On the Capability of Greedy Codeword Assignment Scheme in Finding Binary Fix-Free Codes

Ehsan Ebrahimzadeh, *Student Member, IEEE*, Mohammadali Khosravifard, *Member, IEEE*,
Adel Aghajan, *Student Member, IEEE*, Aaron Gulliver, *Senior Member, IEEE*

***Abstract*— The greedy codeword assignment scheme (GCAS) is guaranteed to find a binary fix-free code for any length vector $(\ell_1, \ell_2, \ldots, \ell_n)$ with Kraft sum not greater than $\frac{5}{8}$. In this paper, we examine the attributes of GCAS from different perspectives. First it is proven that under certain conditions GCAS constructs fix-free codes for length vectors with Kraft sum greater than $\frac{5}{8}$. In order to evaluate the capability of GCAS in finding fix-free codes, the iterative version of GCAS (IGCAS) is applied to all length vectors with Kraft sum $\frac{3}{4}$ for $n \leq 32$. This is also done for the first constructive approach in the literature, i.e., the Harada-Kobayashi (HK) algorithm. It is observed that the Kraft sum of the fix-free codewords obtained using IGCAS (resp. HK) is at least $\frac{23}{32}$ (resp. $\frac{11}{16}$). Combining the construction results from IGCAS and HK, it is shown that there exists a fix-free code for each length vector with Kraft sum $\frac{3}{4}$ and $n \leq 32$.**

I. Introduction

A code is said to be prefix-free if none of the codewords is a prefix of any other. Among prefix-free codes, those which also satisfy the suffix-free property are called fix-free codes. Using fix-free codes, the decoding speed can be doubled and the robustness to transmission errors is increased. Because of these advantages, fix-free codes are used in video standards such as H.263+ and MPEG-4.

It is well-known that the Kraft inequality, i.e. $\sum_{i=1}^{n} 2^{-\ell_i} \leq 1$, is a sufficient condition for the existence of a prefix-free code with lengths $(\ell_1, \ell_2, \ldots, \ell_n)$. In the context of fix-free codes however, finding such a sufficient condition is still a very challenging problem. Initially, Ahlswede *et al.* [1] proposed the following conjecture:

$\frac{3}{4}$*-conjecture:* There exists a fix-free code for the length vector $(\ell_1, \ell_2, \ldots, \ell_n)$ if $\sum_{i=1}^{n} 2^{-\ell_i} \leq \frac{3}{4}$.

Since then, several attempts have been made to prove the $\frac{3}{4}$-conjecture or weaker sufficient conditions. Most of these sufficient conditions involve a constraint on the Kraft sum of the length vector (and possibly an additional constraint). Hence they are referred to as *Kraft-type* sufficient conditions. Using a probabilistic approach, some sufficient conditions are proposed which are not Kraft-type [2][3]. A survey on sufficient conditions for the existence of fix-free codes is provided in [4].

Several of the approaches are constructive in nature [5] [6]. The objective is to assign fix-free codewords $(C_1, C_2, \ldots, C_n)$ to the given length vector $(\ell_1, \ell_2, \ldots, \ell_n)$. Throughout this paper, all length vectors are assumed to be sorted in ascending order, i.e., $\ell_1 \leq \ell_2 \leq \cdots \leq \ell_n$.

In most of the constructive approaches, codeword assignment is accomplished in a sequential manner. This means that codeword C_i is chosen immediately after selecting $C_1, C_2, \ldots, C_{i-1}$ from the *available codewords*, i.e., those for which none of the previously assigned codewords is a prefix or suffix. Denote this set of available codewords by $\psi(i)$. The main difference between the algorithms proposed in the literature is their strategy in selecting C_i from $\psi(i)$. Harada and Kobayashi [5] suggested the first available codeword in the lexicographic order as C_i. Using this algorithm, denoted HK, they proved the following theorem:

Theorem 1: If $|\{\ell_1, \ell_2, \ldots, \ell_n\}| = 2$, then $\sum_{i=1}^{n} 2^{-\ell_i} \leq \frac{3}{4}$ is a sufficient condition for the existence of a binary fix-free code.

The capability of the HK algorithm in code assignment is studied in [7]. Another innovative strategy is to consider the starting and ending symbols of the codewords. With this idea, Yekhanin [8] proved the following theorem:

Theorem 2: $\sum_{i=1}^{n} 2^{-\ell_i} \leq \frac{5}{8}$ is a sufficient condition for the existence of a binary fix-free code.

A greedy version of Yekhanin's approach, called Greedy Codeword Assignment Scheme (GCAS), was used to prove a sufficient condition similar to that of Theorem 2 for D-ary alphabets [6].

Thus far, we have considered GCAS and HK as two algorithms which have been proposed to provide constructive proofs for some Kraft-type sufficient conditions (i.e., Theorems 1 and 2). Following a quite different approach [9][10], Savari *et al.* formulated the problem of the existence of a fix-free code for a given length vector as a boolean satisfiability (SAT) problem. In this way, the satisfiability of the equivalent SAT problem is a necessary and sufficient condition for the existence of a fix-free code. However, this sufficient condition is not Kraft-type. Moreover, since the codeword assignment is not performed in a sequential manner, SAT-based algorithms are not as efficient as their sequential counterparts in terms of time complexity. This is because in the sequential algorithms selection of the target codeword depends only on the codewords assigned beforehand, while in a SAT-based scheme all codelengths influence the codeword assignment. The advantage of SAT-based schemes is that they explicitly determine whether or not a fix-free code exists for a given length vector.

Besides HK and GCAS, several constructive algorithms have been proposed to find an efficient fix-free code for a given probability vector (not length vector) [11] [12] [13]. In these algorithms, the goal is to assign as many fix-free codewords as possible to the associated Huffman codelengths. When no codeword is feasible, the codelengths are changed accordingly. In this way, these algorithms manipulate the initial length vector so that a fix-free code is found. Hence no sufficient conditions can be provided by these algorithms. From this point of view, they are not comparable to GCAS and HK.

In this work, we evaluate the capability of GCAS in constructing *binary* fix-free codes. This involves both proving existence theorems and constructing fix-free codewords for given length vectors. The performance of GCAS is compared with that of HK.

The remainder of this paper is organized as follows. Section II presents some theorems that are proven via GCAS. In Section III, (the iterative version of)GCAS and HK are employed to construct fix-free codewords for all length vectors with Kraft sum $\frac{3}{4}$ for $n \leq 32$. Finally, Section IV concludes the paper.

II. Sufficient Conditions Provided by GCAS

In this section, GCAS is employed to prove some existence theorems. After a brief description of GCAS, we show how under certain conditions the upper bound $\frac{5}{8}$ can be improved. Since the $\frac{3}{4}$-conjecture is proved for $\ell_1 = 1$ [8], here we consider the problem of assigning fix-free codewords $(C_1, C_2, \ldots, C_n)$ to the length vector $(\ell_1, \ell_2, \ldots, \ell_n)$ where $\ell_1 > 1$.

GCAS follows a sequential assignment in a greedy manner as described below. Initially, the algorithm starts with ℓ_1 and assigns the block of ℓ_1 zeros to C_1. Let $\mathcal{A}^*_{x,y}$ denote the set of all finite binary sequences starting with symbol x and ending with symbol y. Therefore, $\psi_{x,y}(i) = \mathcal{A}^*_{x,y} \bigcap \psi(i)$ is the set of all available codewords (elements of $\psi(i)$), starting with x and ending with y, where $x, y \in \{0, 1\}$. In the i-th step of the algorithm, C_i is selected from $\psi_{x_i,y_i}(i)$. In other words, x_i and y_i denote the first and last symbols of the codeword C_i, respectively. In fact, the greediness of GCAS is considered in deciding on (x_i, y_i). If $\psi_{x_{i-1},y_{i-1}}(i) \neq \emptyset$, we set $(x_i, y_i) = (x_{i-1}, y_{i-1})$; otherwise GCAS switches to the next pair of (x, y) according to the lexicographic order (i.e. $(0,0) \rightarrow (0,1) \rightarrow (1,0) \rightarrow (1,1)$). The algorithm terminates unsuccessfully if for some $n' < n$ we have $C_{n'} \in \psi_{1,1}(n')$ but $\psi_{1,1}(n'+1) = \emptyset$. Note that while choosing a codeword from $\psi_{x_i,y_i}(i)$, GCAS selects the *first* available codeword according to the lexicographic order.

For a given length vector $L_n = (\ell_1, \ell_2, \ldots, \ell_n)$, let $\upsilon(L_n)$ denote the last index for which GCAS succeeded in assigning a codeword. Pseudo-code describing GCAS is given below:

Greedy Codeword Assignment Scheme

input: $L_n = (\ell_1, \ell_2, \ldots, \ell_n)$ with $1 < \ell_1 \leq \cdots \leq \ell_n$.
set: $\alpha_1 = \alpha_2 = \beta_1 = \beta_3 = 0$ and $\alpha_3 = \alpha_4 = \beta_2 = \beta_4 = 1$;
initialize: $i = 1$ and $j = 1$
while $i \leq n$ and $j \leq 4$
  $(x_i, y_i) \leftarrow (\alpha_j, \beta_j)$
  if $\psi_{x_i,y_i}(i) \neq \emptyset$
    Choose the first codeword in $\psi_{x_i,y_i}(i)$ as C_i
    $i \leftarrow i + 1$
  else
    $j \leftarrow j + 1$
  endif
endwhile
$\upsilon(L_n) \leftarrow i - 1$
if $\upsilon(L_n) < n$
  GCAS failed to find a binary fix free code for L_n
endif
output: $(C_1, \ldots, C_{\upsilon(L_n)})$

When GCAS is applied to a length vector L_n, define the partial Kraft sums as

$$S_{x,y}(L_n) = \sum_{\substack{m:\ m \leq \upsilon(L_n) \\ C_m \in \mathcal{A}^*_{x,y}}} 2^{-\ell_m}$$

for $x, y \in \{0, 1\}$. Throughout this paper, the argument L_n is omitted wherever it is obvious from the context. In [6], the following theorem on the partial Kraft sums was proven.

Theorem 3: If GCAS is applied to a given length vector L_n and fails to find all the required fix-free codewords, i.e., $\upsilon(L_n) < n$, then the partial Kraft sums satisfy

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0.5 & 1 & 0 & 0 \\ 0.5 & 0 & 1 & 0 \\ 0 & 0.5 & 0.5 & 1 \end{bmatrix} \begin{bmatrix} S_{0,0} \\ S_{0,1} \\ S_{1,0} \\ S_{1,1} \end{bmatrix} \geq \begin{bmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{bmatrix}.$$

■

Corollary 1: For a length vector $L_n = (\ell_1, \ell_2, \ldots, \ell_n)$, if $\sum_{i=1}^{n} 2^{-\ell_i} \leq \frac{1}{2} + \frac{1}{2} S_{0,0}$, then GCAS succeeds in finding the required fix-free codewords for L_n, i.e. $\upsilon(L_n) = n$.

Proof: From Theorem 3 and Lemma 3 in [6] it is easily observed that

$$\text{if} \quad \upsilon(L_n) < n \quad \text{then}$$

$$\sum_{i=1}^{\upsilon(L_n)} 2^{-\ell_i} = S_{0,0} + S_{0,1} + S_{1,0} + S_{1,1} \geq \frac{1}{2} + \frac{1}{2} S_{0,0}. \quad (1)$$

Consequently

$$\text{if} \quad \sum_{i=1}^{\upsilon(L_n)} 2^{-\ell_i} < \frac{1}{2} + \frac{1}{2} S_{0,0} \quad \text{then} \quad \upsilon(L_n) = n. \quad (2)$$

Now we prove that

$$\text{if} \quad \sum_{i=1}^{n} 2^{-\ell_i} = \frac{1}{2} + \frac{1}{2} S_{0,0} \quad \text{then} \quad \upsilon(L_n) = n. \quad (3)$$

This is because $\upsilon(L_n) < n$ and $\sum_{i=1}^{n} 2^{-\ell_i} = \frac{1}{2} + \frac{1}{2} S_{0,0}$ imply that $\sum_{i=1}^{\upsilon(L_n)} 2^{-\ell_i} < \frac{1}{2} + \frac{1}{2} S_{0,0}$ and hence (2) gives $\upsilon(L_n) = n$, which is a contradiction. Also since $\sum_{i=1}^{\upsilon(L_n)} 2^{-\ell_i} \leq \sum_{i=1}^{n} 2^{-\ell_i}$, (2) implies

$$\text{if} \quad \sum_{i=1}^{n} 2^{-\ell_i} < \frac{1}{2} + \frac{1}{2} S_{0,0} \quad \text{then} \quad \upsilon(L_n) = n. \quad (4)$$

Hence, (3) and (4) complete the proof. ■

Noting Corollary 1 and $S_{0,0} \geq \frac{1}{4}$, GCAS is guaranteed to find a fix-free code if $\sum_{i=1}^{n} 2^{-\ell_i} \leq \frac{5}{8}$ [6]. Now, considering an additional constraint, we improve the constructed Kraft-sum from $\frac{5}{8}$ to $\frac{21}{32}$.

Theorem 4: If $\sum_{i:\ell_i=\ell_1} 2^{-\ell_i} = \frac{1}{4}$ and $\sum_{i=1}^{n} 2^{-\ell_i} \leq \frac{21}{32}$, then GCAS finds a fix-free code.

Proof: The case $\ell_1 = 2$ has been considered earlier (see Theorem 5 in [6]). Thus, assume that $\ell_1 \geq 3$. Noting Corollary 1, it is enough to show that $S_{0,0} \geq \frac{5}{16}$. First, define $m = 2^{\ell_1-2}$. Note that GCAS selects all m codewords of length ℓ_1 from $\mathcal{A}^*_{0,0}$. For $i > m$, the inclusion-exclusion principle implies that

$$\begin{aligned} |\psi_{0,0}(i)| \quad \geq \quad & 2^{\ell_i-2} - \sum_{j:\ell_i=\ell_j} 1 - \sum_{j:\ell_i>\ell_j} 2^{\ell_i-\ell_j-1} \\ & - \sum_{j:\ell_i>\ell_j} 2^{\ell_i-\ell_j-1} + 2^{\ell_i-4}. \end{aligned} \tag{5}$$

The above inequality can be justified if we note that:

- The number of codewords of length ℓ_i in $\mathcal{A}^*_{0,0}$ is 2^{ℓ_i-2}.
- The number of codewords of length ℓ_i in $\mathcal{A}^*_{0,0}$ which were previously assigned is $\sum_{j:\ell_i=\ell_j} 1$.
- The number of codewords of length ℓ_i in $\mathcal{A}^*_{0,0}$ for which a previously assigned codeword is a prefix (resp. suffix) is $\sum_{j:\ell_i>\ell_j} 2^{\ell_i-\ell_j-1}$ $\left(\text{resp. } \sum_{j:\ell_i>\ell_j} 2^{\ell_i-\ell_j-1}\right)$.
- Let $\varphi_{0,0}(i)$ denote the set of codewords of length ℓ_i in $\mathcal{A}^*_{0,0}$ for which one of the previously assigned codewords of length ℓ_1 is a prefix and one (not necessarily distinct) is a suffix, i.e., codewords that are excluded twice in the inclusion-exclusion. We have $|\varphi_{0,0}(i)| \geq 2^{\ell_i-4}$. The proof of this assertion is given below for the two possible cases.

  Case 1: $\ell_i \neq 2\ell_1 - 1$
  Since *all* codewords of length ℓ_1 in $\mathcal{A}^*_{0,0}$ have been assigned previously, $\varphi_{0,0}(i)$ contains 2^{ℓ_i-4} codewords of length ℓ_i having the form

  $$\underbrace{0\ \cdots\ 0}_{\ell_1}\ \cdots\ \overbrace{0\ \cdots\ 0}^{\ell_1}, \qquad \text{if} \quad \ell_i > 2\ell_1 - 1$$

  and

  $$\underbrace{0\ \cdots\ \overbrace{0\ \cdots\ 0}^{\ell_1}}_{\ell_1}\ \cdots\ 0, \qquad \text{if} \quad \ell_i < 2\ell_1 - 1.$$

  Case 2: $\ell_i = 2\ell_1 - 1$
  Clearly, in this case there are exactly 2^{ℓ_i-3} codewords of length ℓ_i in $\varphi_{0,0}(i)$ having the form

  $$\underbrace{0\ \cdots\ \overbrace{0}^{\ell_1}}_{\ell_1}\overbrace{\ \cdots\ 0}.$$

Rearranging (5), one may write

$$\begin{aligned} |\psi_{0,0}(i)| \quad & \geq \quad 2^{\ell_i-2} - \sum_{j=1}^{i-1} 2^{\ell_i-\ell_j} + 2^{\ell_i-4} \\ & = \quad 2^{\ell_i-2} - \sum_{j=1}^{m} 2^{\ell_i-\ell_j} - \sum_{j=m+1}^{i-1} 2^{\ell_i-\ell_j} + 2^{\ell_i-4} \\ & = \quad 2^{\ell_i} \left(\frac{1}{16} - \sum_{j=m+1}^{i-1} 2^{-\ell_j} \right), \end{aligned} \tag{6}$$

where (6) is obtained from the fact that $\sum_{j=1}^{m} 2^{-\ell_j} = \frac{1}{4}$. Thus (6) implies $|\psi_{0,0}(i) > 0|$ as long as $\sum_{j=m+1}^{i-1} 2^{-\ell_j} < \frac{1}{16}$. As a result, GCAS terminates assigning codewords from $\mathcal{A}^*_{0,0}$ only if $S_{0,0} \geq \frac{5}{16}$. This completes the proof. ∎

Considering another constraint on the length vector, the upper bound of $\frac{5}{8}$ is improved in the following theorem.

Theorem 5: If $\sum_{i:\ell_i=\ell_1} 2^{-\ell_i} = \frac{1}{8}$ and $\sum_{i=1}^{n} 2^{-\ell_i} \leq \frac{81}{128}$, then GCAS finds a fix-free code.

Proof: The proof is very similar to that of Theorem 4. It is enough to replace 2^{ℓ_i-4} in (5) by 2^{ℓ_i-6} and show that $S_{0,0} \geq \frac{17}{64}$. To do so, it should be shown that $|\varphi_{0,0}(i)| \geq 2^{\ell_i-6}$. Define $m = 2^{\ell_1-3}$. Since GCAS assigns codewords in lexicographic order, all m codewords of length ℓ_1 start with 00 and end with 0. Therefore, $\varphi_{0,0}(i)$ contains all codewords having the form

$$\underbrace{00\ \cdots\ 0}_{\ell_1}\ \cdots\ \overbrace{00\ \cdots\ 0}^{\ell_1}, \qquad \text{if} \quad \ell_i > 2\ell_1 - 1$$

and

$$\underbrace{00\ \cdots\ \overbrace{00\ \cdots\ 0}^{\ell_1}}_{\ell_1}\ \cdots\ 0, \qquad \text{if} \quad \ell_1 + 1 < \ell_i < 2\ell_1 - 2$$

which means that $|\varphi_{0,0}(i)| \geq 2^{\ell_i-6}$. Similarly, it is not hard to verify that for the other cases of ℓ_i, there exists at least 2^{ℓ_i-5} codewords in $\varphi_{0,0}(i)$. Thus for $i > m$, one can write

$$\begin{aligned} |\psi_{0,0}(i)| \quad & \geq \quad 2^{\ell_i-2} - \sum_{j=1}^{i-1} 2^{\ell_i-\ell_j} + 2^{\ell_i-6} \\ & = \quad 2^{\ell_i-2} - \sum_{j=1}^{m} 2^{\ell_i-\ell_j} - \sum_{j=m+1}^{i-1} 2^{\ell_i-\ell_j} + 2^{\ell_i-6} \\ & = \quad 2^{\ell_i} \left(\frac{1}{4} - \frac{1}{8} - \sum_{j=m+1}^{i-1} 2^{-\ell_j} + \frac{1}{64} \right) \end{aligned} \tag{7}$$

and conclude that GCAS terminates assigning codewords from $\mathcal{A}^*_{0,0}$ only if $S_{0,0} \geq \frac{17}{64}$. ∎

Note that following Yekhanin's approach, one cannot guarantee sufficient conditions with Kraft sum greater than $\frac{5}{8}$. In fact, the results of this section are a consequence of the greediness of GCAS, i.e., the inequality given by (1).

III. CONSTRUCTION OF FIX-FREE CODES

In the previous section, GCAS was used to guarantee the existence of a binary fix-free code for length vectors with Kraft sum greater than $\frac{5}{8}$ that also satisfy an additional constraint.

The performance of GCAS is examined in this section from a purely constructive point of view. To do so, it is aimed at employing GCAS for finding fix-free codewords for length vectors with Kraft sum $\frac{3}{4}$ and correspondingly comparing its performance with that of HK. In order to have a meaningful comparison, the codeword assignment procedure in GCAS should be revised. Since GCAS was intended to be used for proving existence theorems, some surplus codewords are deliberately omitted in the codeword assignment procedure. In fact, GCAS terminates once the available codewords in $\mathcal{A}^*_{1,1}$ are exhausted, while there may be available codewords of subsequent lengths in $\mathcal{A}^*_{0,0} \bigcup \mathcal{A}^*_{0,1} \bigcup \mathcal{A}^*_{1,0}$. Thus a natural means of improving GCAS is to applying it *iteratively*. This means that in the iterative version of GCAS, which is denoted by IGCAS henceforth, the algorithm switches to $\psi_{0,0}(i)$ (and then $\psi_{0,1}(i)$) instead of terminating. In other words, the **while** loop in the pseudo-code of GCAS is replaced with the following loop in IGCAS.

while $i \le n$ and $\bigcup_{k=1}^{4} \psi_{\alpha_k,\beta_k}(i) \neq \emptyset$
  $(x_j, y_j) \leftarrow (\alpha_j, \beta_j)$
  if $\psi_{x_i,y_i}(i) \neq \emptyset$
    Choose the first codeword in $\psi_{x_i,y_i}(i)$ as C_i
    $i \leftarrow i+1$
  else
    $j \leftarrow (j \bmod 4) + 1$
  endif
endwhile

In this way, the algorithm terminates only when there exists no available codeword. Therefore, it is now comparable with HK. Obviously, the capability of GCAS is potentially improved if it is applied iteratively. For example, GCAS fails for the length vector $J = (3,3,3,6,6,\ldots,6) \in \mathcal{L}_{\frac{3}{4}}(27)$ for which finding a fix-free code with the HK algorithm is guaranteed by Theorem 1. However, using IGCAS, one obtains the fix-free code $(000, 010, 001, 011011, 011101, 011111, 100100, 100110, 101100, 101110, 110100, 110110, 111100, 111110, 100011, 100101, 100111, 101011, 101101, 101111, 110011, 110101, 110111, 111011, 111101, 111111, 011100)$ for the length vector J. More generally, Theorem 1 which was initially proven using the HK algorithm can also be proven for $D = 2$ using the iterative version of IGCAS. The proof is omitted as a proof of the result already exists.

As stated previously, in this section, the objective is to employing the algorithms for finding fix-free codewords for length vectors with Kraft sum $\frac{3}{4}$.

The set of all such length vectors is denoted by

$$\mathcal{L}_{\frac{3}{4}}(n) = \left\{ (\ell_1, \ldots, \ell_n) \mid \ell_1 \le \ldots \le \ell_n, \sum_{i=1}^{n} 2^{-\ell_i} = \frac{3}{4} \right\}.$$

This set is of particular interest for the following two reasons:

1) The $\frac{3}{4}$-conjecture is proven if it is shown that there exists a fix-free code for each length vector with Kraft sum $\frac{3}{4}$.
2) If the $\frac{3}{4}$-conjecture is proven, to find the minimum redundancy length vector among those for which the existence of a fix-free code is guaranteed, attention should be given to those vectors with Kraft sum equal to $\frac{3}{4}$. This fact is stated formally in the following proposition.

Proposition 1: For any probability vector $P = (p_1, p_2, \ldots, p_n)$ satisfying $p_1 \ge p_2 \ge \cdots \ge p_n > 0$, we have

$$\arg \min_{\substack{\ell_1,\ell_2,\ldots,\ell_n \\ \sum_{i=1}^{n} 2^{-\ell_i} \le \frac{3}{4}}} \sum_{i=1}^{n} p_i \ell_i = \arg \min_{\substack{\ell_1,\ell_2,\ldots,\ell_n \\ \sum_{i=1}^{n} 2^{-\ell_i} = \frac{3}{4}}} \sum_{i=1}^{n} p_i \ell_i.$$

Proof: Let

$$L_n^* = (\ell_1^*, \ell_2^*, \ldots, \ell_n^*) = \arg \min_{\substack{\ell_1,\ell_2,\ldots,\ell_n \\ \sum_{i=1}^{n} 2^{-\ell_i} \le \frac{3}{4}}} \sum_{i=1}^{n} p_i \ell_i$$

and assume that $\sum_{i=1}^{n} 2^{-\ell_i^*} < \frac{3}{4}$. Multiply both sides of this inequality by $2^{\ell_n^*}$. It is clear that $\ell_n^* \ge 2$ and $\ell_i^* \le \ell_n^*$ for $i < n$. As a result, both sides of the inequality $1 + \sum_{i=1}^{n-1} 2^{\ell_n^* - \ell_i^*} < 3 \times 2^{\ell_n^* - 2}$ are integers. Therefore, we can write $2 + \sum_{i=1}^{n-1} 2^{\ell_n^* - \ell_i^*} \le 3 \times 2^{\ell_n^* - 2}$. This shows that the Kraft sum of the length vector $L'_n = (\ell_1^*, \ell_2^*, \ldots, \ell_n^* - 1)$ is not greater than $\frac{3}{4}$. However, the average codelength of L'_n is less than that of L_n^* and this contradicts the optimality of L_n^*. Therefore, it must be that $\sum_{i=1}^{n} 2^{-\ell_i^*} = \frac{3}{4}$. ■

The key to generating the elements of $\mathcal{L}_{\frac{3}{4}}(n)$ is to consider the set of compact length vectors (i.e. those with Kraft sum 1), which has been studied extensively in the literature, e.g. [14][15]. Each $(n+1)$-tuple compact length vector containing length 2 is associated with an element of $\mathcal{L}_{\frac{3}{4}}(n)$, and vice versa.

For a given length vector L_n, let $\sigma(L_n)$ denote the Kraft sum of the codewords which are successfully assigned by the algorithm. The following parameters can be considered in evaluating the capability of the algorithm to find fix-free codes.

- Denote the number of length vectors for which the algorithm fails to find a fix-free code by
$$\mathcal{F}_n = \left| \left\{ L_n \in \mathcal{L}_{\frac{3}{4}}(n) \mid \sigma(L_n) < \frac{3}{4} \right\} \right|.$$
- Denote the minimum Kraft sum $\sigma(L_n)$ among all length vectors in $\mathcal{L}_{\frac{3}{4}}(n)$ by
$$\gamma_n = \min_{L_n \in \mathcal{L}_{\frac{3}{4}}(n)} \sigma(L_n).$$

Table I gives the values of $\mathcal{F}_n$ and γ_n for both IGCAS and HK and $n \leq 32$. This shows that IGCAS finds fix-free codes for a remarkable percentage (more than 99.99%), of length vectors $L_n \in \mathcal{L}_{\frac{3}{4}}(n)$. Moreover, it has far fewer failures than HK.

Interestingly, for $n \leq 32$, there are only two length vectors for which both IGCAS and HK fail to find a fix-free code. These length vectors are:
$J_{22} = (2, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 6, \dots, 6) \in \mathcal{L}_{\frac{3}{4}}(22)$
$J_{32} = (2, 3, 5, 5, 5, 5, 5, 5, 7, \dots, 7) \in \mathcal{L}_{\frac{3}{4}}(32)$.
Fortunately, it is not hard to find fix-free codewords for these length vectors. For instance, the binary representation of the integer vectors

- $K_{22} = (0, 9, 10, 11, 13, 15, 18, 19, 22, 25, 26, 30, 17, 29, 33, 34, 35, 46, 49, 55, 59, 63)$
- $K_{32} = (3, 5, 1, 2, 8, 9, 16, 18, 12, 14, 17, 20, 22, 24, 25, 26, 28, 30, 42, 44, 46, 49, 52, 56, 57, 58, 60, 62, 68, 70, 76, 78)$

provide fix-free codewords for J_{22}, J_{27} and J_{32}, respectively. Thus we can conclude the following theorem.

Theorem 6: If $n \leq 32$, then $\sum_{i=1}^{n} 2^{-\ell_i} = \frac{3}{4}$ is a sufficient condition for the existence of a binary fix-free code. ∎

TABLE I
NUMBER OF FAILURES AND MINIMUM CONSTRUCTED KRAFT SUM FOR GCAS AND HK

n	$\mathcal{L}_{\frac{3}{4}}(n)$	$\mathcal{F}_n^{IGCAS}$	γ_n^{IGCAS}	$\mathcal{F}_n^{HK}$	γ_n^{HK}
3	2	0	3/4	0	3/4
4	2	0	3/4	0	3/4
5	4	0	3/4	0	3/4
6	7	0	3/4	0	3/4
7	11	0	3/4	0	3/4
8	20	0	3/4	0	3/4
9	36	0	3/4	0	3/4
10	63	0	3/4	0	3/4
11	113	1	23/32	0	3/4
12	202	0	3/4	2	23/32
13	360	0	3/4	2	11/16
14	646	0	3/4	2	11/16
15	1157	1	3/4	2	11/16
16	2073	1	3/4	5	11/16
17	3719	1	3/4	7	11/16
18	6668	1	3/4	11	11/16
19	11958	0	3/4	19	11/16
20	21452	1	23/32	34	11/16
21	38480	3	23/32	61	11/16
22	69029	4	23/32	111	11/16
23	123842	5	23/32	192	11/16
24	222175	8	23/32	339	11/16
25	398596	13	23/32	606	11/16
26	715124	22	23/32	1081	11/16
27	1283006	39	23/32	1924	11/16
28	2301866	60	23/32	3445	11/16
29	4129849	108	23/32	6165	11/16
30	7409494	189	23/32	11042	11/16
31	13293651	338	23/32	19796	11/16
32	23850683	604	23/32	35494	11/16

As for the second parameter, i.e. γ_n, Table I reveals that with IGCAS the Kraft sum of the constructed fix-free codewords is at least $\frac{23}{32}$ for $n \leq 32$.

Examining the successes and failures of IGCAS, it was also observed that it finds a fix-free code if each element in $\{\ell_1, \ell_2, \dots, \ell_n\}$ repeats an *even* number of times in the length vector $(\ell_1, \ell_2, \dots, \ell_n)$ with $n \leq 32$. We conjecture that IGCAS finds a fix-free code for all such length vectors.

IV. CONCLUSIONS

The usefulness of the greedy codeword assignment scheme (GCAS) was demonstrated by proving some existence theorems. The capability of GCAS to find binary fix-free codes was examined and compared with the HK algorithm.

To make a fair comparison, the iterative version of GCAS (denoted by IGCAS) is proposed. Exploiting construction results obtained using IGCAS and HK, the existence of a fix-free code for the length vectors with Kraft sum $\frac{3}{4}$ and $n \leq 32$ was proven. Furthermore, implementation of of the algorithms leads to the conjecture that IGCAS finds fix-free codewords for all length vectors with Kraft sum not greater than $\frac{23}{32}$.

In summary, this paper provides further evidence on the validity of the $\frac{3}{4}$-conjecture.

REFERENCES

[1] R. Ahlswede, B. Balkenhol and L. Khachatrian, "Some properties of fix-free codes," *Proc. Int. Sem. on Coding Theory and Combinatorics,* Thahkadzor, Armenia, pp. 20-33, 1996.

[2] C.Ye and R. W.Yeung, "Some basic properties of fix-free codes," *IEEE Trans. Inform. Theory,* vol. 47, no. 1, pp. 72–87, Jan. 2001.

[3] S. A. Savari, "On optimal reversible-variable-length codes," *Proc. IEEE Int. Workshop Inf. Theory Appl.,* San Diego, CA, pp. 311–317, Feb. 2009.

[4] H. Schnettler, "On the $\frac{3}{4}$-conjecture for fix-free codes: A survey," [Online]. Available: //arxiv.org/PS-cache/arxiv/pdf/ 0709/0709.2598v1.pdf

[5] K. Harada and K. Kobayashi, "A note on the fixfree property," *IEICE Trans. Fundamentals*, vol. E82-A, no. 10, pp. 2121-2128, Oct. 1999.

[6] M. Khosravifard, H. Halabian and T. A. Gulliver, "A Kraft-type sufficient condition for the existence of D-ary fix-free codes," *IEEE Trans. Inform. Theory,* vol. 56, no. 6, pp. 2920–2927, June 2010.

[7] M. Khosravifard and T. A. Gulliver, "On the capability of the Harada-Kobayashi algorithm in finding fix-free codewords," *Proc. Int. Symp. on Inform. Theory and its Appl.,* pp. 1–4, Dec. 2008.

[8] S. Yekhanin, "Improved upper bound for the redundancy of fix-free codes," *IEEE Trans. Inform. Theory,*, vol. 50, no. 11, pp. 2815-2818, Nov. 2004.

[9] S. A. Savari, "On minimum-redundancy fix-free codes," *Proc. IEEE Data Compression Conference,* Snowbird, UT, pp. 3–12, Mar. 2009.

[10] N. Abedini, S. P. Khatri and S. A. Savari, "An algorithm for construction of efficient fix-free codes," *Proc. IEEE Data Compression Conference,* Snowbird, UT, pp. 169–178, Mar. 2010.

[11] Y. Takishima, M. Wada and H. Murakami, "Reversible variable length codes," *IEEE Trans. Commun.,* vol. 43, pp. 158–162, Feb.–Apr. 1995.

[12] C. W. Tsai and J. L. Wu, "On constructing the Huffman-code based reversible variable length codes," *IEEE Trans. Commun.,* vol. 49, no. 9, pp. 1506–1509, Sept. 2001.

[13] K. Lakovic and J. Villasenor, "An algorithm for construction of efficient fix-free codes," *IEEE Commun. Letters,* vol. 7, no. 8, pp. 391-393, Aug. 2003.

[14] S. Zaks, "Lexicographic generation of ordered trees," *Theoretical Computer Science*, vol. 10, pp. 63–82, 1980.

[15] M. Khosravifard, M. Esmaeili, H. Saidi and T. A. Gulliver, "A tree based algorithm for generating all possible binary compact codes with N codewords," *IEICE Trans. Fundamentals*, vol. E86-A, no. 10, pp. 2510–2516, Oct. 2003.