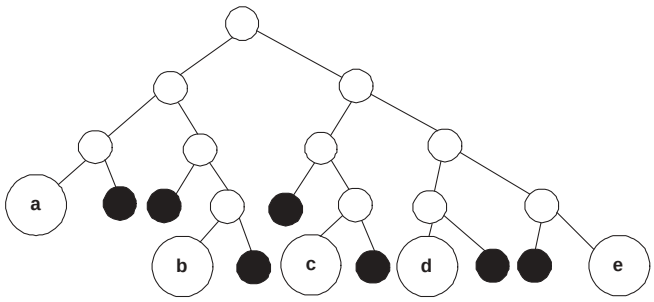

Figure 1: A Hamming-Huffman tree

Hamming-Huffman codes have not received further developments, in our knowledge, and remained open. Motivated by the above problem, we propose a code, called *even code* that is able to detect the occurrence of an odd number of 1 bit errors, in the message. The detection may occur immediately, at the wrong symbol or it may be delayed until the end of the message.

The plan of the paper is as follows. Section 2 describes even codes. In the remainning of the paper, we study optimal even codes. Section 3 presents a method for finding optimal even codes, while a characterization of these codes is described in Section 4. Section 5 is dedicated to comparing classical Huffman codes for equal frequencies with optimal even codes. We show that the maximum size of an encoding in an optimal even code is no more than two bits larger than that of the corresponding encoding in a Huffman code. Moreover, it is always possible to choose an optimal even code in which this difference reduces to one bit. Finally, we compare average sizes. It is proved that the difference between the average encoding size of an optimal even code and the corresponding average

encoding size in a Huffman code lies between 1/3 and 1/2 of a bit. These values represent the overhead in terms of encoding sizes of providing some error detection capability to a Huffman code.

The following definitions are of interest.

Let $\mathscr{S}$ be a set of elements, called *symbols*. An *encoding* e for a symbol $s \in \mathscr{S}$ is a finite sequence of 0's and 1's, associated to s. Each 0 and 1 is a *bit* of e. The bits of the encoding e are labelled $1, 2, \ldots, l$, and l is the *size* of e. A *segment* $e(i, j)$ is the subsequence of e, starting at i and ending at j. A segment $e(1, j)$ is a *prefix* of e. The *parity* of e is the parity of the quantity of 1's contained in e. The set of encodings for all symbols of $\mathscr{S}$ is a *code* $\mathscr{C}$ for $\mathscr{S}$. Two codes $\mathscr{C}$, $\mathscr{C}'$ for $\mathscr{S}$ are *equivalent* when there exists a bijection $\mathscr{C} <=> \mathscr{C}'$, such that corresponding encodings have same size. The *cost* of a code is the sum of the sizes of its encodings. A code is *optimal* when its cost is minimum. A code in wich every encoding does not coincide with a prefix of any other encoding is a *prefix code*.

A *message* M is a sequence of symbols. The *encoded message of* M is the corresponding sequence of encodings. The *parity* of an encoded message is the number of 1's contained in the encodings.

A *binary tree* is a rooted tree T in which every node z, other than the root, is labelled *left* or *right* in such a way that any two siblings have different labels. A *path* of T is a sequence of nodes $z_1, \ldots, z_t$, such that z_q is the parent of z_{q+1}. The value $t - 1$ is the *size* of the path, whereas all z_i are *descendants* of z_1. If z_1 is the root then $z_1, \ldots, z_t$ is a *root path* and, in addition, if z_t is a leaf, then $z_1, \ldots, z_t$ is a *root-leaf path* of T. The *depth* of a node is the size of the root path to it, while the *depth* of the tree is the largest depth among the nodes. The *level* k of T is formed by the nodes whose root paths have size k. For a node z of T, $T(z)$ denotes the *subtree of* T *rooted at* z, that is, the binary tree whose root is z and containing all descendants of z in T. The *left subtree* of z is the subtree $T(z')$, where z' is the left child of z. Similarly, define the *right subtree* of z. The left and right subtrees of the root of T are denoted by T_L and T_R, respectively. A *strictly binary tree* is one in which every node is a leaf or has two children. The edges of T leading to left children are labelled 0, whereas those leading to right children are labelled 1. The *parity* of a node z is the parity of the quantity of 1's among the edges forming the root path to z. A node is *even* or *odd*, according to its parity, respectively. The *path length* $P_T(n)$ of a tree T with n leaves is the sum of the sizes of all root-leaf paths of T. When convenient we may write simply $P(n)$.

A *(binary tree) representation* of a code $\mathscr{C}$ is a binary tree T such that exists a one-to-one correspondence between encodings $e \in \mathscr{C}$ and root-leaf paths p of T in such a way that e is precisely the sequence of labels, 0 or 1, of the edges forming p. A code admits a binary tree representation if and only if it is a prefix code. A *full representation tree* of $\mathscr{C}$ is a binary tree T^* obtained from the representation tree T of $\mathscr{C}$, by adding a new leaf as the second child of every node having exactly one child. The original leaves of T are the *encoding leaves*, whereas the newly introduced leaves, are the *error leaves*. Clearly, in case of Huffman trees, there are no error leaves.

2 Even Codes

In this section, we describe even codes and show how they can detect an odd number of errors in a message.

A *parity code* is a prefix code in which all encodings of the same size have identical parities. An *even code* is a code in which all encodings are even. Similarly, an *even tree* is a tree representation of an even code. Figure 2 ilustrates examples of even trees.

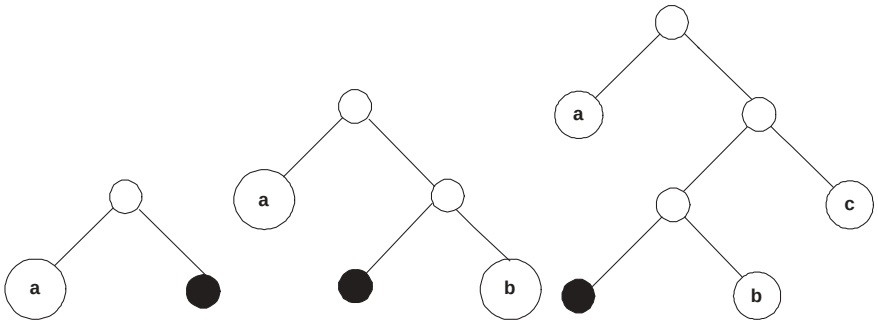

Figure 2: Examples of even trees

The next proposition relates parity codes and even codes.

Theorem 1 *Let $\mathscr{C}$ be a parity code for a set of symbols $\mathscr{S}$. There exists an even code $\mathscr{C}'$ for $\mathscr{S}$, such that $\mathscr{C}$,$\mathscr{C}'$ are equivalent.*

Proof: Let $\mathscr{C}$ be a parity code for $\mathscr{S}$. Construct $\mathscr{C}'$ from $\mathscr{C}$, as follows. Order the encodings of $\mathscr{C}$ according to their sizes and perform the following procedure: Let $d_1, d_2 \ldots, d_k$ be the distinct encoding sizes appearing in $\mathscr{C}$. For $i = 1, \ldots, k$, take the encodings with size d_i and, in case that those encodings have odd parity, change the ith bit in all encodings with size equal to or greater than d_i. Let $\mathscr{C}'$ be the code obtained after iteration k of the procedure. It is clear that all encodings of $\mathscr{C}'$ are even. In addition, $\mathscr{C}$ being itself a prefix code implies that $\mathscr{C}'$ also must be so. In order to verify the latter assertion, let e', e'' be two encodings of $\mathscr{C}$, $|e'| \leq |e''|$. Then whenever bit j of e' changes in the above procedure, bit j of e'' also changes. Consequently, $e' \neq e''(1, |e'|)$, implying that the corresponding encodings of $\mathscr{C}'$ are also distinct. That is, $\mathscr{C}'$ is indeed a prefix code, and consequently an even code. □

The above proposition implies that for our purpose, we can restrict attention to even codes, while handling the larger class of parity codes.

It is simple to conclude that even codes detect the occurrence of an odd number of 1-bit errors in a message as follows. We know that all encodings are even, so the encoded message is also even. By introducing an odd number of errors, the encoded message becomes odd. Since the encodings are even, the latter implies that in the full tree representation of the code, an error leaf would be hit during the decodification process, or otherwise the process terminates at some odd node of the tree. It should be noted that odd codes do not have this property. For example, if we have a code $\mathscr{C} = \{1, 01\}$ and a message 01,

if the first bit is changed, resulting 11, that wrong message would be decodified without signaling error.

The tree representation of an even code is an *even tree*. Finally, a code in which all encoding are odd is an *odd code* and its tree representation an *odd tree*. For a code $\mathscr{C}$, the symmetric of $\mathscr{C}$ is the code obtained from $\mathscr{C}$ by changing the first bit in every encoding. Clearly, a code is even if and only if its symmetric is odd.

3 Optimal even codes

In this section, we describe a method for finding optimal even codes. We know that the cost of an optimal even code $\mathscr{C}$ for n symbols is the minimal path length $P(n)$ among the tree representations of the even codes. The following theorem determines $P(n)$.

Theorem 2

$$P(n) = \begin{cases} 1, & \textit{if } n = 1 \\ 3, & \textit{if } n = 2 \\ n + \min\{P(n-1), \min_{1<i<n-1}\{P(i) + P(n-i)\}\}, & \textit{if } n > 2 \end{cases}$$

Proof: The results for $n = 1$ and 2 can be verified in Figure 2. If $n > 2$, the external minimization in the expression is taken over two cases:

a) the left subtree has only one encoding. Then in this case we have $P(n) = 1 + (n - 1) + P(n-1) = n + P(n-1)$, since at the left side there is only one encoding, and at the right side there is also an optimal odd tree for $n-1$ symbols. Observe now that all the encodings are one level deeper relatively to the root.

b) the left subtree has more than one encoding. Let i be the number of encodings of the left subtree, which is an optimal even tree, $1 < i < n-1$. The right subtree is an optimal odd tree with $n - i$ encodings. All the encodings are one level deeper relatively to the root. The least possible value for $P(n)$ is then given by the internal minimization: $\min_{1<i<n-1}\{i + P(i) + (n-i) + P(n-i)\} = n + \min_{1<i<n-1}\{P(i) + P(n-i)\}$. □

Next, we define a useful family of codes, through its tree representation.

An *equilibrated even(odd) tree* T for n symbols is an even(odd) tree T such that:

- if $n \leq 3$, the equilibrated even trees are those of Figure 2, while the odd ones are the symmetric of these even trees.
- if $n > 3$, the left subtree of T is an equilibrated even tree for $\lfloor n/2 \rfloor$ symbols, while the right subtree is an equilibrated odd tree for $\lceil n/2 \rceil$ symbols.

An example of an equilibrated even tree for 11 symbols is given in Figure 3.

We will prove that equilibrated even trees are optimal. First, we compute the path length of an equilibrated tree.

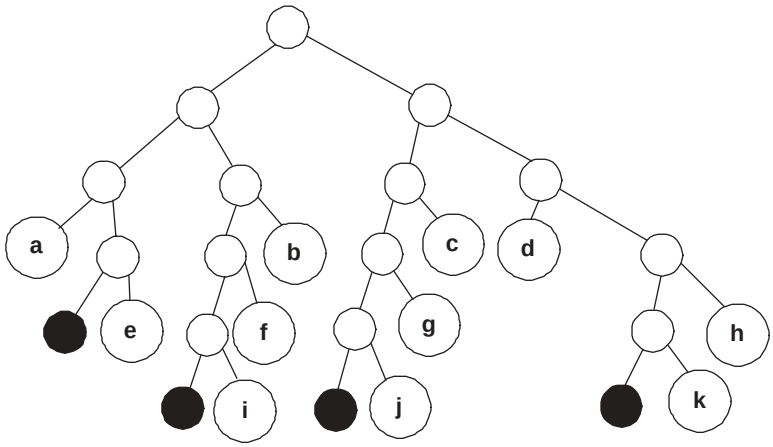

Figure 3: Equilibrated even tree for 11 symbols

Theorem 3 *Let $P(n)$ be the path length of an equilibrated even tree for n symbols. Then:*

$$P(n) = \begin{cases} 1, & \text{if } n = 1 \\ 3, & \text{if } n = 2 \\ n.(\lceil \log \frac{n}{3} \rceil + 3) - 3.2^{\lceil \log \frac{n}{3} \rceil}, & \text{if } n > 2 \end{cases}$$

Proof: If $n \leq 5$, we can verify by hand that the theorem is correct. For $n > 5$, use induction. Assume that the above expression is correct for $2 < j < n$. Applying the definition and the induction hypothesis, it follows:

$P(n) = n + P(\lfloor n/2 \rfloor) + P(\lceil n/2 \rceil)$.

Thus:

$$P(n) = n + \lfloor n/2 \rfloor \left(\left\lceil \log \frac{\lfloor n/2 \rfloor}{3} \right\rceil + 3 \right) - 3.2^{\lceil \log \frac{\lfloor n/2 \rfloor}{3} \rceil} +$$

$$+ \lceil n/2 \rceil \left(\left\lceil \log \frac{\lceil n/2 \rceil}{3} \right\rceil + 3 \right) - 3.2^{\lceil \log \frac{\lceil n/2 \rceil}{3} \rceil}$$

We then have two cases:

a) n is of the form $n = 3.2^{q-1} + 1$.

Then:

$\lceil \log \frac{n}{3} \rceil = q$, $\lceil \log \frac{\lceil n/2 \rceil}{3} \rceil = q - 1$, $\lceil \log \frac{\lfloor n/2 \rfloor}{3} \rceil = q - 2$, $\lfloor n/2 \rfloor = 3.2^{q-2}$.

Hence:

$P(n) = n + \lfloor n/2 \rfloor (q - 2 + 3) - 3.2^{q-2} + \lceil n/2 \rceil (q - 1 + 3) - 3.2^{q-1} = n + (\lfloor n/2 \rfloor + \lceil n/2 \rceil).(q - 1 + 3) - 3.2^{q-2} - 3.2^{q-2} - 3.2^{q-1} = n(q + 3) - 3.2^q = n(\lceil log \frac{n}{3} \rceil + 3) - 3.2^{\lceil log \frac{n}{3} \rceil}$. That is, the result is still valid for n.

b) n is of the form $n = 3.2^q$, or $n = 3.2^{q-1} + p$, $1 < p < 3.2^{q-1}$.

Then:

$\lceil \log \frac{n}{3} \rceil = q$, $\lceil \log \frac{\lceil n/2 \rceil}{3} \rceil = \lceil \log \frac{\lfloor n/2 \rfloor}{3} \rceil = q - 1$.

Hence:

$P(n) = n + \lfloor n/2 \rfloor (q - 1 + 3) - 3.2^{q-1} + \lceil n/2 \rceil (q - 1 + 3) - 3.2^{q-1} = n + (\lfloor n/2 \rfloor + \lceil n/2 \rceil)(q - 1 + 3) - 3.2^{q-1} - 3.2^{q-1} = n + n(q - 1 + 3) - 3.2^{q-1} - 3.2^{q-1} = n(q + 3) - 3.2^q = n(\lceil \log \frac{n}{3} \rceil + 3) - 3.2^{\lceil \log \frac{n}{3} \rceil}$. That is, the result is still valid for n.

In both cases, the result is still valid for n, so the proof is complete. □

Finally, we have

Theorem 4 *Equilibrated even trees are optimal.*

Proof: Let T be an equilibrated even tree for n symbols. For $n \leq 6$, we can verify by hand that the theorem is correct. For $n > 6$ use induction. From Theorem 3, $P(n) = n(\lceil \log \frac{n}{3} \rceil + 3) - 3.2^{\lceil \log \frac{n}{3} \rceil}$.

Induction Hypothesis: Equilibrated even trees for less than n symbols are optimal.

Induction Step: Let T' be an optimal even tree for $n > 6$ symbols. As the subtrees are also optimal and by the induction hypothesis, we can replace the subtrees T'_L and T'_R of T by equilibrated trees, while maintaining T' as optimal. Let i be the number of symbols on the left subtree of T'. Apply Theorem 2.

First, we prove that $i > 2$.

a) By contrary, suppose that $i = 1$.

Hence: $P_{T'}(n) = n + (n - 1)(\lceil \log \frac{n-1}{3} \rceil + 3) - 3.2^{\lceil \frac{n-1}{3} \rceil}$. Consider the following two alternatives.

1) $\lceil \log \frac{n}{3} \rceil = \lceil \log \frac{n-1}{3} \rceil = q$. Then:

$P_{T'}(n) = n + (n - 1)(q + 3) - 3.2^q = n(q + 3) - 3.2^q + n - (q + 3) = P_T(n) + (n - q) - 3$. By using the fact that, in this case, $(n - q) > 3$, we have $P_{T'}(n) > P_T(n)$, an absurd.

2) $\lceil \log \frac{n}{3} \rceil = q$ and $\lceil \log \frac{n-1}{3} \rceil = q - 1$. Then:

$P_{T'}(n) = n + (n - 1)(q - 1 + 3) - 3.2^{q-1} = n(q + 3) - 3.2^{q-1} - (q + 2) = n(q + 3) - 3.2^q + 3.2^{q-1} - (q + 2) = P_T(n) + 3.2^{q-1} - (q + 2)$. By using the fact that, in this case, $q > 1 \Rightarrow 3.2^{q-1} > (q + 2)$, again we arrive to the contradiction $P_{T'}(n) > P_T(n)$. Consequently $i \neq 1$.

b) Let us now try the alternative $i = 2$.

Hence: $P_{T'}(n) = n + 3 + (n - 2)(\lceil \log \frac{n-2}{3} \rceil + 3) - 3.2^{\lceil \frac{n-2}{3} \rceil}$. Examine the possible subcases.

1) $\lceil \log \frac{n}{3} \rceil = \lceil \log \frac{n-2}{3} \rceil = q$. Then:

$P_{T'}(n) = n+3+(n-2)(q+3)-3.2^q = n(q+3)-3.2^q+n-(2q+3) = P_T(n)+n-(2q+3)$. Clearly, this case can only occur for $n > 7$. In this situation, $n > (2q+3)$, implying that $P_{T'}(n) > P_T(n)$, a contradiction.

2) $\lceil \log \frac{n}{3} \rceil = q$, $\lceil \log \frac{n-2}{3} \rceil = q-1$. Then:

$P_{T'}(n) = n+3+(n-2)(q-1+3)-3.2^{q-1} = n(q+3)-3.2^q+3.2^{q-1}-(2q+1) = P_T(n)+3.2^{q-1}-(2q+1)$. By using the fact that in this case $q > 1 \Rightarrow 3.2^{q-1} > (2q+3) > (2q+1)$ we have $P_{T'}(n) > P_T(n)$, again an absurd. Consequently, $i \neq 2$.

c) It was proved that $i > 2$. We will show that writing $P_{T'}(n)$ in terms of a function $f(i)$ yields a non increasing function in the interval $[3, \lfloor \frac{n}{2} \rfloor]$. Consequently, $f(i)$ has a minimum at $i = \lfloor \frac{n}{2} \rfloor$ and T' will be an optimal even tree. This means that T' will be equilibrated and equivalent to T. Hence T must be also optimal.

We have: $f(i) = n + i(\lceil \log \frac{i}{3} \rceil + 3) - 3.2^{\lceil \log \frac{i}{3} \rceil} + (n-i)(\lceil \log \frac{n-i}{3} \rceil + 3) - 3.2^{\lceil \log \frac{n-i}{3} \rceil}$.
By taking two consecutive values for i it follows:

$f(i) - f(i+1) = n + i(\lceil \log \frac{i}{3} \rceil + 3) - 3.2^{\lceil \log \frac{i}{3} \rceil} + (n-i)(\lceil \log \frac{n-i}{3} \rceil + 3) - 3.2^{\lceil \log \frac{n-i}{3} \rceil} - (n + (i+1)(\lceil \log \frac{i+1}{3} \rceil + 3) - 3.2^{\lceil \log \frac{i+1}{3} \rceil} + (n-i-1)(\lceil \log \frac{n-i-1}{3} \rceil + 3) - 3.2^{\lceil \log \frac{n-i-1}{3} \rceil})$.

Let $\lceil \log \frac{i}{3} \rceil = q_1$ and $\lceil \log \frac{n-i}{3} \rceil = q_2$. There are now four subcases because it is possible to have $\lceil \log \frac{i+1}{3} \rceil = q_1$ or $\lceil \log \frac{i+1}{3} \rceil = q_1 + 1$ and $\lceil \log \frac{n-i-1}{3} \rceil = q_2$ or $\lceil \log \frac{n-i-1}{3} \rceil = q_2 - 1$. We should notice that the number of symbols in the left subtree is always supposed to be not greater than that in the right subtree. Hence $q_2 \geq q_1$.

1) $\lceil \log \frac{i+1}{3} \rceil = q_1$ and $\lceil \log \frac{n-i-1}{3} \rceil = q_2$. Then:

$f(i) - f(i+1) = n + i(q_1+3) - 3.2^{q_1} + (n-i)(q_2+3) - 3.2^{q_2} - (n + (i+1)(q_1+3) - 3.2^{q_1} + (n-i-1)(q_2+3) - 3.2^{q_2}) = q_2 - q_1$.
As we have $q_2 \geq q_1$, then $f(i) \geq f(i+1)$.

2) $\lceil \log \frac{i+1}{3} \rceil = q_1$ and $\lceil \log \frac{n-i-1}{3} \rceil = q_2 - 1$. Then:

$f(i) - f(i+1) = n + i(q_1+3) - 3.2^{q_1} + (n-i)(q_2+3) - 3.2^{q_2} - (n + (i+1)(q_1+3) - 3.2^{q_1} + (n-i-1)(q_2-1+3) - 3.2^{q_2-1}) = (n-i-1) - 3.2^{q-1} + q_2 - q_1$.

The transition $\lceil \log \frac{n-i}{3} \rceil = q_2$ and $\lceil \log \frac{n-i-1}{3} \rceil = q_2 - 1$, occurs only when $(n-i)$ is of the form $n - i = 3.2^{q_2-1} + 1$. Then $f(i) - f(i+1) = (n-i-1) - (n-i-1) + q_2 - q_1 = q_2 - q_1$. By using the fact that, in this case, we have $q_2 > q_1$, then $f(i) > f(i+1)$.

3) $\lceil \log \frac{i+1}{3} \rceil = q_1 + 1$ and $\lceil \log \frac{n-i-1}{3} \rceil = q_2$. Then:

$f(i) - f(i+1) = n + i(q_1+3) - 3.2^{q_1} + (n-i)(q_2+3) - 3.2^{q_2} - (n + (i+1)(q_1+1+3) - 3.2^{q_1+1} + (n-i-1)(q_2+3) - 3.2^{q_2}) = 3.2^{q_1} - i + q_2 - q_1 - 1$.

The transition $\lceil \log \frac{i}{3} \rceil = q_1$ and $\lceil \log \frac{i+1}{3} \rceil = (q_1+1)$, occurs only when i is of the form $i = 3.2^{q_1}$. Hence: $f(i) - f(i+1) = i - i + q_2 - q_1 - 1 = q_2 - q_1 - 1$. As we have again in this case $q_2 > q_1$, then $f(i) \geq f(i+1)$.

4) $\lceil \log \frac{i+1}{3} \rceil = q_1 + 1$ and $\lceil \log \frac{n-i-1}{3} \rceil = q_2 - 1$. Then:

$f(i) - f(i+1) = n + i(q_1+3) - 3.2^{q_1} + (n-i)(q_2+3) - 3.2^{q_2} - (n + (i+1)(q_1+3) - 3.2^{q_1} + (n-i-1)(q_2+3) - 3.2^{q_2}) = 3.2^{q_1} - i + n - i - 1 - 3.2^{q_2-1} + q_2 - q_1 - 1.$

The transition $\lceil \log \frac{i}{3} \rceil = q_1$, and $\lceil \log \frac{i+1}{3} \rceil = (q_1+1)$, occurs only when i is of the form $i = 3.2^{q_1}$. Also, the transition $\lceil \log \frac{n-i}{3} \rceil = q_2$, and $\lceil \log \frac{n-i-1}{3} \rceil = (q_2-1)$, occurs only when $n-i$ is of the form $n - i = 3.2^{q_2-1} + 1$. Hence: $f(i) - f(i+1) = i - i + n - i - 1 - (n-i-1) + q_2 - q_1 - 1 = q_2 - q_1 - 1$. By using the fact that, in this case, we have $q_2 > (q_1+1)$, then $f(i) > f(i+1)$.

We proved that, in all cases $f(i) \geq f(i+1)$ in the interval $[3, \lfloor \frac{n}{2} \rfloor]$. Hence, the function $f(i)$ has a minimum at $i = \lfloor \frac{n}{2} \rfloor$ and the even tree with $\lfloor \frac{n}{2} \rfloor$ symbols in the left subtree is optimal and is equilibrated. Consequently, the hypothesis is also valid for n symbols and the induction is complete. □

4 Characterization of optimal even trees

In Section 3, we have defined equilibrated even trees and showed that they are optimal. However, there are optimal even trees that are not equilibrated. In this section, we characterize all optimal even trees.

Let T be a binary tree. Denote by L_T the number of leaves in T.

Theorem 5 *An even tree with n symbols is optimal if and only if*

- *(i) T_L is an optimal even tree and T_R is an optimal odd tree, and*
- *(ii) L_{T_L} or L_{T_R} is an integer in the interval $[i_{min}, \lfloor n/2 \rfloor]$, where*

$$i_{min} = \begin{cases} 3.2^{\lfloor \log \frac{n}{3} \rfloor - 1} & \text{if } n \leq 9.2^{\lfloor \log \frac{n}{3} \rfloor - 1} \\ n - 3.2^{\lfloor \log \frac{n}{3} \rfloor} & \text{otherwise} \end{cases}$$

Proof: Let T be an optimal even tree with n symbols. We show that T satisfies (i) and (ii). If (i) is not satisfied then replacing the left or right subtree of T by an optimal even or odd tree for the same number of symbols would reduce the cost of the tree, a contradiction.
Now we prove that (ii) must hold, using Theorem 4. Without loss of generality, suppose $L_{T_L} \leq L_{T_R}$. We know that $L_{T_L} = i > 2$. Then $n > 5$. We can write n as $n = 3.2^{q-1} + p$, p and q integers, $1 \leq p \leq 3.2^{q-1}$. Let us call $\lceil \log \frac{i}{3} \rceil = q_1$, $\lceil \log \frac{n-i}{3} \rceil = q_2$ and $\lceil \log \frac{n}{3} \rceil = q$. The function $f(i) = P_T(n)$ is non increasing in the interval $[3, \lfloor \frac{n}{2} \rfloor]$ and has a minimum at $i = \lfloor \frac{n}{2} \rfloor$. Consequently, there exists exactly one smallest value i_{min}, such that $f(i)$ remains constant in the interval $[i_{min}, \lfloor \frac{n}{2} \rfloor]$. For each value i of this interval we have a distinct optimal even tree T, such that T_L has i leaves. We determine i_{min}. There are two possible starting points from where $f(i)$ remains constant and q_2 and q_1 should satisfy certain conditions.

a) $q_2 = q_1$.
Then for $i \in [i_{min}, \lfloor \frac{n}{2} \rfloor]$, we have $\lceil \log \frac{i}{3} \rceil = q_1 = \lceil \log \frac{n-i}{3} \rceil = q_2$. Hence $\lceil \log \frac{i_{min}}{3} \rceil = q_1 = \lceil \log \frac{\lfloor \frac{n}{2} \rfloor}{3} \rceil = \lceil \log \frac{n-i}{3} \rceil = q_2$. Now we have two subcases, depending on if n is of the form $n = 3.2^{q-1} + 1$ or no. If it is, then $q_1 = q_2 = q - 2$. If it is not, then $q_1 = q_2 = q - 1$.

1) n is of the form $n = 3.2^{q-1} + 1$. Hence: $q_1 = q_2 = q - 2$.
In this case there is only one possible value for i_{min}:
(I) $i_{min} = \lfloor \frac{n}{2} \rfloor = 3.2^{q-2}$.

2) n is not of the form $n = 3.2^{q-1} + 1$. Hence: $q_1 = q_2 = q - 1$.
In this case, i_{min} should satisfy:
(II) $3.2^{q-2} + 1 \leq i_{min} \leq \lfloor \frac{n}{2} \rfloor$.
(III) $n - i_{min} \leq 3.2^{q-1}$. From (2) and (3), we have:
(IV) $\max\{3.2^{q-2} + 1, n - 3.2^{q-1}\} \leq i_{min} \leq \lfloor \frac{n}{2} \rfloor$.

b) $q_2 = q_1 + 1$. In this case, i_{min} is a transition point of the form $i_{min} = 3.2^{q_1}$. We have two subcases. In the first one we will prove that n is not of the form $n = 3.2^q$.

1) Suppose that $n = 3.2^q$.
Let us see the possible values for q_1. We know that $q_1 < q$. We can not have $q_1 = q - 1$ because this would imply that $n - i_{min} = 3.2^{q-1}$ and $\lceil \log \frac{n-i_{min}}{3} \rceil = q_2 = q - 1 = q_1$. But we also can not have $q_1 < q - 1$ because this would imply that $n - i_{min} > 3.2^{q-1}$ and $\lceil \log \frac{n-i_{min}}{3} \rceil = q_2 = q$. Hence $q_2 > q_1 + 1$. We do not have possible values for q_1. Consequently, n can not be of the form $n = 3.2^q$.

2) $n = 3.2^{q-1} + p$, $1 \leq p < 3.2^{q-1}$.
Let us see the possible values for q_1. We can not have $q_1 = q$ because this would imply that $n - 3.2^{q-1} = p$ and $\lceil \log \frac{p}{3} \rceil = q_2 < q - 1 = q_1$. We can not have $q_1 < q - 2$ because this would imply that $q_2 \geq q - 1$ and hence $q_2 > q_1 + 1$. But we can have $q_1 = q - 2$, the only possible value for q_1. In this case, we have $n - i_{min} = 3.2^{q-2} + p$ and $q_2 = \lceil \log \frac{3.2^{q-2}+p}{3} \rceil = q - 1 = q_1 + 1$.

We can resume this situation stating the conditions for i_{min}:
(V) $3.2^{q-2} = i_{min} < \lfloor \frac{n}{2} \rfloor$, and
(VI) $n - i_{min} \leq 3.2^{q-1} \Rightarrow n \leq 3.2^{q-1} + 3.2^{q-2} = 9.2^{q-2}$

We determined the conditions that i_{min} should satisfy for the existence of an optimal even tree having i_{min} leaves in its left subtree. These conditions leads to the determination of the exact values of i_{min} as follows. Consider the possible values that n can assume.

1) $n = 3.2^q$.
Then $n = 6.2^{q-1} < 9.2^{q-1} = 9.2^{\lfloor \log \frac{n}{3} \rfloor - 1}$.
This situation corresponds to subcase a.2). By applying (IV), we have:
$i_{min} = \max\{3.2^{q-2} + 1, 3.2^q - 3.2^{q-1}\} = 3.2^{q-1} = \lfloor \frac{n}{2} \rfloor$. In this case there is only one element in the interval. We can write: $i_{min} = 3.2^{q-1} = 3.2^{\lfloor \log \frac{n}{3} \rfloor - 1}$, which corresponds to the first situation of the theorem.

2) $n = 3.2^{q-1} + 1$.
Then $n = 6.2^{q-1} + 1 < 9.2^{q-1} = 9.2^{\lfloor \log \frac{n}{3} \rfloor - 1}$.
This situation corresponds to case a.1). By applying (I), we have:

$i_{min} = 3.2^{q-2}$. In this case there is also only one element in the interval. We can write: $i_{min} = 3.2^{q-2} = 3.2^{\lfloor \log \frac{n}{3} \rfloor - 1}$, which also corresponds to the first situation of the theorem.

3) $n \leq 9.2^{\lfloor \log \frac{n}{3} \rfloor - 1}$ and is not cases 1) nor 2).
This is equivalent to say that n is of the form $n = 3.2^{q-1} + p$, $2 \leq p \leq 3.2^{q-2}$. The smallest value for i_{min} occurs by applying (V), $i_{min} = 3.2^{q-2}$ because, if we would apply (IV) we would have a worst value $i_{min} \geq 3.2^{q-2} + 1$.
We can rewrite this condition as: $i_{min} = 3.2^{\lfloor \log \frac{n}{3} \rfloor - 1}$, because here we have $\lfloor \log \frac{n}{3} \rfloor = q - 1 = \lceil \log \frac{n}{3} \rceil - 1$. This case concludes the first situation of the theorem.

4) $n > 9.2^{\lfloor \log \frac{n}{3} \rfloor - 1}$ and is not cases 1) nor 2) nor 3).
In this situation we must have $n = 3.2^{q-1} + p$, $3.2^{q-2} < p < 3.2^{q-1}$, since $9.2^{q-2} = 3.2^{q-1} + 3.2^{q-2} \Rightarrow \lfloor \log \frac{n}{3} \rfloor = q - 1 = \lceil \log \frac{n}{3} \rceil - 1$.
Hence, this is equivalent to $n > 9.2^{\lceil \log \frac{n}{3} \rceil - 1}$. There is only one possible way to satisfy the condition (IV) of case a). That is:
$\lfloor \frac{n}{2} \rfloor \geq i_{min} \geq \max\{3.2^{q-2} + 1, n - 3.2^{q-1}\}$. But we have $n - 3.2^{q-1} > 9.2^{q-2} - 3.2^{q-1} = 3.2^{q-2}$. Then $i_{min} = n - 3.2^{q-1}$, that is, $i_{min} = n - 3.2^{\lfloor \log \frac{n}{3} \rfloor}$, corresponding to the second situation of the theorem. This completes the proof of the necessary condition.

For the suffiency, suppose that T is an even tree satisfying (i) and (ii). The proof of the necessity and Theorem 4 imply that an optimal even tree satisfies (i) and (ii). Moreover, there exists an optimal tree T' satisfying (i) and (ii) and also $L_{T'} = L_T$. Consequently, T' and T have the same cost, impying that T is indeed optimal. □

The above theorem provides a direct recognition of optimal even trees. It also describes a method for generating all optimal trees for a given set of symbols.

5 Optimal even trees and Huffman trees

In this section, we compare optimal even trees with Huffman trees. Here, Huffman trees correspond to complete strictly binary trees. First, we compare the sizes of the maximum encodings of these trees. Finally, we compare the average sizes of an encoding in the trees.

Let us determine the minimum and maximum depths of optimal even trees for n symbols. It is rather intuitive that these parameters can be obtained from the following ideas:

1. Optimal even trees with minimum depth can be recursively constructed by using left subtrees with maximum number of symbols, provided that this number is not greater than the number of symbols in the right subtree. Consequently, the left subtree contains $\lfloor \frac{n}{2} \rfloor$ symbols. As it was defined before, these are the equilibrated even trees.

2. An optimal even tree T with maximum depth can also be recursively constructed by using left subtrees with minimum number of symbols. That is, the left subtree of T has i_{min} leaves, where i_{min} is given by Theorem 5.

Let $D_{min}(n)$ and $D_{max}(n)$ be respectively the minimum and maximum depth of an optimal even tree for n symbols. Then

$$D_{min} = \begin{cases} 1, & \text{if } n = 1 \\ 1 + D_{min}(\lceil n/2 \rceil), & \text{if } n > 1 \end{cases} \quad \text{and}$$

$$D_{max} = \begin{cases} 1, & \text{if } n = 1 \\ 1 + D_{max}(n - i_{min}), & \text{if } n > 1 \end{cases}$$

Solving the above recurrences, we obtain

$$D_{min} = \begin{cases} 1, & \text{if } n = 1, \\ \lceil \log n \rceil + 1, & \text{if } n > 1 \end{cases} \quad \text{and}$$

$$D_{max} = \begin{cases} 1, & \text{if } n = 1 \\ \lceil \log \frac{n}{3} \rceil + 3, & \text{if } n > 1 \end{cases}$$

In Huffman trees, the minimum and the maximum depths coincide. Let H be a Huffman tree for n symbols and D_H its depth. Then $D_H = \lceil \log n \rceil$. Comparing D_{min} and D_H, we conclude that we can always construct an optimal even tree whose largest encoding is one bit larger than the corresponding one for a Huffman tree. From the values of D_{max} and D_H, we can additionally obtain that the maximum encoding in an arbitrary optimal even tree is at most two bits larger than that in a Huffman tree.

In the sequel, we consider average sizes. In fact, we consider average costs of the trees, which in this case correspond to average root-leaf path sizes, i.e average encoding sizes. The average is over the number of symbols n.

The following theorem shows that the average encoding size of an optimal even code is at most 0.5 bits larger that that of a corresponding Huffman code.

Theorem 6 *The difference between the average encoding sizes of an optimal even tree and a Huffman tree, with $n > 1$ symbols, lies in the interval $[1/3, 1/2]$. It is minimum when $n = 3.2^k$, and maximum when $n = 2^k$, for some k.*

Proof: The average encoding size of an optimal even tree (Theorem 4) is

$$\frac{n(\lceil \log \frac{n}{3} \rceil + 3) - 3.2^{\lceil \log \frac{n}{3} \rceil}}{n}$$

while that of an optimal Huffman tree [5] is

$$\frac{n(\lceil \log n \rceil + 1) - 2^{\lceil \log n \rceil}}{n} \quad .$$

Then

$$d = \frac{n(\lceil \log \frac{n}{3} \rceil + 3) - 3.2^{\lceil \log \frac{n}{3} \rceil} - (n(\lceil \log n \rceil + 1) - 2^{\lceil \log n \rceil})}{n}$$

First, we prove that $d \leq 1/2$. Consider the folowing two cases.

1) n is of the form $n = 2^k$, $k \geq 1$. Then $\lceil \log n \rceil = k$ and $\lceil \log \frac{n}{3} \rceil = \lceil \log ((2^{k-2})(\frac{4}{3})) \rceil = \lceil (k-2) + \log \frac{4}{3} \rceil = k - 2 + 1 = k - 1$.

$$d = \frac{2^k(k-1+3) - 3.2^{k-1} - (2^k(k+1) - 2^k)}{2^k} = 1/2$$

2) $n = 2^k + p$, $1 < p < 2^k$. Then $\lceil \log n \rceil = k + 1$ and $\lceil \log \frac{n}{3} \rceil$ may be equal to $k - 1$ or k. We have two subcases:

2.1) If $\lceil \log \frac{n}{3} \rceil = k - 1$,

$$d = \frac{n(k-1+3) - 3.2^{k-1} - (n(k+1+1) - 2^{k+1})}{n} = \frac{2^{k-1}}{2^k + p}$$

Using the fact that $p > 1$, we conclude that $d < 1/2$.

2.2) If $\lceil \log \frac{n}{3} \rceil = k$, we have:

$$d = \frac{n(k+3) - 3.2^k - (n(k+1+1) - 2^{k+1})}{n} = \frac{n - 2^k}{n} = \frac{p}{2^k + p}$$

Using the fact that $p < 2^k$, we obtain $d < 1/2$.

Thus, the maximum difference between the encodings is $1/2$.

In the sequel, we show that d is at least $1/3$. Consider two more cases.

3) $n = 3.2^k$, $k \geq 0$. Then $\lceil \log n \rceil = \lceil k + \log 3 \rceil = (k + 2)$ and $\lceil \log \frac{n}{3} \rceil = \lceil \log 2^k \rceil = k$. Then

$$d = \frac{3.2^k k + 9.2^k - 3.2^k - 3.2^k k - 9.2^k + 2^{k+2}}{3.2^k} = \frac{-3.2^k + 2^{k+2}}{3.2^k} = 1/3$$

4) $n = 3.2^k + p$, $1 < p < 3.2^k$. Then $\lceil \log n \rceil$ may be $(k+2)$ or $(k+3)$, and $\lceil \log \frac{n}{3} \rceil = k+1$. Consider two additional subcases:

4.1) If $\lceil \log n \rceil = k + 2$, we have:

$$d = \frac{n(k+1+3) - 3.2^{k+1} - (n(k+2+1) - 2^{k+2})}{n} = \frac{n - 2^{k+1}}{n} = \frac{2^k + p}{3.2^k + p}$$

Since $p > 1$, we conclude that $d > 1/3$.

4.2) If $\lceil \log n \rceil = k + 3$, we have:

$$d = \frac{n(k+1+3) - 3.2^{k+1} - (n(k+3+1) - 2^{k+3})}{n} = \frac{2^{k+1}}{n} = \frac{2.2^k}{3.2^k + p}$$

By using the fact that $p < 3.2^k$, we conclude that $d > 1/3$.

Thus, in the two subcases above, the minimum difference between the encodings is 1/3. The proof is complete. □

The limits 1/3 and 1/2 can be explained in terms of the equilibrated even trees:

a) When n is of the form $n = 2^k$, the Huffman tree is a full tree, where all the leaves have the same depth $\lceil \log n \rceil$. On the other hand, the equilibrated even tree is the composition of 2^{k-1} subtrees with two encodings. In this tree, half of the leaves have depth $\lceil \log n \rceil$ and half have depth $\lceil \log n \rceil + 1$. Then, the difference of the average encoding lengths is 1/2.

b) When n is of the form $n = 3.2^k$, the Huffman tree is a complete tree, where there are 2.2^k leaves whose depths are $\lceil \log n \rceil$, and 2^k leaves whose depths are all $\lceil \log n \rceil - 1$. On the other hand, the equilibrated even tree is the composition of 2^k subtrees with three encodings. In this case, we have 2^k leaves whose depths are all $\lceil \log n \rceil + 1$, 2^k leaves whose depths are $\lceil \log n \rceil$, and 2^k leaves whose depths are $\lceil \log n \rceil - 1$. This situation leads to the difference of 1/3 in the average encoding lenghts. In the remaining cases for n, the corresponding trees have intermediate configurations between these limits.

6 Conclusion

Even codes have been defined as those whose encodings are all even. A characterization of optimal even codes has been described. It has been shown that an optimal even code for n symbols can always be found, such that the largest encoding size is just one bit larger than that for corresponding Huffman code (with equal frequencies). Furthermore, the difference is at most 2 bits, for any optimal even tree. In addition, the average size of an encoding in the optimal even code is at least 1/3 bit larger and at most 1/2 larger than that of an Huffman code. Optimal even codes for n symbols can be constructed in $O(n)$ time, aplying the definition of equilibrated trees in Section 3.

References

[1] N. Faller. *An adaptative Method for Data Compression* . Record of the 7th Asilomar Conference on Circuits, Systems and Computers, Naval Postgraduate School, Monterrey, Ca., pp. 593-597, 1973.

[2] R. G. Gallager. *Variations on a Theme by Huffman* . IEEE Transactions on Information Theory, 24(1978), pp. 668-674.

[3] R. W. Hamming.*Coding And Information Theory.* Prentice Hall, 1980.

[4] D. A. Huffman. *A Method for the Construction of Minimum Redundancy Codes.* Proceedings of the IRE, 40:1098-1101, 1951.

[5] D. E. Knuth. *The Art of Computer Programming.* Addison Wesley, 1973.

[6] D. E. Knuth. *Dynamic Huffman Coding.* Journal of Algorithms, 6(1985), pp. 163-180.

[7] E. S. Laber *Um algoritmo eficiente para construção de códigos de prefixo com restrição de comprimento.* Master's thesis, PUC-RJ, Rio de Janeiro, 1997.

[8] L. L. Larmore and D. S. Hirshberg. *A fast algorithm for optimal length-limited Huffman codes.* JACM, Vol. 37 No 3, pp. 464-473, Jul. 1990.

[9] R. L. Milidiu, E. S. Laber and A. A. Pessoa. *Improved Analysis of the FGK Algorithm.* Journal of Algorithms, Vol. 28, pp. 195-211, 1999.

[10] R. L. Milidiu and E. S. Laber. *The Warm-up Algorithm: A Lagrangean Construction of Length Restricted Huffman Codes.* Siam Journal on Computing, Vol. 30 No 5, pp. 1405-1426, 2000.

[11] R. L. Milidiu and E. S. Laber. *Improved Bounds on the Ineficiency of Length Restricted Codes.* Algorithmica, Vol. 31 No 4, pp. 513-529, 2001.

[12] A. Turpin and A. Moffat. *Practical length-limited coding for large alphabeths.* Computer J., Vol. 38, No 5, pp. 339-347, 1995.

[13] E. S. Schwartz. *An Optimum Encoding with Minimal Longest Code and Total Number of Digits.* Information and Control, 7(1964), pp. 37-44.