Algorithmica **12** 409–419

Complexity Aspects of Guessing Prefix Codes

Aviezri S. Fraenkel[1] and Shmuel T. Klein[2]

[1] Dept. of Applied Math. and CS, The Weizmann Institute of Science, Rehovot 76100, Israel
[2] Dept. of Mathematics and Computer Science, Bar-Ilan University, Ramat Gan 52900, Israel

Abstract: Given a natural language cleartext and a ciphertext obtained by Huffman coding, the problem of guessing the code is shown to be NP-complete for various variants of the encoding process.

One of the best known compression techniques is due to Huffman [3], which is optimal for any given probability distribution in the sense that it achieves a minimum redundancy code, provided each codeword consists of an integral number of bits. The aspect of using Huffman codes also as an encryption method has been considered in [6] and recently in [4], where it was motivated by an application to storing a large textual database on a CD-ROM. The text of the database had not only to be compressed, but also to be encrypted to prevent illegal use of copyrighted material. In this paper we show that various decoding problems involving variable length prefix codes, of which Huffman codes are a special case, are NP-complete, and suggest some methods how this could be exploited to increase the cryptographic security of using such codes. We do not, however, seek absolute secrecy as might be required in some military applications. Since we consider mainly literary cleartexts which are in the public domain, all we need is to make the cryptanalysis difficult enough so that the cost of the decryption effort exceeds the potential profit of "breaking the code".

In the sequel, we refer to the elements $\sigma_1, \ldots, \sigma_n$ of an alphabet Σ as *letters*, and to the elements of a text as *characters*, i.e., letters counting their multiplicity.

Thus the text `people` consists of 6 characters and 4 letters.

We consider the problem of "breaking a code". First, we consider it for a model where every character or run of identical characters is encoded individually. Given is a cleartext $T = t_1 \cdots t_h$, where the t_i are the elements to be encoded, and a ciphertext S, which is a binary sequence of which the opponent knows that it is a prefix encoding of T; that is, he knows T and S and that there is a partition of S into codewords $c(t_1), \ldots, c(t_h)$ which satisfy the following two conditions:

(1) the set of the different codewords in the sequence $\{c(t_1), \ldots, c(t_h)\}$ is a prefix set, i.e., no $c(t_i)$ is the proper prefix of any other (the $\{c(t_i)\}$ are not necessarily a complete code, as not all the characters of the alphabet need appear in T);

(2) the encoding defined by the sequence is consistent, that is, $c(t_i) = c(t_j)$ if and only if $t_i = t_j$, for all $1 \leq i, j \leq h$.

The opponent's objective is to find the code, i.e., the function $c()$.

In many applications, the cleartext T is a small subset of a large database $\mathcal{T}$. In [4], $\mathcal{T}$ is the *Trésor de la Langue Française*, a French database with 112 million words of running text. Recently, the French government, which owns the database, has decided to store it on CD-ROM, and to restrict the printout of retrieved locations to at most 300 characters of cleartext. The small subset of cleartext T might be the short context of some keyword retrieved in response to a query. Then a portion of ciphertext containing the encryption of this context is typically downloaded from the CD-ROM into RAM, and we may therefore assume that the opponent has access to it. In this case, the decryption attempt of the opponent could be based on the statistics of the letter frequencies in the full text $\mathcal{T}$, which are well known, rather than on analyzing only the given cleartext T. These statistics enable one to guess the codeword lengths within a small error. Each such guess induces a partition of the ciphertext. This partition is then subjected to the above tests (1) and (2). Any partition passing these two tests successfully can be accepted by the opponent as constituting a valid function $c()$.

A way to prevent such decryption attempts is by adding some random bits to the ciphertext, increasing its size by a small constant factor. Suppose we decide to

encode our text T using some code $\{c(\sigma_i)\}$, with $|c(\sigma_i)| = \ell_i$ for $i = 1, \ldots, n$, where, here and below, $|x|$ denotes the length (number of characters) of the string x. We then choose certain indices j and (small) integers ℓ'_j, and adjoin to each occurrence of $c(\sigma_j)$ a suffix of ℓ'_j random bits, that is, different occurrences of $c(\sigma_j)$ may be followed by different suffixes of length ℓ'_j; this does not affect the prefix property of the code, as for decoding, the ℓ'_j bits following an occurrence of $c(\sigma_j)$ are simply skipped. It follows that σ_j is now encoded by $\ell_j + \ell'_j$ bits, but the above indicated attack for constructing $c()$ would fail, because condition (2) is not true anymore. If the opponent knows about this strategy, he must now guess which bits to skip. The following theorem shows that a similar problem is NP-complete.

Define a *meta-character* to be either a character or a run of adjacent identical characters. Define SPC, the Subsequence Prefix Code problem, as follows:

SPC (Subsequence Prefix Code). *Input*: $\ell \in \mathcal{Z}^+$, $p \in \mathcal{Z}^+$, a sequence $T = \{t_1, \ldots, t_h\}$ of text-characters over the letters $\{\sigma_0, \ldots, \sigma_n\}$, with fixed $n \geq 1$, and a binary sequence S. *Question*: Is there a subsequence S' of S, $|S'| = \ell$, such that S' can be partitioned into codewords, the lengths of which belong to $\{s, s+1, \ldots, s+p\}$ for some $s \in \mathcal{Z}^+$, such that the set of these codewords forms a prefix code and such that each codeword in S' encodes a meta-character of T?

Theorem 1. SPC is NP-complete.

Proof: Given the subsequence S' and its partition into codewords, a linear scan of S' can verify that the lengths of the codewords are indeed in the required range. Given the partition of T into meta-characters $\{\tau_1, \ldots, \tau_{h'}\}$, we have to check that the partition of S' induces a consistent encoding of T, that is, $\tau_i = \tau_j$ if and only if $c(\tau_i) = c(\tau_j)$, for $1 \leq i < j \leq h'$. This can be checked by a linear scan of T and S' using a table of size $O(h)$. Moreover, it can be verified in at most $O(h^2)$ steps that $c(\tau_i)$ is not a prefix of $c(\tau_j)$ unless $i = j$. Thus SPC $\in NP$.

For the reduction, we need the following problem, which is known to be NP-complete (see Garey & Johnson [2]).

SUS (Subset Sum). *Input*: $(a_1, \ldots, a_m) \in (\mathcal{Z}^+)^m$, $B \in \mathcal{Z}^+$ with $B \leq \sum_{i=1}^m a_i$. *Question*: Is there a binary vector $(\varepsilon_1, \ldots, \varepsilon_m)$ such that $\sum_{i=1}^m \varepsilon_i a_i = B$?

We show SUS $\propto$ SPC. Let $(a_1, \ldots, a_m) \in (\mathcal{Z}^+)^m$ and $B \in \mathcal{Z}^+$ be an instance of SUS. Let $A = \sum_{i=1}^m a_i$. We construct an instance of SPC. We let $k = 1 + \lceil \log_2(m+1) \rceil$ and create the $m+1$ codewords $d_0, \ldots, d_m$, where d_0 is a string of $k+1$ 1's, and for $1 \le i \le m$, d_i is the k-bit binary representation of $i-1$ (i.e., the binary representation of $i-1$, padded on the left with leading 0's to fill to length k), followed by a trailing 0-bit. Replacing any d_i by its k-bit prefix preserves the prefix property.

Define now $m+1$ meta-characters $\tau_0, \ldots, \tau_m$ by

$$\tau_0 = \sigma_0 \qquad \tau_i = (\sigma_{1+(i-1) \bmod n})^{\lfloor (i-1)/n \rfloor + 1} \quad \text{for } 0 < i \le m,$$

where x^y designates, here and below, the concatenation of the string x by itself y times, and $x_1 x_2$ is the concatenation of x_1 followed by x_2. In other words, $\tau_i = \sigma_i$ for $0 \le i \le n$, then $\tau_i = (\sigma_{i-n})^2$ for $n < i \le 2n$, etc.

Define the text T and the binary sequence S by

$$T = \left\{ (\tau_0\tau_1)^{a_1} (\tau_0\tau_2)^{a_2} \cdots (\tau_0\tau_m)^{a_m} \right\}, \qquad S = \left\{ (d_0 d_1)^{a_1} (d_0 d_2)^{a_2} \cdots (d_0 d_m)^{a_m} \right\}.$$

Note that $2A \le |T| \le (\lceil m/n \rceil + 1)A$ and that $|S| = 2(k+1)A$, which are exponential in the input length $\Omega(\sum_{i=1}^m \log a_i)$. This follows from the fact that SUS is solvable in pseudo-polynomial time (see [2]), so in its NP-complete instances, the weights a_i are exponentially large with respect to the number m of the weights. Letting $m = qn + r$, $0 \le r < n$, we construct succinct representations T'' and S'' of T and S:

$$T'' = \{\sigma_0, \sigma_1, 1, a_1; \cdots; \sigma_0, \sigma_n, 1, a_n; \sigma_0, \sigma_1, 2, a_{n+1}; \cdots; \sigma_0, \sigma_n, 2, a_{2n}; \cdots; \sigma_0, \sigma_1, q,$$
$$a_{(q-1)n+1}; \cdots; \sigma_0, \sigma_n, q, a_{qn}; \sigma_0, \sigma_1, q+1, a_{qn+1}; \cdots; \sigma_0, \sigma_r, q+1, a_m\}$$

$$S'' = \{d_0, d_1, a_1; \cdots; d_0, d_n, a_n; d_0, d_{n+1}, a_{n+1}; \cdots; d_0, d_{2n}, a_{2n}; \cdots;$$
$$d_0, d_{(q-1)n+1}, a_{(q-1)n+1}; \cdots; d_0, d_{qn}, a_{qn}; d_0, d_{qn+1}, a_{qn+1}; \cdots; d_0, d_m, a_m\},$$

where the σ_i are encoded by the logarithms of their indices and all numbers are encoded in binary. In particular, all the a_i are encoded in binary (with $\lceil \log_2 a_i \rceil$ bits). Then $|T''|$ and $|S''|$ are linear in the input length. Also A can be written with $\log_2 A \le \sum_{i=1}^m \log_2 a_i$ bits.

We complete the construction by letting $p = 1$ and $\ell = 2kA + A + B$ (thus S' is obtained by deleting some $A - B$ bits from S).

Suppose there is $(\varepsilon_1, \ldots, \varepsilon_m) \in \{0,1\}^m$ with $\sum_{i=1}^m \varepsilon_i a_i = B$. We define the codewords c as follows: $c(\tau_0) = d_0$; $c(\tau_i)$ is the k-bit prefix of d_i (and the rightmost 0-bit deleted) if $\varepsilon_i = 0$; $c(\tau_i) = d_i$ if $\varepsilon_i = 1$ $(1 \leq i \leq m)$. Then $\{c(\tau_i)\}_{i=0}^m$ is a prefix code and $S' = \left\{(c(\tau_0)c(\tau_1))^{a_1}(c(\tau_0)c(\tau_2))^{a_2} \cdots (c(\tau_0)c(\tau_m))^{a_m}\right\}$ is an encoding of T of length $|S'| = \sum_{i=1}^m (k+1+k+\varepsilon_i)a_i = 2kA + A + B = \ell$. The codewords have lengths k or $k+1$, so $p = 1$ as required.

Conversely, suppose $S' \subseteq S$, $|S'| = \ell$, that the set of codewords in S' is prefix and that S' is an encoding of T each of whose codewords has length s or $s+1$ for some $s \in \mathcal{Z}^+$.

Claim: The partition of T into meta-characters to be encoded consists of a_i occurrences of $\tau_0\tau_i$, $1 \leq i \leq m$, and the codeword corresponding to τ_i is either d_i or the k-bit prefix of d_i, $0 \leq i \leq m$.

Proof of the Claim: By induction on i. As to the partition of T into meta-characters to be encoded, since a meta-character consists only of a run of identical characters, and in the prefix $(\tau_0\tau_1)^{a_1}$ of T, σ_0 and σ_1 alternate, there is no other possibility to parse this prefix into meta-characters.

As to the lengths of $c(\tau_0)$ and $c(\tau_1)$, consider the prefix $\sigma_0\sigma_1\sigma_0\sigma_x$ of T, where $x = 1$ or $x = 2$, depending on whether $a_1 > 1$ or not. Consider the prefix $1^{k+1}0^{k+1}1^{k+1}0^{k-1}$ of S. Because of consistency, the lengths of both $c(\sigma_0)$ and $c(\sigma_1)$ must be larger than $(k+1)/2$, otherwise both $c(\sigma_0)$ and $c(\sigma_1)$ would consist only of 1's. If the length of $c(\sigma_0)$ is larger than $(k+1)/2$ but smaller than k, then $c(\sigma_0)$ consists of a string of 1's of length $< k$, so that due to the assumption that codeword lengths can differ by at most 1, $c(\sigma_1)$ is also of length at most k (but has at least one 0 in its suffix), but then the encoding $c(\sigma_0)$ of the third character would start with a 0, contradicting consistency. If the length of $c(\sigma_0)$ is larger than $k+1$, then $|c(\sigma_1)|$ must be at least $k+1$, so that $c(\sigma_1)$ would include some 1's of the second block of 1's, but then the encoding of the third and first characters are inconsistent because the former has less 1's as prefix. Thus $|c(\sigma_0)|$ is either $k+1$ (i.e., $c(\sigma_0) = d_0$) or k (i.e., $c(\sigma_0)$ is the k-bit prefix of d_0). In either case, the first $c(\sigma_1)$ begins with the first 0 of S. It follows that $|c(\sigma_1)|$ cannot be less than k or more than $k+1$, as otherwise the encoding of the third character would either start with a zero, or have at least one zero in its suffix. Thus $c(\sigma_1)$ is either d_1 or its k-bit

prefix, proving the induction claim for $i \leq 1$.

Suppose the claim is true for $i < j$, so that the partition of a prefix of T into meta-characters consists of a_i occurrences of $\tau_0\tau_i$ for $i < j$, but that the immediately following substring $(\tau_0\tau_j)^{a_j}$ of T is not parsed as a_j occurrences of $\tau_0\tau_j$. Suppose that a prefix of $(\tau_0\tau_j)^{a_j}$ is parsed as $(\tau_0\tau_j)^a\tau_0\tau_u$, with $0 \leq a < a_j$ and $u \neq j$. It is not possible that $u < j$, because τ_u already appeared earlier in the partition of T and has, by the inductive hypothesis, d_u or its k-bit prefix as the corresponding codeword; however, the corresponding bitstring in S is different from d_u, contradicting consistency. But $u > j$ is also impossible, since this would mean that more than $\lfloor (j-1)/n \rfloor + 1$ adjacent occurrences of $\sigma_{1+(j-1) \bmod n}$ are encoded here as a single meta-character, whereas the text doesn't contain so many. Thus $u = j$, which proves the first part of the claim.

As to the second part, we know from the inductive hypothesis that τ_i is encoded by d_i or its k-bit prefix, for $i < j$, so that the encoding of the substring of T starting with $(\tau_0\tau_j)^{a_j}$ starts with $d_0d_jd_0$. Note that d_0 consists of $k+1$ 1's, while d_j has a leading and trailing 0. It follows that $|c(\tau_j)|$ cannot be less than k or more than $k+1$, as otherwise the encoding of the second τ_0 of the block $(\tau_0\tau_j)^{a_j}$ would either start with a zero, or have at least one zero in its suffix. Thus $c(\tau_j)$ is either d_j or its k-bit prefix, proving the induction claim. ∎

It follows from the claim that $s = k$. The codeword $c(\tau_0)$ cannot be the k-bit prefix of d_0, because then even if $|c(\tau_i)| = k+1$ for all $0 < i \leq m$, the length of S' would only be $2kA + A < \ell$. Thus $c(\tau_0) = d_0$. It follows that for $1 \leq i \leq m$, there is an $\varepsilon_i \in \{0, 1\}$ such that the length of $c(\tau_i)$ is $k + \varepsilon_i$. We have

$$2kA + A + B = \ell = |S'| = \sum_{i=1}^{m}(|c(\tau_0)| + |c(\tau_i)|)a_i = \sum_{i=1}^{m}((k+1) + (k+\varepsilon_i))a_i,$$

thus $\sum_{i=1}^{m} \varepsilon_i a_i = B$, which shows that $(\varepsilon_i, \ldots, \varepsilon_m)$ is a solution of the given SUS-instance. ∎

If inserting new bits into the ciphertext makes decoding more difficult, it reduces, on the other hand, the compression efficiency. In the following theorem we show that decoding may still be difficult, even if no new bits are inserted, if we do not insist on encoding each meta-character of the cleartext T individually. The

idea is that if one can use also general variable length strings in T, this puts the additional burden on the opponent of guessing not only the partition of S, but also that of T. This is one of the defenses against a cryptographic attack suggested in [6]. In [5] and [4], some heuristics are suggested for choosing the variable length strings. Choosing a set of strings so as to maximize compression may be intractable, since if its elements are restricted to be the prefixes or suffixes of words in the text, the problem has been shown in [1] to be NP-complete. Thus even if the opponent quite naturally assumes that we try to optimize compression, he still has to guess the set of elements used.

We could, for example, decide that this set should include the single (upper and lower case) letters, k_1 out of the n_1 most frequent bigrams, k_2 out of the n_2 most frequent trigrams, k_3 out of the n_3 most frequent words of any length, and other frequent strings. It is easy to compile a list of the most promising variable length strings of this kind, and compression efficiency will barely be affected by the preference of a specific subset over another, if these subsets include a large enough number of elements. The corresponding Huffman codes, however, can be completely different. Thus the exact set could be chosen at random, with $\binom{n_1}{k_1}$ choices for the bigrams, $\binom{n_2}{k_2}$ for the trigrams, etc. The indices j_i of the chosen subsets are our secret key ($1 \leq j_i \leq \binom{n_i}{k_i}$). The number of choices is maximized by setting $k_i \approx n_i/2$, yielding a number of possibilities of order $2^{\Sigma n_i}$. Thus the possibility of the opponent guessing the key, which would also let him guess the partition of the cleartext and the lengths of the corresponding codewords, can be ruled out.

Though the determination of the secret key is difficult, there might presumably be a direct way of decrypting the given ciphertext. We now show that also any such attempt seems to be very hard. We first consider the case where in the partition of T only simple meta-characters or pairs consisting of a meta-character followed by a character are allowed and where the lengths of the codewords in S are restricted to certain values. Define the following problem:

RPE (Restricted Prefix Encoding). *Input*: $z \in \mathcal{Z}^+$, a sequence $T = \{t_1, \ldots, t_h\}$ of text-characters over the letters $\{\sigma_0, \sigma_1, \ldots, \sigma_n\}$ with fixed $n \geq 1$, and a binary sequence S. *Question*: Is S a prefix encoding of T satisfying the restrictions stated

below, in other words, can T be partitioned into ℓ elements and S into ℓ codewords such that the set C of these codewords forms a prefix code and such that the sequence of codewords in the partition of S encodes the sequence of elements in the partition of T? The additional restrictions are:

1. each element in the partition of T consists either of a meta-character, or of a meta-character followed by a single character;
2. each codeword in the partition of S has length 1, s or $s+1$ for some $s \in \mathcal{Z}^+$;
3. there are precisely z occurrences of codewords of length 1 in the partition of S.

Theorem 2. RPE is NP-complete.

Proof: Given sequences T and S and their partitions into elements and codewords, a linear scan of S can verify that there are precisely z codewords of length 1. Let $f_1, \ldots, f_\ell$ denote the elements of the partition of T, where each f is either a single meta-character or a meta-character followed by a character. There are at most $O(h^2)$ different elements in the partition of T. We have to check that the partition of S induces a consistent encoding of the partition of T, that is, $f_i = f_j$ if and only if $c(f_i) = c(f_j)$. This can be checked by a linear scan of T and S using a table of size $O(h^2)$. Moreover, it can be verified in at most $O(h^4)$ steps that $c(f_i)$ is not a prefix of $c(f_j)$ unless $i = j$. Thus RPE $\in NP$.

We show SUS $\propto$ RPE. Let $(a_1, \ldots, a_m) \in (\mathcal{Z}^+)^m$ and $B \in \mathcal{Z}^+$ $(B \leq \sum_{i=1}^m a_i)$ be an instance of SUS. We construct an instance of RPE. Let k be defined by $k = \lceil \log_2(m+1) \rceil$. For $1 \leq i \leq m$, let d_i be the k-bit binary representation of $i-1$ (i.e., the binary representation of $i-1$, padded on the left with leading 0's to fill to length k), preceded by a leftmost 0-bit and followed by a rightmost 1-bit; then the set of m codewords $d_1, \ldots, d_m$ is a (non-complete) prefix code, with all of its codewords of length $k+2$. The prefix property is preserved if the codeword 1 is added to the set, or if any $(k+2)$-bit codeword is replaced by its $(k+1)$-bit prefix.

Let $A = \sum_{i=1}^m a_i$. As in the proof of Theorem 1, we define $m+1$ meta-characters $\tau_0, \ldots, \tau_m$ by

$$\tau_0 = \sigma_0 \qquad \tau_i = (\sigma_{1+(i-1) \bmod n})^{\lfloor (i-1)/n \rfloor + 1} \quad \text{for } 0 < i \leq m.$$

We define the text T and the binary string S by

$$T = \{\tau_0^{4(m+2)A}(\tau_1\tau_0)^{4a_1}(\tau_2\tau_0)^{4a_2}\cdots(\tau_m\tau_0)^{4a_m}\},$$

$$S = \{1^{4(m+2)A}d_1^{4a_1}d_2^{4a_2}\cdots d_m^{4a_m}\}.$$

We have $4(m+4)A \le |T| \le 4(m+\lceil m/n\rceil+3)A$ and $|S| = 4(k+m+4)A$, since $|d_i| = k+2$ $(1 \le i \le m)$. Thus $|T|$ and $|S|$ are exponential in the input length, but can be rewritten succinctly as in the proof of Theorem 1. The construction is completed by letting $z = 4(m+2)A + 4B$.

Suppose SUS has a solution $(\varepsilon_1, \ldots, \varepsilon_m) \in \{0,1\}^m$ with $\sum_{i=1}^m \varepsilon_i a_i = B$. We define the codewords $c \in C$ by $c(\tau_0) = 1$, and by $c(\tau_i\tau_0) = d_i$ if $\varepsilon_i = 0$; $c(\tau_i)c(\tau_0) = d_i$ if $\varepsilon_i = 1$, i.e., $c(\tau_i)$ is the $(k+1)$-bit prefix of d_i, and it is followed by $c(\tau_0) = 1$. Then C is a prefix code, S is an encoding of T with $s = k+1$, and the number of occurrences of $c(\tau_0)$ is

$$4(m+2)A + \sum_{i=1}^{m} 4a_i\varepsilon_i = 4(m+2)A + 4B = z.$$

Conversely, suppose S is an encoding of T using elements of a prefix code C whose codewords have length 1, s or $s+1$ for some $s \in \mathcal{Z}^+$, and that there are precisely z occurrences of codewords of length 1 in the partition of S. Let $R = \tau_0^{4(m+2)A}$ be the prefix of length $4(m+2)A$ of T, and denote by T' the suffix of T obtained by deleting the prefix R, and by S' the suffix of S obtained by deleting the prefix $1^{4(m+2)A}$.

Claim 1: In the partition of R into meta-characters and pairs of the form (meta-character, character) to be encoded, each τ_0 in R is encoded by itself, and $c(\tau_0) = 1$.

Proof of Claim 1: The string T' contains at most $4(\lceil m/n\rceil+1)A < z$ characters. Hence some element in R is necessarily encoded by a single bit. If this element is a meta-character of the form τ_0^j or a pair of the form (τ_0^{j-1}, τ_0) for some $j > 1$, then there are at most $4(m+2)A/j < z$ such elements, because a string of more than one consecutive τ_0 appears only in R. Thus the element which is encoded by a single bit must be τ_0.

There are $4(k+2)A$ bits in S', so the total number of zeros in S is less than $4(k+2)A \le z$ (since $k \le m$). It follows that $c(\tau_0) = 1$. But then it is not possible

that any meta-character of the form τ_0^j or any pair of the form (τ_0^{j-1}, τ_0) for some $j > 1$ is encoded, nor is it possible that a suffix τ_0^j of R, for some $j > 0$, is encoded together with the following τ_1, because the corresponding codewords in the partition of S would start with a 1, violating the prefix property. Thus each of the $4(m+2)A$ first τ_0's is encoded by the single bit 1. ∎

Claim 2: In the partition of T' into elements to be encoded: **(1)** every meta-character is of maximal length, i.e., in any substring $\sigma_0\sigma_i^j\sigma_0$ of $\sigma_0 T'$, the run σ_i^j is parsed as one block, either as a single meta-character, or as the pair $\sigma_i^j\sigma_0$; **(2)** there is no pair of the form $\sigma_0\sigma_i$; and **(3)** the bits in S corresponding to $\tau_i\tau_0$ are d_i, in the sense that either $\tau_i\tau_0$ is encoded as one element by d_i, or τ_i is encoded by the $(k+1)$-bit prefix of d_i and τ_0 is encoded by 1.

Proof of Claim 2: By induction on i. Suppose that there is an element in the partition of T' that is encoded by more than $k+2$ bits. Then we are left with at least $4A-1$ elements in T', each of which is encoded by at least $k+2$ bits, so that the length of the encoding of T' is at least $(4A-1)(k+2)+(k+3) > 4A(k+2) = |S'|$. Thus the length of the encoding of no element can exceed $k+2$.

Consider the first few characters $(\tau_1\tau_0)^4$ of T', and consider the prefix $(0^{k+1}1)^4$ of S'. Suppose that the first τ_1 of T' is encoded on its own and that $|c(\tau_1)| = k+2$. Then the following τ_0 in T' is not encoded on its own, since the following bit in S' is 0, and we know by Claim 1 that $c(\tau_0) = 1$. But if $\tau_0\tau_1$ is encoded as one element, then $|c(\tau_0\tau_1)|$ must be $k+2$ or $k+1$, contradicting either consistency or the prefix property. Thus if the first τ_1 is encoded on its own, $|c(\tau_1)|$ must be less than $k+2$. It cannot be less than $(k+2)/2$, since then the following codeword would also consist only of zeros. If $|c(\tau_1)| = k'$, with $(k+2)/2 \le k' \le k$, then again the following bit in S' is 0, so that the second character τ_0 must be encoded together with the following τ_1, and a prefix of $c(\tau_0\tau_1)$ is $0^{k+1-k'}$. But $|c(\tau_0\tau_1)|$ is then at most $k+1$, so that the next codeword must start with at least $k+2-k'$ zeros; on the other hand, the next characters in T' are again $\tau_0\tau_1$ (because there are at least 4 occurrences of $\tau_1\tau_0$ in the prefix of T'), and τ_0 cannot be encoded on its own, so the next codeword is also $c(\tau_0\tau_1)$, violating consistency. If $|c(\tau_1)| = k+1$, the next codeword starts with a 1, and because of the prefix property, the next codeword must then be 1, so it is $c(\tau_0)$.

Suppose now that the first τ_1 is encoded together with τ_0, and suppose that the second pair $\tau_1\tau_0$ is also encoded together. If $|c(\tau_1\tau_0)| < k+2$, then $c(\tau_0\tau_1)$ must consist only of zeros, so $k' = |c(\tau_1\tau_0)|$ is at most $(k+1)/2$. The partition of the prefix of S' into codewords consists therefore of $r \geq 2$ occurrences of $0^{k'}$, followed by a codeword containing a 1, followed again by a codeword consisting only of zeros. The length of this codeword cannot be different from k' because of the prefix property, so it must be the encoding of $\tau_1\tau_0$ because of consistency. However, there is no possibility to parse the prefix $(\tau_1\tau_0)^{4a_1}$ of T' into elements to be encoded such that the first r and the $(r+2)$-nd elements are $\tau_1\tau_0$, but the $(r+1)$-st element being different. Thus if both the first and the second pair $\tau_1\tau_0$ are encoded together, we must have $|c(\tau_1\tau_0)| = k+2$.

Still assuming that the first pair $\tau_1\tau_0$ is encoded together, suppose now that the following τ_1 is encoded on its own. Suppose $|c(\tau_1\tau_0)| = k' < k+2$, and consider the fourth character, τ_0. If it is encoded together with the next (the third) τ_1, then the first three codewords satisfy $|c(\tau_1\tau_0)| < k+2$, $|c(\tau_1)| \leq k+2$ and $|c(\tau_0\tau_1)| \leq k+2$. Thus $c(\tau_0\tau_1)$ starts with at least $k+2-k'$ zeros and must include the second 1 (the $2k+4$-th bit) of S', otherwise it would consist only of zeros. But then the next codeword starts with a 0, so it cannot be $c(\tau_0)$, therefore it must be $c(\tau_0\tau_1)$ (since there are at least 4 occurrences of $\tau_1\tau_0$ at the beginning of T'), but the corresponding bits in S' start with at least $k+3-k'$ zeros, a contradiction. It follows that also in the case that the second τ_1 is encoded on its own, we have $|c(\tau_1\tau_0)| = k+2$.

Summarizing, we have shown that the first characters $\tau_1\tau_0$ of T' are either encoded together by d_1, or τ_1 is encoded on its own by the $(k+1)$-bit prefix of d_1, and τ_0 is then encoded on its own by 1. Suppose this is true for the first $u-1$ pairs $\tau_1\tau_0$, $0 < u-1 < 4a_1$, then we know that the bits corresponding to the encoding of the following characters in T' start with $0^{k+1}1$, which implies, by consistency and the prefix property, that the u-th pair $\tau_1\tau_0$ of T' is also either encoded together by d_1, or τ_1 is encoded on its own by the $(k+1)$-bit prefix of d_1, and τ_0 is then encoded on its own by 1. This concludes the proof for $i = 1$.

Assume the claim is true for $i < v$ and let T'' be the suffix of T' starting with $(\tau_v\tau_0)^{4a_v}$. By the induction hypothesis, the encoding of T'' starts with $d_v^{a_v}$.

Consider the partition of T'' into elements to be encoded. Each such element is encoded either by $k+2$ or $k+1$ bits. Consider the first element encoded by $k+1$ bits, if any; since the first bit of the codeword in S' corresponding to the next element is 1, this must be the codeword 1 because of the prefix property, so the following encoded element in T'' must be τ_0. If D is the number of elements τ_0 in the partition of T'', it follows that exactly D elements are encoded by $k+1$ bits, D are encoded by 1 bit and the others by $k+2$ bits.

Suppose that the first τ_v is not parsed as a single meta-character, but that its prefix $\tau_{v'}$, with $v' < v$, is encoded as a single unit, either as the meta-character $\tau_{v'}$, or as a pair, together with the following character. The length of the corresponding codeword must then be $k+2$, because it cannot be less than $k+1$ or more than $k+2$, and if it were $k+1$, the next codeword would start with a 1, contradicting either consistency or the prefix property, since the following character in T'' is not τ_0. Thus the codeword corresponding to $\tau_{v'}$ or to the pair $\tau_{v'}$ followed by the next character, is d_v. We are then left with at least $A' = \sum_{i=v}^{m} 4a_i$ elements in the partition of T'', which are encoded by at least $(A'-D)(k+2) + D + A'(k+1) = (k+2)A'$ bits, which is impossible, since only $(k+2)(A'-1)$ bits remain in S'. It follows that the first τ_v is parsed as a single meta-character.

Suppose that the first τ_v is encoded on its own and that $c(\tau_v) = d_v$. Then the next codeword must also be d_v because of the prefix property, which means that the next element should again be τ_v; however, the next element starts with τ_0. Thus if the first τ_v is encoded on its own, we have $|c(\tau_v)| = k+1$, and the next element is τ_0. If, on the other hand, the first τ_v is encoded together with the following τ_0, $|c(\tau_v\tau_0)|$ cannot be $k+1$, as the next bit in S' would be 1 and the next element is not τ_0, so we have $|c(\tau_v\tau_0)| = k+2$. As above, consistency and the prefix property imply that this is also true for subsequent pairs $\tau_v\tau_0$, showing that the claim is true for $i = v$. ∎

By Claim 2 we know that in the partition of T', all the pairs $\tau_i\tau_0$ are either encoded as a single element, or as two elements: the meta-character τ_i followed by the character τ_0. For every $i \in \{1, \ldots, m\}$ the encoding of $\tau_i\tau_0$ may thus be either by the single codeword d_i of length $k+2$, or by two codewords $c(\tau_i)c(\tau_0)$, where $c(\tau_i)$ is the $(k+1)$-bit prefix of d_i. Now the prefix property implies that for every fixed i,

precisely *one* of these two possibilities prevails for all the $4a_i$ occurrences of d_i. Thus we may define $(\varepsilon_1, \ldots, \varepsilon_m) \in \{0,1\}^m$ such that $\varepsilon_i = 0$ if and only if $\tau_i\tau_0$ is encoded by the single codeword d_i $(1 \leq i \leq m)$. The number of occurrences of the codeword τ_0 is thus $4(m+2)A + \sum_{i=1}^{m} 4a_i\varepsilon_i$ which must be equal to $z = 4(m+2)A + 4B$. Thus $B = \sum_{i=1}^{m} \varepsilon_i a_i$, so that $(\varepsilon_1, \ldots, \varepsilon_m)$ is a solution of the given instance of SUS. ∎

Note that though z is given, the problem is NP-complete. The point is that even if the opponent guesses the element encoded by a single bit, not all its occurrences are necessarily encoded by a single bit, due to a different parsing.

We now define a less restrictive version of RPE.

PE (Prefix Encoding). *Input*: Positive integers z, ℓ_1, p and q, a sequence $T = \{t_1, \ldots, t_h\}$ of text-characters over the letters $\{\sigma_0, \sigma_1, \ldots, \sigma_n\}$ with fixed $n \geq 1$, and a binary sequence S. *Question*: Is S a prefix encoding of T in the sense defined in RPE, satisfying:
1. each element in the partition of T consists either of a single meta-character or of up to $p-1$ consecutive meta-characters followed by a character;
2. the lengths of the codewords in the partition of S belong to the set $\{\ell_1, \ell_2, \ldots, \ell_q\}$, where $\ell_1 < \ell_2 < \cdots < \ell_q$ and $\ell_1 = \ell_j - \ell_i$ for some $1 < i < j \leq q$;
3. there are precisely z occurrences of codewords of length ℓ_1 in the partition of S.

Corollary. PE is NP-complete.

Proof: Restricting PE to the values $\ell_1 = 1$, $p = 2$ and $q = 3$ we get RPE, which is NP-complete by Theorem 2. ∎

Returning to our application of storing, in encrypted form, a large literary database on a CD-ROM, we conclude that there is probably no polynomial algorithm for breaking the code, unless $P = NP$. The price of applying a non-polynomial algorithm will, on the other hand, even for subtexts of moderate length, soon exceed the value an opponent might gain by infringing on the copyright. Thus Huffman coding, which is widely known to yield efficient compression, can also, in certain applications, be useful as an encryption method.

References

[1] **Fraenkel A.S., Mor M., Perl Y.,** Is text compression by prefixes and suffixes practical? *Acta Informatica* **20** (1983) 371–389.

[2] **Garey M.R., Johnson D.S.,** *Computers and Intractability: A Guide to the Theory of* NP-*Completeness*, W.H. Freeman, San Francisco (1979).

[3] **Huffman D.,** A method for the construction of minimum redundancy codes, *Proc. of the IRE* **40** (1952) 1098–1101.

[4] **Klein S.T., Bookstein A., Deerwester S.,** Storing text retrieval systems on CD-ROM: compression and encryption considerations, *ACM Trans. on Information Systems* **7** (1989) 230–245.

[5] **Rubin F.,** Experiments in text file compression, *Comm. ACM* **19** (1976) 617–623.

[6] **Rubin F.,** Cryptographic aspects of data compression codes, *Cryptologia* **3** (1979) 202–205.