Compressing multisets using tries

Vincent Gripon, Michael Rabbat, Vitaly Skachek and Warren J. Gross
Department of Electrical and Computer Engineering
McGill University, 3480 University St., Montréal, QC H3A 2A7, Canada
Emails: vincent.gripon@ens-cachan.org, michael.rabbat@mcgill.ca, vitaly.skachek@gmail.com, warren.gross@mcgill.ca

***Abstract*—We consider the problem of efficient and lossless representation of a multiset of m words drawn with repetition from a set of size 2^n. One expects that encoding the (unordered) multiset should lead to significant savings in rate as compared to encoding an (ordered) sequence with the same words, since information about the order of words in the sequence corresponds to a permutation. We propose and analyze a practical multiset encoder/decoder based on the trie data structure. The act of encoding requires $O(m(n+\log m))$ operations, and decoding requires $O(mn)$ operations. Of particular interest is the case where cardinality of the multiset scales as $m = \frac{1}{c}2^n$ for some $c > 1$, as $n \to \infty$. Under this scaling, and when the words in the multiset are drawn independently and uniformly, we show that the proposed encoding leads to an arbitrary improvement in rate over encoding an ordered sequence with the same words. Moreover, the expected length of the proposed codes in this setting is asymptotically within a constant factor of $\frac{5}{3}$ of the lower bound.**

I. Introduction

Many commonly encountered data sources produce sequences of data where the sequential information is meaningful. Characters must be placed in a particular order to form understandable words; the order of words follows syntactic rules to reflect the meaning of a sentence; a song is recognizable if its notes are played in the right order.

However, in a number of relevant data processing and communication tasks, the order of information is not required. For example, many natural language processing methods operate on the so-called "bags-of-words" representation of a document, discarding all order information and only keeping record of which words appear in the document and how many times each appears. In mathematical terms, the bag-of-words is a multiset—a set in which some elements may appear multiple times. Similarly, when storing the items in an inventory, the order in which entries are recorded does not matter as long as the items and counts are all preserved[1].

The order information which is discarded by the non-injective mapping from a sequence to the corresponding multiset is equivalent to a permutation (when a multiset has m words, approximately $\log_2 m!$ bits are required to encode such a permutation), and so one would expect that encoding only the multiset could result in significant savings [1].

In this paper we propose and analyze a practical method for compressing a multiset. Our method is built upon a data structure called a *trie* (also known as a prefix tree). Tries were introduced independently by La Briandais [2] and Fredkin [3], with Fredkin suggesting the name to indicate their use for information re*trie*val while simultaneously evoking the tree-like structure. Tries naturally capture the set of elements appearing in a multiset. We augment this structure with counts of how many times each element appears in the multiset. When encoding a multiset containing m words drawn (with repetition) from a set of size 2^n, the computational complexity of our encoder is $O(m(n+\log(m)))$ and the complexity of our decoder is $O(mn)$. For the case where words are drawn from an i.i.d. Bernoulli $1/2$ source, we derive an expression for the expected code length and show that it asymptotically is within a constant of a lower bound on the multiset entropy when $m, n \to \infty$, $m = 2^n/c$ for some $c > 1$. Our results are based on combinatorial arguments related to the structure of random tries.

A. Related work

Cover [4] proposes a uniquely decipherable compression scheme for sets of binary sequences based on index functions. Later, Lempel [5] observes that order is not always important in a message and proposes the development of multiset decipherable codes—codes which are not necessarily uniquely decipherable but for which every decoding results in the same multiset. In contrast, the codes developed in this paper are uniquely decipherable (i.e., prefix codes).

In the series of papers [6], [7], [8], Varshney and Goyal study information theoretic limits and universal codes for unordered information. For the case where the words to be encoded are produced by a finite-alphabet source, they point out that encoding the multiset is equivalent to encoding the corresponding histogram. Reznik [9] introduces an elegant combinatorial approach to encoding and enumerating histograms. When considering a histogram of m elements in 2^n bins, the encoding complexity is $\Omega(m2^n)$ operations.

Reznik [10], [11] considers the problem of encoding a *set* of words. These codes have recently been applied to the problem of efficiently encoding sets of image features in a system for mobile visual search [12]. Reznik's approach uses *digital search trees* (DSTs) to organize the set. Each vertex in the DST corresponds to one word in the set. The code is a concatenation of the structure of the resulting DST, which encapsulates the prefix of each word, and the word suffixes, which are not encoded by the tree. The resulting code is shown to offer improvements close to the $\log m!$ rate savings one would hope for when the source is memoryless. Interestingly, the

[1]On the other hand, the order and way in which data is stored may have important implications for the computational complexity of decoding, searching or processing the encoded data set.

structure of the DST, and subsequently its encoding, depends on the order in which words of the set are processed during its formation. This would seem to suggest that there is some inherent redundancy to these codes.

II. Coding of sequences

Suppose we have m discrete random variables $X_1, \ldots, X_m$, where each X_i takes values in the space $\mathcal{X}$ of size $|\mathcal{X}| = 2^n$. Assume that all X_i are independent and identically distributed (i.i.d.), with distribution $\mathbb{P}(X_i = x) = p(x) = p$ for all X_i and all $x \in \mathcal{X}$. We are interested in efficiently encoding information about these variables.

A classical result of Shannon [13] shows that the number of bits required to encode the random variable X with values in $\mathcal{X}$ is bounded from below by its entropy. More specifically, let $C : \mathcal{X} \to \mathcal{D}^*$ be a code such that a realization x of a random variable X is encoded using strings of arbitrary length over the alphabet $\mathcal{D}$. Let $C(x)$ be the codeword corresponding to x, and let $l(x)$ denote the length of this codeword. Here we will be primarily interested in the binary case, $\mathcal{D} = \{0, 1\}$. The code C is lossless or non-singular if each $x \in \mathcal{X}$ is mapped to a unique value in $\mathcal{D}^*$.

A well-known consequence of the Kraft inequality [14] is that the expected length of the uniquely decipherable source code C is lower-bounded by the entropy of X, i.e.,

$$L = \sum_{x \in \mathcal{X}} p(x) l(x) \geq H(X) \, , \tag{1}$$

where $H(X) = -\sum_{x \in \mathcal{X}} p(x) \log_2 p(x)$ is the binary entropy function. In fact, one can make the stronger statement that, out of all possible codes, the one with minimum expected length L_X satisfies

$$H(X) \leq L_X \leq H(X) + 1 \, . \tag{2}$$

Hence, if we assume that the X_i are uniformly distributed, then $H(X) = n$, and since $X_1, \ldots, X_m$ are i.i.d., the number of bits needed to transmit the ordered sequence $S = (X_1, \ldots, X_m)$ is at least $L_S \geq mn$. Of course, this bound is achieved when considering the representation that simply lists all codewords in the sequence, and so $L_S = mn$.

III. Lower-bound for coding multisets

Let $Y = \{X_1, X_2, \ldots X_m\}$ denote the random multiset obtained by taking the values appearing in S in an arbitrary order. We are now interested in finding the minimum number of bits required to encode this multisetset.

Let $\mathcal{Y}_m$ denote the collection of all multisets containing m objects drawn (with repetition) from $\mathcal{X}$, where $m < \frac{1}{2} \cdot 2^n = 2^{n-1}$. Let $c \triangleq 2^n/m > 2$. The size of $\mathcal{Y}_m$ is given by the multiset coefficient,

$$\begin{aligned} \left(\binom{2^n}{m}\right) &= \binom{2^n + m - 1}{m} \\ &= \frac{\prod_{k=1}^{m} 2^n + k - 1}{m!} \, . \end{aligned} \tag{3}$$

Although all X_i are assumed to be i.i.d. and uniform, the induced distribution on $\mathcal{Y}_m$ is not uniform. Nevertheless, we have the following result.

Theorem 1. *For sufficiently large n, the binary entropy of the random vector Y as above is bounded from below by*

$$H(Y) \geq 2^n \cdot \left(\log_2(c) + \log_2\left(1 - \frac{1}{c}\right)\right) \cdot \left(1 - e^{-\frac{1}{c}}\right) - \epsilon, \tag{4}$$

for any small $\epsilon > 0$.

Proof: Start by using the fact that $H(Y) \geq H(Z(Y))$ where $Z(Y)$ is the set containing all the elements in Y without repetition. Note that possible sets $Z(Y)$ are uniform. Then lower bound $H\big(Z(Y)\big||Z(Y)| = m'\big)$ using $\log_2(m'!) \leq m' \log_2(m')$ and Stirling's approximation. Use the fact that $\log(m')$ is increasing with m' to substitute m' with m where possible. Finally, sum $H\big(Z(Y)\big||Z(Y)| = m'\big)p(|Z(Y)| = m')$ for all $m' \leq m$ to obtain the result. ∎

Remark. Note that when c grows, the lower bound in (4) becomes

$$H(Y) \geq \underbrace{mn - \log_2(m!) - \epsilon}_{\text{Denote by: } H^-(Y)} \, .$$

On the other hand, $H(Y)$ can be upper bounded using (3) and some elementary manipulations, which leads to

$$\begin{aligned} H(Y) &\leq \log_2\left(\left(\left(1 + \frac{m-1}{2^n}\right) 2^n\right)^m\right) - \log_2(m!) \\ &\leq \underbrace{mn + m \log_2\left(1 + \frac{m-1}{2^n}\right) - \log_2(m!)}_{\text{Denote by: } H^+(Y)} \, . \end{aligned} \tag{5}$$

Given the upper bound on $H(Y)$, we can quantify the rate savings obtained by encoding the multiset Y instead of the sequence S by examining a bound on the ratio of their expected lengths,

$$\frac{L_Y}{L_S} \leq 1 + \frac{m \log_2(1 + \frac{m-1}{2^n}) - \log_2(m!)}{mn} \, . \tag{6}$$

Note that this ratio can also be lower bounded as follows:

$$\frac{L_Y}{L_S} \geq 1 - \frac{\log_2(m!)}{mn} \, . \tag{7}$$

When $c = 2^n/m$ is large, the upper and lower bounds on the multiset entropy are tight:

$$H^-(Y) \underset{n \to \infty}{\sim} H^+(Y) \underset{n \to \infty}{\sim} m(n - \log_2(m)) \, . \tag{8}$$

It follows that

$$\frac{L_Y}{L_S} \underset{n \to \infty}{\sim} 1 - \frac{\log_2(m)}{n} \underset{n \to \infty}{\sim} \frac{\log_2(c)}{n} \, , \tag{9}$$

which means that the gain in the expected lengths when comparing the ordered sequence S to the corresponding unordered multiset Y can be made arbitrarily large.

In the next section we introduce an algorithm to encode such unordered multisets Y with a short expected length.

IV. A CONSTRUCTIVE ENCODING USING TRIES

In this section, we develop an encoding technique based on tries to efficiently encode multisets. Let us fix $\mathcal{D} = \{0, 1\}$, and let $C : \mathcal{X} \to \{0, 1\}^n$ be a bijective mapping from each element in $\mathcal{X}$ to a length-n binary string. We consider compressing the binary representation C_Y of the multiset $Y \in \mathcal{Y}_m$, where $C_Y = \{C(X_1), C(X_2), \ldots, C(X_m)\}$. Given a multiset C_Y, we denote by $\mu(w), w \in C_Y$ the number of occurrences of w in C_Y, which we call the degree of w. In particular, we have $\sum_{w \in C_Y} \mu(w) = m$.

A trie associated with a set of words W over $\mathcal{D}$ is an edge-labeled tree that contains exactly $|W|$ maximum branches, with labels corresponding to the words in W. For example, the trie associated with the set of words $W = \{00000, 01000, 10000, 01001, 01101\}$ is depicted in Figure 1.

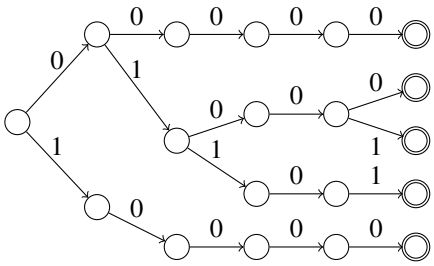

Figure 1. Trie associated with the set of words $W = \{00000, 01000, 10000, 01001, 01101\}$. It contains 5 maximum branches corresponding to the words in W.

We extend this representation to multisets by annotating each leaf (equivalently, each branch) in the trie with the degree corresponding to the associated word. We call such a structure a multitrie. Note that this association between multisets and multitries is a bijection; in the remainder of the paper we do not distinguish a multiset from its associated multitrie.

To encode a multitrie, we use the following algorithm, named **AlgI**:

1) The labels of all branches (together with their corresponding degrees) are listed in lexicographical order.
2) For all words, except for the first word, each bitstring is replaced by its longest suffix that differs from the corresponding suffix contained in the previous word.
3) Any occurrence of the pair of bits `01` in the remaining words is replaced by the chain `0101`.
4) The chain `01` is added at the end of each word.
5) If the degree of the word in the multiset is greater than one, a string of `0`'s is appended to the end of each word. The number of `0`'s appended is equal to the degree of that word in the multiset.
6) Finally, all obtained words are concatenated to form the encoded string representing the multiset.

Note that this algorithm has computational complexity that is $O(m(n + \log(m)))$, since Steps 2–6 involve traversing all words in the set, and Step 1 involves sorting the words[2]. We denote by f the function that associates a multitrie with its encoding using this algorithm.

Step	Content
1	$[\texttt{00000:}1, \texttt{01000:}1, \texttt{01001:}2, \texttt{01101:}1, \texttt{10000:}1]$
2	$[\texttt{00000:}1, \texttt{1000:}1, \texttt{1:}2, \texttt{101:}1, \texttt{10000:}1]$
3	$[\texttt{00000:}1, \texttt{1000:}1, \texttt{1:}2, \texttt{10101:}1, \texttt{10000:}1]$
4	$[\texttt{0000001:}1, \texttt{100001:}1, \texttt{101:}2, \texttt{1010101:}1, \texttt{1000001:}1]$
5	$[\texttt{0000001}, \texttt{100001}, \texttt{10100}, \texttt{1010101}, \texttt{1000001}]$
6	`00000011000011010010101011000001`

Figure 2. The six steps corresponding to the encoding of the multitrie $C_Y = \{00000, 01000, 10000, 01001, 01001, 01101\}$. When relevant, the degree of a branch is added at the end of its labels.

For example, consider the multiset C_Y which contains the same words as in W with the difference that `01001` occurs twice. Figure 2 illustrates the six steps corresponding to the encoding of C_Y.

This encoding is lossless, which means that f is injective. Moreover, each step in **AlgI** is reversible. Indeed, to see that, please observe that after Step 2, each remaining suffix (except for the first word in the lexicographical order) must start with `1`. Therefore, the maximal substring of the form $(01)^{2i+1}$ can only appear at the end of each word (when the subsequent `0`'s are discarded). We use this property to design the decoding algorithm **AlgII** which computes the inverse mapping f^{-1}:

1) Search for all *maximal* occurrences of a pattern of the form $(01)^{2i+1}$ or $(01)^{2i+1}0(0)^+$ (end of the word), and split the string after each such occurence. Each obtained sub-string corresponds to a distinct word in the multiset.
2) Remove the `0`'s at the end of each obtained word and set its degree to be the number of `0`'s. If there are no `0`'s, then set its degree to one.
3) Remove the two last symbols (`01`) for each word.
4) Replace maximal occurrences of $(0101)^i$ in each word by $(01)^i$.
5) For all words, starting with the second word and up to the last one, add the prefix using the previous word. The resulting words are the branches characterizing the multitrie.

Note that the complexity of this decoding algorithm is $O(mn)$ since each step involves traversing the entire bitstring. Figure 3 illustrates the five steps corresponding to the decoding of the string obtained at the last step of Figure 2.

The next section analyzes the expected length of encoded multisets using **AlgI**.

V. PERFORMANCE OF ALGI FOR MULTITRIE ENCODING

Let us fix a multiset with the parameters of Section IV. Throughout this section we assume that the words w in the multiset C_Y are i.i.d. samples from a Bernoulli $1/2$ source.

[2]We assume the machine performing the encoding operates on words of length at least n bits. If this is not the case, then each comparison in Step 1 has complexity $O(n)$ and so the overall encoding complexity is $O(mn \log m)$.

Step	Content
0	00000011000011010011010101011000001
1	[0000001, 100001, 10100, 1010101, 1000001]
2	[0000001:1, 100001:1, 101:2, 1010101:1, 1000001:1]
3	[00000:1, 1000:1, 1:2, 10101:1, 10000:1]
4	[00000:1, 1000:1, 1:2, 101:1, 10000:1]
5	[00000:1, 01000:1, 01001:2, 01101:1, 10000:1]

Figure 3. The five steps corresponding to the decoding of the string obtained in Step 6 in Figure 2. When relevant, the degree of a branch is added at the end of its labels.

The expected length of the encoded string associated with such a multiset using **AlgI** can be expressed as a function of n and m. More precisely, this expected length L can be decomposed into four terms,

$$L = L_2 + 2L_3 + 2L_4 + L_5 \ , \tag{10}$$

where L_2 corresponds to the expected total number of bits in the strings obtained after Step 2, L_3 to the the expected number of duplicated pairs 01 in Step 3, L_4 to the expected number of chains 01 added in Step 4, and L_5 to the expected number of 0's added in Step 5 of **AlgI**.

In the following lemmas, we develop the expected values of L_2 and L_3.

Lemma 2. Let $E_k = 2^k(1-(1-2^{-k})^m)$. Then

$$L_2 = \sum_{k=1}^{n} E_k \ .$$

Proof: A multitrie contains at most 2^k edges at depth k. Based on the source statistics, the probability that any of these edges is used by a word is $p_k = 2^{-k}$. Given m messages, the probability that the multitrie contains a particular edge is obtained by considering the converse of the probability that this edge is not used, the latter being $(1-p_k)^m$. Thus the expected number of edges at depth k in the multitrie is given by E_k. The expected total number of bits after Step 2 is equal to the number of edges in the multitrie which is obtained by summing E_k over all levels $k = 1, \ldots, n$. ∎

Lemma 3.

$$L_3 \leq \frac{L_2 - E_n}{3} + \frac{n}{6} + \frac{1}{3} \ . \tag{11}$$

Proof: We want to bound the number of appearances of 01 in any string obtained after Step 2 of **AlgI**. In Step 2, the first word in lexicographical order is left untouched, and all remaining words are shortened. We estimate the expected length ℓ of each of the shortened strings using Lemma 2. Note that E_n is the number of unique words appearing in the multiset, and thus the number of leafs in the multitrie. Consequently,

$$\ell = \frac{L_2 - n}{E_n - 1}. \tag{12}$$

Next, note that the distribution of pairs of bits in these words is not uniform. More precisely, the probabilities (a) to observe the pair 00 and (b) to observe the pair 10 in the produced words are both higher than that of observing the pair 01. To show (a), note that the word of the form a01b can remain as a result of applying Step 2, only if there is no word of the form a00c appearing beforehand. Therefore it is more likely to observe 00 than 01.

Moreover, to show (b), note that by construction of the multitrie, except for the first word, each suffix remaining after Step 2 begins with a 1. Thus, each word that contains 01 also contains 10. Moreover, between each pair of occurrences of 01 within one word, 10 also appears at least once. Hence, it is also more likely to observe 10 than 01.

Denote by $p_{\mathtt{XY}}$ the probability to observe the pair XY. It follows that $3p_{01} + p_{11} \leq 1$, and thus $p_{01} \leq \frac{1}{3}$.

Let L_3' denote the expected number of pairs 01 in any bitstring except for the first one obtained after Step 2 of **AlgI**. Since there are $\ell - 1$ pairs of symbols in a word of length ℓ, it follows that

$$L_3' = (E_n - 1)(\ell - 1)/3 \tag{13}$$

$$\leq \frac{L_2 - n - E_n + 1}{3} \ . \tag{14}$$

Finally, the first word contains a maximum of $n/2$ 01's. Thus, $L_3 = \frac{n}{2} + L_3'$, and we obtain the desired result. ∎

Using these results, we can bound the total expected length to code a multiset of m words drawn independently and uniformly from a set of size $|\mathcal{X}| = 2^n$.

Theorem 4.

$$L \leq \frac{5}{3}\sum_{k=1}^{n} E_k + \frac{4m}{3} + \frac{2n}{3} + \frac{2}{3} \ . \tag{15}$$

Proof: It follows directly from the definitions of Steps 4 and 5 in **AlgI** that $L_4 = E_n$ and $L_5 = m - E_n$. Use this in equation (10), together with Lemmas 2 and 3 and the fact that $m \geq E_n$, to obtain the claim. ∎

Armed with this bound on the expected code length, we can compare the performance of the proposed multitrie code to the bounds discussed in Section III. Note that

$$\sum_{k=1}^{n} E_k \leq 2^{n+1} \ , \tag{16}$$

and so Theorem 4 gives:

$$L \leq \frac{5}{3}2^{n+1} + \frac{4m}{3} + \frac{n}{3} + \frac{2}{3} \ . \tag{17}$$

It follows that

$$\frac{L}{L_S} \leq \frac{5 \times 2^{n+1}}{3mn} + \frac{4}{3n} + \frac{1}{3m} + \frac{2}{3mn} \ . \tag{18}$$

In particular, if there exists $c > 2$ such that $m = 2^n/c$ we obtain

$$\frac{L}{L_S} \leq \frac{10c}{3n} + \frac{4}{3n} + \frac{c}{3 \cdot 2^n} + \frac{2c}{3n \cdot 2^n} \underset{n\to\infty}{\longrightarrow} 0 \ . \tag{19}$$

In other words, and not surprisingly, the multitrie compression technique can result in an arbitrarily large gain in the expected

length when compared to that of a corresponding ordered sequence S.

Under the same scaling behavior, and provided that there exists an integer $\ell \geq 0$ such that $c = 2^\ell$, the bound in (16) can be sharpened to obtain a stronger result. Note that the condition $m = 2^{n-l}$ is not restrictive, since $\sum_{k=1}^{n} E_k$ is non-increasing with c and thus the bound for any c is given by rounding up to the next largest power of two.

Theorem 5. *Let ℓ be a non-negative integer, and $c = 2^\ell$. Then for any $\epsilon > 0$, there exists a positive integer n_0 such that for any $n \geq n_0$ we have*

$$\frac{L}{L_Y} \leq \frac{5}{3}\left[\frac{2}{\log_2(c)} + c(1-e^{-\frac{1}{c}})\right] + \epsilon \underset{c\to\infty}{\longrightarrow} \frac{5}{3} + \epsilon \,. \quad (20)$$

Proof: Let us first split the sum into two parts at the index $k' = n - \ell$:

$$\sum_{k=1}^{n} E_k = \sum_{k=1}^{k'} E_k + \sum_{k=k'+1}^{n} E_k \,. \quad (21)$$

The first part can be upper bounded by $2^{k'+1}$. In the second part, note that there are ℓ terms in the sum and each term is bounded by $E_k \leq E_n$. Thus, $\sum_{k=k'+1}^{n} E_k \leq \ell E_n$. Moreover, since $(1-\frac{1}{n})^r \leq e^{r/n}$,

$$E_n \underset{n\to\infty}{\sim} 2^n(1-e^{-\frac{1}{c}}) \,. \quad (22)$$

Thus, we obtain

$$\sum_{k=1}^{n} E_k \leq L_2^+ \,, \quad (23)$$

where $L_2^+ \underset{n\to\infty}{\sim} 2^{n+1}/c + \ell 2^n(1-e^{-\frac{1}{c}})$.

We conclude by incorporating these results together with (9) into (15). ∎

It follows from Theorem 5 that for a fixed value of $c > 2$, the ratio between the expected length of the string compressed using the multitrie method and the lower bound approaches constant when n grows. This ratio can be made arbitrarily close to $5/3$ by taking c large enough.

Figure 4 depicts (a) the ratio of L obtained by simulations to theoretical values of L_s; (b) the ratio of theoretical values of L_Y to L_S. Both ratios are functions of n, for $c = 1$ and $c = 10$.

VI. Conclusion

We introduced an algorithm (**AlgI**) to compress multisets of binary words obtained using a Bernoulli 1/2 source. When the number of words in the multiset scales as $m = 2^n/c$, $c > 1$ as $n \to \infty$, with n being the length of words, we proved this algorithm is asymptotically at a constant factor (at most 5/3) from the lower bound introduced in Section III. Moreover, this algorithm has a very low computational complexity ($O(m(n+\log m))$ for encoding and $O(mn)$ for decoding), which makes it scalable for large inputs.

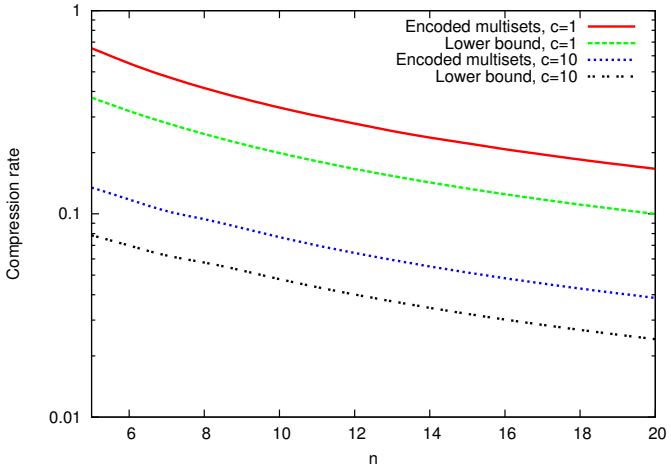

Figure 4. The figure presents: (a) the ratio of the value of L obtained by simulations to the theoretical value of L_S; (b) the ratio of the theoretical values of L_Y to L_S. Both ratios are functions of n, for $c = 1$ and $c = 10$.

We believe the results presented in this manuscript can easily be extended to more general sources by first compressing all the individual words, and then applying **AlgI**.

Future work includes providing algorithms for common operations on the encoded multisets—including union, intersection, lookup—with limited complexity, and considering lossy algorithms to further approach the lower bound.

Acknowledgements: The authors thank Luc Devroye for an illuminating discussion and Yuriy Reznik for pointing out useful related work. This work is supported in part by the Natural Sciences and Engineering Research Council of Canada.

References

[1] L. Varshney, "Optimal information storage: Nonsequential sources and neural channels," S.M. Thesis, Dept. Electrical Engineering and Computer Science, MIT, Boston, MA, Jun. 2006.

[2] R. de La Briandais, "File searching using variable length keys," in *Proc. Western Joint Computer Conference*, 1959, pp. 295–298.

[3] E. Fredkin, "Trie memory," *Communications of the ACM*, vol. 3, no. 9, pp. 490–499, 1960.

[4] T. Cover, "Enumerative source encoding," *IEEE Trans. on Inf. Theory*, vol. 19, no. 1, pp. 73–77, 1963.

[5] A. Lempel, "On multiset decipherable codes," *IEEE Trans. Inf. Theory*, vol. 32, no. 5, pp. 714–716, Sep. 1986.

[6] L. Varshney and V. Goyal, "Toward a source coding theory for sets," in *Proc. IEEE DCC*, Snowbird, UT, Mar. 2006.

[7] ——, "Ordered and disordered source coding," in *Proc. ITA*, San Diego, CA, Feb. 2006.

[8] ——, "On universal coding of unordered data," in *Proc. ITA*, San Diego, CA, Jan. 2007.

[9] Y. Reznik, "An algorithm for quantization of discrete probability distributions," in *Proc. IEEE DCC*, Snowbird, UT, Mar. 2011.

[10] ——, "Coding of sets of words," in *Proc. IEEE DCC*, Snowbird, UT, Mar. 2011.

[11] ——, "Codes for unordered sets of words," in *Proc. IEEE ISIT*, St. Petersburg, Russia, Jul. 2011.

[12] V. Chandrasekhar, S. Tsai, Y. Reznik, G. Takacs, D. Chen, and B. Girod, "Compressing features sets with digital search trees," in *Proc. IEEE Int. Workshop on Mobile Vision (IWMV)*, Barcelona, Spain, Nov. 2011.

[13] C. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, Jul., Oct. 1948.

[14] L. Kraft, "A device for quantizing, grouping, and coding amplitude modulated pulses," M.S. Thesis, Dept. Electrical Engineering and Computer Science, MIT, 1949.