Random Permutation Codes
Lossless Source Coding of Non-Sequential Data

Daniel Severo

The Edward S. Rogers Sr. Department of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

VECTOR INSTITUTE

Advisors: Ashish Khisti and Alireza Makhzani

October 7, 2024

Outline

Papers

(NeurIPS 2024) **Severo**, Khisti, Makhzani. *Random Cycle Coding: Lossless Compression of Cluster Assignments via Bits-Back Coding*.

(Under Review) **Severo**, Su, Liu, Johnson, Karrer, Van den Broeck, Muckley, Ullrich. *Enhancing and Evaluating Probabilistic Circuits for High-Resolution Lossless Image Compression*.

(NeurIPS 2024) Kunze, **Severo**, Zani, van de Meent, Townsend. *Entropy Coding of Large Unordered Data Structures.*

(ICLR 2024) **Severo**, Theis, Ballé. *The Unreasonable Effectiveness of Linear Prediction as a Perceptual Metric* . `https://arxiv.org/abs/2310.05986`

(ICLR 2024) Kunze, **Severo**, Zani, van de Meent, Townsend. *Entropy Coding of Unordered Data Structures.* **Oral (top 12% of accepted papers at ICML NCW Workshop).** `https://openreview.net/forum?id=afQuNt3Ruh`

(ICML 2023) Neklyudov, Brekelmans, **Severo**, Makhzani. *Action Matching: A Variational Method for Learning Stochastic Dynamics from Samples.* `https://arxiv.org/abs/2210.06662`

(ICML 2023) **Severo**, Townsend, Khisti, Makhzani. *Random Edge Coding: One-Shot Bits-Back Coding of Large Labeled Graphs*. `https://arxiv.org/abs/2305.09705`

(JSAIT 2023) **Severo**, Townsend, Khisti, Makhzani, Ullrich. *Compressing Multisets with Large Alphabets using Bits-Back Coding*. **Best Paper Award at NeurIPS DGM Workshop 2021.** `https://arxiv.org/abs/2107.09202`

(ICASSP 2022) Domanovitz, **Severo**, Khisti, Yu. *Data-Driven Optimization for Zero-Delay Lossy Source Coding with Side Information.* `https://ieeexplore.ieee.org/document/9747823`

(ICML 2021) Ruan*, Ullrich*, **Severo***, Townsend, Khisti, Doucet, Makhzani, Maddison. *Improving Lossless Compression Rates via Monte Carlo Bits-Back Coding*. **Long Talk (top 15% of accepted papers).** `https://arxiv.org/abs/2102.11086`

(BSC 2021) **Severo**, Elad Domanovitz, Ashish Khisti. *Regularized Classification-Aware Quantization*. `https://arxiv.org/abs/2107.09716`

Papers

(NeurIPS 2024) **Severo**, Khisti, Makhzani. *Random Cycle Coding: Lossless Compression of Cluster Assignments via Bits-Back Coding*.

(Under Review) **Severo**, Su, Liu, Johnson, Karrer, Van den Broeck, Muckley, Ullrich. *Enhancing and Evaluating Probabilistic Circuits for High-Resolution Lossless Image Compression.*

(NeurIPS 2024) Kunze, **Severo**, Zani, van de Meent, Townsend. *Entropy Coding of Large Unordered Data Structures.*

(ICLR 2024) **Severo**, Theis, Ballé. *The Unreasonable Effectiveness of Linear Prediction as a Perceptual Metric* . https://arxiv.org/abs/2310.05986

(ICLR 2024) Kunze, **Severo**, Zani, van de Meent, Townsend. *Entropy Coding of Unordered Data Structures.* **Oral (top 12% of accepted papers at ICML NCW Workshop).** https://openreview.net/forum?id=afQuNt3Ruh

(ICML 2023) Neklyudov, Brekelmans, **Severo**, Makhzani. *Action Matching: A Variational Method for Learning Stochastic Dynamics from Samples.* https://arxiv.org/abs/2210.06662

(ICML 2023) **Severo**, Townsend, Khisti, Makhzani. *Random Edge Coding: One-Shot Bits-Back Coding of Large Labeled Graphs*. https://arxiv.org/abs/2305.09705

(JSAIT 2023) **Severo**, Townsend, Khisti, Makhzani, Ullrich. *Compressing Multisets with Large Alphabets using Bits-Back Coding*. **Best Paper Award at NeurIPS DGM Workshop 2021.** https://arxiv.org/abs/2107.09202

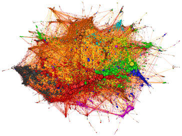
(ICASSP 2022) Domanovitz, **Severo**, Khisti, Yu. *Data-Driven Optimization for Zero-Delay Lossy Source Coding with Side Information.* https://ieeexplore.ieee.org/document/9747823

(ICML 2021) Ruan*, Ullrich*, **Severo***, Townsend, Khisti, Doucet, Makhzani, Maddison. *Improving Lossless Compression Rates via Monte Carlo Bits-Back Coding*. **Long Talk (top 15% of accepted papers).** https://arxiv.org/abs/2102.11086

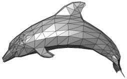
(BSC 2021) **Severo**, Elad Domanovitz, Ashish Khisti. *Regularized Classification-Aware Quantization.* https://arxiv.org/abs/2107.09716

Motivation

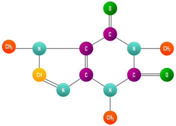
Non-sequential data is everywhere



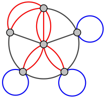
Social networks



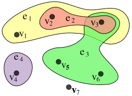
3D meshes



Molecules



Graphs



Hypergraphs

$$\{\{a, b\}, \{c, \{\ldots\}\}\}$$

Problem setting

Problem Setting

Given data modelled by a discrete random variables $X \sim P_X$, with a **large** alphabet $\mathcal{X}$,

Problem Setting

Given data modelled by a discrete random variables $X \sim P_X$, with a **large** alphabet $\mathcal{X}$, find a lossless code

$$C \colon \mathcal{X} \mapsto \{0, 1, 00, 01, \dots\},$$

Problem Setting

Given data modelled by a discrete random variables $X \sim P_X$, with a **large** alphabet $\mathcal{X}$, find a lossless code

$$C \colon \mathcal{X} \mapsto \{0, 1, 00, 01, \dots\},$$

such that the **single-letter** code-length is close to the information content for all $x \in \mathcal{X}$

$$|C(x)| \approx -\log P_X(x).$$

Problem Setting

Given data modelled by a discrete random variables $X \sim P_X$, with a **large** alphabet $\mathcal{X}$, find a lossless code

$$C \colon \mathcal{X} \mapsto \{0, 1, 00, 01, \dots\},$$

such that the **single-letter** code-length is close to the information content for all $x \in \mathcal{X}$

$$|C(x)| \approx -\log P_X(x).$$

Later, X will be a non-sequential data type (e.g., set, graph).

Problem Setting

Given data modelled by a discrete random variables $X \sim P_X$, with a **large** alphabet $\mathcal{X}$, find a lossless code

$$C\colon \mathcal{X} \mapsto \{0, 1, 00, 01, \dots\},$$

such that the **single-letter** code-length is close to the information content for all $x \in \mathcal{X}$

$$|C(x)| \approx -\log P_X(x).$$

Later, X will be a non-sequential data type (e.g., set, graph). Assume $\mathcal{X}$ is too large to be held in memory.

Random Order Coding

Random Order Coding: Problem Statement

Given a sequence of i.i.d. discrete random variables

$$Z^n = (Z_1, \dots, Z_n),$$

Random Order Coding: Problem Statement

Given a sequence of i.i.d. discrete random variables

$$Z^n = (Z_1, \ldots, Z_n),$$

perform lossless coding of the **multiset**

$$X = \mathcal{M} = \{Z_1, \ldots, Z_n\},$$

Random Order Coding: Problem Statement

Given a sequence of i.i.d. discrete random variables

$$Z^n = (Z_1, \ldots, Z_n),$$

perform lossless coding of the **multiset**

$$X = \mathcal{M} = \{Z_1, \ldots, Z_n\},$$

with single-letter code-length

$$C(\mathcal{M}) \approx -\log P_{\mathcal{M}}(\mathcal{M}) = -\log P_{Z^n}(z^n) - \log M, \tag{1}$$

Random Order Coding: Problem Statement

Given a sequence of i.i.d. discrete random variables

$$Z^n = (Z_1, \ldots, Z_n),$$

perform lossless coding of the **multiset**

$$X = \mathcal{M} = \{Z_1, \ldots, Z_n\},$$

with single-letter code-length

$$C(\mathcal{M}) \approx -\log P_{\mathcal{M}}(\mathcal{M}) = -\log P_{Z^n}(z^n) - \log M, \qquad (1)$$

where the constant M is known as the *multinomial coefficient* of $\mathcal{M}$

$$M = \frac{n!}{\prod_{z \in \mathcal{Z}} \mathcal{M}(z)!} \leq n!. \qquad (2)$$

Random Order Coding: encode $\mathcal{M}$ w/ $-\log P_{\mathcal{M}}(\mathcal{M})$ bits

Construct order information iteratively by "sampling without replacement" from $\mathcal{M}$. Alternate:

1. Decode sample (w.o. replacement) from $\mathcal{M}$
2. Encode sampled element using P_Z

until $\mathcal{M}$ is depleted.

Random Order Coding: encode $\mathcal{M}$ w/ $-\log P_{\mathcal{M}}(\mathcal{M})$ bits

Construct order information iteratively by “sampling without replacement” from $\mathcal{M}$. Alternate:

1. Decode sample (w.o. replacement) from $\mathcal{M}$
2. Encode sampled element using P_Z

until $\mathcal{M}$ is depleted.

$\{\mathtt{a}, \mathtt{b}, \mathtt{b}\}$

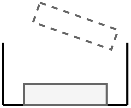
$L(\mathcal{M}) = \varepsilon$

Random Order Coding: encode $\mathcal{M}$ w/ $-\log P_{\mathcal{M}}(\mathcal{M})$ bits

Construct order information iteratively by "sampling without replacement" from $\mathcal{M}$. Alternate:

1. Decode sample (w.o. replacement) from $\mathcal{M}$
2. Encode sampled element using P_Z

until $\mathcal{M}$ is depleted.



$\{\mathtt{a}, \mathtt{b}, \mathtt{b}\}$

$$L(\mathcal{M}) = \varepsilon - \log \frac{1}{2/3}$$

Random Order Coding: encode $\mathcal{M}$ w/ $-\log P_{\mathcal{M}}(\mathcal{M})$ bits

Construct order information iteratively by “sampling without replacement” from $\mathcal{M}$. Alternate:

1. Decode sample (w.o. replacement) from $\mathcal{M}$
2. Encode sampled element using P_Z

until $\mathcal{M}$ is depleted.

$\{\mathtt{a}, \mathtt{b}\}$

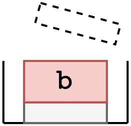
$$L(\mathcal{M}) = \varepsilon - \log \frac{1}{2/3} + \log \frac{1}{P_Z(\mathtt{b})}$$

Random Order Coding: encode $\mathcal{M}$ w/ $-\log P_{\mathcal{M}}(\mathcal{M})$ bits

Construct order information iteratively by "sampling without replacement" from $\mathcal{M}$. Alternate:

1. Decode sample (w.o. replacement) from $\mathcal{M}$
2. Encode sampled element using P_Z

until $\mathcal{M}$ is depleted.



$\{\mathtt{a}, \mathtt{b}\}$

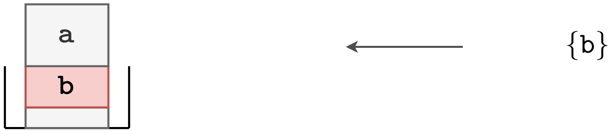
$$L(\mathcal{M}) = \varepsilon - \log \frac{1}{2/3} + \log \frac{1}{P_Z(\mathtt{b})} - \log \frac{1}{1/2}$$

Random Order Coding: encode $\mathcal{M}$ w/ $-\log P_{\mathcal{M}}(\mathcal{M})$ bits

Construct order information iteratively by "sampling without replacement" from $\mathcal{M}$. Alternate:

1. Decode sample (w.o. replacement) from $\mathcal{M}$
2. Encode sampled element using P_Z

until $\mathcal{M}$ is depleted.



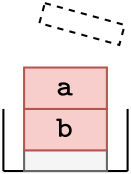
$$L(\mathcal{M}) = \varepsilon - \log \frac{1}{2/3} + \log \frac{1}{P_Z(\mathtt{b})} - \log \frac{1}{1/2} + \log \frac{1}{P_Z(\mathtt{a})}$$

Random Order Coding: encode $\mathcal{M}$ w/ $-\log P_{\mathcal{M}}(\mathcal{M})$ bits

Construct order information iteratively by "sampling without replacement" from $\mathcal{M}$. Alternate:

1. Decode sample (w.o. replacement) from $\mathcal{M}$
2. Encode sampled element using P_Z

until $\mathcal{M}$ is depleted.



$\{\texttt{b}\}$

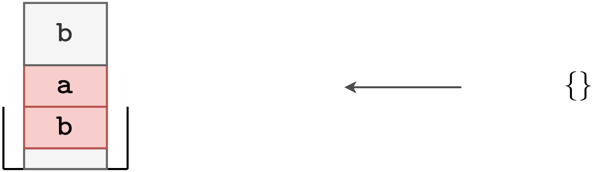
$$L(\mathcal{M}) = \varepsilon - \log \frac{1}{2/3} + \log \frac{1}{P_Z(\texttt{b})} - \log \frac{1}{1/2} + \log \frac{1}{P_Z(\texttt{a})} - \log \frac{1}{1/1}$$

Random Order Coding: encode $\mathcal{M}$ w/ $-\log P_{\mathcal{M}}(\mathcal{M})$ bits

Construct order information iteratively by "sampling without replacement" from $\mathcal{M}$. Alternate:

1. Decode sample (w.o. replacement) from $\mathcal{M}$
2. Encode sampled element using P_Z

until $\mathcal{M}$ is depleted.



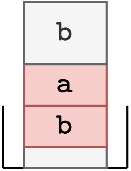
$$L(\mathcal{M}) = \varepsilon - \log \frac{1}{2/3} + \log \frac{1}{P_Z(\mathtt{b})^2} - \log \frac{1}{1/2} + \log \frac{1}{P_Z(\mathtt{a})} - \log \frac{1}{1/1}$$

Random Order Coding: encode $\mathcal{M}$ w/ $-\log P_{\mathcal{M}}(\mathcal{M})$ bits

Construct order information iteratively by "sampling without replacement" from $\mathcal{M}$. Alternate:

1. Decode sample (w.o. replacement) from $\mathcal{M}$
2. Encode sampled element using P_Z

until $\mathcal{M}$ is depleted.



$\{\}$

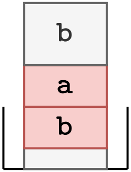
$$L(\mathcal{M}) = \varepsilon + \log \frac{1}{P_Z(\mathtt{b})^2 P_Z(\mathtt{a})} - \log \frac{1}{(2/3)(1/2)(1/1)}$$

Random Order Coding: encode $\mathcal{M}$ w/ $-\log P_{\mathcal{M}}(\mathcal{M})$ bits

Construct order information iteratively by "sampling without replacement" from $\mathcal{M}$. Alternate:

1. Decode sample (w.o. replacement) from $\mathcal{M}$
2. Encode sampled element using P_Z

until $\mathcal{M}$ is depleted.



$\{\}$

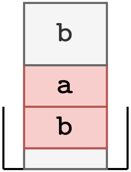
$$L(\mathcal{M}) = \varepsilon + \log \frac{1}{P_{Z^n}(\texttt{bab})} - \log M$$

Random Order Coding: encode $\mathcal{M}$ w/ $-\log P_{\mathcal{M}}(\mathcal{M})$ bits

Construct order information iteratively by “sampling without replacement” from $\mathcal{M}$. Alternate:

1. Decode sample (w.o. replacement) from $\mathcal{M}$
2. Encode sampled element using P_Z

until $\mathcal{M}$ is depleted.



$\{\}$

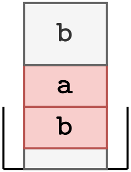
$$L(\mathcal{M}) = \varepsilon + \log \frac{1}{P_{\mathcal{M}}(\{\mathtt{a}, \mathtt{b}, \mathtt{b}\})}$$

Random Order Coding: encode $\mathcal{M}$ w/ $-\log P_{\mathcal{M}}(\mathcal{M})$ bits

Construct order information iteratively by "sampling without replacement" from $\mathcal{M}$. Alternate:

1. Decode sample (w.o. replacement) from $\mathcal{M}$
2. Encode sampled element using P_Z

until $\mathcal{M}$ is depleted.



$\{\}$

$$L(\mathcal{M}) = \varepsilon + \log \frac{1}{P_{\mathcal{M}}(\{\mathtt{a}, \mathtt{b}, \mathtt{b}\})}$$

Complexity: $\mathcal{O}(n \cdot P_Z + n \cdot \log m)$

Random Order Coding: what about ϵ?

How many initial bits ε do we need?

Random Order Coding: what about ϵ?

How many initial bits ε do we need?

Let z^n be the sequence generated by ROC (e.g., $z^n =$ *bab*)

Random Order Coding: what about ϵ?

How many initial bits ε do we need?

Let z^n be the sequence generated by ROC (e.g., $z^n =$ *bab*)

Problem: the stack can deplete at any decoding/sampling step!

Random Order Coding: what about ϵ?

How many initial bits ε do we need?

Let z^n be the sequence generated by ROC (e.g., $z^n = \textit{bab}$)

Problem: the stack can deplete at any decoding/sampling step!

However, the average increase at any step is positive:

$$\begin{aligned}
\mathbb{E}\left[\Delta_i \,|\, \mathcal{M}\right] &= \mathbb{E}\left[\log P_{Z_i \,|\, \overline{\mathcal{M}}_i}(Z_i \,|\, \overline{\mathcal{M}}_i) - \log P_Z(Z_i) \middle| \mathcal{M}\right] && (3)\\
&= D_{\mathrm{KL}}(P_{Z_i \,|\, \overline{\mathcal{M}}_i}(\cdot \,|\, \overline{\mathcal{M}}_i) \,\|\, P_Z) && (4)\\
&\geq 0 && (5)
\end{aligned}$$

Multisets as Equivalence Classes

Multisets as Equivalence Classes

Recall the problem statement ...

Multisets as Equivalence Classes

Recall the problem statement ...

Given a sequence of i.i.d. discrete random variables

$$Z^n = (Z_1, \ldots, Z_n),$$

perform lossless coding of the **multiset**

$$X = \mathcal{M} = \{Z_1, \ldots, Z_n\},$$

Multisets as Equivalence Classes

Recall the problem statement ...

Given a sequence of i.i.d. discrete random variables

$$Z^n = (Z_1, \ldots, Z_n),$$

perform lossless coding of the **multiset**

$$X = \mathcal{M} = \{Z_1, \ldots, Z_n\},$$

Let $\mathcal{Z}^n$ be the alphabet of Z^n.

Multisets as Equivalence Classes

Recall the problem statement ...

Given a sequence of i.i.d. discrete random variables

$$Z^n = (Z_1, \ldots, Z_n),$$

perform lossless coding of the **multiset**

$$X = \mathcal{M} = \{Z_1, \ldots, Z_n\},$$

Let $\mathscr{Z}^n$ be the alphabet of Z^n.

Partition $\mathscr{Z}^n$ into subsets $x \subset \mathscr{Z}^n$ of equivalent sequences.

Multisets as Equivalence Classes

Recall the problem statement ...

Given a sequence of i.i.d. discrete random variables

$$Z^n = (Z_1, \ldots, Z_n),$$

perform lossless coding of the **multiset**

$$X = \mathcal{M} = \{Z_1, \ldots, Z_n\},$$

Let $\mathcal{Z}^n$ be the alphabet of Z^n.

Partition $\mathcal{Z}^n$ into subsets $x \subset \mathcal{Z}^n$ of equivalent sequences.

Each x can be mapped uniquely to some $\mathcal{M}$ such that

$$P_{\mathcal{M}}(\mathcal{M}) = P_X(x).$$

Multisets as Equivalence Classes

Formally, ...

Given any two $z^n, w^n \in \mathcal{Z}^n$,

Multisets as Equivalence Classes

Formally, ...

Given any two $z^n, w^n \in \mathcal{Z}^n$, let $z^n \sim w^n$ if, and only if, there exists a permutation σ such that

Multisets as Equivalence Classes

Formally, ...

Given any two $z^n, w^n \in \mathcal{Z}^n$, let $z^n \sim w^n$ if, and only if, there exists a permutation σ such that

$$(z_1, \ldots, z_n) = (w_{\sigma(1)}, \ldots, w_{\sigma(n)}).$$

Multisets as Equivalence Classes

Formally, ...

Given any two $z^n, w^n \in \mathcal{Z}^n$, let $z^n \sim w^n$ if, and only if, there exists a permutation σ such that

$$(z_1, \ldots, z_n) = (w_{\sigma(1)}, \ldots, w_{\sigma(n)}).$$

In the equivalence class view, a multiset $X = \mathcal{M}$ is a random variable with alphabet equal to the **quotient set**: $\mathcal{X} = \mathcal{Z}^n / \sim$.

Multisets as Equivalence Classes: Example

Let $\mathscr{Z} = \{\blacktriangle, \bigstar, \square\}, n = 3$.

Multisets as Equivalence Classes: Example

Let $\mathcal{Z} = \{\blacktriangle, \bigstar, \square\}, n = 3$.

The alphabet $X = \mathcal{Z}^3/\!\sim$ is shown below alongside the corresponding multiset.

Multisets as Equivalence Classes: Example

Let $\mathscr{Z} = \{\blacktriangle, \bigstar, \square\}, n = 3$.

The alphabet $X = \mathscr{Z}^3/\sim$ is shown below alongside the corresponding multiset.

Multiset	Equivalence class in $\mathscr{Z}^3/\sim$
$\{\blacktriangle, \blacktriangle, \blacktriangle\}$	$\{\blacktriangle\blacktriangle\blacktriangle\}$
$\{\bigstar, \bigstar, \bigstar\}$	$\{\bigstar\bigstar\bigstar\}$
$\{\square, \square, \square\}$	$\{\square\square\square\}$

Multisets as Equivalence Classes: Example

Let $\mathscr{Z} = \{\blacktriangle, \bigstar, \square\}, n = 3$.

The alphabet $X = \mathscr{Z}^3/\!\sim$ is shown below alongside the corresponding multiset.

Multiset	Equivalence class in $\mathscr{Z}^3/\!\sim$
$\{\blacktriangle, \blacktriangle, \blacktriangle\}$	$\{\blacktriangle\blacktriangle\blacktriangle\}$
$\{\bigstar, \bigstar, \bigstar\}$	$\{\bigstar\bigstar\bigstar\}$
$\{\square, \square, \square\}$	$\{\square\square\square\}$
$\{\blacktriangle, \blacktriangle, \bigstar\}$	$\{\blacktriangle\blacktriangle\bigstar, \blacktriangle\bigstar\blacktriangle, \bigstar\blacktriangle\blacktriangle\}$

Multisets as Equivalence Classes: Example

Let $\mathcal{Z} = \{\blacktriangle, \bigstar, \square\}, n = 3$.

The alphabet $X = \mathcal{Z}^3/\sim$ is shown below alongside the corresponding multiset.

Multiset	Equivalence class in $\mathcal{Z}^3/\sim$
$\{\blacktriangle, \blacktriangle, \blacktriangle\}$	$\{\blacktriangle\blacktriangle\blacktriangle\}$
$\{\bigstar, \bigstar, \bigstar\}$	$\{\bigstar\bigstar\bigstar\}$
$\{\square, \square, \square\}$	$\{\square\square\square\}$
$\{\blacktriangle, \blacktriangle, \bigstar\}$	$\{\blacktriangle\blacktriangle\bigstar, \blacktriangle\bigstar\blacktriangle, \bigstar\blacktriangle\blacktriangle\}$
$\{\blacktriangle, \blacktriangle, \square\}$	$\{\blacktriangle\blacktriangle\square, \blacktriangle\square\blacktriangle, \square\blacktriangle\blacktriangle\}$
$\{\blacktriangle, \bigstar, \bigstar\}$	$\{\blacktriangle\bigstar\bigstar, \bigstar\blacktriangle\bigstar, \bigstar\bigstar\blacktriangle\}$
$\{\bigstar, \bigstar, \square\}$	$\{\bigstar\bigstar\square, \bigstar\square\bigstar, \square\bigstar\bigstar\}$
$\{\bigstar, \square, \square\}$	$\{\bigstar\square\square, \square\bigstar\square, \square\square\bigstar\}$
$\{\blacktriangle, \square, \square\}$	$\{\blacktriangle\square\square, \square\blacktriangle\square, \square\square\blacktriangle\}$
$\{\blacktriangle, \bigstar, \square\}$	$\{\blacktriangle\bigstar\square, \blacktriangle\square\bigstar, \bigstar\blacktriangle\square, \bigstar\square\blacktriangle, \square\blacktriangle\bigstar, \square\bigstar\blacktriangle\}$

Multisets as Equivalence Classes: Info. Content

Let $[z^n] \subset \mathcal{Z}^n$ be the equivalence class of z^n.

Multisets as Equivalence Classes: Info. Content

Let $[z^n] \subset \mathcal{Z}^n$ be the equivalence class of z^n.

For any $[z^n] \subset \mathcal{Z}^n$ the multinomial coefficient M is equal $|[z^n]|$.

Multisets as Equivalence Classes: Info. Content

Let $[z^n] \subset \mathcal{Z}^n$ be the equivalence class of z^n.

For any $[z^n] \subset \mathcal{Z}^n$ the multinomial coefficient M is equal $|[z^n]|$.

The information content of the multiset $\mathcal{M}$,

$$-\log P_{\mathcal{M}}(\mathcal{M}) = -\log P_{Z^n}(z^n) - \log M,$$

Multisets as Equivalence Classes: Info. Content

Let $[z^n] \subset \mathcal{Z}^n$ be the equivalence class of z^n.

For any $[z^n] \subset \mathcal{Z}^n$ the multinomial coefficient M is equal $|[z^n]|$.

The information content of the multiset $\mathcal{M}$,

$$-\log P_{\mathcal{M}}(\mathcal{M}) = -\log P_{Z^n}(z^n) - \log M,$$

can be rewritten as,

$$-\log P_X([z^n]) = -\log P_{Z^n}(z^n) - \log|[z^n]|.$$

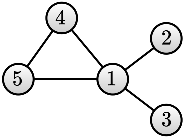
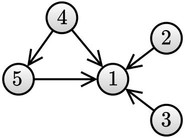
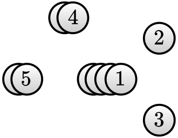
Combinatorial Random Variables

Combinatorial Random Variables

Different $\sim$ result in different non-sequential objects.

Combinatorial Random Variables

Different $\sim$ result in different non-sequential objects.

	Undirected Graphs	Directed Graphs	Multisets
X			
$[Z^n]$	$21, 31, 41, 51, 45$	$21, 31, 41, 51, 45$	2131415145
	$21, 14, 13, 15, 45$	$21, 31, 51, 41, 45$	1111234455
	$\ldots$	$\ldots$	$\ldots$

Combinatorial Random Variables

Restricting the class of $\sim$ allows for efficient algorithms! (see thesis)

Combinatorial Random Variables

Restricting the class of $\sim$ allows for efficient algorithms! (see thesis)

$\sim'$ is *finer* than $\sim$ if any eq. class under $\sim'$ is a subset of some eq. class under $\sim$

Combinatorial Random Variables

Restricting the class of $\sim$ allows for efficient algorithms! (see thesis)

$\sim'$ is *finer* than $\sim$ if any eq. class under $\sim'$ is a subset of some eq. class under $\sim$

Definition (Combinatorial Random Variables - CRVs)

A CRV is a random variable with alphabet equal to the quotient set $\mathcal{Z}^n/\sim$

Combinatorial Random Variables

Restricting the class of $\sim$ allows for efficient algorithms! (see thesis)

$\sim'$ is *finer* than $\sim$ if any eq. class under $\sim'$ is a subset of some eq. class under $\sim$

Definition (Combinatorial Random Variables - CRVs)

A CRV is a random variable with alphabet equal to the quotient set $\mathcal{Z}^n/\!\sim$, where the equivalence relation is *finer* than that of multisets.

Combinatorial Random Variables: Random Perm. Codes

Random Permutation Codes optimally code CRVs.

Combinatorial Random Variables: Random Perm. Codes

Random Permutation Codes optimally code CRVs.

Sets and multisets $\mapsto$ Random Order Coding (ROC)

Combinatorial Random Variables: Random Perm. Codes

Random Permutation Codes optimally code CRVs.

Sets and multisets $\mapsto$ Random Order Coding (ROC)

Graphs, hyper-graphs, multigraphs $\mapsto$ Random Edge Coding (REC)

Combinatorial Random Variables: Random Perm. Codes

Random Permutation Codes optimally code CRVs.

Sets and multisets $\mapsto$ Random Order Coding (ROC)

Graphs, hyper-graphs, multigraphs $\mapsto$ Random Edge Coding (REC)

Partitions and cluster assignments $\mapsto$ Random Cycle Coding (RCC)

Applications

Applications

Savings, compared to coding a sequence, are bounded by

$$\log|[z^n]| \leq \log n!$$

Applications

Savings, compared to coding a sequence, are bounded by

$$\log|[z^n]| \leq \log n!$$

This quantity is non-negligible if the cost of coding Z_i is small.

Applications

Savings, compared to coding a sequence, are bounded by

$$\log|[z^n]| \leq \log n!$$

This quantity is non-negligible if the cost of coding Z_i is small.

Most fruitful application so far: compressing vector database indices.

Applications

Savings, compared to coding a sequence, are bounded by

$$\log |[z^n]| \leq \log n!$$

This quantity is non-negligible if the cost of coding Z_i is small.

Most fruitful application so far: compressing vector database indices.
e.g., FAISS: `https://github.com/facebookresearch/faiss`

Applications

Savings, compared to coding a sequence, are bounded by

$$\log|[z^n]| \le \log n!$$

This quantity is non-negligible if the cost of coding Z_i is small.

Most fruitful application so far: compressing vector database indices.
e.g., FAISS: `https://github.com/facebookresearch/faiss`
Between 15% and 70% savings in realistic use cases!

Thank you!

Random Edge Coding

We want to losslessly compress a (large) labeled graph G at the one-shot rate

$$\log 1/P(G)$$

Random Edge Coding

We want to losslessly compress a (large) labeled graph G at the one-shot rate

$$\log 1/P(G)$$

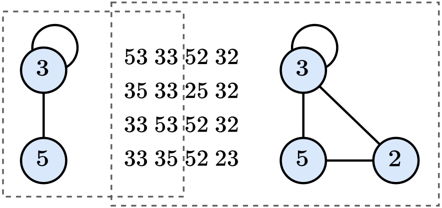
Graphs can be represented as sequences of vertices

Random Edge Coding

We want to losslessly compress a (large) labeled graph G at the one-shot rate

$$\log 1/P(G)$$

Graphs can be represented as sequences of vertices

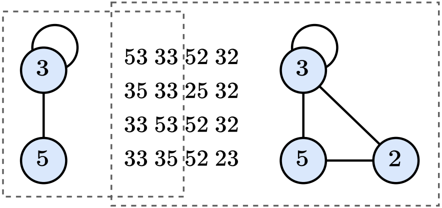


Random Edge Coding

We want to losslessly compress a (large) labeled graph G at the one-shot rate

$$\log 1/P(G)$$

Graphs can be represented as sequences of vertices



Equivalent sequences $\mathbf{v} \sim \mathbf{w}$ map to the same graph

$$53\ 33 \sim 33\ 53 \sim 33\ 35 \sim 35\ 33$$

Random Edge Coding

Graphs are equivalence classes over sequences

Definition (Graphs as Equivalence Classes)

Let $V = \{1, \ldots, n\}$ be the set of nodes,

Random Edge Coding

Graphs are equivalence classes over sequences

Definition (Graphs as Equivalence Classes)

Let $V = \{1, \ldots, n\}$ be the set of nodes, and $\mathbf{v}, \mathbf{w} \in V^{2k}$ sequences of $2k$ elements.

Random Edge Coding

Graphs are equivalence classes over sequences

Definition (Graphs as Equivalence Classes)

Let $V = \{1, \ldots, n\}$ be the set of nodes, and $\mathbf{v}, \mathbf{w} \in V^{2k}$ sequences of $2k$ elements. A graph is an element of $V^{2k}/\sim$,

Random Edge Coding

Graphs are equivalence classes over sequences

Definition (Graphs as Equivalence Classes)

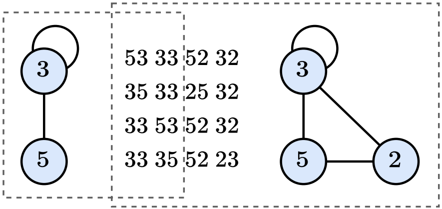
Let $V = \{1, \ldots, n\}$ be the set of nodes, and $\mathbf{v}, \mathbf{w} \in V^{2k}$ sequences of $2k$ elements. A graph is an element of $V^{2k} / \sim$, where $\mathbf{v} \sim \mathbf{w}$ if we can permute edges, and vertices within an edge, of $\mathbf{v}$ to get $\mathbf{w}$.

Random Edge Coding

Graphs are equivalence classes over sequences

Definition (Graphs as Equivalence Classes)

Let $V = \{1, \ldots, n\}$ be the set of nodes, and $\mathbf{v}, \mathbf{w} \in V^{2k}$ sequences of $2k$ elements. A graph is an element of $V^{2k}/\sim$, where $\mathbf{v} \sim \mathbf{w}$ if we can permute edges, and vertices within an edge, of $\mathbf{v}$ to get $\mathbf{w}$.



Random Edge Coding

If $P(\mathbf{v}) = P(\mathbf{w})$ for any $v \sim w$,

Random Edge Coding

If $P(\mathbf{v}) = P(\mathbf{w})$ for any $v \sim w$, then

$$\log 1/P(G) = \log 1/P(\mathbf{v}) - \underbrace{\log\left(\#\text{ of equivalent sequences}\right)}_{\text{excess bits}}$$

Random Edge Coding

If $P(\mathbf{v}) = P(\mathbf{w})$ for any $v \sim w$, then

$$\log 1/P(G) = \log 1/P(\mathbf{v}) - \underbrace{\log(\#\text{ of equivalent sequences})}_{\text{excess bits}}$$

Given $\mathbf{v}$, how many equiv. seqs. are there for non-simple graphs?

Random Edge Coding

If $P(\mathbf{v}) = P(\mathbf{w})$ for any $v \sim w$, then

$$\log 1/P(G) = \log 1/P(\mathbf{v}) - \underbrace{\log(\#\text{ of equivalent sequences})}_{\text{excess bits}}$$

Given $\mathbf{v}$, how many equiv. seqs. are there for non-simple graphs?

We can,

- Permute edges $\rightarrow |E|!$

Random Edge Coding

If $P(\mathbf{v}) = P(\mathbf{w})$ for any $v \sim w$, then

$$\log 1/P(G) = \log 1/P(\mathbf{v}) - \underbrace{\log(\#\text{ of equivalent sequences})}_{\text{excess bits}}$$

Given $\mathbf{v}$, how many equiv. seqs. are there for non-simple graphs?

We can,

- Permute edges $\rightarrow |E|!$
- Permute vertices within an edge $\rightarrow 2^{|E|}$

Random Edge Coding

If $P(\mathbf{v}) = P(\mathbf{w})$ for any $v \sim w$, then

$$\log 1/P(G) = \log 1/P(\mathbf{v}) - \underbrace{\log(\#\text{ of equivalent sequences})}_{\text{excess bits}}$$

Given $\mathbf{v}$, how many equiv. seqs. are there for non-simple graphs?

We can,

- Permute edges $\rightarrow |E|!$
- Permute vertices within an edge $\rightarrow 2^{|E|}$

Excess bits $= \log(|E|!) + |E|$

Random Edge Coding

What are good choices for $P(\mathbf{v})$?

Random Edge Coding

What are good choices for $P(\mathbf{v})$?

Pólya's Urn $\rightarrow$ 0-parameters, fast, and well-studied

Random Edge Coding

What are good choices for $P(\mathbf{v})$?

Pólya's Urn $\rightarrow$ 0-parameters, fast, and well-studied

For $\mathbf{v} = (v_1, v_2, \ldots, v_{2k})$

Random Edge Coding

What are good choices for $P(\mathbf{v})$?

Pólya's Urn $\rightarrow$ 0-parameters, fast, and well-studied

For $\mathbf{v} = (v_1, v_2, \ldots, v_{2k})$

$$P(v_{i+1} \,|\, v^i) \propto d_{v^i}(v_{i+1}) + 1, \tag{6}$$

where $d_{v^i}(v) = \sum_{j=1}^{i} 1\{v = v_j\}$ is the count of vertex v in v^i.

Random Edge Coding

What are good choices for $P(\mathbf{v})$?

Pólya's Urn $\rightarrow$ 0-parameters, fast, and well-studied

For $\mathbf{v} = (v_1, v_2, \ldots, v_{2k})$

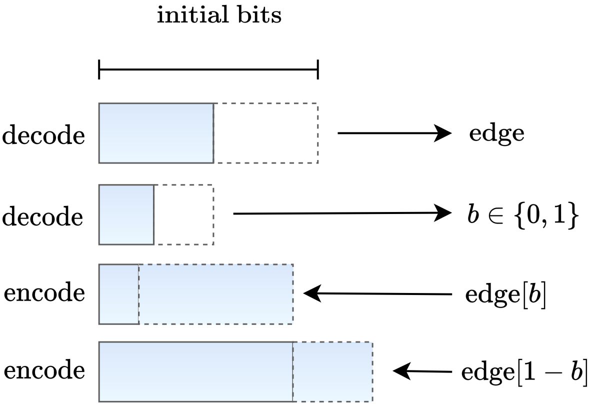
$$P(v_{i+1} \,|\, v^i) \propto d_{v^i}(v_{i+1}) + 1, \tag{6}$$

where $d_{v^i}(v) = \sum_{j=1}^{i} 1\{v = v_j\}$ is the count of vertex v in v^i.

The joint distribution can be expressed as

$$P(\mathbf{v}) = \frac{1}{n^{\uparrow k}} \prod_{v \in [n]} d_{v^k}(v)!, \tag{7}$$

Random Edge Coding



Random Edge Coding

	Social Networks				Others	
	YouTube	FourSq.	Digg	Gowalla	Skitter	DBLP
# Nodes	3,223,585	639,014	770,799	196,591	1,696,415	317,080
# Edges	9,375,374	3,214,986	5,907,132	950,327	11,095,298	1,049,866
$10^6 \times$Density	1.8	15.8	19.8	50.2	7.7	20.9
(Ours) PU w/ REC	**15.19**	9.96	10.62	12.19	14.26	15.92
Pool Comp.	15.38	**9.23**	11.59	**11.73**	7.45	**8.78**
Slashburn	17.03	10.67	**9.82**	11.83	12.75	12.62
Backlinks	17.98	11.69	12.56	15.56	11.49	10.79
List Merging	15.80	9.95	11.92	14.88	**8.87**	14.13