Insertion and Deletion Correction in Polymer-based Data Storage

Anisha Banerjee[1]

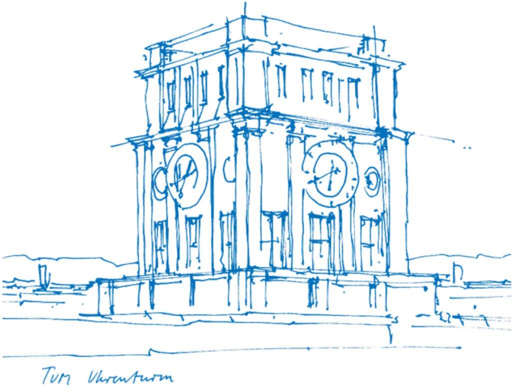
Joint work with

Antonia Wachter-Zeh[1], Eitan Yaakobi[2]

[1]Technical University of Munich
Institute for Communications Engineering

[2]Technion – Israel Institute of Technology
Department of Computer Science

June 28, 2022



Introduction

Problem Statement & Results

Conclusion

Motivation

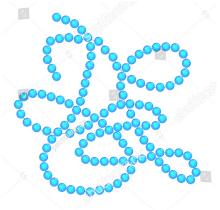
- Need for high-density and reliable data storage media.
  - Molecular storage paradigms, e.g. DNA storage

[OACL17] A. Al Ouahabi, J.-A. Amalian, L. Charles, and J.-F. Lutz, “Mass spectrometry sequencing of long digital polymers facilitated by programmed inter-byte fragmentation”, *Nature Communications*, vol. 8, no. 1, p. 967, Dec. 2017

Motivation

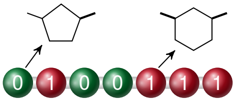
- Need for high-density and reliable data storage media.
  - Molecular storage paradigms, e.g. DNA storage

- One potential candidate is polymer-based data storage[OACL17].
  - Low readout latency.
  - Low storage cost.
  - Common analytical equipment usable.



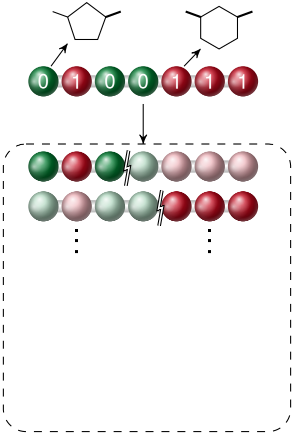
[OACL17] A. Al Ouahabi, J.-A. Amalian, L. Charles, and J.-F. Lutz, “Mass spectrometry sequencing of long digital polymers facilitated by programmed inter-byte fragmentation”, *Nature Communications*, vol. 8, no. 1, p. 967, Dec. 2017

Background



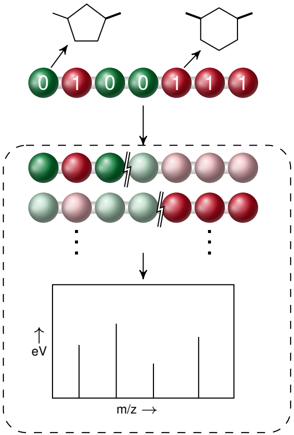
- Encode bits into synthetic macromolecules of distinct masses.

Background



- Encode bits into synthetic macromolecules of distinct masses.
- Read using tandem mass spectrometer.
  - Breaks links & generates fragments.

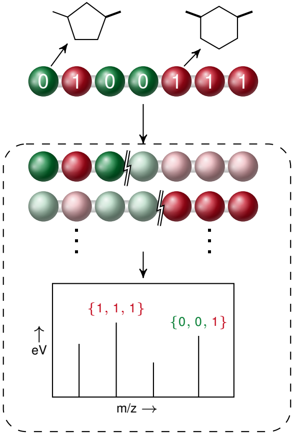
Background



- Encode bits into synthetic macromolecules of distinct masses.

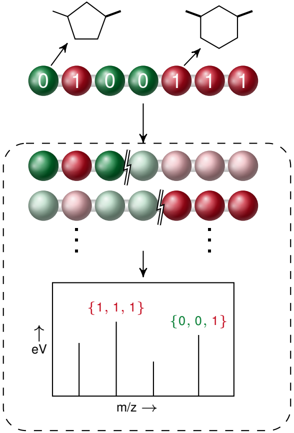
- Read using tandem mass spectrometer.
  - Breaks links & generates fragments.
  - Reports masses of fragments via peak series.

Background



- Encode bits into synthetic macromolecules of distinct masses.
- Read using tandem mass spectrometer.
  - Breaks links & generates fragments.
  - Reports masses of fragments via peak series.
  - Mass suggests number of 0s and 1s in fragment.

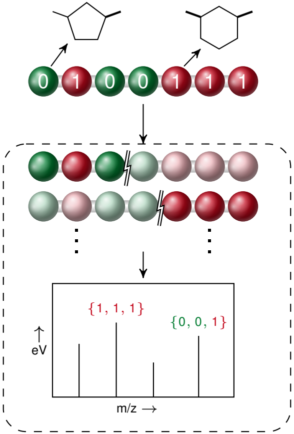
Background



- Encode bits into synthetic macromolecules of distinct masses.
- Read using tandem mass spectrometer.
  - Breaks links & generates fragments.
  - Reports masses of fragments via peak series.
  - Mass suggests number of 0s and 1s in fragment.

composition

Background



- Encode bits into synthetic macromolecules of distinct masses.
- Read using tandem mass spectrometer.
  - Breaks links & generates fragments.
  - Reports masses of fragments via peak series.
  - Mass suggests $\underbrace{\text{number of 0s and 1s}}_{\textbf{composition}}$ in fragment.
- Assume we get compositions of all contiguous substrings.

Model

- For $\boldsymbol{s} \in \{0,1\}^n$, define **multiset** $C_k(\boldsymbol{s})$ as

$$C_k(\boldsymbol{s}) = \{\{s_i, s_{i+1}, \ldots, s_{i+k-1}\} : 1 \leq i \leq n-k+1\}$$

Model

- For $\boldsymbol{s} \in \{0,1\}^n$, define **multiset $C_k(\boldsymbol{s})$** as

$$C_k(\boldsymbol{s}) = \{\{s_i, s_{i+1}, \ldots, s_{i+k-1}\} : 1 \leq i \leq n-k+1\}$$

- Define **composition multiset $C(\boldsymbol{s})$** as

$$C(\boldsymbol{s}) = \{C_k(\boldsymbol{s}) : 1 \leq k \leq n\}$$

Model

- For $\boldsymbol{s} \in \{0,1\}^n$, define **multiset $C_k(\boldsymbol{s})$** as

$$C_k(\boldsymbol{s}) = \{\{s_i, s_{i+1}, \ldots, s_{i+k-1}\} : 1 \le i \le n-k+1\}$$

- Define **composition multiset $C(\boldsymbol{s})$** as

$$C(\boldsymbol{s}) = \{C_k(\boldsymbol{s}) : 1 \le k \le n\}$$

Example

Consider $\boldsymbol{s} = (0,1,0,1)$.

Model

- For $\boldsymbol{s} \in \{0,1\}^n$, define **multiset $C_k(\boldsymbol{s})$** as

$$C_k(\boldsymbol{s}) = \{\{s_i, s_{i+1}, \ldots, s_{i+k-1}\} : 1 \leq i \leq n-k+1\}$$

- Define **composition multiset $C(\boldsymbol{s})$** as

$$C(\boldsymbol{s}) = \{C_k(\boldsymbol{s}) : 1 \leq k \leq n\}$$

Example

Consider $\boldsymbol{s} = (0,1,0,1)$.

$$C_2(\boldsymbol{s})$$

Model

- For $\boldsymbol{s} \in \{0,1\}^n$, define **multiset** $C_k(\boldsymbol{s})$ as

$$C_k(\boldsymbol{s}) = \{\{s_i, s_{i+1}, \ldots, s_{i+k-1}\} : 1 \leq i \leq n-k+1\}$$

- Define **composition multiset** $C(\boldsymbol{s})$ as

$$C(\boldsymbol{s}) = \{C_k(\boldsymbol{s}) : 1 \leq k \leq n\}$$

Example

Consider $\boldsymbol{s} = (0, 1, 0, 1)$.

$$C_2(\boldsymbol{s}) = \{\{0,1\},$$

Model

- For $\boldsymbol{s} \in \{0,1\}^n$, define **multiset** $C_k(\boldsymbol{s})$ as

$$C_k(\boldsymbol{s}) = \{\{s_i, s_{i+1}, \ldots, s_{i+k-1}\} : 1 \leq i \leq n-k+1\}$$

- Define **composition multiset** $C(\boldsymbol{s})$ as

$$C(\boldsymbol{s}) = \{C_k(\boldsymbol{s}) : 1 \leq k \leq n\}$$

Example

Consider $\boldsymbol{s} = (0, 1, 0, 1)$.

$$C_2(\boldsymbol{s}) = \{\{0,1\}, \{1,0\},$$

Model

- For $\boldsymbol{s} \in \{0,1\}^n$, define **multiset $C_k(\boldsymbol{s})$** as

$$C_k(\boldsymbol{s}) = \{\{s_i, s_{i+1}, \ldots, s_{i+k-1}\} : 1 \leq i \leq n-k+1\}$$

- Define **composition multiset $C(\boldsymbol{s})$** as

$$C(\boldsymbol{s}) = \{C_k(\boldsymbol{s}) : 1 \leq k \leq n\}$$

Example

Consider $\boldsymbol{s} = (0, 1, 0, 1)$.

$$C_2(\boldsymbol{s}) = \{\{0,1\}, \{1,0\}, \{0,1\}\}.$$

Model

- For $\boldsymbol{s} \in \{0,1\}^n$, define **multiset $C_k(\boldsymbol{s})$** as

$$C_k(\boldsymbol{s}) = \{\{s_i, s_{i+1}, \ldots, s_{i+k-1}\} : 1 \leq i \leq n-k+1\}$$

- Define **composition multiset $C(\boldsymbol{s})$** as

$$C(\boldsymbol{s}) = \{C_k(\boldsymbol{s}) : 1 \leq k \leq n\}$$

Example

Consider $\boldsymbol{s} = (0,1,0,1)$.

$$C_2(\boldsymbol{s}) = \{\{0,1\},\{1,0\},\{0,1\}\}.$$
$$C(\boldsymbol{s})$$

Model

- For $\boldsymbol{s} \in \{0,1\}^n$, define **multiset** $C_k(\boldsymbol{s})$ as

$$C_k(\boldsymbol{s}) = \{\{s_i, s_{i+1}, \ldots, s_{i+k-1}\} : 1 \leq i \leq n-k+1\}$$

- Define **composition multiset** $C(\boldsymbol{s})$ as

$$C(\boldsymbol{s}) = \{C_k(\boldsymbol{s}) : 1 \leq k \leq n\}$$

Example

Consider $\boldsymbol{s} = (0,1,0,1)$.

$$C_2(\boldsymbol{s}) = \{\{0,1\},\{1,0\},\{0,1\}\}.$$
$$C(\boldsymbol{s}) = \{\{0\},\{1\},\{0\},\{1\}, \leftarrow C_1(\boldsymbol{s})$$

Model

- For $\boldsymbol{s} \in \{0,1\}^n$, define **multiset** $C_k(\boldsymbol{s})$ as

$$C_k(\boldsymbol{s}) = \{\{s_i, s_{i+1}, \ldots, s_{i+k-1}\} : 1 \leq i \leq n-k+1\}$$

- Define **composition multiset** $C(\boldsymbol{s})$ as

$$C(\boldsymbol{s}) = \{C_k(\boldsymbol{s}) : 1 \leq k \leq n\}$$

Example

Consider $\boldsymbol{s} = (0,1,0,1)$.

$$\begin{aligned} C_2(\boldsymbol{s}) &= \{\{0,1\},\{1,0\},\{0,1\}\}. \\ C(\boldsymbol{s}) &= \{\{0\},\{1\},\{0\},\{1\}, \\ &\quad \{0,1\},\{1,0\},\{0,1\}, \leftarrow C_2(\boldsymbol{s}) \end{aligned}$$

Model

- For $\boldsymbol{s} \in \{0,1\}^n$, define **multiset $C_k(\boldsymbol{s})$** as

$$C_k(\boldsymbol{s}) = \{\{s_i, s_{i+1}, \ldots, s_{i+k-1}\} : 1 \le i \le n-k+1\}$$

- Define **composition multiset $C(\boldsymbol{s})$** as

$$C(\boldsymbol{s}) = \{C_k(\boldsymbol{s}) : 1 \le k \le n\}$$

Example

Consider $\boldsymbol{s} = (0,1,0,1)$.

$$\begin{aligned} C_2(\boldsymbol{s}) &= \{\{0,1\},\{1,0\},\{0,1\}\}. \\ C(\boldsymbol{s}) &= \{\{0\},\{1\},\{0\},\{1\}, \\ &\quad \{0,1\},\{1,0\},\{0,1\}, \\ &\quad \{0,1,0\},\{1,0,1\}, \leftarrow C_3(\boldsymbol{s}) \end{aligned}$$

Model

- For $\boldsymbol{s} \in \{0,1\}^n$, define **multiset** $C_k(\boldsymbol{s})$ as

$$C_k(\boldsymbol{s}) = \{\{s_i, s_{i+1}, \ldots, s_{i+k-1}\} : 1 \leq i \leq n-k+1\}$$

- Define **composition multiset** $C(\boldsymbol{s})$ as

$$C(\boldsymbol{s}) = \{C_k(\boldsymbol{s}) : 1 \leq k \leq n\}$$

Example

Consider $\boldsymbol{s} = (0,1,0,1)$.

$$\begin{aligned} C_2(\boldsymbol{s}) &= \{\{0,1\},\{1,0\},\{0,1\}\}. \\ C(\boldsymbol{s}) &= \{\{0\},\{1\},\{0\},\{1\}, \\ &\quad \{0,1\},\{1,0\},\{0,1\}, \\ &\quad \{0,1,0\},\{1,0,1\},\{0,1,0,1\} \leftarrow C_4(\boldsymbol{s}) \end{aligned}$$

Model

- For $\boldsymbol{s} \in \{0,1\}^n$, define **multiset $C_k(\boldsymbol{s})$** as

$$C_k(\boldsymbol{s}) = \{\{s_i, s_{i+1}, \ldots, s_{i+k-1}\} : 1 \leq i \leq n-k+1\}$$

- Define **composition multiset $C(\boldsymbol{s})$** as

$$C(\boldsymbol{s}) = \{C_k(\boldsymbol{s}) : 1 \leq k \leq n\}$$

Example

Consider $\boldsymbol{s} = (0,1,0,1)$.

$$\begin{aligned} C_2(\boldsymbol{s}) &= \{\{0,1\},\{1,0\},\{0,1\}\}. \\ C(\boldsymbol{s}) &= \{\{0\},\{1\},\{0\},\{1\}, \\ &\quad \{0,1\},\{1,0\},\{0,1\}, \\ &\quad \{0,1,0\},\{1,0,1\},\{0,1,0,1\}\}. \end{aligned}$$

Model

- For $\boldsymbol{s} \in \{0,1\}^n$, define **multiset $C_k(\boldsymbol{s})$** as

$$C_k(\boldsymbol{s}) = \{\{s_i, s_{i+1}, \ldots, s_{i+k-1}\} : 1 \leq i \leq n-k+1\}$$

- Define **composition multiset $C(\boldsymbol{s})$** as

$$C(\boldsymbol{s}) = \{C_k(\boldsymbol{s}) : 1 \leq k \leq n\}$$

Example

Consider $\boldsymbol{s} = (0,1,0,1)$.

$$\begin{aligned} C_2(\boldsymbol{s}) &= \{\{0,1\},\{1,0\},\{0,1\}\}. \\ C(\boldsymbol{s}) &= \{\{0\},\{1\},\{0\},\{1\}, \\ &\quad \{0,1\},\{1,0\},\{0,1\}, \\ &\quad \{0,1,0\},\{1,0,1\},\{0,1,0,1\}\}. \end{aligned}$$

Note: If $\widetilde{\boldsymbol{s}} = (1,0,1,0)$, $C(\widetilde{\boldsymbol{s}}) = C(\boldsymbol{s})$.

Model

- For $\boldsymbol{s} \in \{0,1\}^n$, define **multiset** $C_k(\boldsymbol{s})$ as

$$C_k(\boldsymbol{s}) = \{\{s_i, s_{i+1}, \ldots, s_{i+k-1}\} : 1 \leq i \leq n-k+1\}$$

- Define **composition multiset** $C(\boldsymbol{s})$ as

$$C(\boldsymbol{s}) = \{C_k(\boldsymbol{s}) : 1 \leq k \leq n\}$$

Example

Consider $\boldsymbol{s} = (0,1,0,1)$.

$$\begin{aligned} C_2(\boldsymbol{s}) &= \{\{0,1\},\{1,0\},\{0,1\}\}. \\ C(\boldsymbol{s}) &= \{\{0\},\{1\},\{0\},\{1\}, \\ &\quad \{0,1\},\{1,0\},\{0,1\}, \\ &\quad \{0,1,0\},\{1,0,1\},\{0,1,0,1\}\}. \end{aligned}$$

Note: If $\widetilde{\boldsymbol{s}} = (1,0,1,0)$, $C(\widetilde{\boldsymbol{s}}) = C(\boldsymbol{s})$. $\rightarrow$ A string and its reversal have identical composition multisets!

Previous Work

- [ADMOP15] showed that strings of length one less than a prime or twice a prime are uniquely reconstructable, up to reversal.

[ADMOP15] J. Acharya, H. Das, O. Milenkovic, A. Orlitsky, and S. Pan, "String Reconstruction from Substring Compositions", *SIAM Journal on Discrete Mathematics*, vol. 29, no. 3, pp. 1340–1371, Jan. 2015

Previous Work

- [ADMOP15] showed that strings of length one less than a prime or twice a prime are uniquely reconstructable, up to reversal.
- [PGM19] proposed a code of $O(\log n)$ redundancy to allow unique reconstruction for all lengths.

Construction

$$
\begin{aligned}
\mathcal{S}_R(n) = \{\boldsymbol{s} \in \{0,1\}^n, s_1 = 0, s_n = 1, \text{ and } \\
\exists I \subset \{2, \ldots, n-1\} \text{ such that } \\
\text{for all } i \in I, s_i \neq s_{n+1-i}, \text{ for all } i \notin I, s_i = s_{n+1-i}, \\
\boldsymbol{s}_{[n/2] \cap I} \text{ has each prefix with strictly more 0s than 1s.}\}
\end{aligned}
$$

[PGM19] S. Pattabiraman, R. Gabrys, and O. Milenkovic, "Reconstruction and Error-Correction Codes for Polymer-Based Data Storage", in *2019 IEEE Information Theory Workshop (ITW)*, Visby, Sweden: IEEE, Aug. 2019, pp. 1–5

Previous Work

- [GPM20] proposed codes to correct composition substitutions.

[GPM20] R. Gabrys, S. Pattabiraman, and O. Milenkovic, “Mass Error-Correction Codes for Polymer-Based Data Storage”, in *2020 IEEE International Symposium on Information Theory (ISIT)*, Los Angeles, CA, USA: IEEE, Jun. 2020, pp. 25–30
[GPM20a] R. Gabrys, S. Pattabiraman, and O. Milenkovic, *Reconstructing Mixtures of Coded Strings from Prefix and Suffix Compositions*, Number: arXiv:2010.11116 arXiv:2010.11116 [cs, math], Oct. 2020
[GM22] U. Gupta and H. Mahdavifar, “A New Algebraic Approach for String Reconstruction from Substring Compositions”, *arXiv:2201.09955 [cs, math]*, Jan. 2022, arXiv: 2201.09955

Previous Work

- [GPM20] proposed codes to correct composition substitutions.
- [GPM20a] introduced codes to reconstruct multiple strings from a mixture of prefix and suffix compositions, simultaneously.

[GPM20] R. Gabrys, S. Pattabiraman, and O. Milenkovic, "Mass Error-Correction Codes for Polymer-Based Data Storage", in *2020 IEEE International Symposium on Information Theory (ISIT)*, Los Angeles, CA, USA: IEEE, Jun. 2020, pp. 25–30

[GPM20a] R. Gabrys, S. Pattabiraman, and O. Milenkovic, *Reconstructing Mixtures of Coded Strings from Prefix and Suffix Compositions*, Number: arXiv:2010.11116 arXiv:2010.11116 [cs, math], Oct. 2020

[GM22] U. Gupta and H. Mahdavifar, "A New Algebraic Approach for String Reconstruction from Substring Compositions", *arXiv:2201.09955 [cs, math]*, Jan. 2022, arXiv: 2201.09955

Previous Work

- [GPM20] proposed codes to correct composition substitutions.
- [GPM20a] introduced codes to reconstruct multiple strings from a mixture of prefix and suffix compositions, simultaneously.
- [GM22] devised a new algebraic algorithm to reconstruct strings from their composition multisets.

[GPM20] R. Gabrys, S. Pattabiraman, and O. Milenkovic, "Mass Error-Correction Codes for Polymer-Based Data Storage", in *2020 IEEE International Symposium on Information Theory (ISIT)*, Los Angeles, CA, USA: IEEE, Jun. 2020, pp. 25–30

[GPM20a] R. Gabrys, S. Pattabiraman, and O. Milenkovic, *Reconstructing Mixtures of Coded Strings from Prefix and Suffix Compositions*, Number: arXiv:2010.11116 arXiv:2010.11116 [cs, math], Oct. 2020

[GM22] U. Gupta and H. Mahdavifar, "A New Algebraic Approach for String Reconstruction from Substring Compositions", *arXiv:2201.09955 [cs, math]*, Jan. 2022, arXiv: 2201.09955

Introduction

Problem Statement & Results

Conclusion

Problem Statement

1. Are composition-insertion-correcting codes equivalent to composition-deletion-correcting codes?

Problem Statement

1. Are composition-insertion-correcting codes equivalent to composition-deletion-correcting codes?

   $\rightarrow$Yes!

Problem Statement

1. Are composition-insertion-correcting codes equivalent to composition-deletion-correcting codes?

   $\rightarrow$Yes!

2. How robust is $\mathcal{S}_R(n)$ to insertions and deletions in composition multisets?

Problem Statement

1. Are composition-insertion-correcting codes equivalent to composition-deletion-correcting codes?

   $\rightarrow$Yes!

2. How robust is $\mathcal{S}_R(n)$ to insertions and deletions in composition multisets?

   $\rightarrow$ Can correct deletion of at most one complete multiset.

Problem Statement

1. Are composition-insertion-correcting codes equivalent to composition-deletion-correcting codes?

   $\rightarrow$Yes!

2. How robust is $\mathcal{S}_R(n)$ to insertions and deletions in composition multisets?

   $\rightarrow$ Can correct deletion of at most one complete multiset.

   $\rightarrow$ Similar construction exists to correct deletions of more multisets.

Insertion-Deletion Equivalence

Lemma

A code can correct the deletion of t multisets, if and only if it can correct composition insertion errors in those t multisets.

Insertion-Deletion Equivalence

Lemma

A code can correct the deletion of t multisets, if and only if it can correct composition insertion errors in those t multisets.

Proof idea:

- Define ***t*-multiset deletion ball** of $\boldsymbol{s}$ as

$$D_t(\boldsymbol{s}) = \{C'(\boldsymbol{s}) = \bigcup_{i \in \{1,\ldots,n\} \setminus \mathcal{I}} C_i(\boldsymbol{s}) : \mathcal{I} \subseteq \{1,\ldots,n\}, |\mathcal{I}| \leq t\}$$

Insertion-Deletion Equivalence

Lemma

A code can correct the deletion of t multisets, if and only if it can correct composition insertion errors in those t multisets.

Proof idea:

- Define t-multiset deletion ball of $\boldsymbol{s}$ as

$$D_t(\boldsymbol{s}) = \{C'(\boldsymbol{s}) = \bigcup_{i \in \{1,\ldots,n\} \setminus \mathcal{I}} C_i(\boldsymbol{s}) : \mathcal{I} \subseteq \{1,\ldots,n\}, |\mathcal{I}| \leq t\}$$

- Consider a $\boldsymbol{v}$ such that $D_t(\boldsymbol{s}) \cap D_t(\boldsymbol{v}) \neq \emptyset$. So, for some $\mathcal{I} \subseteq \{1,\ldots,n\}$ where $|\mathcal{I}| \leq t$,

$$C_j(\boldsymbol{s}) \neq C_j(\boldsymbol{v}) \quad \forall\, j \in \mathcal{I}$$

Insertion-Deletion Equivalence

Lemma

A code can correct the deletion of t multisets, if and only if it can correct composition insertion errors in those t multisets.

Proof idea:

- Define t-multiset deletion ball of $\boldsymbol{s}$ as

$$D_t(\boldsymbol{s}) = \{C'(\boldsymbol{s}) = \bigcup_{i \in \{1,\ldots,n\} \setminus \mathcal{I}} C_i(\boldsymbol{s}) : \mathcal{I} \subseteq \{1, \ldots, n\}, |\mathcal{I}| \leq t\}$$

- Consider a $\boldsymbol{v}$ such that $D_t(\boldsymbol{s}) \cap D_t(\boldsymbol{v}) \neq \emptyset$. So, for some $\mathcal{I} \subseteq \{1, \ldots, n\}$ where $|\mathcal{I}| \leq t$,

$$C_j(\boldsymbol{s}) \neq C_j(\boldsymbol{v}) \quad \forall\, j \in \mathcal{I}$$

$$C_j(\boldsymbol{s}) = C_j(\boldsymbol{v}) \quad \forall\, j \in \{1, \ldots, n\} \setminus \mathcal{I}$$

Insertion-Deletion Equivalence

- Let $\boxed{C'_k(\boldsymbol{s})} = C_k(\boldsymbol{s}) \cup C_k(\boldsymbol{v}) = C'_k(\boldsymbol{v})$.

$C_k(\boldsymbol{s})$ corrupted by insertion of compositions $C_k(\boldsymbol{v}) \setminus C_k(\boldsymbol{s})$.

Insertion-Deletion Equivalence

- Let $\boxed{C'_k(\boldsymbol{s})} = C_k(\boldsymbol{s}) \cup C_k(\boldsymbol{v}) = C'_k(\boldsymbol{v})$.

  $C_k(\boldsymbol{s})$ corrupted by insertion of compositions $C_k(\boldsymbol{v}) \setminus C_k(\boldsymbol{s})$.

- Define ***t*-multiset insertion ball** as

$$I_t(\boldsymbol{s}) = \{C'(\boldsymbol{s}) = \bigcup_{i \in \{1,\dots,n\}} C'_i(\boldsymbol{s}) : \mathcal{I} \subseteq \{1,\dots,n\},\ |\mathcal{I}| \leq t,$$
$$C_i(\boldsymbol{s}) \subset C'_i(\boldsymbol{s}) \ \ \forall i \in \mathcal{I}, C'_i(\boldsymbol{s}) = C_i(\boldsymbol{s}) \ \ \forall i \in \{1,\dots,n\} \setminus \mathcal{I}\}.$$

Insertion-Deletion Equivalence

- Let $C'_k(\boldsymbol{s}) = C_k(\boldsymbol{s}) \cup C_k(\boldsymbol{v}) = C'_k(\boldsymbol{v})$.

  $C_k(\boldsymbol{s})$ corrupted by insertion of compositions $C_k(\boldsymbol{v}) \setminus C_k(\boldsymbol{s})$.

- Define t-multiset insertion ball as

$$I_t(\boldsymbol{s}) = \{C'(\boldsymbol{s}) = \bigcup_{i \in \{1,\dots,n\}} C'_i(\boldsymbol{s}) : \mathcal{I} \subseteq \{1, \dots, n\},\ |\mathcal{I}| \leq t,$$
$$C_i(\boldsymbol{s}) \subset C'_i(\boldsymbol{s}) \ \ \forall i \in \mathcal{I}, C'_i(\boldsymbol{s}) = C_i(\boldsymbol{s}) \ \ \forall i \in \{1, \dots, n\} \setminus \mathcal{I}\}.$$

- Thus

$$\bigcup_{k \in \mathcal{I}} C'_k(\boldsymbol{s}) \cup \bigcup_{j \in \{1,\dots,n\} \setminus \mathcal{I}} C_j(\boldsymbol{s}) = \bigcup_{j \in \{1,\dots,n\} \setminus \mathcal{I}} C_j(\boldsymbol{v}) \cup \bigcup_{k \in \mathcal{I}} C'_k(\boldsymbol{v}).$$

Insertion-Deletion Equivalence

- Let $\boxed{C'_k(\mathbf{s})} = C_k(\mathbf{s}) \cup C_k(\mathbf{v}) = C'_k(\mathbf{v})$.

  $C_k(\mathbf{s})$ corrupted by insertion of compositions $C_k(\mathbf{v}) \setminus C_k(\mathbf{s})$.

- Define t-multiset insertion ball as

$$I_t(\mathbf{s}) = \{C'(\mathbf{s}) = \bigcup_{i \in \{1,\ldots,n\}} C'_i(\mathbf{s}) : \mathcal{I} \subseteq \{1,\ldots,n\},\ |\mathcal{I}| \leq t,$$
$$C_i(\mathbf{s}) \subset C'_i(\mathbf{s})\ \ \forall i \in \mathcal{I}, C'_i(\mathbf{s}) = C_i(\mathbf{s})\ \ \forall i \in \{1,\ldots,n\} \setminus \mathcal{I}\}.$$

- Thus

$$\bigcup_{k \in \mathcal{I}} C'_k(\mathbf{s}) \cup \bigcup_{j \in \{1,\ldots,n\} \setminus \mathcal{I}} C_j(\mathbf{s}) = \bigcup_{j \in \{1,\ldots,n\} \setminus \mathcal{I}} C_j(\mathbf{v}) \cup \bigcup_{k \in \mathcal{I}} C'_k(\mathbf{v}).$$

  $\rightarrow$ Equal by definition of $D_t(\mathbf{s})$.

Insertion-Deletion Equivalence

- Let $\boxed{C'_k(\mathbf{s})} = C_k(\mathbf{s}) \cup C_k(\mathbf{v}) = C'_k(\mathbf{v})$.

  $C_k(\mathbf{s})$ corrupted by insertion of compositions $C_k(\mathbf{v}) \setminus C_k(\mathbf{s})$.

- Define t-multiset insertion ball as

$$I_t(\mathbf{s}) = \{C'(\mathbf{s}) = \bigcup_{i \in \{1,\dots,n\}} C'_i(\mathbf{s}) : \mathcal{I} \subseteq \{1,\dots,n\},\ |\mathcal{I}| \le t,$$
$$C_i(\mathbf{s}) \subset C'_i(\mathbf{s})\ \ \forall i \in \mathcal{I}, C'_i(\mathbf{s}) = C_i(\mathbf{s})\ \ \forall i \in \{1,\dots,n\} \setminus \mathcal{I}\}.$$

- Thus

$$\bigcup_{k \in \mathcal{I}} C'_k(\mathbf{s}) \cup \bigcup_{j \in \{1,\dots,n\} \setminus \mathcal{I}} C_j(\mathbf{s}) = \bigcup_{j \in \{1,\dots,n\} \setminus \mathcal{I}} C_j(\mathbf{v}) \cup \bigcup_{k \in \mathcal{I}} C'_k(\mathbf{v}).$$

$\rightarrow$ Equal since $C'_k(\mathbf{s}) = C'_k(\mathbf{v})$.

Insertion-Deletion Equivalence

- Let $\boxed{C'_k(\mathbf{s})} = C_k(\mathbf{s}) \cup C_k(\mathbf{v}) = C'_k(\mathbf{v})$.

  $C_k(\mathbf{s})$ corrupted by insertion of compositions $C_k(\mathbf{v}) \setminus C_k(\mathbf{s})$.

- Define t-multiset insertion ball as

$$I_t(\mathbf{s}) = \{C'(\mathbf{s}) = \bigcup_{i \in \{1,\dots,n\}} C'_i(\mathbf{s}) : \mathcal{I} \subseteq \{1,\dots,n\},\ |\mathcal{I}| \leq t,$$
$$C_i(\mathbf{s}) \subset C'_i(\mathbf{s})\ \ \forall i \in \mathcal{I}, C'_i(\mathbf{s}) = C_i(\mathbf{s})\ \ \forall i \in \{1,\dots,n\} \setminus \mathcal{I}\}.$$

- Thus

$$\underbrace{\bigcup_{k \in \mathcal{I}} C'_k(\mathbf{s}) \cup \bigcup_{j \in \{1,\dots,n\} \setminus \mathcal{I}} C_j(\mathbf{s}) = \bigcup_{j \in \{1,\dots,n\} \setminus \mathcal{I}} C_j(\mathbf{v}) \cup \bigcup_{k \in \mathcal{I}} C'_k(\mathbf{v})}_{\in I_t(\mathbf{s}) \cap I_t(\mathbf{v})}.$$

Insertion-Deletion Equivalence

- Let $\boxed{C'_k(\boldsymbol{s})} = C_k(\boldsymbol{s}) \cup C_k(\boldsymbol{v}) = C'_k(\boldsymbol{v})$.

  $C_k(\boldsymbol{s})$ corrupted by insertion of compositions $C_k(\boldsymbol{v}) \setminus C_k(\boldsymbol{s})$.

- Define t-multiset insertion ball as

$$I_t(\boldsymbol{s}) = \{C'(\boldsymbol{s}) = \bigcup_{i \in \{1,\ldots,n\}} C'_i(\boldsymbol{s}) : \mathcal{I} \subseteq \{1, \ldots, n\}, \ |\mathcal{I}| \leq t,$$

$$C_i(\boldsymbol{s}) \subset C'_i(\boldsymbol{s}) \ \ \forall i \in \mathcal{I}, C'_i(\boldsymbol{s}) = C_i(\boldsymbol{s}) \ \ \forall i \in \{1, \ldots, n\} \setminus \mathcal{I}\}.$$

- Thus

$$\underbrace{\bigcup_{k \in \mathcal{I}} C'_k(\boldsymbol{s}) \cup \bigcup_{j \in \{1,\ldots,n\} \setminus \mathcal{I}} C_j(\boldsymbol{s}) = \bigcup_{j \in \{1,\ldots,n\} \setminus \mathcal{I}} C_j(\boldsymbol{v}) \cup \bigcup_{k \in \mathcal{I}} C'_k(\boldsymbol{v})}_{\in I_t(\boldsymbol{s}) \cap I_t(\boldsymbol{v})}.$$

- $D_t(\boldsymbol{s}) \cap D_t(\boldsymbol{v}) \neq \emptyset \iff I_t(\boldsymbol{s}) \cap I_t(\boldsymbol{v}) \neq \emptyset$

Multiset Deletions

Definition

An **asymmetric multiset deletion** occurs in the composition multiset $C(\boldsymbol{s})$ of a string $\boldsymbol{s} \in \{0,1\}^n$, if for some $i \in [n]$, the multiset $C_i(\boldsymbol{s})$ is entirely missing, while $C_{n-i+1}(\boldsymbol{s})$ is not corrupted.

Multiset Deletions

Definition

An **asymmetric multiset deletion** occurs in the composition multiset $C(\boldsymbol{s})$ of a string $\boldsymbol{s} \in \{0, 1\}^n$, if for some $i \in [n]$, the multiset $C_i(\boldsymbol{s})$ is entirely missing, while $C_{n-i+1}(\boldsymbol{s})$ is not corrupted.

Lemma

$\mathcal{S}_R(n)$ can correct a single asymmetric multiset deletion.

Asymmetric Multiset Deletions

Construction

$$\begin{aligned}\mathcal{S}_{DA}^{(t)}(n) = \{ \boldsymbol{s} \in \{0,1\}^n : s_1 = 0, s_n = 1, \text{ and } \\ \exists I \subset \{2, \ldots, \frac{n}{2}\},\ |I| \geq t, \text{ such that } \\ \forall\, i \in I, s_i \neq s_{n+1-i}, \text{ and } \forall i \notin I, s_i = s_{n+1-i}, \\ \boldsymbol{s}_{[n/2] \cap I} \text{ is a string wherein each } \\ \text{prefix has at least } t \text{ more 0s than 1s} \}.\end{aligned}$$

Asymmetric Multiset Deletions

Construction

$$\mathcal{S}_{DA}^{(t)}(n) = \Big\{ \boldsymbol{s} \in \{0,1\}^n : s_1 = 0, s_n = 1, \text{ and } \exists I \subset \{2, \ldots, \frac{n}{2}\}, \ |I| \geq t, \text{ such that } \forall\, i \in I, s_i \neq s_{n+1-i}, \text{ and } \forall i \notin I, s_i = s_{n+1-i}, \ \boldsymbol{s}_{[n/2] \cap I} \text{ is a string wherein each prefix has at least } t \text{ more 0s than 1s} \Big\}.$$

- Is an extension of $\mathcal{S}_R(n)$ [PGM19] .
- Involves at most $0.5 \log(n - 2t) + 2t + 3$ redundant bits.

[PGM19] S. Pattabiraman, R. Gabrys, and O. Milenkovic, “Reconstruction and Error-Correction Codes for Polymer-Based Data Storage”, in *2019 IEEE Information Theory Workshop (ITW)*, Visby, Sweden: IEEE, Aug. 2019, pp. 1–5

Asymmetric Multiset Deletions

Theorem

$\mathcal{S}_{DA}^{(t)}(n)$ can correct up to t-asymmetric multiset deletions.

- Proof idea [PGM21]:

[PGM21] S. Pattabiraman, R. Gabrys, and O. Milenkovic, "Coding for Polymer-Based Data Storage", *arXiv:2003.02121 [cs, math]*, Jun. 2021, arXiv: 2003.02121

Asymmetric Multiset Deletions

Theorem

$\mathcal{S}_{DA}^{(t)}(n)$ can correct up to t-asymmetric multiset deletions.

- Proof idea [PGM21]:
  1. For $\boldsymbol{s} \in \mathcal{S}_{DA}^{(t)}(n)$, build a set of all strings $\boldsymbol{v} \in \mathcal{S}_{DA}^{(t)}(n)$ such that $D_{t+1}(\boldsymbol{s}) \cap D_{t+1}(\boldsymbol{v}) \neq \emptyset$.

[PGM21] S. Pattabiraman, R. Gabrys, and O. Milenkovic, "Coding for Polymer-Based Data Storage", *arXiv:2003.02121 [cs, math]*, Jun. 2021, arXiv: 2003.02121

Asymmetric Multiset Deletions

Theorem

$\mathcal{S}_{DA}^{(t)}(n)$ can correct up to t-asymmetric multiset deletions.

- Proof idea [PGM21]:
  1. For $\boldsymbol{s} \in \mathcal{S}_{DA}^{(t)}(n)$, build a set of all strings $\boldsymbol{v} \in \mathcal{S}_{DA}^{(t)}(n)$ such that $D_{t+1}(\boldsymbol{s}) \cap D_{t+1}(\boldsymbol{v}) \neq \emptyset$.
  -Includes all $\boldsymbol{v}$ such that if $(\boldsymbol{s}_1^k, \boldsymbol{s}_{n-k+1}^n) = (\boldsymbol{v}_1^k, \boldsymbol{v}_{n-k+1}^n)$ and $s_{k+1} \neq v_{k+1}$, then for $i \in \{1, \ldots, t+1\}$
  $$|C_{n-k-i}(\boldsymbol{s}) \setminus C_{n-k-i}(\boldsymbol{v})| \leq 2.$$

[PGM21] S. Pattabiraman, R. Gabrys, and O. Milenkovic, “Coding for Polymer-Based Data Storage”, *arXiv:2003.02121 [cs, math]*, Jun. 2021, arXiv: 2003.02121

Asymmetric Multiset Deletions

Theorem

$\mathcal{S}_{DA}^{(t)}(n)$ can correct up to t-asymmetric multiset deletions.

- Proof idea [PGM21]:
  1. For $\boldsymbol{s} \in \mathcal{S}_{DA}^{(t)}(n)$, build a set of all strings $\boldsymbol{v} \in \mathcal{S}_{DA}^{(t)}(n)$ such that $D_{t+1}(\boldsymbol{s}) \cap D_{t+1}(\boldsymbol{v}) \neq \emptyset$.
  -Includes all $\boldsymbol{v}$ such that if $(\boldsymbol{s}_1^k, \boldsymbol{s}_{n-k+1}^n) = (\boldsymbol{v}_1^k, \boldsymbol{v}_{n-k+1}^n)$ and $s_{k+1} \neq v_{k+1}$, then for $i \in \{1, \ldots, t+1\}$
  $$|C_{n-k-i}(\boldsymbol{s}) \setminus C_{n-k-i}(\boldsymbol{v})| \leq 2.$$
  2. We infer that no $\boldsymbol{v}$ simultaneously satisfies
  $$C_{n-k-t-1}(\boldsymbol{s}) = C_{n-k-t-1}(\boldsymbol{v}).$$

[PGM21] S. Pattabiraman, R. Gabrys, and O. Milenkovic, "Coding for Polymer-Based Data Storage", *arXiv:2003.02121 [cs, math]*, Jun. 2021, arXiv: 2003.02121

Asymmetric Multiset Deletions

- Thus, if $C_{n-k-1}(\boldsymbol{s}), \ldots, C_{n-k-t}(\boldsymbol{s})$ are deleted, each $\boldsymbol{v} \in \mathcal{S}_{DA}^{(t)}(n)$ upholds

$$C_{n-k-t-1}(\boldsymbol{s}) \neq C_{n-k-t-1}(\boldsymbol{v})$$

Asymmetric Multiset Deletions

- Thus, if $C_{n-k-1}(\boldsymbol{s}), \ldots, C_{n-k-t}(\boldsymbol{s})$ are deleted, each $\boldsymbol{v} \in \mathcal{S}_{DA}^{(t)}(n)$ upholds

$$
\begin{aligned}
& C_{n-k-t-1}(\boldsymbol{s}) \neq C_{n-k-t-1}(\boldsymbol{v}) \\
\implies & D_t(\boldsymbol{s}) \cap D_t(\boldsymbol{v}) = \emptyset.
\end{aligned}
$$

Asymmetric Multiset Deletions

- Thus, if $C_{n-k-1}(\boldsymbol{s}), \ldots, C_{n-k-t}(\boldsymbol{s})$ are deleted, each $\boldsymbol{v} \in \mathcal{S}_{DA}^{(t)}(n)$ upholds

$$
\begin{aligned}
& C_{n-k-t-1}(\boldsymbol{s}) \neq C_{n-k-t-1}(\boldsymbol{v}) \\
\implies \; & D_t(\boldsymbol{s}) \cap D_t(\boldsymbol{v}) = \emptyset.
\end{aligned}
$$

$\rightarrow$ Thus, $\boldsymbol{s}$ can be recovered uniquely.

Asymmetric Multiset Deletions

- Thus, if $C_{n-k-1}(\boldsymbol{s}), \ldots, C_{n-k-t}(\boldsymbol{s})$ are deleted, each $\boldsymbol{v} \in \mathcal{S}_{DA}^{(t)}(n)$ upholds

$$
\begin{aligned}
& C_{n-k-t-1}(\boldsymbol{s}) \neq C_{n-k-t-1}(\boldsymbol{v}) \\
\implies & D_t(\boldsymbol{s}) \cap D_t(\boldsymbol{v}) = \emptyset.
\end{aligned}
$$

$\rightarrow$ Thus, $\boldsymbol{s}$ can be recovered uniquely.

- When deleted multisets are nonconsecutive, proof follows similarly.

Introduction

Problem Statement & Results

Conclusion

Summary of Results

- A code that can correct t multiset deletions, iff correct composition insertions in t multisets.

Summary of Results

- A code that can correct t multiset deletions, iff correct composition insertions in t multisets.
- $\mathcal{S}_R(n)$ can correct the deletion of a single multiset.

Summary of Results

- A code that can correct t multiset deletions, iff correct composition insertions in t multisets.
- $\mathcal{S}_R(n)$ can correct the deletion of a single multiset.
- To correct the deletion of t asymmetric multisets, $0.5 \log(n-2t) + 2t + 3$ redundant bits suffice.

Summary of Results

- A code that can correct t multiset deletions, iff correct composition insertions in t multisets.
- $\mathcal{S}_R(n)$ can correct the deletion of a single multiset.
- To correct the deletion of t asymmetric multisets, $0.5\log(n-2t)+2t+3$ redundant bits suffice.
- For any $1 \leq i \leq n$, $\mathcal{S}_R(n)$ can correct the deletion of multisets $C_i(\boldsymbol{s})$ and $C_{n-i+1}(\boldsymbol{s})$.

Future Work

- **Upper bounds** on codes for unique reconstruction from composition multisets.

Future Work

- **Upper bounds** on codes for unique reconstruction from composition multisets.
- Extension to **non-binary** alphabets.

Future Work

- **Upper bounds** on codes for unique reconstruction from composition multisets.
- Extension to **non-binary** alphabets.
- Extension to **circular polymers**, i.e. bits on a ring.

Thank you!

References

- A. Al Ouahabi, J.-A. Amalian, L. Charles, and J.-F. Lutz, "Mass spectrometry sequencing of long digital polymers facilitated by programmed inter-byte fragmentation", *Nature Communications*, vol. 8, no. 1, p. 967, Dec. 2017.
- J. Acharya, H. Das, O. Milenkovic, A. Orlitsky, and S. Pan, "String Reconstruction from Substring Compositions", *SIAM Journal on Discrete Mathematics*, vol. 29, no. 3, pp. 1340–1371, Jan. 2015.
- S. Pattabiraman, R. Gabrys, and O. Milenkovic, "Reconstruction and Error-Correction Codes for Polymer-Based Data Storage", in *2019 IEEE Information Theory Workshop (ITW)*, Visby, Sweden: IEEE, Aug. 2019, pp. 1–5.
- R. Gabrys, S. Pattabiraman, and O. Milenkovic, "Mass Error-Correction Codes for Polymer-Based Data Storage", in *2020 IEEE International Symposium on Information Theory (ISIT)*, Los Angeles, CA, USA: IEEE, Jun. 2020, pp. 25–30.
- ——,*Reconstructing Mixtures of Coded Strings from Prefix and Suffix Compositions*, Number: arXiv:2010.11116 arXiv:2010.11116 [cs, math], Oct. 2020.
- U. Gupta and H. Mahdavifar, "A New Algebraic Approach for String Reconstruction from Substring Compositions", *arXiv:2201.09955 [cs, math]*, Jan. 2022, arXiv: 2201.09955.
- S. Pattabiraman, R. Gabrys, and O. Milenkovic, "Coding for Polymer-Based Data Storage", *arXiv:2003.02121 [cs, math]*, Jun. 2021, arXiv: 2003.02121.