Your Presentation Title

Subtitle or Conference Name

Author Name
Institution or Affiliation

February 2026

Outline

1. Background & Motivation
2. Problem Formulation
3. Methodology
4. Results
5. Conclusion

Background

- Way to index massive bacterial datasets
- Each genome is a color. Each k-mer is associated with a set of colors.
- The color set of a k-mer is the set of color associated with it.
- Interesting object: the set of distinct color sets
- Key to compressing the data structure

Color matrix

- Color matrix: rows are k-mers, column are colors
- So we want to build to distinct rows
- It's easy to build the matrix column by column.
- But the query is row by row.

Color matrix

	G_1	G_2	G_3	G_4	G_5	G_6	G_7	G_8
`ACGTA`	1	0	1	1	0	0	1	0
`CGTAC`	1	1	0	0	1	0	0	1
`GTACG`	0	1	1	0	0	1	0	0
`TACGT`	1	0	1	1	0	0	1	0
`ACGTG`	0	0	1	0	1	1	0	1
`TGCAA`	1	1	0	1	0	0	0	0
`GCAAC`	0	1	1	0	0	1	0	0
`CAACT`	0	0	0	1	1	0	1	1
`TTGCA`	1	0	1	1	0	0	1	0
`AACGT`	0	1	1	0	1	0	1	0
`CGTAT`	0	0	1	0	1	1	0	1
`GTATC`	0	0	1	0	0	1	1	0
`TATCG`	1	1	0	0	1	0	0	0
`ATCGA`	0	1	1	1	0	0	1	1
`CGAAC`	0	0	1	0	1	1	0	1

Color matrix

	G_1	G_2	G_3	G_4	G_5	G_6	G_7	G_8
`ACGTA`	1	0	1	1	0	0	1	0
`CGTAC`	1	1	0	0	1	0	0	1
`GTACG`	0	1	1	0	0	1	0	0
`TACGT`	1	0	1	1	0	0	1	0
`ACGTG`	0	0	1	0	1	1	0	1
`TGCAA`	1	1	0	1	0	0	0	0
`GCAAC`	0	1	1	0	0	1	0	0
`CAACT`	0	0	0	1	1	0	1	1
`TTGCA`	1	0	1	1	0	0	1	0
`AACGT`	0	1	1	0	1	0	1	0
`CGTAT`	0	0	1	0	1	1	0	1
`GTATC`	0	0	1	0	0	1	1	0
`TATCG`	1	1	0	0	1	0	0	0
`ATCGA`	0	1	1	1	0	0	1	1
`CGAAC`	0	0	1	0	1	1	0	1

	G_1	G_2	G_3	G_4	G_5	G_6	G_7	G_8	**Fingerprint**
ACGTA	1	0	1	1	0	0	1	0	1011010011
CGTAC	1	1	0	0	1	0	0	1	0110100101
GTACG	0	1	1	0	0	1	0	0	1100011110
TACGT	1	0	1	1	0	0	1	0	1011010011
ACGTG	0	0	1	0	1	1	0	1	0101110100
TGCAA	1	1	0	1	0	0	0	0	1001001010
GCAAC	0	1	1	0	0	1	0	0	1100011110
CAACT	0	0	0	1	1	0	1	1	0010111001
TTGCA	1	0	1	1	0	0	1	0	1011010011
AACGT	0	1	1	0	1	0	1	0	1110001100
CGTAT	0	0	1	0	1	1	0	1	0101110100
GTATC	0	0	1	0	0	1	1	0	0011000111
TATCG	1	1	0	0	1	0	0	0	1000110010
ATCGA	0	1	1	1	0	0	1	1	0100101011
CGAAC	0	0	1	0	1	1	0	1	0101110100

Fingerprint function

- Build colors in parallel column by column, and for each row of the color matrix, store a fingerprint of the colors on that row
- Requirements for the fingerprint function f:
  - f takes in a fingerprint and a color, and adds the color to the fingerprint
  - Given set $\{c_1, c_2, c_3\}$ the fingerprint is $f(f(f(\text{initial}, c_1), c_2), c_3)$
  - Order-invariant: $f(f(f(\text{initial}, c_1), c_2), c_3) = f(f(f(\varnothing, c_3), c_2), c_1)$
  - Atomically updateable: $x \leftarrow f(x, c)$ is an atomic CPU operation
  - Collision-resistant (ideally a universal hash family)

Collision analysis

The fingerprint function is **universal hash family**. By the union bound:

Lemma 2. *Given a set of distinct sets* $A_0, \ldots, A_{N-1}$*, the probability that there exists two sets* $A_i \neq A_j$ *such that* $F(A_i) = F\left(A_j\right)$ *is at most* $\frac{N^2}{2^{\ell+1}}$*, where* l *is the length of a fingerprint.*

For example, for $\ell = 128$ and $N = 10^9$, we have a collision probability of at most $10^{18}\ /\ 2^{129} \approx 1.47 \cdot 10^{-21}$.

Conclusion

Summary

- We proposed a method for...
- Achieves state-of-the-art on...
- Theoretical guarantees under mild assumptions

Future Work

- Extend to non-convex settings
- Scale to larger datasets ($n > 10^6$)

Questions?

author@university.edu