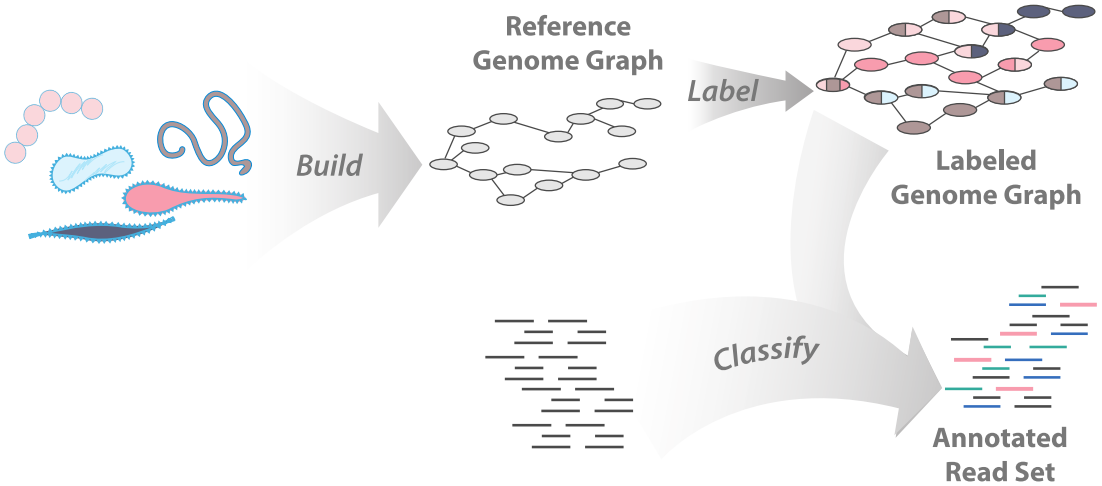
ETH zürich

Sparse Binary Relation Representations for Genome Graph Annotation

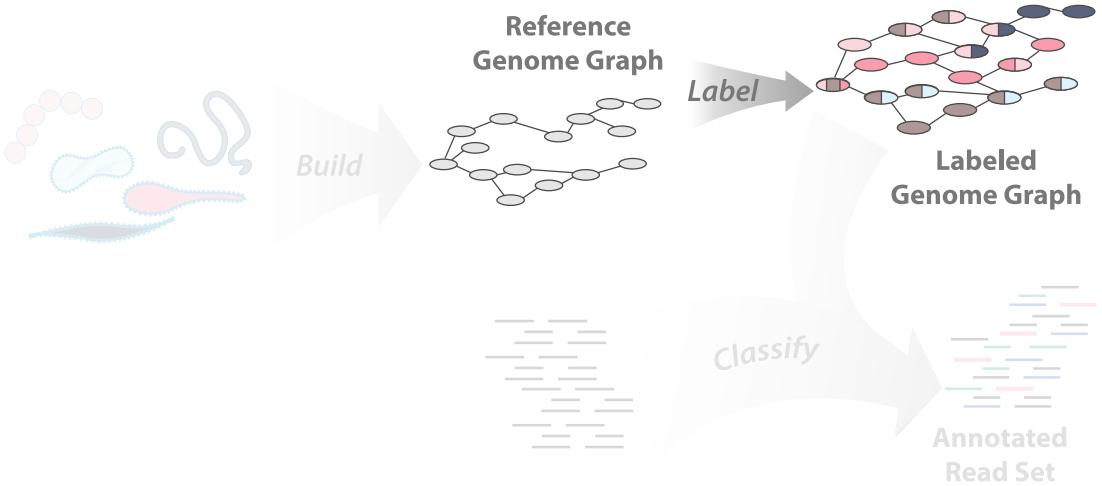
Mikhail Karasikov, Harun Mustafa, Amir Joudaki, Sara Javadzadeh-No, Gunnar Rätsch, André Kahles

BIOMEDICAL INFORMATICS

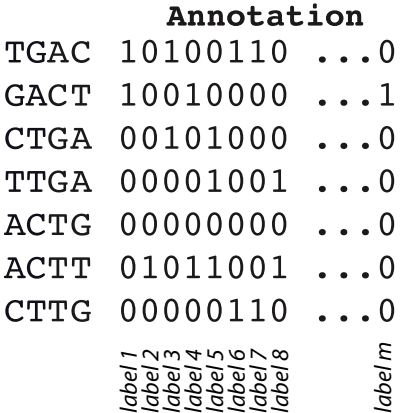
Motivation


Reference
Genome Graph
Label
Build
Labeled
Genome Graph
Classify
Annotated
Read Set

Motivation

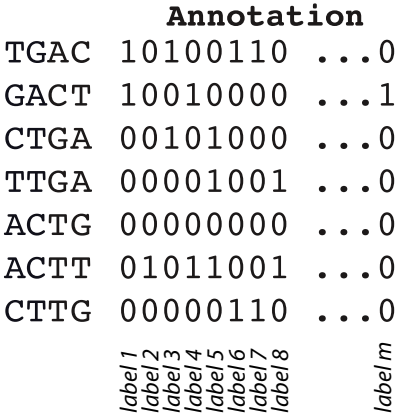

Reference
Genome Graph
Label
Build
Labeled
Genome Graph
Classify
Annotated
Read Set

Graph annotation: relations between k-mers and labels



$$\text{Annotation} \subset k\text{-mers} \times \text{labels}$$

Graph annotation: relations between k-mers and labels



$$\text{Annotation} \subset k\text{-mers} \times \text{labels}$$

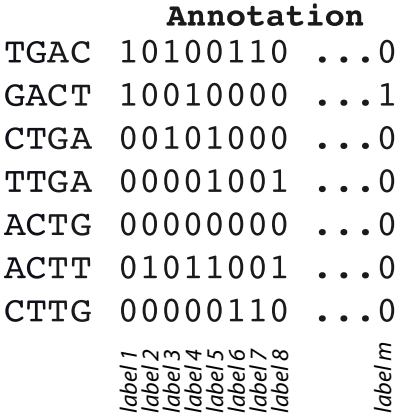
Annotation matrix $A \in \{0,1\}^{n \times m}$

n : #k-mers

m : #labels

$A_{ij} = 1$ iff k-mer i has label j

Graph annotation: relations between k-mers and labels



$$\text{Annotation} \subset k\text{-mers} \times \text{labels}$$

Annotation matrix $A \in \{0,1\}^{n \times m}$

n : #k-mers

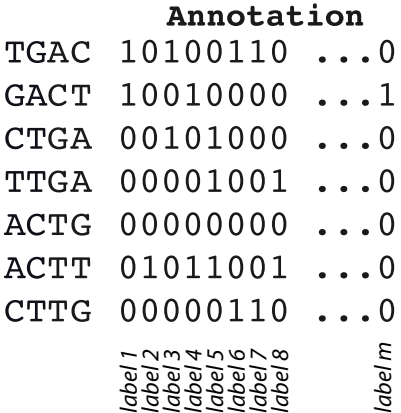
m : #labels

$A_{ij} = 1$ iff k-mer i has label j

$$n \sim 10^9\text{–}10^{11}$$

$$m \sim 10^3\text{–}10^6$$

Graph annotation: relations between k-mers and labels



$$\text{Annotation} \subset k\text{-mers} \times \text{labels}$$

Annotation matrix $A \in \{0,1\}^{n \times m}$

n : #k-mers

m : #labels

$A_{ij} = 1$ iff k-mer i has label j

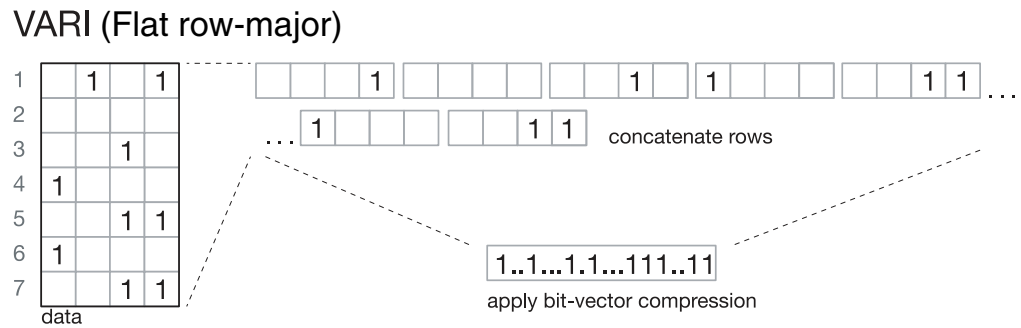
$$n \sim 10^9\text{–}10^{11}$$

$$m \sim 10^3\text{–}10^6$$

How do we compress this?

State-of-the-art row-major representations

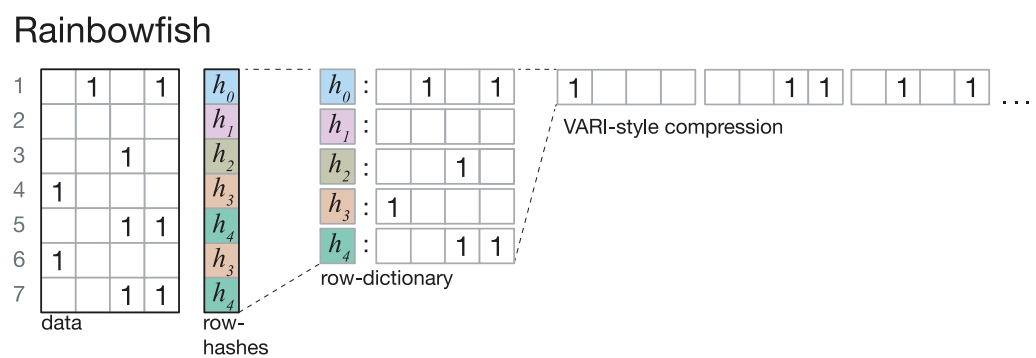
Muggli *et al.* (2017)



Row query: $O(1)$ Column query: $O(n)$

State-of-the-art row-major representations

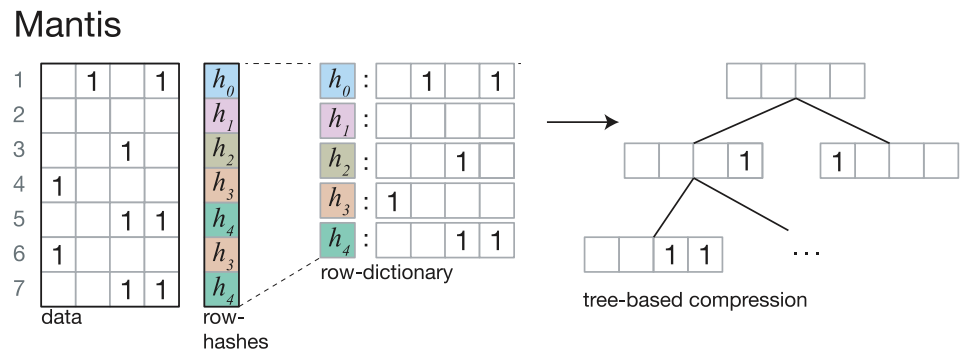
Almodaresi *et al.* (2017)



Row query: $O(1)$ Column query: $O(n)$

State-of-the-art row-major representations

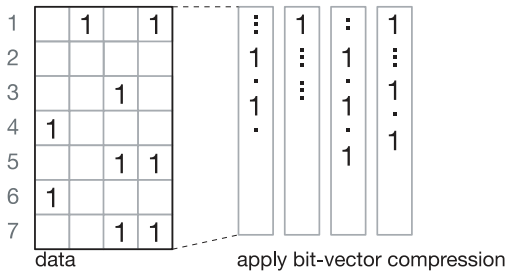
Almodaresi *et al.* (2019)



Row query: $O(\text{tree height})$ Worst-case

Column query: $O(n)$

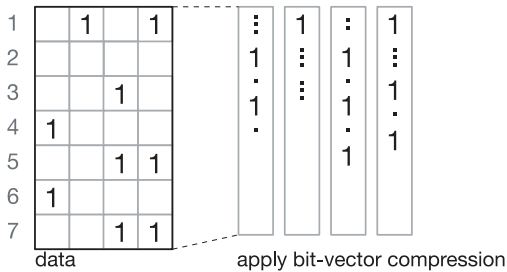
Column-major binary relation representation



Column compressed annotation enables

- independent construction of each column
- fast column queries

Column-major binary relation representation



Column compressed annotation enables

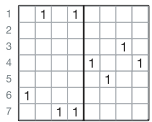
- independent construction of each column
- fast column queries

Converting to hierarchical compression

- takes advantage of column similarities
- allows for sub-$O(m)$ row query time

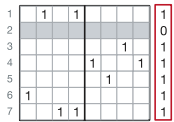
Binary relation wavelet trees (BRWT)

Barbay *et al.* (2013)



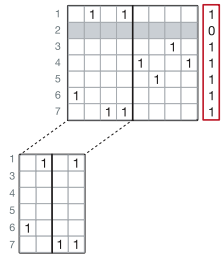
Binary relation wavelet trees (BRWT)

Barbay *et al.* (2013)



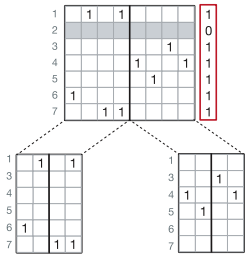
Binary relation wavelet trees (BRWT)

Barbay *et al.* (2013)



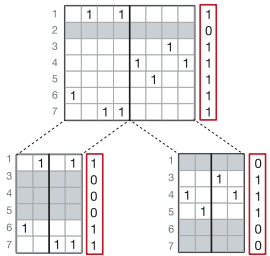
Binary relation wavelet trees (BRWT)

Barbay *et al.* (2013)



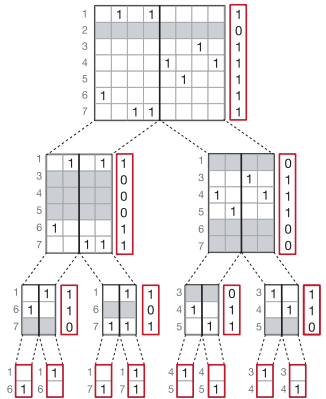
Binary relation wavelet trees (BRWT)

Barbay *et al.* (2013)

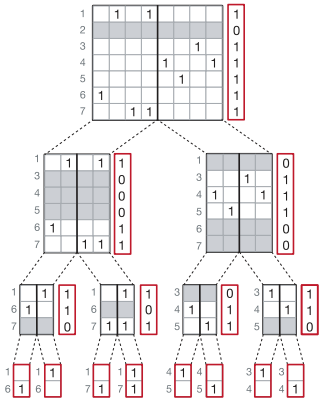


Binary relation wavelet trees (BRWT)

Barbay *et al.* (2013)



Binary relation wavelet trees (BRWT)

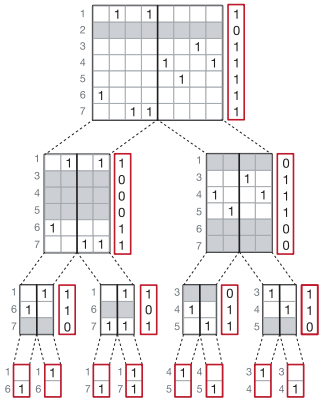


Hierarchical column compression

Each step removes zero-rows and splits matrix vertically

A similar concept is used in Split Sequence Bloom Trees (SSBT), where each column represents a Bloom filter (Solomon and Kingsford (2018))

Binary relation wavelet trees (BRWT)



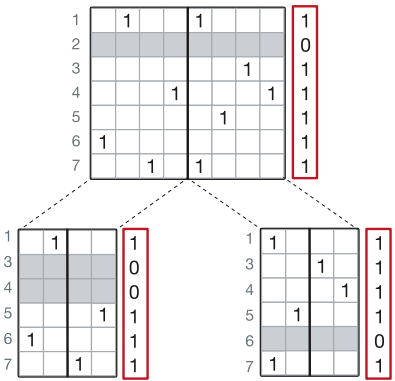
Hierarchical column compression

Each step removes zero-rows and splits matrix vertically

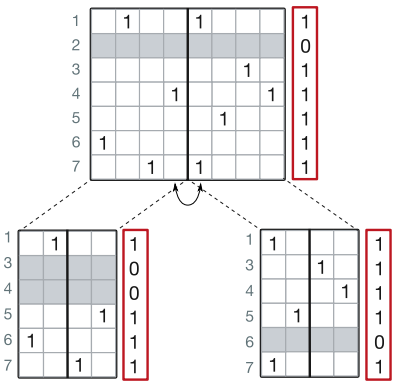
A similar concept is used in Split Sequence Bloom Trees (SSBT), where each column represents a Bloom filter (Solomon and Kingsford (2018))

Let's generalize and improve this!

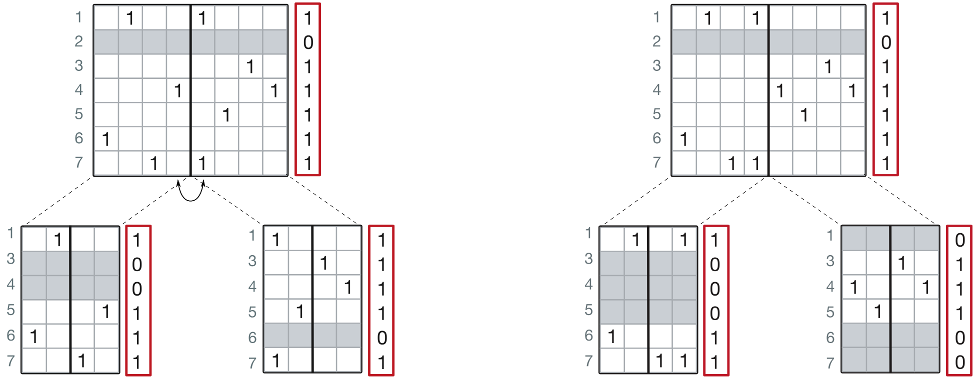
Generalizing BRWT: optimized column partitioning



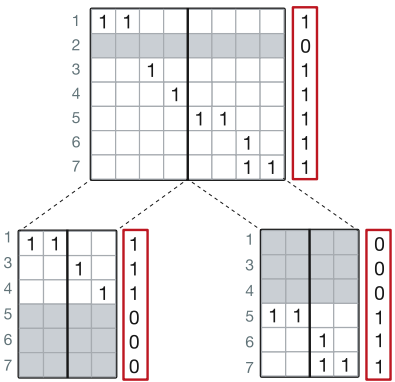
Generalizing BRWT: optimized column partitioning



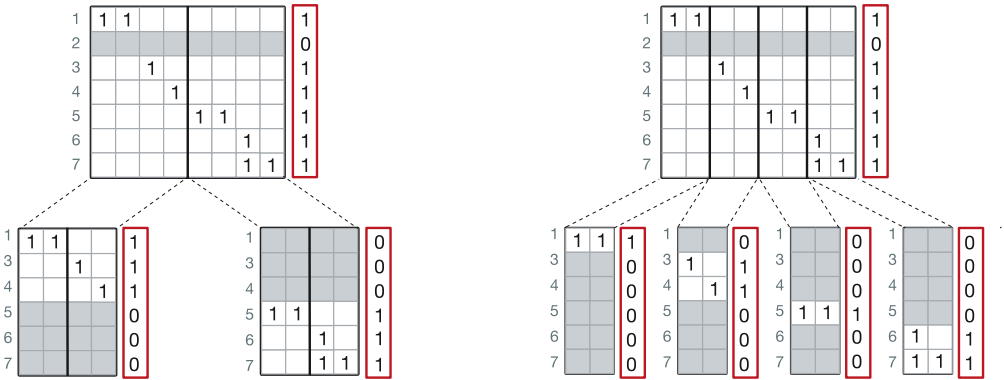
Generalizing BRWT: optimized column partitioning



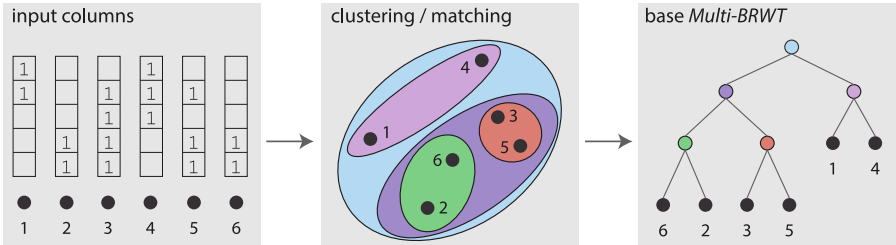
Generalizing BRWT: increasing tree arity



Generalizing BRWT: increasing tree arity



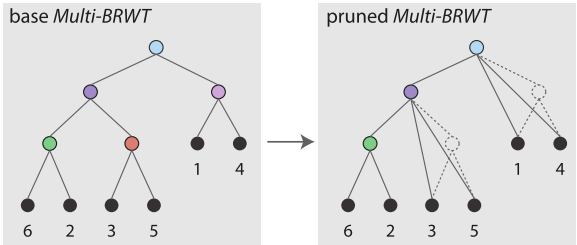
Technique 1: Multi-BRWT Construction via column clustering



Each cluster defines a node in the BRWT tree

Proposed implementation: Greedy pairwise matching of columns (GPM)

Technique 2: Arity relaxation

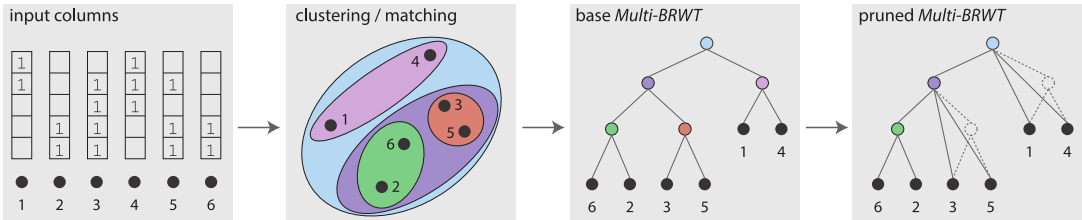


Remove internal node v and assign Children(v) to their grandparent if:

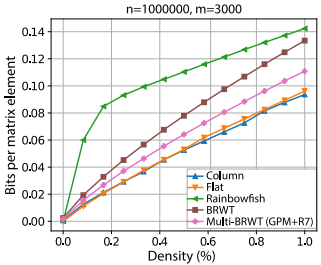
$$\text{Space}(\widehat{\text{Children}(v)}) < \text{Space}(v) + \text{Space}(\text{Children}(v)).$$

Apply greedily up the tree starting from the leaves

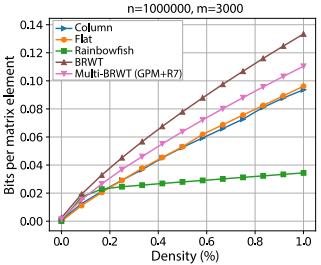
Construction of our Multi-BRWTs



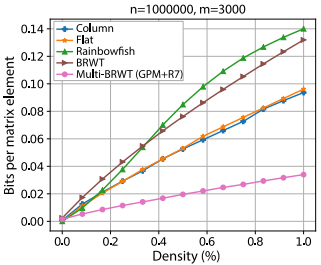
Results: simulated binary relation matrices



Random rows
Significant overhead



Duplicated rows
Rainbowfish performs well



Duplicated columns
Multi-BRWT scales best

Real-world data: genome graph annotations

Kingsford (Solomon and Kingsford (2018))

- Low variability
- Graph: Human RNA-Seq
- $n = 3,693,178,415$
- Labels: SRA accession IDs
- $m = 2,586$
- density: 0.19%

RefSeq (Agarwala *et al.* (2017))

- High variability
- Graph: 79,448 reference genomes
- $n = 1,073,741,824$
- Labels: Taxonomic family
- $m = 3,173$
- density: 3.8%

Results: genome graph annotations

Methods	Kingsford	RefSeq
Column	36.56	80.18
Flat	41.21	121.60
Rainbowfish	23.16	136.65
BRWT	14.05	57.24

Sizes measured in gigabytes (Gb).

Results: genome graph annotations

Methods	Kingsford	RefSeq
Column	36.56	80.18
Flat	41.21	121.60
Rainbowfish	23.16	136.65
BRWT	14.05	57.24
Multi-BRWT (5-ary)	13.01	53.09

Sizes measured in gigabytes (Gb).

Results: genome graph annotations

Methods	Kingsford	RefSeq
Column	36.56	80.18
Flat	41.21	121.60
Rainbowfish	23.16	136.65
BRWT	14.05	57.24
Multi-BRWT (5-ary)	13.01	53.09
Multi-BRWT (GPM)	10.60	50.13
Multi-BRWT (GPM + relax up to 20-ary)	**9.95**	**43.62**

Sizes measured in gigabytes (Gb).

Conclusions

- Column-major hierarchical compression provides
  - Up to 29% improvement in compression ratio over baseline BRWT
  - Up to 68% improvement over the state-of-the-art
  - Good row query, much faster column query
- More robust to unique k-mer growth for increasing number of columns
  - Row similarity a by-product of column similarity
- Future work
  - Improvements to column clustering
  - Caching to improve query time
  - Hybrid compression schemes
  - Dynamic operations (insert, delete, rearrange, etc.)

Acknowledgements

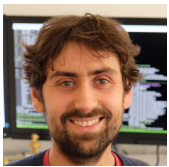
Mikhail Karasikov

Harun Mustafa

Amir Joudaki

Sara Javadzadeh-No

Gunnar Rätsch



André Kahles

Thanks to:
Mario Stanke
Torsten Hoefler
Biomedical Informatics

References

Agarwala, R., Barrett, T., Beck, J., Benson, D. A., Bollin, C., Bolton, E., Bourexis, D., Brister, J. R., Bryant, S. H., Canese, K., Charowhas, C., Clark, K., DiCuccio, M., Dondoshansky, I., Feolo, M. *et al.* (2017). Database Resources of the National Center for Biotechnology Information. *Nucleic Acids Research*.

Almodaresi, F., Pandey, P., and Patro, R. (2017). Rainbowfish: A Succinct Colored de Bruijn Graph Representation. In R. Schwartz and K. Reinert, editors, *17th International Workshop on Algorithms in Bioinformatics (WABI 2017)*, volume 88 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 18:1–18:15, Dagstuhl, Germany. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

Almodaresi, F., Pandey, P., Ferdman, M., Johnson, R., and Patro, R. (2019). An efficient, scalable and exact representation of high-dimensional color information enabled via de bruijn graph search. In *International Conference on Research in Computational Molecular Biology*, pages 1–18. Springer.

Barbay, J., Claude, F., and Navarro, G. (2013). Compact binary relation representations with rich functionality. *Information and Computation*.

Kokot, M., Długosz, M., and Deorowicz, S. (2017). KMC 3: counting and manipulating k-mer statistics. *Bioinformatics*, **33**(17), 2759–2761.

Muggli, M. D., Bowe, A., Noyes, N. R., Morley, P. S., Belk, K. E., Raymond, R., Gagie, T., Puglisi, S. J., and Boucher, C. (2017). Succinct colored de Bruijn graphs. *Bioinformatics*.

Solomon, B. and Kingsford, C. (2018). Improved Search of Large Transcriptomic Sequencing Databases Using Split Sequence Bloom Trees. *Journal of Computational Biology*, **25**(7), 755–765.

Appendix

Construction time, RAM usage

Methods	Kingsford	RefSeq
Flat	2.2h 75GB	29h 201GB
Rainbowfish	5h 287GB	100h 1.6TB
BRWT (p=4)	9h 82GB	21h 83GB
Multi-BRWT (5-ary, p=4)	4h 44.5GB	10h 82.6GB
Multi-BRWT (GPM) (p=30)	3h 99GB	10.5h 103GB
Multi-BRWT (GPM + relax 20)	5.5h(2.5h) 99GB(12GB)	22.5h(12h) 103GB(47GB)

Sizes measured in gigabytes (Gb).

Graph construction

Kingsford data

- Extracted k-mers using KMC Kokot *et al.* (2017)
- Filtered k-mers using approach in Almodaresi *et al.* (2017)
- Added k-mers to a hash table
- Annotate only canonical k-mers

RefSeq data

- Extract k-mers using in-house encoder
- Populate succinct bit vector indexed by k-mer encodings
- Annotate forward k-mers

Simulated data: fewer columns