Weighted Adaptive Huffman Coding

Aharon Fruchtman* Yoav Gross* Shmuel T. Klein** Dana Shapira*

** Bar Ilan University, Israel * Ariel University, Israel

Data Compression Conference

Contribution

A new generic coding method is defined, extending the known static and dynamic variants and including them as special cases. This leads then to the formalization of a new adaptive coding method, which is shown to be always at least as good as the best dynamic variant known to date, and in particular, always better than static Huffman coding. We present empirical results that show improvements achieved by the proposed method, even when the encoded file includes the model description.

Introduction

Huffman coding is known to be optimal in case the alphabet is known in advance, the set of codewords is fixed and each codeword consists of an integral number of bits. If one of these conditions is violated, optimality is not guaranteed. In the *dynamic* variant of Huffman coding the encoder and decoder maintain identical copies of the model; at each position, the model consists of the frequencies of the elements processed so far. After each processed element σ, the model is updated by incrementing the frequency of σ by 1, while the other frequencies remain the same.
An enhanced dynamic Huffman coding named *forward looking coding* [2] starts with the full frequencies, similar to the static variant, and then decreases them progressively. For this method, after each processed element σ, the model is altered by *decrementing* the frequency of σ by 1, while the other frequencies remain the same. Forward looking Huffman coding has been shown to be always better by at least $m-1$ bits than static Huffman coding. A hybrid method, exploiting both the classical backward and the new forward approaches is proposed in [1], and has been shown to be always at least as good as the forward looking Huffman coding.

Positional Coding

Positional coding is a special case of a forward weight, with $\mathcal{L} \equiv g(i) = n - i + 1$ for $1 \le i \le n$, where $n = |T|$. We shall use the notation $p_\sigma(i)$ to denote $W(\mathcal{L}, \sigma, i, n)$.

Example

Let $T =$ `c c a b b b c a a a` as a small running example. The details for the Positional encoding are presented in the following table. The function $\mathcal{L}$, given on the second line, enumerates the indices in reverse order starting at $n = 10$ down to 1. At the first position $i = 1$, the values of $p_{\mathtt{a}}(1)$, $p_{\mathtt{b}}(1)$ and $p_{\mathtt{c}}(1)$ are 14, 18 and 23, respectively, as shown in the first column of the last three rows. In a left to right scan, the values $p_\sigma(i)$ only change at indices i for which $T[i-1] = \sigma$. Light gray therefore refers to the non-changed values (starting, in a left to right scan, just after an occurrence of σ and ending at the rightmost position where σ occurs in T).

i	1	2	3	4	5	6	7	8	9	10
T	c	c	a	b	b	b	c	a	a	a
$\mathcal{L}(i)$	10	9	8	7	6	5	4	3	2	1
$p_{\mathtt{a}}(i)$	14	14	14	6	6	6	6	6	3	1
$p_{\mathtt{b}}(i)$	18	18	18	18	11	5	0	0	0	0
$p_{\mathtt{c}}(i)$	23	13	4	4	4	4	4	0	0	0

Definitions

Given is a file $T = T[1, n]$ of n characters over an alphabet Σ. We define a general weight $W(g, \sigma, \ell, u)$ based on four parameters, in which

- $g : [1, n] \longrightarrow \mathbb{R}^+$ is a non negative function defined on the integers that assigns a positive real number as a weight to each position $i \in [1, n]$ within T;
- $\sigma \in \Sigma$ is a character of the alphabet;
- ℓ and u are the boundaries of an interval, $1 \le \ell \le u \le n$, serving to restrict the domain of the function g.

The value of the weight $W(g, \sigma, \ell, u)$ is defined for each character $\sigma \in \Sigma$, as the sum of the values of the function g for all positions j in the range $[\ell, u]$ at which σ occurs, that is $T[j] = \sigma$:

$$W(g, \sigma, \ell, u) = \sum_{\{\ell \le j \le u \mid T[j] = \sigma\}} g(j).$$

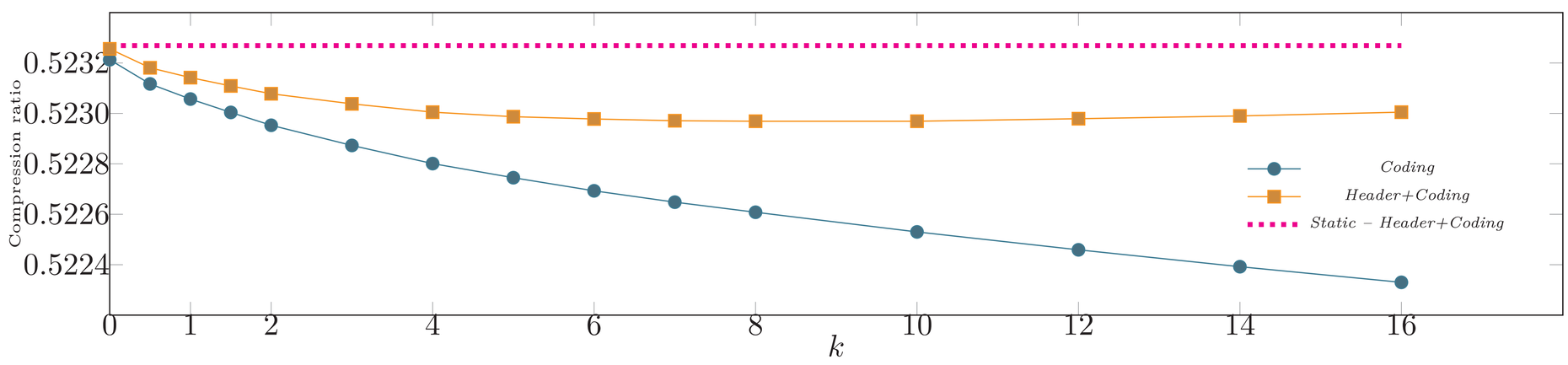
Interesting are weights defined relatively to a current position, *Backward* and *Forward* looking that are implemented by means of the interval $[\ell, u]$ used to restrict the considered range.

The rationale

Characters that are close to the current position are assigned a higher priority than those farther away. The rationale is that the close by characters are those that we are about to encode, so we concentrate on how to reduce the lengths of their codewords, even at the price of having to lengthen the codewords of more distant characters in the text, since, anyway, the encoding of those will be reconsidered by the adaptive process once we get closer to them.

Experimental Results

We considered weighted coding corresponding to functions of the form $g(i) = (n - i + 1)^k$. The figure presents the compression ratio, defined as the size of the compressed divided by the size of the original file, for integer values of k ranging from 0 to 16, as well as for $k = 0.5$ and $k = 1.5$ using both Huffman coding. In particular, Forward is the special case $k = 0$, and Positional encoding corresponds to $k = 1$.



References

[1] Aharon Fruchtman, Shmuel T. Klein, and Dana Shapira. Bidirectional adaptive compression. In *Proceedings of the Prague Stringology Conference 2019*, pages 92–101, 2019.

[2] Shmuel T. Klein, Shoham Saadia, and Dana Shapira. Forward looking Huffman coding. In *The 14th Computer Science Symposium in Russia, CSR, Novosibirsk, Russia, July 1-5*, 2019.