Smooth Representation of Rankings

(Invited Paper)

Arya Mazumdar
Department of ECE
University of Minnesota– Twin Cities
Minneapolis, MN 55455
email: arya@umn.edu

Olgica Milenkovic
Department of ECE
University of Illinois
Urbana, IL 61801
email: milenkov@illinois.edu

***Abstract*—An encoding of data for digital storage is called *smooth*, if, for any small change in the raw data, only a proportionately small change has to be made in the encoded data. In this paper, we consider the problem of smooth encoding for ordinal data, i.e., *ranking* of objects. It is shown that, simple efficient smooth encoding for storage (as well as compression) of rankings is possible with respect to the well-known Kendall τ metric on permutations.**

I. Introduction

Ordinal data conveys information about the relational properties of data instead of their absolute numerical values. In various applications, such as in social sciences or economics, the true value of data is frequently missing, erroneous or misleading, while on the other hand relative ordering or ranking of data is easily computable.

Ranking of data is represented usually by a permutation. However digital storage in databases with conventional media is done by writing integers from a discrete alphabet (most frequently, binary) in the memory. Hence, for storage purposes, ordinal datasets, namely rankings, are mapped to binary vectors. This encoding is straight-forward in most of the cases, and it is well-known that to store a permutation of n elements, one needs about $n \log_2 n$ bits of storage.

In many cases, the complete ranking is too big to store, and only some partial information might be of interest. For example, in a recommendation system for restaurants (such as YELP.com), the ranking at the top matters more than the intricacies at the bottom of the list. For these reasons, a ranking may be compressed allowing some distortion in recovery. In such cases, the amount of storage needed, and the error in recovery allowed, naturally present a rate-distortion theory of permutations [1].

Most of the ordinal data, generated from recommender systems and search engines and/or stored in distributed storage systems (such as, cloud storage), undergoes small, but frequent changes which have to be accounted for in the representation/compressed format. *Smooth representation* techniques satisfy the need for fast processing of such volatile data. Updating stored digital data requires access and time, and therefore bandwidth and energy. If the changes in ordinal data are large, then it may be inevitable to update a large number of stored symbols. Nevertheless, it is desirable to have sublinear (in storage amount) updates in encoded data for comparably small changes in the original ordinal data, especially in BigData applications. This problem, under the name of *update-efficient error-correcting codes*, was recently addressed in [2] in the context of binary data and channel coding. For compression of standard binary data, the smoothness of lossless compression was studied in [3].

It is not clear in advance if smooth compression is plausible for ordinal data, given that near-optimal sorting and compression algorithms map points in S_n to points $\{0,1\}^n$ that obey a near-uniform distribution and may lie at a large average Hamming distance from each other. In this paper, we show that such smooth representation and compression is possible for permutations with respect to the very popular and useful Kendall τ distance and its generalizations. In the following section, we present some preliminaries regarding the space of permutations. In Sections III and IV, we, respectively for the lossless and the lossy cases, show the smoothness property of a storage algorithm.

II. Preliminaries

A *permutation* is a bijection $\sigma : [n] \to [n]$, that is, for any $i, j \in [n]$, $i \neq j$, one has $\sigma(i) \neq \sigma(j)$. We let S_n denote the set of all permutations of the set $[n]$, i.e., the symmetric group of order $n!$. For any $\sigma \in S_n$, we write $\sigma = (\sigma(1), \sigma(2), \cdots, \sigma(n))$, the *vector representation* of a permutation, where $\sigma(i)$ is the image of $i \in [n]$ under the permutation σ. The inverse σ^{-1} of a permutation σ is a permutation in which an element and the position that it occupies are exchanged.

To store a permutation on n elements, one needs $\log_2(n!) \approx n \log_2(n/e)$ bits. However, to directly store the vector representation naively, one ends up using $n \log_2 n$ bits. To get rid of this extra $O(n)$ bits, we may encode the permutation to its Lehmer code first.

A permutation $\sigma \in S_n$ may be uniquely encoded to its *Lehmer code* (also called the *inversion vector*), $\mathbf{x}_\sigma \in \mathcal{H}_n \triangleq [0,1] \times [0,2] \times \cdots \times [0, n-1]$, where $\mathbf{x}_\sigma(i) = |\{j \in [n] : j < i+1, \sigma^{-1}(j) > \sigma^{-1}(i+1)\}|$, $i = 1, \ldots, n-1$. In words, $\mathbf{x}_\sigma(i), i = 1, \ldots, n-1$ is the number of inversions (pairs out of order) in the permutation σ for which $i+1$ is the first element. For instance, we have

σ	$\mathbf{x}_\sigma$
2 1 6 4 3 7 5 9 8	1 0 1 0 3 1 0 1

It is well known that the Lehmer code is bijective, and simple reconstruction algorithms are known [4]. They are of particular interest for compression, given that they perform a zero-distortion transform from the domain of ordinals to the domain of quantitative data.

Now, the ith coordinate of a Lehmer code can be stored using $\lceil \log_2(i+1) \rceil$ bits without any loss. Hence, a total of $\sum_{i=1}^{n-1} \lceil \log_2(i+1) \rceil \approx \log_2(n!)$ bits are used for storage.

An representation $f : S_n \to \{0,1\}^m$ is termed as a *source code* here. When $m \geqslant \log_2(n!)$ the representation is lossless, and otherwise the representation is a *lossy compression*.

Definition 1: A source code $f : S_n \to \{0,1\}^m$ is said to be (u,t)-smooth with respect to a distortion measure $d : S_n \times S_n \to \mathbb{R}_+ \cup \{0\}$, if for any $\pi, \sigma \in S_n$, $d(\pi,\sigma) \leqslant u$ implies $d_H(f(\pi), f(\sigma)) \leqslant t$, where $d_H(\cdot,\cdot)$ denotes the Hamming distance.

The metric on permutation that is of interest to us in this paper is the Kendall τ metric. Kendall τ distance is a natural metric on permutations, that was introduced in [5] for application in statistics, and then adapted as a suitable measure for various application in computer science and bioinformatics (e.g., [6], [7]) and most recently in error-correcting codes [8], [9].

Definition 2: The Kendall τ distance $d_\tau(\cdot,\cdot)$ between any two permutations is the minimum number of pairwise adjacent swaps needed to convert one to other. Formally, let $I(\sigma)$ denote the number of inversions in $\sigma \in S_n$. For any two permutations σ, π,

$$\begin{aligned} d_\tau(\sigma,\pi) &= I(\sigma\pi^{-1}) \\ &= |\{(i,j) \in [n]^2 : \pi^{-1}(i) > \pi^{-1}(j), \sigma^{-1}(i) < \sigma^{-1}(j)\}|. \end{aligned}$$

In what follows, we consider a (u,t)-smooth codes in the context of Kendall metric. A practical generalization of d_τ, called *weighted Kendall metric* has recently been proposed in [10], [11]. The generalization of the result presented in this paper to weighted Kendall metric will appear in the full version of this paper.

Finally, other interesting and practical distances on permutations, such as the Ulam metric [12], may be considered to construct smooth codes [1].

III. Lossless representation

In this section, we exhibit a smooth mapping for the Kendall τ distance which illustrates the approach to be pursued for more general distance measures. The ideas behind the techniques were used in the context of constructing error-correcting codes in [14].

[1]Another popular metric, called the Spearman's footrule, does not lead to any interesting question beyond the case for Kendall τ metric, because of their equivalence within a constant factor [13].

Define the ℓ_1 distance function on $\mathcal{H}_n$ as

$$d_1(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{n-1} |\mathbf{x}(i) - \mathbf{y}(i)|, \qquad (\mathbf{x},\mathbf{y} \in \mathcal{H}_n) \tag{1}$$

with the computations performed over the set of integers. For instance, if $\sigma_1 = (2,1,4,3)$ and $\sigma_2 = (2,3,4,1)$, then the Lehmer codes of σ_1 and σ_2 equal $\mathbf{x}_{\sigma_1} = (1,0,1)$ and $\mathbf{x}_{\sigma_2} = (1,1,1)$. To compute the distance $d_\tau(\sigma_1,\sigma_2)$, we note that $\sigma_1^{-1} = \sigma_1$ and so $I(\sigma_2\sigma_1^{-1}) = I((1,4,3,2)) = 3$. Observe that the mapping $\sigma \to \mathbf{x}_\sigma$ is a weight-preserving bijection between S_n and $\mathcal{H}_n$, although it is not distance preserving. Indeed, $d_\tau(\sigma_1,\sigma_2) = 3$ while $d_1(\mathbf{x}_{\sigma_1}, \mathbf{x}_{\sigma_2}) = 1$. It can actually be shown [8] that $d_\tau(\sigma_1,\sigma_2) \geqslant d_1(\mathbf{x}_{\sigma_1}, \mathbf{x}_{\sigma_2})$.

We make use of the binary Gray code, which is a mapping ϕ_s from the ordered set of integers $[0, 2^s - 1]$ to $\{0,1\}^s$, with the property that the images of two successive integers differ in exactly one bit. Suppose that $b_{s-1}b_{s-2}\ldots b_0$, $b_i \in \{0,1\}, 0 \leqslant i < s$, is the binary representation of an integer $u \in [0, 2^s - 1]$. By definition, set $b_s = 0$ and construct $\phi_s(u) = (g_{s-1}, g_{s-2}, \ldots, g_0)$, where

$$g_j = (b_j + b_{j+1}) \pmod 2, \qquad j = 0, 1, \ldots s-1. \tag{2}$$

The Gray map for the first 10 integers looks as follows:

$$\begin{array}{rcc} 0| & & 00000000 \\ 1| & & 00000001 \\ 2| & & 00000011 \\ 3| & & 00000010 \\ \hline 4| & & 00000110 \\ 5| & \longrightarrow & 00000111 \\ 6| & & 00000101 \\ 7| & & 00000100 \\ \hline 8| & & 00001100 \\ 9| & & 00001101 \\ \vdots & & \vdots \end{array}$$

Note the "reflective" nature of the map: the last 2 bits of the second block of four are a reflection of the last 2 digits of the first block with respect to the horizontal line; the last 3 bits of the second block of eight follow a similar rule, and so on.

It is straightforward to see that $|i-j| \geqslant d_H(\phi_s(i), \phi_s(j))$ for any two $i,j \in \{0,1,\ldots,2^s-1\}$.

Let us now describe the representation mapping for permutations as follow. Assume that one is given $\sigma \in S_n$, with Lehmer code $\mathbf{x}_\sigma = (\mathbf{x}_\sigma(1), \ldots, \mathbf{x}_\sigma(n-1))$. Let the representation map $f : S_n \to \{0,1\}^m$ be of the form

$$f(\sigma) = (\phi_{m_1}(\mathbf{x}_\sigma(1)), \ldots, \phi_{m_{n-1}}(\mathbf{x}_\sigma(n-1))),$$

where $m_i \equiv \lceil \log_2 i \rceil$ and $m = \sum_{i=1}^{n-1} m_i < \log_2(n!) + n$. The underlying mapping is (u,u)-smooth for any integer u. In addition, the overhead of the method compared to the optimal (not necessarily smooth) compression scheme is at most n bits.

The above analysis works when $u = o(n \log n)$. But, the largest possible value of the Kendall distance is $\binom{n}{2}$. Indeed, this maximum occurs when two permutations are exactly in the reverse order in the vector notation (e.g., $(1,2,\ldots,n)$ and (n, n-1, …, 1)). Hence a small change in ranking may mean

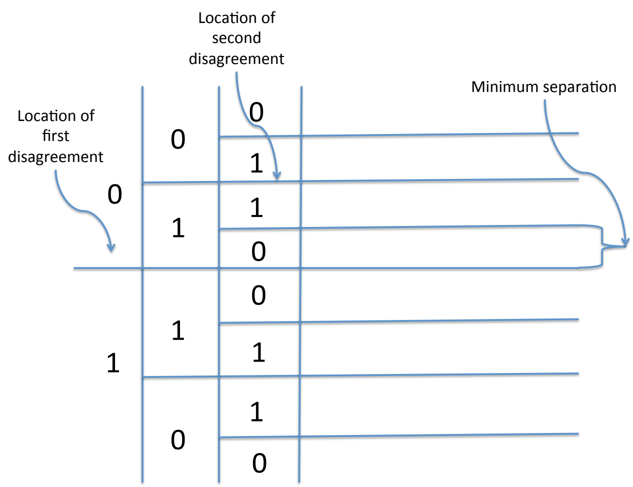

Fig. 1. The table of Gray map showing that the difference between two integer is $2^{\ell-1}+1$ where ℓ is the location (indexed from right) of the second left-most coordinate where the Gray codes differ.

an update with Kendall τ distance $u = O(n^{3/2})$. However, because $m < n\log_2 n$, this will mean that almost all bits needs to be changed in the digital representation, and we end up being far from smooth. In the following we will show that, this is not the case. The mapping remains (u,t)-smooth even when $u = \Omega(n \log n)$, with a t much less than m.

To show this we need to use the "reflective" property [14] of the Gray map.

Lemma 1: If $|i-j| \leqslant w$, then $d_H(\phi_s(i), \phi_s(j)) \leqslant \lceil \log_2 w \rceil + 1$, for all integer $w \geqslant 1$.

Proof: Suppose, $d_H(\phi_s(i), \phi_s(j)) = t$. Let $i_0 < i_1 < \cdots < i_{t-1}$ are the coordinates where $\phi_s(i)$ and $\phi_s(j)$ differ (the coordinates start with 0 and ends at $s-1$). From the reflective property of the Gray map (see, Fig. 1) the difference between i and j is at least $2^{i_{t-2}}+1$. Now, the minimum value of i_{t-2} is $t-2$. Hence, whenever, $d_H(\phi_s(i), \phi_s(j)) \geqslant t$, we must have $|i-j| \geqslant 2^{t-2}+1$.

Let us now substitute for $t = \lceil \log_2 w \rceil + 2$ above. Hence, $d_H(\phi_s(i), \phi_s(j)) > \lceil \log_2 w \rceil + 1$ implies $|i-j| \geqslant 2^{\lceil \log_2 w \rceil} + 1 \geqslant w+1$. Therefore, whenever, $|i-j| \leqslant w$, we must have, $d_H(\phi_s(i), \phi_s(j)) \leqslant \lceil \log_2 w \rceil + 1$. ∎

The next theorem shows how this lemma translate to the smoothness of entire representation of permutation.

Theorem 2: Suppose, for $\pi, \sigma \in S_n$, $d_\tau(\pi, \sigma) = w$. Then,

$$d_H(f(\sigma), f(\pi)) \leqslant (n-1)\Big(\log_2 \frac{w}{n-1} + 2\Big) \tag{3}$$

Proof: Assume, for σ and π the corresponding two inversion vectors are $\mathbf{x}_\sigma = (\mathbf{x}_\sigma(1), \dots, \mathbf{x}_\sigma(n-1))$ and $\mathbf{x}_\pi = (\mathbf{x}_\pi(1), \dots, \mathbf{x}_\pi(n-1))$. Now,

$$d_1(\mathbf{x}_\sigma, \mathbf{x}_\pi) \leqslant w,$$

and

$$d_1(\mathbf{x}_\sigma(i), \mathbf{x}_\pi(i)) = w_i, \quad i = 1, \dots, n-1.$$

Clearly, $\sum_{i=1}^{n-1} w_i \leqslant w$, and from Lemma 1,

$$d_H(\phi_{m_i}(\mathbf{x}_\sigma(i)), \phi_{m_i}(\mathbf{x}_\pi(i))) \leqslant \lceil \log_2 w_i \rceil + 1,$$

whenever $w_i \geqslant 1$. Hence, assuming total number of nonzero w_is to be N,

$$\begin{aligned} d_H(f(\sigma), f(\pi)) &\leqslant \sum_{i=1 \text{ to } n-1: w_i \neq 0} (\lceil \log_2 w_i \rceil + 1) \\ &\leqslant \sum_{i=1 \text{ to } n-1: w_i \neq 0} \log_2 w_i + 2N \\ &\leqslant N \log_2 \frac{\sum_{i=1}^{n-1} w_i}{N} + 2N \\ &\leqslant (n-1)\Big(\log_2 \frac{w}{n-1} + 2\Big), \end{aligned}$$

where we have used the concavity of the log function and Jensen's inequality. ∎

From the above, we claim that the representation of permutations by f defined above is smooth everywhere. First of all, the mapping is (u,t)-smooth, with $t = \min\Big\{u, (n-1)\Big(\log_2 \frac{u}{n-1} + 2\Big)\Big\}$. In particular, when $u = O(n)$, $t = u$. But, recall, Kendall distance can be as large as $\binom{n}{2} = \Omega(n^2)$. The above theorem tells us when $u = \Omega(n^{1+\delta})$, $t = \delta n \log_2 n + O(n) = \delta m + o(m)$, for any $\delta > 0$.

In the next section we show that, not only the lossless representation, but a the scalar quantization lossy source coding algorithm [1, Sec. V] for compression of rankings, is also smooth.

IV. Lossy compression

Assume, we want to construct a (u,t)-smooth code that compresses the permutations with a worst-case distortion guarantee D. In the lossy compression, a source code $\mathcal{C} \in S_n$ is a set of permutation such that, given any $\sigma \in S_n$, there exists an $\pi \in \mathcal{C}$ such that $d_\tau(\sigma, \pi) \leqslant D$. The elements of $\mathcal{C}$ are then represented and stored in binary. A lossy compression algorithm, a simple scalar quantization, is presented in [1, Sec. V].

The algorithm is a generalization of the lossless representation above. Given any permutation σ, in the first step of the algorithm, its Lehmer code $\mathbf{x}_\sigma$ is found. Then each coordinate of $\mathbf{x}_\sigma$ is independently quantized with uniform quantization levels. For example, the ith coordinate may take value in $0, 1, \dots, i$. We divide this coordinate in ℓ_i different levels with any two levels uniformly separated by $2D_i$. We must have, $D_i = \frac{2(i+1)D}{(n+1)(n-2)}$, so that, $\sum_{i=1}^{n-1} D_i = D$. For details, we refer the reader to [1] [2].

Suppose this compression algorithm is (u,t)-smooth. We will find the values of t given u next. Suppose, u_i is the absolute difference in each coordinate of the Lehmer code from a permutation σ and its updated version σ' (that is, $|\mathbf{x}_\sigma(i) - \mathbf{x}_{\sigma'}(i)| = u_i$). Clearly, $\sum_{i=1}^{n-1} u_i = d_1(\mathbf{x}_\sigma, \mathbf{x}_{\sigma'}) \leqslant$

[2] The distortion here is measured as the ℓ_1 distance of Lehmer code, and not the Kendall τ distance. However, it was shown in [15] recently that this distance is very close to Kendall τ distance and the average case rate-distortion properties are same for these two distances.

$d_\tau(\sigma, \sigma') = u$. Now if q is the quantizer function in the ith coordinate, then clearly, $|q(x_\sigma(i)) - q(x_{\sigma'}(i))| \leqslant u_i + 2D_i$, or difference in each coordinate is upper bounded by $u_i + 2D_i$.

But from Lemma 1,

$$\begin{aligned} t &\leqslant \sum_{i=1 \text{ to } n-1: u_i + 2D_i \neq 0} \lceil \log_2(u_i + 2D_i) \rceil + 1 \\ &\leqslant \sum_{i=1 \text{ to } n-1: u_i + 2D_i \neq 0} \log_2(u_i + 2D_i) + 2(n-1) \\ &\leqslant (n-1) \log_2 \frac{\sum_{i=1}^{n-1}(u_i + 2D_i)}{n-1} + 2(n-1) \\ &\leqslant (n-1)\Big(\log_2 \frac{u + 2D}{n-1} + 2 \Big). \end{aligned}$$

And, hence, conclusions similar to the previous section can be drawn. In particular, when $u = O(n^{1+\delta})$ for some $0 < \delta \leqslant 1$, and $u > D$, then the amount of update is only $\delta n \log_2 n + O(n)$.

Acknowledgement: A. Mazumdar's work was supported in part by NSF CCF1318093 and a grant from University of Minnesota.

REFERENCES

[1] Da Wang, Arya Mazumdar, and G.W. Wornell. A rate-distortion theory for permutation spaces. In *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*, pages 2562–2566. IEEE, 2013.

[2] Arya Mazumdar, Gregory W. Wornell, and Venkat Chandar. Update efficient codes for error correction. In *Proc. Int. Symp. Inform. Theory*, pages 1558–1562, Cambridge, MA, July 2012.

[3] Andrea Montanari and Elchanan Mossel. Smooth compression, Gallager bound and nonlinear sparse-graph codes. In *Proc. Int. Symp. Inform. Theory*, pages 2474–2478, Toronto, Canada, July 2008.

[4] Donald Ervin Knuth. *The art of computer programming, Volume 3: sorting and searching*. Addison-Wesley, 1998.

[5] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 1938.

[6] John G Kemeny. Mathematics without numbers. *Daedalus*, 88(4):577–591, 1959.

[7] V Yu Popov. Multiple genome rearrangement by swaps and by element duplications. *Theoretical Computer Science*, 385(1):115–126, 2007.

[8] Anxiao Jiang, Moshe Schwartz, and Jehoshua Bruck. Error-correcting codes for rank modulation. In *Information Theory, 2008. ISIT 2008. IEEE International Symposium on*, pages 1736–1740. IEEE, 2008.

[9] Alexander Barg and Arya Mazumdar. Codes in permutations and error correction for rank modulation. *Information Theory, IEEE Transactions on*, 56(7):3158–3165, 2010.

[10] Farzad Farnoud and Olgica Milenkovic. Sorting of permutations by cost-constrained transpositions. *Information Theory, IEEE Transactions on*, 58(1):3–23, 2012.

[11] Fardad Raisali, Farzad Farnoud Hassanzadeh, and Olgica Milenkovic. Weighted rank aggregation via relaxed integer programming. In *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*, pages 2765–2769. IEEE, 2013.

[12] Daniele Mundici. The logic of ulams game with lies. *Knowledge, Belief, and Strategic Interaction (Castiglioncello, 1989), Cambridge Stud. Probab. Induc. Decis. Theory, Cambridge Univ. Press, Cambridge*, pages 275–284, 1992.

[13] Persi Diaconis and Ronald L Graham. Spearman's footrule as a measure of disarray. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 262–268, 1977.

[14] Arya Mazumdar, Alexander Barg, and Gilles Zemor. Constructions of rank modulation codes. *IEEE transactions on information theory*, 59(2):1018–1029, 2013.

[15] Da Wang, Arya Mazumdar, and Gregory W Wornell. Lossy compression of permutations. In *preprint*, 2014.