New Bounds on D-ary Optimal Codes

Gonzalo Navarro [a] Nieves Brisaboa [b]

[a] *Center for Web Research, Dept. of Computer Science, Univ. of Chile, Chile.*
gnavarro@dcc.uchile.cl.

[b] *Database Lab., Univ. da Coruña, Facultade de Informática, Spain.*
brisaboa@udc.es.

Abstract

We propose a simple method that, given a symbol distribution, yields upper and lower bounds on the average code length of a D-ary optimal code over that distribution. Thanks to its simplicity, the method permits deriving analytical bounds for families of parametric distributions. We demonstrate this by obtaining new bounds, much better than the existing ones, for Zipf and exponential distributions when $D > 2$.

Key words: Optimal codes, Huffman coding, redundancy, frequency distributions, D-ary codes, dense codes, information retrieval.

1 Introduction

Huffman coding [10] is one of the most widely used coding methods. Given a probability distribution $\{p_i\}_{0 \le i < n}$ for a set of source symbols, Huffman coding assigns a minimal *prefix code* to the set of source symbols. That is, Huffman assigns a *codeword* (sequence of target symbols) to each source symbol such that no code is a prefix of another and $L = \sum_{0 \le i < n} p_i \ell_i$ is minimal, where ℓ_i is the length of the codeword assigned to symbol i. If the alphabet of the target symbols is of size D, then we say that the Huffman code is D-ary. D-ary codes are useful for large source alphabets, where they provide competitive compression ratios compared to binary codes, yet permit faster decoding. For example, byte-oriented Huffman is useful in compressed text databases [14]. On infinite source alphabets, one can still speak of *optimal codes*, that is, prefix codes that minimize the redundancy.

⋆ Supported by CYTED VII.19 RIBIDI Project and, for the first author, Millennium Nucleus Center for Web Research, Grant P04-067-F, Mideplan, Chile.

Several studies have been carried out on the problem of bounding L given $\{p_i\}$. This is useful to predict the compression ratios that can be achieved by optimal codes on specific sources or for families of parametric distributions. Although a lot of work has been carried out on improving upper and lower bounds on the redundancy, most of it focuses on the case $D = 2$ (see an excellent survey by Abrahams [1], especially Section 2.2).

In this paper we present a new method to lower and upper bound the redundancy of optimal D-ary codes when applied to a family of parametric distributions. Those bounds are analytic functions of the distribution parameters and, unlike numeric pointwise estimations that can be obtained with other methods, permit making general assertions on the compression of those families. We demonstrate this by analyzing the Zipf and exponential distributions, obtaining much better analytical bounds than the existing ones for the D-ary case.

2 Related Work

The most widely known bound to Huffman compression, coming from the Noiseless Coding Theorem [15], is $H_D \leq L < H_D + 1$, where

$$H_D = \sum_{0 \leq i < n} p_i \log_D \frac{1}{p_i} \tag{1}$$

is the D-ary zero-order entropy of the distribution. The difference between L and the entropy, $r = L - H_D$, is called the *redundancy* of the code. Hence Shannon's theorem establishes that the redundancy is between 0 and 1.

This bound is tight (that is, there are distributions for which the bounds are reached), but it is independent of the distribution. Tighter bounds can be obtained if we take into account the distribution. As explained, most of the huge amount of work on bounding the redundancy of D-ary optimal codes applies to the case $D = 2$ [1]. The only bounds we could find for general D are given in [13,4,5].

From now on we will assume, w.l.o.g., that $\{p_i\}$ is sorted in decreasing order. In [13,4] it is shown that

$$r \geq s - (1 - p_1) \log_D(D^s - 1) - \kappa(p_1), \tag{2}$$

where $\kappa(x) = -x \log_D(x) - (1-x)\log_D(1-x)$ and the positive integer s is 1 if $\log_D(D+1) \leq 1/(1-p_1)$ and otherwise it satisfies

$$\log_D \frac{D^s - 1}{D^{s-1} - 1} \quad > \quad \frac{1}{1-p_1} \quad \geq \quad \log_D \frac{D^{s+1} - 1}{D^s - 1},$$

and they prove that the bound is tight.

In [5] it is shown that, if ϕ_i is the tight lower bound as a function of p_i, then $\phi_{i-1} \leq \phi_i$, that is, the bound improves as we consider less probable source symbols. This makes the above bound valid for any p_i, not just p_1. They prove another bound for any $D > 2$:

$$\begin{aligned} r &\geq m - (1-p_n)\log_D(D^m - 1) - \kappa(p_n), \text{ if } \alpha_{m+1} \leq p_n < D^{-m} \\ r &\geq m - (1-Dp_n)\log_D(D^m - 1) - \kappa(Dp_n), \text{ if } D^{-m-1} < p_n \leq \alpha_m \end{aligned}$$

where p_n is the least probable symbol of the distribution and α_m is the unique zero of $\kappa(x) - \kappa(Dx) + x\log_D(D^{m-1} - 1)$ in $x \in (0, 1/D)$.

With respect to upper bounds, in [13] they improve a bit a previous bound of [6] and obtain

$$r \quad \leq \quad \sigma + \frac{Dp_1}{e \ln D}, \tag{3}$$

where $\sigma = \log_D(D-1) + \log_D(\log_D e) - \log_D e + 1/(D-1)$. For the case $D > 2$ and $p_1 \geq 1/2$ they show that

$$r \quad \leq \quad 1 - \kappa(p_1), \tag{4}$$

and a result proved in [4] is useful for $p_1 < 1/2$:

$$r \quad \leq \quad 1 - \kappa(p_1) + 2(1-p_1)/D. \tag{5}$$

There are also some related results, yet not giving analytical bounds to the redundancy. For example, in [11,12] they consider the problem of the existence of optimal D-ary codes for a given infinite distribution. Their methods also permit numerically deriving bounds for infinite distributions, by successive approximations over finite truncations thereof. Those numerical methods are less powerful than the analytical methods we consider in this paper, as the former are pointwise and cannot be used to derive analytical (and possibly parametric) formulas that bound the redundancy of families of distributions. In [8] they consider the construction of optimal codes for geometric distributions.

3 Dense Coding

In [3] we proposed *Dense Coding* as a more efficient alternative to *Tagged Huffman Coding* [14] for direct compressed text searching on natural language texts. This dense coding, however, is interesting by itself as a bound for the compression that can be obtained with a Huffman code. In this section we present this coding and some of its properties, generalizing the previous proposal of [3].

Definition 1: Given source symbols with probabilities $\{p_i\}_{0 \le i < n}$, an (s, c) stop-cont code (where c and s are integers larger than zero) assigns to each source symbol i a unique code of target symbols formed by a base-c digit sequence terminated by a digit between c and $c + s - 1$. The target alphabet is thus $[0, c + s - 1]$.

It should be clear that a stop-cont coding is just a base-c numerical representation, with the exception that the last digit is between c and $c + s - 1$, i.e., the last digit is a base-s number that is distinguished from previous digits by adding c. Digits between 0 and $c-1$ are called "continuers" and those between c and $c + s - 1$ are called "stoppers".

An alternative useful view is to consider a tree of arity $D = s + c$, where the first c children are internal nodes reserved for continuers and the last s children are leaves corresponding to stoppers (although it is possible for a node not to use all its continuers or all its stoppers). Then each codeword is a root-to-leaf path in the tree. The next property clearly follows.

Property 1: Any (s, c) stop-cont code is a prefix code.

Proof: If one code were a prefix of the other, since the shorter code must have a final digit of value at least c, then the longer code must have an intermediate digit which is not smaller than c. A contradiction. $\square$

Among all the possible (s, c) stop-cont codes for a given probability distribution, the *dense code* is one that minimizes the average symbol length.

Definition 2: Given source symbols with *decreasing* probabilities $\{p_i\}_{0 \le i < n}$, the corresponding (s, c)-dense code is an (s, c) stop-cont code where the codewords are assigned as follows: Let $k \geq 1$ be such that

$$s\,\frac{c^{k-1}-1}{c-1} \;\leq\; i \;<\; s\,\frac{c^k-1}{c-1},$$

then the code corresponding to source symbol i is formed by $k-1$ base-c digits and a final digit between c and $c+s-1$. If $k=1$ then the code is simply the stopper $c+i$. Otherwise the code is formed by the number $\lfloor x/s \rfloor$ written in base c, followed by $c+(x \bmod s)$, where $x = i - \frac{sc^{k-1}-s}{c-1}$.

Example 1: The codes assigned to symbols $i = 0 \ldots 15$ by a $(2,3)$-dense coding are as follows: $\langle 3\rangle$, $\langle 4\rangle$, $\langle 0,3\rangle$, $\langle 0,4\rangle$, $\langle 1,3\rangle$, $\langle 1,4\rangle$, $\langle 2,3\rangle$, $\langle 2,4\rangle$, $\langle 0,0,3\rangle$, $\langle 0,0,4\rangle$, $\langle 0,1,3\rangle$, $\langle 0,1,4\rangle$, $\langle 0,2,3\rangle$, $\langle 0,2,4\rangle$, $\langle 1,0,3\rangle$, and $\langle 1,0,4\rangle$.

Note that the code does not depend on the exact symbol probabilities, but just on their ordering by frequency. We now prove that the dense coding is an optimal stop-cont coding.

Property 2: The average length of a (s,c)-dense code is minimal with respect to any other (s,c) stop-cont code.

Proof: Let us consider an arbitrary (s,c) stop-cont code, and let us write down all the possible codewords in numerical order, as in Example 1, together with the symbol they encode, if they encode one. Then it is clear that (i) any unused code in the middle could be used to represent the source symbol with longest codeword, hence a compact assignment of target symbols is optimal; and (ii) if a less probable symbol with a shorter code is swapped with a more probable symbol with a longer code, then the average code length decreases, and hence sorting the symbols by decreasing frequency is optimal. $\square$

In [3] we use dense coding as an alternative compression method. In this paper we are interested in its ability to give simple lower and upper bounds to the compression given by a D-ary optimal code.

4 The Bounds

Since any (s,c)-dense code is a prefix code, its average code length is not smaller than a D-ary optimal code, where $D = s + c$. This is valid for any choice of s and c, so different bounds can be obtained for a given D.

On the other hand, a D-ary optimal code can be seen as a (D, D) stop-cont code, provided we add D to the last digit of all the codewords. Hence, its average code length cannot be smaller than a (D, D)-dense code.

Hence, if we call $L_{h(D)}$ the average code length using D-ary optimal code and $L_{d(s,c)}$ the average code length using (s, c)-dense code, then we have the following bounds for any $1 \leq s < D$:

$$L_{d(D,D)} \quad \leq \quad L_{h(D)} \quad \leq \quad L_{d(s,D-s)}.$$

The first inequality gives us a simple method for lower bounding the average codeword length of a D-ary optimal code: Add up 1 for the first D probabilities, 2 for the next D^2, 3 for the next D^3, and so on. The second inequality gives us an upper bound that can be computed the same way, but this is different for each choice of s.

The simplicity of the bounds opens to door to the possibility of deriving *analytic* bounds for a family of distributions, if there exists an analytical expression of the symbol frequencies. We start with a generic development than can be applied to any distribution. In the following sections, we complete the analysis for specific examples of distributions that are interesting in real life. Recall that we assume that symbols are sorted in decreasing frequency order.

Since $c^{k-1}s$ different codewords can be coded using k digits, let us call

$$w_k \quad = \quad \sum_{j=1}^{k} s c^{j-1} \quad = \quad s\,\frac{c^k - 1}{c - 1}$$

(where $w_0 = 0$) the number of source symbols that can be coded with up to k digits. Let us also call

$$f_k \quad = \quad \sum_{j=w_{k-1}+1}^{w_k} p_j$$

the overall probability of source symbols coded with k digits.

Then, the average codeword length for the (s, c)-dense coding is

$$\begin{aligned} L_{d(s,c)} &= \sum_{k=1}^{K} k f_k \; = \; \sum_{k=1}^{K} k \sum_{j=w_{k-1}+1}^{w_k} p_j \\ &= 1 + \sum_{k=1}^{K-1} k \sum_{j=w_k+1}^{w_{k+1}} p_j \; = \; 1 + \sum_{k=1}^{K-1} \sum_{j=w_k+1}^{w_K} p_j \qquad (6) \end{aligned}$$

where $K = \lfloor \log_c \left(1 + \frac{n(c-1)}{s}\right) - 1 \rfloor$.

5 Generalized Zipf Distribution

An interesting particular case is a distribution typical of natural language texts, although it holds for many other processes as well [16]. It is well known [2] that, in natural language texts, the vocabulary distribution closely follows a generalized Zipf's law [16], that is, $p_i = A/i^\theta$ and $n = \infty$, for suitable constants A and $\theta > 1$. The latter depends on the text, while

$$A \;=\; \frac{1}{\sum_{i\geq 1} 1/i^\theta} \;=\; \frac{1}{\zeta(\theta)}$$

makes sure that the distribution adds up 1.[1] From Eq. (1), the entropy of this distribution is

$$H_D \;= \sum_{i\geq 1} p_i \log_D \frac{1}{p_i} \;= A\theta \sum_{i\geq 1} \frac{\log_D i}{i^\theta} - \log_D A \;= \frac{-\theta\zeta'(\theta)/\zeta(\theta) + \ln\zeta(\theta)}{\ln D},$$

and this is of course a lower bound for $L_{h(D)}$.

On the other hand, from Eq. (6) we have

$$L_{d(s,c)} \;=\; 1 + A \sum_{k\geq 1} \sum_{j \geq w_k+1} 1/j^\theta.$$

At this point we resort to integration to get lower and upper bounds. Since $1/j^\theta$ decreases with j and since $c^k - 1 \geq (c-1)c^{k-1}$, we have that the above summation is upper bounded as follows

$$\begin{aligned} L_{d(s,c)} &\leq 1 + A \sum_{k\geq 1} \int_{w_k}^{\infty} 1/x^\theta dx \;=\; 1 + \frac{A(c-1)^{\theta-1}}{(\theta-1)s^{\theta-1}} \sum_{k\geq 1} \frac{1}{(c^k-1)^{\theta-1}} \\ &\leq 1 + \frac{A(c-1)^{\theta-1}}{(\theta-1)s^{\theta-1}} \, \frac{c^{1-\theta}}{1-c^{1-\theta}} \, (1-1/c)^{1-\theta} \\ &= 1 + \frac{1}{(\theta-1)\zeta(\theta)s^{\theta-1}(1-c^{1-\theta})} \end{aligned}$$

[1] We are using the Zeta function $\zeta(x) = \sum_{i>0} 1/i^x$. We will also use $\zeta'(x) = \partial\zeta(x)/\partial x$.

In order to obtain the optimum upper bound for $L_{d(s,c)}$ we substitute $c = D - s$ in the latter formula and differentiate with respect to s. It turns out that the minimum upper bound is reached for $s = D - D^{1/\theta}$ and $c = D^{1/\theta}$. The upper bound becomes

$$\min_s L_{d(s,D-s)} \quad \leq \quad 1 + \frac{1}{(\theta-1)\zeta(\theta)(D-D^{1/\theta})^{\theta-1}(1-D^{1/\theta-1})}.$$

A lower bound can be obtained similarly, as follows:

$$L_{d(s,c)} \geq 1 + A \sum_{k\geq 1} \int_{w_k+1}^{\infty} 1/x^\theta dx = 1 + \frac{A(c-1)^{\theta-1}}{\theta-1} \sum_{k\geq 1} \frac{1}{(sc^k - s + c - 1)^{\theta-1}},$$

where the last term is difficult to bound. However, since we need this only for the case $s = c = D$, we substitute and go on

$$\begin{aligned} L_{d(D,D)} &\geq 1 + \frac{A(D-1)^{\theta-1}}{\theta-1} \sum_{k\geq 1} \frac{1}{(D^{k+1}-1)^{\theta-1}} \\ &\geq 1 + \frac{A(D-1)^{\theta-1}}{\theta-1} \sum_{k\geq 1} \frac{1}{(D^{\theta-1})^{k+1}} \\ &= 1 + \frac{(D-1)^{\theta-1}}{(\theta-1)\zeta(\theta)D^{\theta-1}(D^{\theta-1}-1)} \end{aligned}$$

In order to demonstrate the relevance of this result, we have compared our upper and lower bounds against the best previous bounds we could find, which are expressed in Eqs. (2) to (5). The lower bounds of [5] were left out because in this case $p_n \to 0$. We chose the range $\theta \in [1,2]$, since it is the interesting one in most real-life cases. We have experimented with D values 2, 4, 8, 16, 32, 64, 128 and 256. Our bounds are better than others as D increases, but for D as low as 4 we give already competitive upper bounds. For larger D values our bounds are by far the tightest, and at some point our lower and upper bound are so close that they permit predicting the Huffman redundancy with high precision without actually running the algorithm. For space limitations we show in Figure 1 only a sample of the D values tried.

We also include a line called "Real value". This is the result of running the Huffman algorithm on the first part of the sequence, up to $p_i = 10^{-9}$ (this meant up to 54 million symbols). At this point the redundancy results were stable up to precision 10^{-3}. It can be seen that, for not very low D values, our upper bound is actually a good predictor of the real redundancy. For $D = 128$ the two curves are almost indistinguishable. For low D values, on the other hand, the lower bound of Eq. (2) turns out to be a better predictor.

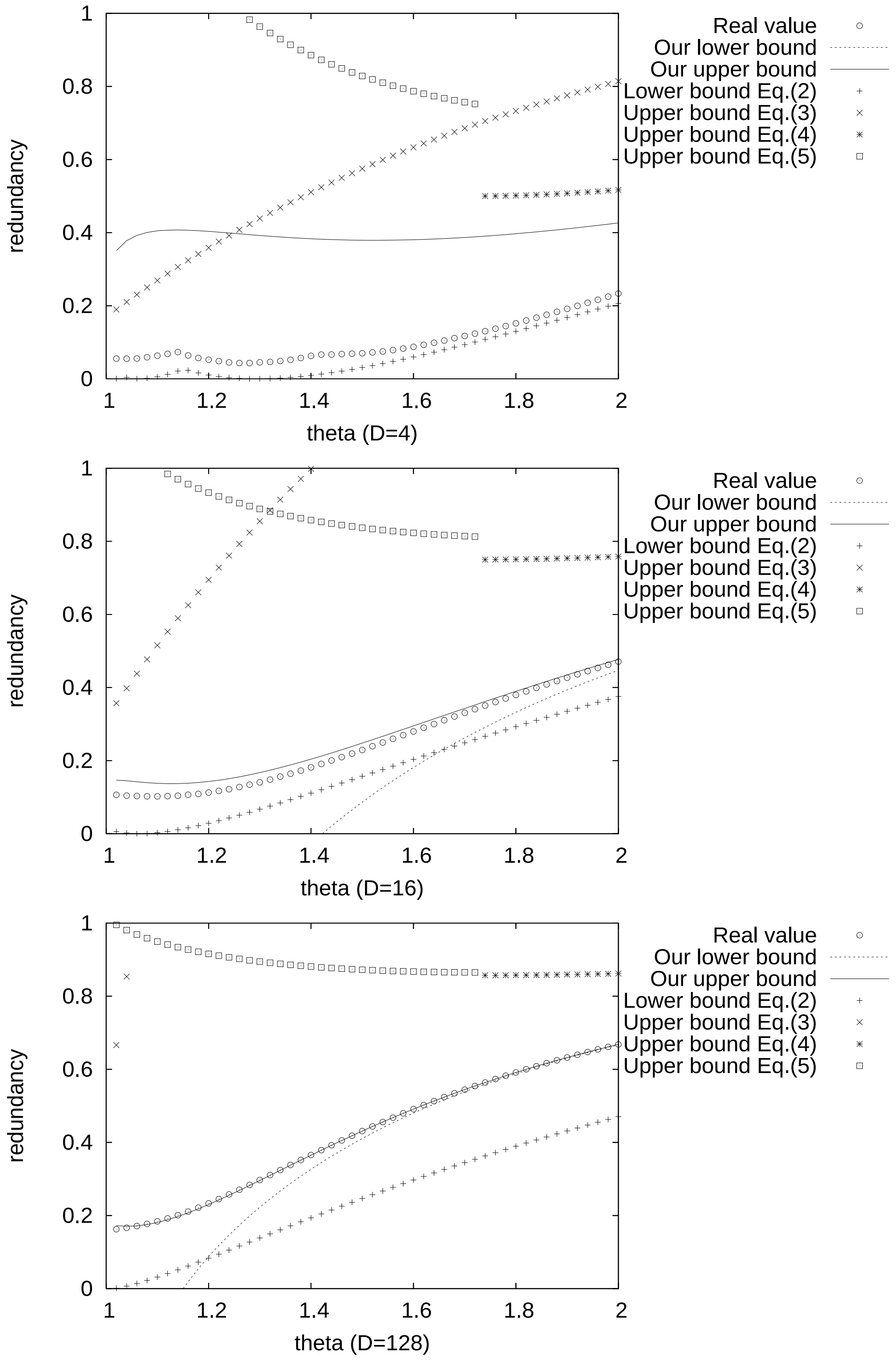

Fig. 1. Comparison between our and previous lower and upper bounds on the redundancy of a D-ary optimal coding over a Zipf distribution with parameter $\theta \in [1, 2]$. From top to bottom we show $D = 4$, $D = 16$ and $D = 128$. Note that our lower bound is totally out of range for $D = 4$.

This result is particularly important for compressed text databases, as it permits predicting their compressed size from an analytical model of the word distribution in some language.

We remark that there exists previous work on binary coding and $\theta = 1$ [9], which is out of our scope ($D > 2$, $\theta > 1$).

6 Exponential Distribution

Let us now consider a distribution of the form $p_j = (1-a)a^{j-1}$, $j \geq 1$. The D-ary entropy of this distribution is easily computed:

$$H_D = \frac{-a\log_D(a) - (1-a)\log_D(1-a)}{1-a}.$$

An (s,c)-dense code has the following average code length, following Eq. (6):

$$L_{d(s,c)} = 1 + (1-a)\sum_{k\geq 1}\sum_{j\geq w_k+1} a^{j-1} = 1 + \sum_{k\geq 1} a^{w_k};$$

by replacing w_k we get

$$L_{d(s,c)} = 1 + \sum_{k\geq 1} a^{\frac{s(c^k-1)}{c-1}} = 1 + a^s + a^{s(1+c)} + a^{s(1+c+c^2)} + \ldots.$$

We could not find a closed expression for the above, but the terms become superexponentially less important as we go on. For example, a very tight lower bound is given by $L_{d(D,D)} \geq 1 + a^D + a^{D(D+1)} + a^{D(D^2+D+1)}$.

With respect to the upper bound, we used a simple bound such as $L_{d(s,c)} \leq 1 + a^s + a^{s(c+1)} + a^{s(c^2+c+1)}/(1-a)$. We could not find the optimum analytically, but only numerically for each case. However, a reasonable solution is $s = c = D/2$. In fact, the effect of not using the optimum setup is noticeable only for a very close to 1.

As a sample of the results, Figure 2 shows the case $D = 8$. Again, these results become better for larger D. As it can be seen, our method gives by far the tightest lower and upper bounds for the cases when a is not very close to 1 (up to 0.8 in the case $D = 8$).

Again, we remark that there exists previous work on binary coding for exponential (or geometric) distributions [7], which is out of our scope ($D > 2$).

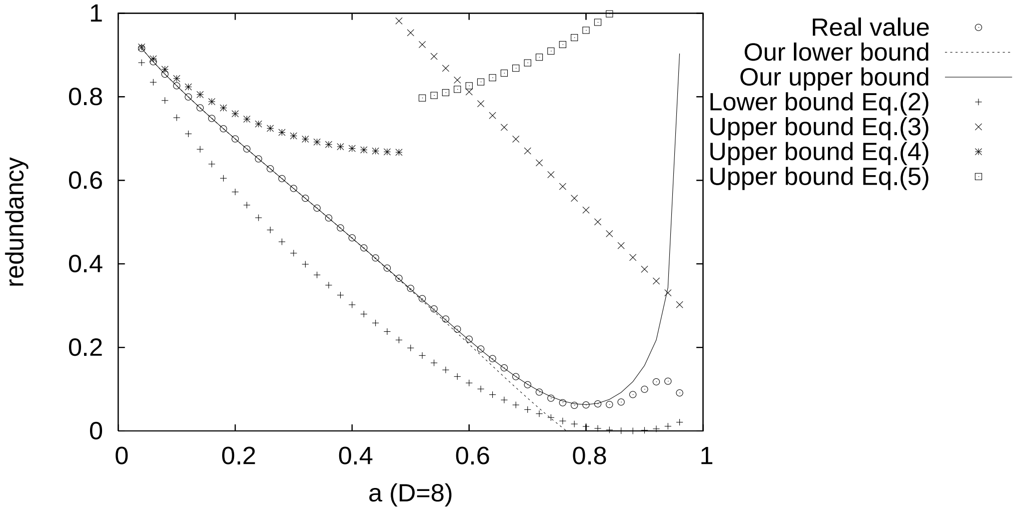

Fig. 2. Comparison between our and previous lower and upper bounds on the redundancy of a D-ary optimal coding over an exponential distribution of the form $p_j = (1 - a)a^{j-1}$, $j \geq 1$, for the case $D = 8$.

7 Conclusions

We have presented a new method to lower and upper bound the redundancy of a D-ary optimal code. The method is based on analyzing the performance of a Dense code, which is a less efficient variant of Huffman, yet much simpler to analyze. By changing the parameters of the Dense codes, we get lower and upper bounds on D-ary optimal codes.

The technique is useful for deriving analytical bounds for distribution families where an analytical (and possibly parametric) expression exists for the probabilities. We have demonstrated the technique by applying it to Zipfian and exponential distributions, showing that our technique usually obtains lower and upper bounds that are by far better than what can be obtained with previous analytical methods, for D-ary codes. The results on Zipf models is particularly relevant to compressed text databases, as it can be used to estimate the size of the compressed collection or the performance of search algorithms based on a parameterized Zipfian model of the text distribution [14].

Acknowledgements

We wish to thank the anonymous reviewers, that helped us improve the paper.

References

[1] J. Abrahams. Code and parse trees for lossless source encoding. *Communications in Information and Systems*, 1(2):113–146, April 2001.

[2] T. C. Bell, J. G. Cleary, and I. H. Witten. *Text Compression.* Prentice Hall, 1990.

[3] N. Brisaboa, A. Fariña, G. Navarro, and M. Esteller. (S,C)-Dense Coding: An optimized compression code for natural language text databases. In *Proc. 10th Intl. Symp. on String Processing and Information Retrieval (SPIRE'03)*, LNCS 2857, pages 122–136. Springer, 2003.

[4] R. Capocelli and A. de Santis. A note on D-ary Huffman codes. *IEEE Trans. on Inf. Theory*, 37(1):174–179, January 1991.

[5] R. de Prisco and A. de Santis. On lower bounds for the redundancy of optimal codes. In *Proc. Data Compression Conference 1998*, 1998.

[6] R. Gallager. Variations on a theme by Huffman. *IEEE Trans. on Inf. Theory*, 24(6):668–674, November 1978.

[7] R. Gallager and D. van Voorhis. Optimal source codes for geometrically distributed alphabets. *IEEE Trans. on Inf. Theory*, 21(2):228–230, 1975.

[8] M. Golin and K. Ma. Algorithms for infinite Huffman-codes. In *Proc. 15th Symp. on Discrete Algorithms (SODA'04)*, pages 758–767, 2004.

[9] M. Gutman. Fixed prefix-encoding of the integers can be Huffman-optimal. *IEEE Trans. on Inf. Theory*, 36(4):936–938, 1990.

[10] D. Huffman. A method for the construction of minimum-redundancy codes. In *Proc. of the Institute of Electrical and Radio Engineers*, volume 40, pages 1090–1101, 1952.

[11] A. Kato, T. Han, and H. Nagaoka. Huffman coding with an infinite alphabet. *IEEE Trans. on Inf. Theory*, 42(3):977–984, 1996.

[12] T. Linder, V. Tarokh, and K. Zeger. Existence of optimal prefix codes for infinite source alphabets. *IEEE Trans. on Inf. Theory*, 43(6):2026–2028, 1997.

[13] D. Manstetten. Tight bounds on the redundancy of Huffman codes. *IEEE Trans. on Inf. Theory*, 38(1):144–151, January 1992.

[14] E. Moura, G. Navarro, N. Ziviani, and R. Baeza-Yates. Fast and flexible word searching on compressed text. *ACM Trans. on Inf. Systems (TOIS)*, 18(2):113–139, 2000.

[15] C. Shannon. A mathematical theory of communication. *Bell Syst. Tech. J.*, 27:398–403, 1948.

[16] G. Zipf. *Human Behaviour and the Principle of Least Effort.* Addison-Wesley, 1949.