On the construction of statistically synchronizable codes *

R. M. Capocelli[1], *A. De Santis*[2,3], *L. Gargano*[2], and *U. Vaccaro*[2]

[1] Dipartimento di Matematica, Università di Roma, 00185 Roma, Italy

[2] Dipartimento di Informatica ed Applicazioni, Università di Salerno, 84081 Baronissi (SA), Italy

Abstract

We consider the problem of constructing statistically synchronizable codes over arbitrary alphabets and for any finite source. We show how to efficiently construct a statistically synchronizable code whose average codeword length is whithin the least likely codeword probability from that of the Huffman code for the same source.

Moreover, we give a method for constructing codes having a synchronizing codeword. The codes we construct present high synchronizing capability and low redundancy.

[3] Part of this work was done while visiting IBM T. J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York, 10598.

*This work was partially supported by the Italian Ministry of the University and Scientific Research, within the framework of the Project: Progetto ed Analisi di Algoritmi. Part of this work has been presented at the 1990 IEEE International Symposium on Information Theory, San Diego, CA, Jan. 1990.

1 Introduction

A basic problem in information transmission is to maintain the synchronization between the encoder and the decoder. An error occurring in the information stream as well as a decoder misfunctioning can lead to a loss of synchronization and consequently to an incorrect decoding of the code message. It would be then desirable to encode the source output in such a way that the decoder could easily recover the synchronization once it has been lost. In this paper we consider suboptimal codes that admit decoders able to self-synchronize with high probability provided the sequence of code symbols is long enough.

Let s be the number of code symbols that the decoder must observe to recover synchronization. Generally, s is a random variable depending both on the source and the code. If s is upper bounded by some finite constant S independent from the source, the code is called *synchronizable*, see for example [8], [10], [13], [14], [27] and references therein quoted. Unfortunately, the synchronizability property is an additional requirement for a code that leads to a loss of efficiency in terms of average codeword length. In fact, apart from trivial cases, no optimal binary code is synchronizable [27]. Therefore, it is interesting to consider codes that satisfy the weaker property of admitting decoders able to self-synchronize with probability approaching 1 as the length of the code message goes to infinity. Codes exhibiting this property are called *statistically synchronizable*.

Definition 1 [4] [28] *A code is called statistically synchronizable for a given source if*

$$\lim_{S\to\infty} Pr\{s \leq S\} = 1.$$

The statistical synchronizability property of the code implies the occurrence, in the input of the decoder, of a sequence of codewords which allows the decoder to recover synchronization [4-6], [15], [16], [18-27]. Such sequences are called *synchronizing*. The construction of statistically synchronizable codes having optimal average codeword length has been widely studied. For instance, Rudner [20] gave a method to construct binary optimal codes having shortest possible synchronizing sequence. Rudner's method, however, applies only to a limited class of probability distributions. Relations between the existence of a synchronizing sequence and the distribution of the codewords lengths in a code have been studied by Schützenberger [25].

An interesting particular case is when the synchronizing sequence is formed by a single codeword. Such codes, called *synchronous*, deserve attention for two main reasons: **a)** they allow always the decoder to recover synchronization; **b)** they have a small average synchronization delay. Ferguson and Rabinowitz [5] were the first to consider synchronous codes. They characterized classes of probability distributions that admit binary synchronous Huffman codes. Subsequently, Montgomery and Abrahams [17] gave a method to construct suboptimal synchronous binary codes for several classes of probability distributions.

In this paper we show that, when a source admits no statistically synchronizable optimal code, one can efficiently construct a statistically synchronizable code whose average

codeword length is within the least likely codeword probability from the optimal possible for that source.

Moreover, we give a method to construct synchronous codes for *arbitrary* code alphabet and for *any* finite source. The codes we obtain have two major advantages. First, the synchronizing performances of these codes often greatly surpass the synchronizing performances of any synchronous optimal code, as we show for several practical sources like natural languages. Second, their average codeword length is close to the optimal: we provide an upper bound which shows that their redundancy is less than $1 + 1/(e \cdot d \cdot \ln d)$, where d is the code alphabet size and e is the base of natural logarithms. It should be added that in the binary case our codes reduce to "1-ended codes", recently introduced by Berger and Yeung [2] and further analyzed by Capocelli and De Santis [3], providing thus a novel application of 1-ended codes.

The paper is organized as follows. In Section 2 we give the basic definitions. In Section 3 we provide an algorithm for constructing codes with synchronizing sequences. In Section 4 we give an efficient algorithm for constructing synchronous codes and then we provide the upper bound on the average codeword length.

2 Notations and definitions

Let X be a finite set of code symbols of cardinality d. Denote by X^n the set of all sequences obtained by concatenating n symbols of X. Let $X^+ = \bigcup_{n \geq 1} X^n$ be the set of all finite sequences of elements of X and $X^* = X^+ \cup \{\lambda\}$, λ denoting the empty sequence. We call the elements $w \in X^+$ *words* and denote by $l(w)$ their lengths, i.e., if $w \in X^n$, we have $l(w) = n$. A finite subset C of X^+ is a *code*, its elements are called codewords.

Given a word $w \in X^+$, if $w = ps$ then $p \in X^*$ is called a *prefix* of w and $s \in X^*$ is called a *suffix* of w. Given a code C, let $Suffix(C)$ be the set of all suffixes of codewords in C and $Prefix(C)$ be the set of all prefixes of codewords in C. A code is called *prefix* if no codeword is prefix of any other codeword. A prefix code C is maximal if for each $x \in X^+$, the code $C \cup \{x\}$ is not prefix.

Given a word $w \in X^+$ and a positive integer r, with w^r we denote the word obtained by concatenating r copies of w. A *code message* $w_1 \dots w_k$ is the word obtained by the concatenation of the codewords $w_i \in C$, for $i = 1, \dots, k$.

Given a code C, we define $L_C = \max\{l(w) | w \in C\}$. When no ambiguity is possible we shall omit the subscript C. A *length vector* $\alpha = (\alpha_1, \dots, \alpha_L)$ of a code C is a vector such that L is the maximum length of any codeword of C and α_i is the number of codewords with length i in C, for $i = 1, \dots, L$. Two codes are *equivalent* if they have the same length vector.

Given a probability distribution $(p_1, \dots, p_n)$ and a code $C = \{w_1, \dots, w_n\}$, where p_i is the probability associated to the codeword w_i, for $i = 1, \dots, n$, the average codeword length of C is defined as $E(C) = \sum_{i=1}^{n} p_i l(w_i)$. If $E(C)$ is minimum among all prefix codes for the probability distribution $(p_1, \dots, p_n)$ then C is called *optimal*. The redundancy r of

C is defined as $r = E(C) - H$, where $H = H(p_1, \ldots, p_n) = -\sum p_i \log p_i$ [1] is the entropy of the distribution $(p_1, \ldots, p_n)$.

3 Statistically synchronizable codes

In this section we consider the problem of efficiently constructing statistically synchronizable prefix codes with average codeword length within p_n from the minimum possible for the same source.

Throughout this section, we assume that the information source is a stationary random process $\ldots U_{t-1}U_tU_{t+1}\ldots$, with the property that there exists a real number $\epsilon > 0$ such that

$$Pr\{U_t = a_t\} \geq \epsilon \text{ and } Pr\{U_t = a_t | U_{t-1} = a_{t-1}, \ldots, U_{t-j} = a_{t-j}\} \geq \epsilon$$

for all integers t and j and source symbols $a_{t-j}, \ldots, a_t$. Such a source is called *ϵ–guaranteed* [4], [28]. If a source is ϵ–guaranteed and we encode its output with codewords of a code C, then the probability that a code message contains a given sequence of codewords α approaches 1 as the length of the code message increases, for every sequence α. This property is useful in that it makes the statistical synchronizability depend only on combinatorial properties of the code and not on the source.

Definition 2 *Given a prefix code C, a code message $w_1 \ldots w_k \in C^+$ is a* synchronizing sequence *for C if there are no $p, q \in Prefix(C) - C$ and $v_1, \ldots, v_r \in C$, $r \geq 0$, such that*

$$pw_1 \ldots w_k = v_1 \ldots v_r q.$$

We stress that the above definition is the simplified version, for prefix codes, of the synchronizing sequence introduced in [4]. The following result holds [4], [28].

Theorem 1 *Let C be a code used to encode an ϵ–guaranteed source. The code C is statistically synchronizable if and only if it has at least a synchronizing sequence.*

Schützenberger first afforded the problem of the construction of statistically synchronizable maximal prefix codes (or synchronizing codes, in his terminology) and proved the following result [25].

Theorem 2 *There exists a d-ary statistically synchronizable maximal prefix code with length vector $\alpha = (\alpha_1, \ldots, \alpha_L)$ if and only if the following two conditions hold*

1. $\sum_{i=1}^{L} \alpha_i d^{-i} = 1$ (condition for maximality), *and*
2. $\gcd\{i \mid \alpha_i > 0,\ i = 1, \ldots, L\} = 1$.

[1] Unless otherwise specified, the log's are in base d.

Schützenberger gave also a method to construct statistically synchronizing codes with length vector satisfying above Theorem 2. The following theorem provides the simplest construction method [1] for such a codes.

Theorem 3 *Let $C \subseteq X^+$ be a maximal prefix code. If the lengths of the codewords formed by all a's, $a \in X$, are relatively prime, then C has at least a synchronizing sequence.*

Therefore, a statistically synchronizable maximal prefix code can be obtained by constructing a code in which there exist two words a^i and b^j with i and j relatively prime and $a, b \in X$.

The problem of constructing statistically synchronizable codes with minimum average codeword length is thus open if: **i)** the length vector of any optimal code has greatest common divisor greater than one; **ii)** optimal codes are not maximal codes (this happens when $d > 2$ and the number of codewords is different from $d + h(d-1)$, for any integer h). We show that also in these cases it is possible to construct a statistically synchronizable code at the expenses of only a small increase on the optimal average codeword length.

Without loss of generality we suppose that, if the size of X is d then the elements of X are the integers $0, \ldots, d-1$. The following theorem shows how to construct such a code from an optimal code.

Theorem 4 *Let $P = (p_1, \ldots, p_n)$ be a probability distribution with $p_i \geq p_{i+1}$, $i = 1, \ldots n-1$, C an optimal d–ary prefix code for P, and $\alpha = (\alpha_1, \ldots, \alpha_L)$ its length vector. Then there exists a d-ary statistically synchronizable prefix code D having length vector $\beta = (\beta_1, \ldots, \beta_{L+1})$ with $\beta_i = \alpha_i$, for $i = 1, \ldots, L-1$, $\beta_L = \alpha_L - 1$, $\beta_{L+1} = 1$, and average codeword length $E(D) = E(C) + p_n$.*

Proof. Given the optimal code C, construct a code B equivalent to C and containing the word 0^L. Since B is also optimal, there is a word $0^{L-1}a \in B$ for some $a \neq 0$ (otherwise the code $B - \{0^L\} \cup \{0^{L-1}\}$ would be a prefix code having average codeword length smaller than that of the optimal code C). Consider the code $D = B - \{0^L\} \cup \{0^L a\}$ obtained appending the code symbol a to 0^L. We show now that the sequence $(0^L a)^{L-1}$ is synchronizing for D. To this aim, we shall prove that for each $p \in Prefix(D)$ it holds $p(0^L a)^{L-1} \notin Prefix(D^+) - D^+$. For that, since $0^L \notin Suffix(D)$, it is sufficient to prove that for each integer $i = 1, \ldots, L-1$, it holds

$$0^{L-i}a(0^L a)^i \notin Prefix(D^+) - D^+. \tag{1}$$

The proof of (1) is by induction. For $i = 1$, $0^{L-1}a \in D$ and (1) holds. Suppose that (1) holds for each integer $\leq i-1$ and consider $0^{L-i}a(0^L a)^i$. If there exists no $w \in D$ prefix of $0^{L-i}a(0^L a)^i$, then (1) holds; if such a w exists then $w = 0^{L-i}a0^j$, for some $0 \leq j \leq i-1$. For $j = 0$, we obtain $(0^{L-i}a)(0^L a)^i \in D^+$ and (1) holds. For $j > 0$, we obtain

$$0^{L-i}a(0^L a)^i = 0^{L-i}a(0^j 0^{L-j}a)(0^L a)^{i-1} = (0^{L-i}a0^j)(0^{L-j}a)(0^L a)^j(0^L a)^{i-1-j}.$$

By using the inductive hypothesis on j relation (1) follows.

Assigning longer codewords of D to smaller probabilities gives $E(D) = E(C) + p_n$.

□

Statistically synchronizable codes constructed according to the method outlined in the proof of Theorem 4 are not maximal. We stress that maximality is a very desirable feature in coding theory. Indeed, if the used code is maximal and statistically synchronizable, *any* decoder will recover synchronization as soon as it reads a synchronizing sequence [27] while, if the code is not maximal, a specially designed decoder is needed [4].

In the next theorem we will show that in the binary case it is possible to construct statistically synchronizable maximal codes suitably modifying a *binary* optimal code. Notice that since binary optimal codes are maximal, because of Theorem 2 we need only to consider the case in which the given source admits only optimal codes with codeword lengths having greatest common divisor greater than or equal to two.

Theorem 5 *Let $P = (p_1, \ldots, p_n)$ be a probability distribution, with $p_i \geq p_{i+1}$, $i = 1, \ldots n-1$, C be an optimal binary prefix code for P with length vector $\alpha = (\alpha_1, \ldots, \alpha_L)$ such that $\gcd\{i \mid \alpha_i > 0,\ i = 1, \ldots, L\} \geq 2$. Then there exists a binary statistically synchronizable maximal prefix code with average codeword length $\leq E(C) + p_n$ and length vector $\beta = (\beta_1, \ldots, \beta_{L+1}) = (\alpha_1, \ldots, \alpha_{L-2}, 1, \alpha_L - 3, 2)$.*

Proof. First notice that $\alpha_{L-1} = 0$ since $\gcd\{i \mid \alpha_i > 0,\ i = 1, \ldots, L\} \geq 2$ and $\alpha_L > 0$. This, together with the maximality of C, implies $\alpha_L \geq 4$.
To show that β is the length vector of a statistically synchronizable maximal prefix code, it is enough to prove that conditions *1.* and *2.* of Theorem 2 hold for β. Condition *2.* is true since words of length $L-1$ and L exist. To verify condition *1.* merely compute

$$\sum_{i=1}^{L+1} \beta_i 2^{-i} = \sum_{i=1}^{L-2} \alpha_i 2^{-i} + 2^{-(L-1)} + (\alpha_L - 3)2^{-L} + 2 \cdot 2^{-(L+1)} = \sum_{i=1}^{L-2} \alpha_i 2^{-i} + \alpha_L 2^{-L} = 1.$$

From Theorem 2 there exists a maximal statistically synchronizable code D having β as length vector. Recalling that, in an optimal code, longer codewords are associated to smaller probabilities, we immediately get that there exists an encoding of P by D such that $E(D) - E(C) = p_n + p_{n-1} - p_{n-2} \leq p_n$. □

4 d-ary synchronous codes

In the previous section we have shown how to construct codes with a synchronizing sequence whose average codeword length is within p_n from that of the Huffman code for the same source. Synchronizing sequences containing more than one codeword may have two drawbacks. First, they assure statistically synchronizability only when the source sequences associated to the synchronizing sequences has nonzero probability of occurrence, for instance, when the source is is ϵ–guaranteed [4] [28]. Second, the average decoder synchronization delay, that is the average number of code symbols before the occurrence of a synchronizing sequence, increases with the number of codewords in the sequence. These arguments have motivated researchers to look for codes having a synchronizing codeword,

i.e., *synchronous codes*. Ferguson and Rabinowitz [5] gave sufficient conditions for a source to admit of a binary synchronous Huffman code. Subsequently, Montgomery and Abrahams [17] have developed a procedure for constructing suboptimal synchronous binary codes for gapless sources, i.e, sources whose Huffman code length vector $(\alpha_1, \ldots, \alpha_L)$ has the property that $\alpha_j \neq 0$ implies $\alpha_i \neq 0$ for all $j \leq i \leq L$. They have also shown that their algorithm may be adapted to apply to classes of gapped sources.

In this section we give an efficient algorithm to construct d-ary synchronous codes. Our algorithm can be applied to any source. Moreover, our algorithm works for any cardinality d of the code alphabet. We give an upper bound on the redundancy of the codes which shows that their average codeword length is close to the optimal. The synchronizing performances of our codes often greatly surpasses the performances of the previous proposed synchronous codes.

We first consider the easy case in which the source has an optimal code with minimum codeword length 1. We show that such a source admits an optimal synchronous code. This result has been proved in [5] for binary codes. The proof in the d–ary case is similar, however we report it for the reader's convenience.

Theorem 6 *If $\alpha = (\alpha_1, \ldots, \alpha_L)$, with $\alpha_1 > 0$, is the length vector of an optimal code then there exists an equivalent synchronous code.*

Proof. Let C be a prefix d-ary code with length vector α and let a be a codeword of length 1. We first prove that there exists an equivalent code C' having a as codeword and with the property that if a^i is the longest string of a's in a codeword of C' then a^i can be only a suffix of a codeword, that is, for each $x \in X^*$

$$xa^iy \in C' \quad \text{implies} \quad y = \lambda. \tag{2}$$

Assume C does not satisfy above property, that is there exists $xa^iy \in C$ with $y \neq \lambda$. If $l(y) \geq 2$, from the optimality of C we get that there exists $xa^ia \in Prefix(C)$, contradicting the assumption that i is the maximum number of consecutive a's in a codeword. If $y \in X$, the code C' obtained from C changing xa^iy into xa^ia is equivalent to C and satisfies (2).

We show now that each codeword $xa^i \in C'$ is synchronizing. By Definition 2, it is enough to prove that if for some $p, q \in Prefix(C)$ and $v_1, \ldots, v_r \in C'$ it holds

$$pxa^i = v_1 \ldots v_r q$$

then $q \in C'^+$.

If $v_1 \ldots v_r$ is a prefix of px, we get that $q = x'a^i$, for some $x' \in X^*$, which from (2) implies $q \in C$. If px is a prefix of $v_1 \ldots v_r$ we get, for some j, that $q = a^j$ which, since $a \in C$, implies $q \in C^+$. ∎

Example 1 *A 4-ary optimal synchronous code for the Latin alphabet, constructed along the line of the previous theorem, is given in Table 1. The frequencies are taken from* [12]. *According to Theorem 6 all codewords with suffix 3 are synchronizing. We have considered several cases, experimental results have shown that the construction presented in Theorem 6 is likely to produce more synchronizing codewords than those guaranteed by its proof. In this case it is easy to see that also the codewords* 101,*and* 102, *are synchronizing. The sum of the frequencies of the letters corresponding to synchronizing words is* 421/831 *which corresponds to a synchronization within an average of about* 1.97 *codewords.*

We consider now the construction of synchronous codes for general sources. Let us define the following class of prefix codes.

Definition 3 *A d–ary* feasible *code is a prefix code in which no codeword has* 0 *as last letter.*

Berger and Yeung [2] introduced and gave an analysis of feasible codes for the case $d = 2$. They named these codes "1–ended", since the requirement of the definition forces the codewords to end with 1. Subsequently, Capocelli and De Santis [3] improved some of the results given in [2]. [2] and [3] do not consider the case of arbitrary d.

Definition 4 *Let C be a code with maximum codeword length L. A codeword of the form $0^{L-1}x$, where $x \in X - \{0\}$, is called a* s–word.

The term s–word is due to the fact that s–words of feasible codes are synchronizing codewords, as the following theorem shows.

Theorem 7 *A s–word of a feasible code is a synchronizing codeword.*

Proof. Let C be a feasible code with maximum length L. Let $0^{L-1}a$ be a s–word in C. Suppose by contradiction that $0^{L-1}a$ is not a synchronizing word. By definition, there exists $p \in Prefix(C)$ such that $p0^{L-1}a = v_1 \dots v_r q$ for some $v_1, \dots, v_r \in C$ and $q \in Prefix(C) - C$. Since L is the maximum length, we get that $r > 0$ and $v_1 = p0^i$ for some i, which contradicts the definition of a feasible code. □

Notice that, in general, a feasible code with a s–word is likely to have many synchronizing words that are not s-words. For instance, with the same technique used in Theorem 7, it is possible to show that all the codewords $0^{L-2}a$, with $a \neq 0$, and $a0^{L-2}b$, with $a, b \neq 0$, if any, are synchronizing.

Example 2 *Consider the binary feasible code* $C_1 = \{0001, 001, 0101, 011, 1001, 101, 11\}$. *The word* 0001 *is a s–word and* 0001, 001, 1001 *are synchronizing codewords. Consider the ternary feasible code*

$$C_2 = \{w_1, \dots, w_{13}\} = \{00001, 0001, 0002, 001, 002, 01, 02, 1001, 101, 102, 201, 21, 22\}.$$

The word 00001 *is a s-word and* $w_1, \dots, w_{11}$ *are synchronizing for* C_2.

For a feasible code, to have a s–word is not a strong requirement. Indeed, from *any* feasible code it is possible to derive an equivalent feasible code having s–words.

Theorem 8 *Let C be a feasible code. Then there exists a feasible code equivalent to C which has at least one s–word.*

Proof. Let C be a feasible code of maximum length L which has no s–words. Let $0^j ay$, where $a \in X - \{0\}$, $y \in X^+$, be a codeword of C of length L. Consider the code C' constructed from C by changing codewords $0^j au$ into $0^{j+1}u$ and codewords $0^{j+1}u$ into $0^j au$ and leaving the other codewords unchanged. It is easy to see that C' is a feasible code equivalent to C. By iterating above transformation (at most $l(y)$ times) we end up with a feasible code having at least one s–word. □

Notice that above proof of the theorem is constructive and provides a procedure that can be implemented to run in time $O(L)$. Since for optimum feasible codes $L \leq |C| + 1$ the time complexity is $O(|C|)$. Because of Theorems 7 and 8, an efficient method for constructing feasible codes translates immediately into an efficient method to construct synchronous codes. Notice that there is no guarantee on the optimality of the codes. However, we shall see that it is possible to construct codes with redundancy very close to 1. We trade a small increment on the average codeword length for the additional property of having a synchronizing codeword. To date, there is no efficient algorithm to construct optimal feasible (in particular 1-ended) codes. It is not even known whether a polynomial time algorithm exists [2],[3]. If we relax the requirement of optimality, we can efficiently construct almost optimal feasible codes. Indeed, there is a simple strategy for the construction of feasible codes whith average codeword length close to the optimal.

Let $P = \{p_1, \ldots, p_n\}$ be a probability distribution with entropy $H = H(p_1, \ldots, p_n) = -\sum p_i \log p_i$. The encoding of P by its Huffman code C specifies an association of codewords to source letters. The probability of $c \in C$ is given by $Pr(c) = p_s$, if c is associated with the source letter s whose probability is p_s. For each $p \in Prefix(C)$ define $Pr(p)$ as the sum of probabilities $Pr(c)$ of all codewords in $\{c : p \text{ is a prefix of } c\}$. Let L_C be the maximum length of codewords in C, and for $y \in Prefix(C)$ let $C_y = \{w \mid yw \in C\}$ be the code consisting of the suffixes of elements of C with prefix y.

Definition 5 *Let $a < b$ be two code letters and P be a probability distribution. A Huffman code C for P is called* LS–code *(for* Left Skewed *code) if all the following conditions are met*

a) $xa \in C$ implies $xb \in C$;

b) $xa, xb \in Prefix(C)$ implies $L_{C_{xa}} \geq L_{C_{xb}}$;

c) $xa, xb \in Prefix(C)$ and $L_{C_{xa}} = L_{C_{xb}}$ imply $Pr(xa) \leq Pr(xb)$.

It is easy to see that any probability distribution P has at least an LS–code. Let us now investigate the complexity of constructing LS–codes.

Recall that the Huffman procedure is bottom-up, and it is usually visualized as the construction of a tree. The n probabilities of the distribution are initially assigned to nodes, which constitute the initial set of subtrees. Then the roots of subtrees with smallest probabilities are made children of the same node, whose probability becomes the sum of those of the children. At this stage code letters are assigned to the links entering the parent. The procedure is iterated until only one root is left. Details can be found in any textbook (see [6], for example).

To construct LS–codes, we modify the above algorithm by assigning in each merging code letters according to Definition 5. More formally consider the following procedure. Let n be the number of source symbols and assume $n = k(d-1)+1$, for an integer k. The extension to arbitrary n is immediate.

1. Assign each probability of P to a single node.

2. Let $q_1 \leq q_2 \leq \ldots \leq q_d$ be the d smallest probabilities among the roots of the subtrees constructed until now, $r_1, r_2, \ldots, r_d$ be the corresponding roots, and let h_i be the height of the subtree rooted at r_i.

   Make all the r_i's children of the same node r. Label the links between the r_i's and r by assigning letters $0, 1, \ldots, d-1$ in order of decreasing height h_i's. (If two subtrees have the same height, assign the smaller label to the link whose endpoint is the least likely root.) Set $q_1 + q_2 + \ldots + q_d$ as probability of the new root r.

3. Repeat Step 2 until only one root is left.

It is easy to verify that the above procedure constructs a LS–code. The complexity of the algorithm does not increase significantly with respect to the usual Huffman codes construction. For a fixed alphabet the procedure can be implemented to run in time $O(n \log n)$.

Let P_a be the sum of probabilities of letters encoded by codewords ending with a.

Lemma 1 *For any LS–code, it holds* $P_0 \leq P_1 \leq \ldots \leq P_{d-1}$.

Proof. For each $a \in X$, let $\Pi(a) = \{x \mid xa \in C\}$. Then the probability P_a can be written as

$$P_a = \sum_{x \in \Pi(a)} Pr(xa).$$

Consider two code symbols $i < j$. By Property a) of LS–codes we have $\Pi(i) \subseteq \Pi(j)$. Moreover, for each $x \in \Pi(i)$, by Property c) of LS–codes we get $Pr(xi) \leq Pr(xj)$. Hence the lemma. ∎

Definition 6 *Let C be a LS–code for a probability distribution P. The* augmented *code of C is the code obtained by appending the code symbol 1 to codewords in C which have 0 as last symbol.*

The following lemma is immediate from above definition, since for an LS–code C with maximum length L we have $0^{L-1}(d-1) \in C$, and if $0^L \in C$ then $0^L 1$ is in the augmented code.

Lemma 2 *Any augmented code is feasible and has at least a s–word.*

Example 3 *Consider the English alphabet. The frequencies of letters and space (as given in* [9]*) and the binary augmented code C are given in Table 2. C has 14 synchronizing codewords (marked by $*$ in the table). The average codeword length of the Huffman code is* 4.1195*. The average codeword length of C is* 4.2965*. The sum of the probabilities of letters encoded with a synchronizing word is* 0.2193*, which corresponds to a synchronization within* 4.6 *codewords, on average. It is possible to improve the synchronizing performances of the code by associating more likely source letters to shorter codewords, and among codewords of same length associating more likely source letters to synchronizing codewords. In our example the resulting encoding has still average codeword length* 4.2965 *but the sum of the probabilities of synchronizing codewords raises to* 0.2923*, which corresponds to a resynchronization within* 3.42 *codewords, on average.*

From Ferguson and Rabinowitz [5] one has that there exists a synchronous Huffman code, but in this code only the longest codeword is synchronizing. Applying the method of Montgomery and Abrahams [17] we get a code with average codeword length equal to $4.1195 + Pr(z) = 4.1200$, but only the 8 longest codewords are synchronizing and the sum of the probabilities of synchronizing words is only 0.0695*.*

Example 4 *Consider the probability distribution* $(1/3, 1/8, 1/8, 1/8, 1/8, 1/12, 1/12)$*. The only length vector corresponding to an optimal binary code is* $(0, 1, 6)$*. There is no binary synchronous code with length vector* $(0, 1, 6)$ [5]*. The average codeword length of Huffman code is* 3*. The augmented code is the code C_1 of Example 2, which has average codeword length* 3.3*. The sum of the probabilities of synchronizing codewords is equal to* $1/4$*, which corresponds to resynchronization every 4 codewords, on average. Applying the method in* [17] *one has an increment of* $1/12$ *on the average codeword length, but only the longest codeword, having probability* $1/12$*, is synchronizing.*

Example 5 *The ternary augmented code D for the Italian alphabet with the frequencies of* [12] *is given in Table 3. The code D has 20 synchronizing codewords (marked by $*$). The average codeword length of the Huffman code is* 2.559*. The average codeword length of the augmented code is* 2.673*, while the sum of the probabilities of the synchronizing words is* 0.655 *which corresponds to synchronization within an average of about* 1.5 *codewords. As in Example 3, a rearrangement of the correspondence between source letters and codewords increases these numbers to* 0.780 *and* 1.28*, respectively, and preserving the average codeword length of* 2.673*.*

The following lemma tells us that the optimal average codeword length of augmented codes is always within $1/d$ to that of the Huffman code for the same source.

Lemma 3 *Let P be a probability distribution and E_h be the average codeword length of the corresponding Huffman code. Then the average codeword length E of the augmented code satisfies*

$$E \leq E_h + 1/d.$$

Proof. By construction, the average codeword length of the augmented code is $E = E_h + P_0$. By Lemma 1 we find $P_0 \leq P_1 \leq \ldots \leq P_{d-1}$. Since $\sum_{j=0}^{d-1} P_j = 1$, it follows $P_0 \leq 1/d$. □

We prove now an upper bound on the average codeword length of augmented codes.

Theorem 9 *Let $P = (p_1, \ldots, p_n)$ be a probability distribution with entropy H. Then the average codeword length E of the d–ary, $d \geq 3$, augmented code satisfies*

$$E \leq H + 1 + \frac{1}{d} - \frac{d}{d(d-1)+1}\left(1 - \frac{1}{e \cdot \ln d}\right). \tag{3}$$

Before proving the Theorem, we report hereafter some known results that will be useful in the proof (see [7], [15]).

Property 1. If q_a and q_b are the probabilities of nodes a and b at the same level in a d–ary Huffman code tree, and b is not a leaf, then $dq_a \geq q_b$.

Property 2. Let P be a probability distribution with entropy H. The redundancy $r = E_h - H$ of a d–ary Huffman code satisfies

$$r \leq l - H(q_1, q_2, \ldots, q_{d^l}) + \frac{q'd}{e \cdot \ln d}$$

where l is some level at which the code tree is full, $q_1 \geq q_2 \geq \ldots \geq q_{d^l}$ are the probabilities of the d^l nodes at level l, and q' is the probability of the most likely node at level $l+1$.

Proof of Theorem 9. Let $C = \{c_1, \ldots, c_n\}$ be a LS–code for P, and E_h be its average codeword length. Denote by C' the augmented code obtained by adding 1 to codewords of C ending with 0.

First assume that $n < d$. Thus each codeword provided by the Huffman procedure consists of a single symbol. Any Huffman code for P satisfying Property a) of LS–codes, cannot have 0 as codeword. Thus, the augmented code has $E = E_h = 1$.

Assume now $n = d$. The augmented code is $\{01, 1, \ldots, (d-1)\}$ and its average codeword length E satisfies

$$E - H = 1 - H(p_1, p_2, \ldots, p_d) + q,$$

where $q = \min p_i$. Gallager [7] proved that

$$H(x_1, x_2, \ldots, x_d) \geq dx \tag{4}$$

where $x = \min x_i$ and $\sum x_i = 1$. Applying (4) we find that

$$E - H \leq 1 - q(d-1)$$

and hence the theorem in the case $n = d$.

Finally consider the case $n > d$.

Let α be the length vector of C. Let q_i, $i = 0, 1, \ldots, d-1$, be the probability of occurrence of the code symbol i as prefix in C, i.e. $q_i = \sum_{c_j \in iX^*} p_j$. Let S_i be the probability distribution obtained by considering all probabilities in P which have assigned a codeword $c \in C \cap iX^*$, and then normalizing (dividing by q_i).

The average codeword length E_h of encoding P by C can be written as

$$E_h = 1 + \sum_{i=0}^{d-\alpha_1-1} q_i E(C_i) \tag{5}$$

where $E(C_i)$ is the average codeword length of C_i if one encodes S_i with C_i. $E(C_i)$ is also equal to the average codeword length of the Huffman code for S_i. Let C_i' be the code augmented from C_i.

The average codeword length E of the augmented code can be written as

$$E = 1 + \sum_{i=0}^{d-\alpha_1-1} q_i E(C_i').$$

Applying Lemma 3 we get

$$E \leq 1 + \sum_{i=0}^{d-\alpha_1-1} q_i \Big(E(C_i) + 1/d \Big)$$

and using (5) we have

$$E \leq E_h + \sum_{i=0}^{d-\alpha_1-1} \frac{q_i}{d}.$$

Making use of Property 2 with $l = 1$ we obtain

$$E \leq H + 1 - H(q_0, q_1, \ldots, q_{d-1}) + \frac{q'd}{e \cdot \ln d} + \sum_{i=0}^{d-\alpha_1-1} \frac{q_i}{d}.$$

Applying inequality (4) we get

$$E \leq H + 1 - dq + \frac{q'd}{e \cdot \ln d} + \sum_{i=0}^{d-\alpha_1-1} \frac{q_i}{d},$$

where q denotes the minimum among the q_i's. Since q' and q are probabilities of nodes at level two and one, respectively, we have $q \geq q'$. Hence

$$E \leq H + 1 - dq \left(1 - \frac{1}{e \cdot \ln d} \right) + \sum_{i=0}^{d-\alpha_1-1} \frac{q_i}{d}. \tag{6}$$

Assume now that $\alpha_1 \geq 1$. From Property 1 we have $q \geq q_i/d$, for $i = 0, \ldots, d-\alpha_1-1$, that summing over i yields $(d-\alpha_1)q \geq (\sum_{i=0}^{d-\alpha_1-1} q_i)/d$ and consequently $dq \geq (\sum_{i=0}^{d-\alpha_1-1} q_i)/(d-1)$. Substituting this into (6) leads to

$$E \leq H + 1 - \sum_{i=0}^{d-\alpha_1-1} \frac{q_i}{d-1}\left(1 - \frac{1}{e \cdot \ln d}\right) + \sum_{i=0}^{d-\alpha_1-1} \frac{q_i}{d}.$$

Using the fact that $\sum_{i=0}^{d-\alpha_1-1} q_i \leq 1$ we have

$$E \leq H + 1 - \frac{1}{d-1}\left(1 - \frac{1}{e \cdot \ln d}\right) + \frac{1}{d}. \tag{7}$$

Finally, consider the case $\alpha_1 = 0$. Then $\sum_{i=0}^{d-1} q_i = 1$ and (6) can be written as

$$E \leq H + 1 - dq\left(1 - \frac{1}{e \cdot \ln d}\right) + \frac{1}{d}. \tag{8}$$

Let q, the minimum probability at level one, be q_k. From Property 1 we have that q satisfies $q \geq q_j/d$, $j = 0, 1, \ldots, d-1$, that summing over j, $j \neq k$, yields $(d-1)q \geq \sum_{j \neq k} q_j/d = (1-q)/d$ and consequently

$$q \geq \frac{1}{d(d-1)+1}.$$

Substituting this into (8) yields

$$E \leq H + 1 - \frac{d}{d(d-1)+1}\left(1 - \frac{1}{e \cdot \ln d}\right) + \frac{1}{d}. \tag{9}$$

Taking the maximum between the two bounds (7) and (9), obtained for the cases $\alpha_1 \geq 1$ and $\alpha_1 = 0$ respectively, gives (3) and then the theorem. □

For example, the bound gives $E \leq H + 1.048$ for $d = 3$, $E \leq H + 1.023$ for $d = 4$, and $E \leq H + 1.016$ for $d = 5$.

The bound provided by above theorem is rather cumbersome. A simpler (but also weaker) bound is the following:

$$E \leq H + 1 + \frac{1}{e \cdot d \cdot \ln d}.$$

Its validity can be directly derived by the expression (3) or, alternatively, by using $q \geq \sum_{i=0}^{d-\alpha_1-1} q_i/d^2$ and then $\sum_{i=0}^{d-\alpha_1-1} q_i \leq 1$ in the inequality (6).

Capocelli and De Santis [3] proved that in the binary case ($d = 2$) the average codeword length of augmented codes satisfies the following stronger bound

$$E < H + 1.$$

We conjecture that above bound holds in the d–ary case, as well.

5 Conclusions

In this paper we have considered the construction of almost–optimal statistically synchronizable codes for any probability distribution and any size of the code alphabet. We gave efficient methods to construct both codes with a synchronizing sequence and codes with a synchronizing codeword.

In the former case we have shown that a code with a synchronizing sequence can be always constructed with an increase, with respect to the average codeword length of the Huffman code, of at most p_n (the probability of the least likely letter).

In the latter case we have considered the construction of synchronous codes over arbitrary alphabets. The method applies to any probability distribution. In the binary case, the comparison with [17], when it applies, shows that our codes trade a small redundancy for a smaller average synchronization delay.

References

[1] J. Berstel and D. Perrin, *Theory of codes*, Academic Press, 1985.

[2] T. Berger and R. Yeung, "Optimum '1'-ended binary prefix codes", *IEEE Trans. Inform. Theory*, vol. IT-**36**, n. 6, pp. 1435–1441, Nov. 1990.

[3] R. M. Capocelli and A. De Santis, "'1'-ended binary prefix codes", 1990 IEEE International Symposium on Information Theory, San Diego, CA, Jan. 1990.

[4] R.M. Capocelli, L. Gargano, and U. Vaccaro, "On the characterization of statistically synchronizable variable-length codes," *IEEE Trans. Inform. Theory*, vol. IT-**34**, pp. 817–825, July 1988.

[5] T.J. Ferguson and J.H. Rabinowitz, "Self-synchronizing Huffman codes," *IEEE Trans. Inform. Theory*, vol. IT-**30**, pp. 687–693, July 1984.

[6] R. G. Gallager, *Information Theory and Reliable Communication*, New York: Wiley, 1968.

[7] R. G. Gallager, "Variation on a theme by Huffman," *IEEE Trans. Inform. Theory*, vol. IT-**24**, n. 6, pp. 668–674, Nov. 1978.

[8] E.N. Gilbert, "Synchronization of binary messages," *IRE Trans. Inform. Theory*, vol. IT-**6**, pp. 470–477, Sept. 1960.

[9] E.N. Gilbert and E.F. Moore, "Variable length binary encodings," *Bell Syst. Tech. J.*, vol. **38**, pp. 933–967, 1960.

[10] S.W. Golomb and B. Gordon, "Codes with bounded synchronization delay," *Inform. Contr.*, vol. **8**, pp. 355–372, 1965.

[11] D. A. Huffman, "A method for the construction of minimum redundancy codes," *Proc. IRE*, **40**, n. 2, 1098–1101, Sept. 1952.

[12] A. Lange and E.A. Soudart, *Treatise on cryptography*, Aegean Park Press, 1981.

[13] V. I. Levenshtein, "Certain properties of code systems," *Soviet. Phis. Doklady*, vol. **6**, pp. 858–860, 1962.

[14] V. I. Levenshtein, "Some properties of coding and self adjusting automata for decoding messages," *Probl. Kibernetiki*, vol. **11**, pp. 63–121, 1964.

[15] D. Manstetten, "Tight bounds on the redundancy of Huffman codes," *IEEE Trans. Inform. Theory*, to appear.

[16] J. C. Maxted and J. P. Robinson, "Error recovery for variable length codes," *IEEE Trans. Inform. Theory*, vol. IT-**31**, n. 6, pp. 794–801, Nov. 1985.

[17] B. L. Montgomery and J. Abrahams, "Synchronization of binary source codes," *IEEE Trans. Inform. Theory*, vol. IT-**32**, pp. 851–854, Nov. 1986.

[18] P.G. Neumann, "Efficient error-limiting variable-length codes," *IRE Trans. Inform. Theory*, vol. IT-**8**, pp. 292–304, July 1962.

[19] D. Perrin, "Codes asynchrones," *Bull. Soc. Math. France*, vol. **105**, pp. 385–404, 1977.

[20] B. Rudner, "Construction of minimum-redundancy codes with an optimum synchronizing Property," *IEEE Trans. Inform. Theory*, vol. IT-**17**, pp. 478–487, July 1971.

[21] R.A. Sholtz and R.M. Storwick, "Block codes for statistical synchronization," *IEEE Trans. Inform. Theory*, vol. IT-**16**, pp. 432–438, July 1970.

[22] R.A. Sholtz and R.L. Welch, "Mechanization of codes with bounded synchronization delay," *IEEE Trans. Inform. Theory*, vol. IT-**16**, pp. 438–446, July 1970.

[23] M.P. Schützenberger, "On the synchronizing properties of certain prefix codes," *Inform. Contr*, vol. **7**, pp. 23–36, 1964.

[24] M.P. Schützenberger, "On an application of semigroup methods to some problems in coding," *IRE Trans. Inform. Theory*, vol. **2**, pp. 47–60, Sept. 1965.

[25] M.P. Schützenberger, "On synchronizing prefix codes," *Inform. Contr*, vol. **11**, pp. 396–401, 1967.

[26] L.R. Stanfel, "Mathematical optimization and the synchronizing properties of encodings," *Information and Computation*, vol. **77**, pp. 57–76, 1988.

[27] J.J. Stiffler, *Theory of synchronous communications*, Prentice Hall, Englewood Cliffs, NJ, 1971.

[28] V.K.W. Wei and R.A. Sholtz, "On the characterization of statistically synchronizable codes," *IEEE Trans. Inform. Theory*, vol. IT-**26**, pp. 733–735, Nov. 1980.

Letter	Frequency×851	Codeword
h	5	0000
x	6	0001
v	7	0002
f	9	100
b	12	001
q	13	002
g	14	*003
d	17	*101
l	21	*102
p	30	*103
c	33	11
m	34	12
o	44	01
n	60	02
r	67	20
s	68	21
a	72	22
t	72	*03
u	74	*13
e	92	*23
i	101	*3

Table 1

Letter	Frequency×1000	Codeword
space	1859	011
e	1031	111
t	796	0101
a	642	*0001
o	632	0011
i	575	1101
n	574	11001
s	514	1011
r	484	10101
h	467	1001
l	321	*00001
d	317	00101
u	228	*10001
c	218	*100001
f	208	010011
m	198	0100101
w	175	*010001
y	164	*0100001
p	152	001001
g	152	*000001
b	127	*0010001
v	83	*0000001
k	49	*00000001
x	13	*000000001
j	8	*0000000001
q	8	*00000000001
z	5	*000000000001

Table 2

Letter	Frequency× 1000	Codeword
q	6	*00001
f	8	*0001
z	9	*0002
b	9	*10001
h	11	*1001
v	15	*1002
g	20	*001
m	26	*002
u	30	*101
p	32	*102
d	38	*2001
c	43	*201
s	61	*202
t	61	*2101
l	66	*211
n	66	*212
r	67	*01
o	87	*02
a	103	11
i	116	12
e	126	22

Table 3

Captions to Tables.
Table 1: Optimal 4-ary synchronous code for the Latin alphabet.
Table 2: Derived binary code for the English alphabet.
Table 3: Derived ternary code for the Italian alphabet.