Recent advancements in approximate coloured compacted de Bruijn graph representations

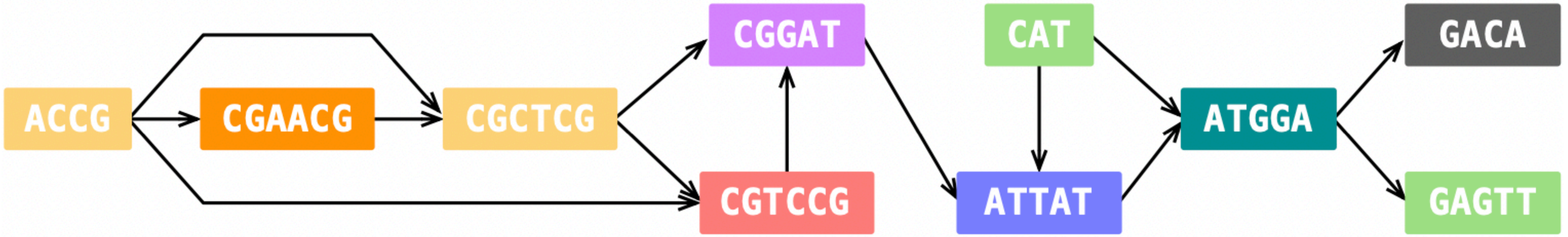
Work-in-progress

Yoshihiro Shibuya, Pierre Peterlongo, Giulio Ermanno Pìbiri — SeqBIM 2023, Lille, France

Overview

- Introduction
  - Colored compacted de Bruijn graphs (ccdBGs)
  - Exact representations
  - Approximate representations
- Main question
- Idea
- Future directions

Colored (compacted) de-Bruijn graphs



- Given a set of references $\mathscr{R} = \{R_1, R_2, \ldots, R_n\}$, the de Bruijn graph (dBG) of $\mathscr{R}$ is a directed graph where:
  - Nodes are all the distinct k-mers of $\mathscr{R}$
  - There is an edge between node u and v iff the (k-1)-length suffix of u is equal to the (k-1) prefix of v
- Non-branching paths can be compacted into contiguous strings called “unitigs” (cdBG)
- Annotating each k-mer with the set of references it appears in, and by compacting only the non-branching paths with the same colours, we obtain the coloured compacted de Bruijn Graph (ccdBG)

Applications

- Pangenomics
- Metagenomic classification
- Abundance estimation
- Long-term storage

NOTE:
In this talk we will focus on static data structures (no updates),
that support queries (no long-term storage)

Exact ccdBG representations

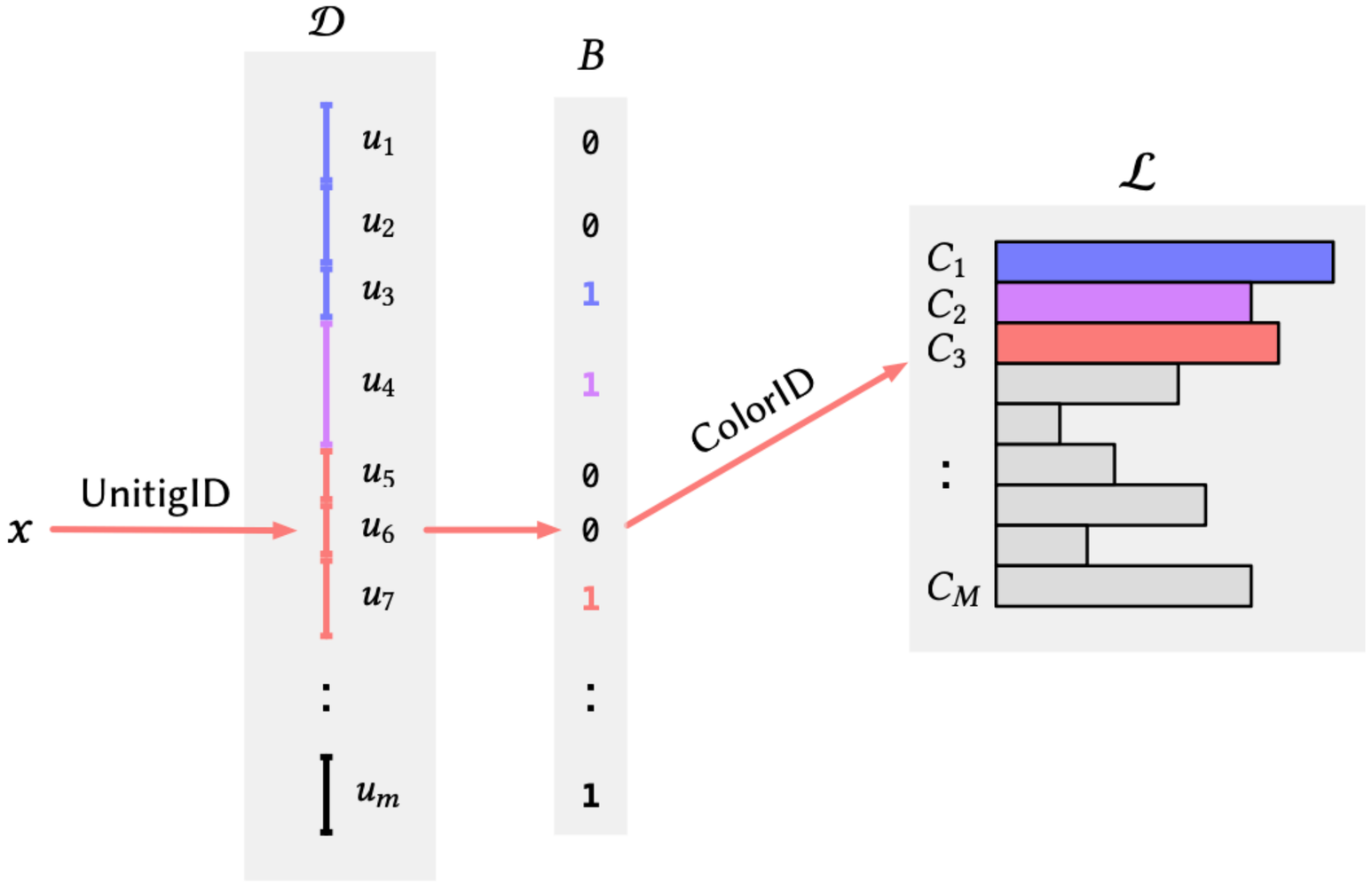
- Given a k-mer K,
    - if $K \in \mathscr{R}$ then return its color id
    - Otherwise return “null”
- Most solutions are divided into 3 parts:
    1. K-mer storage
    2. Colours storage
    3. Mapping

Fulgor (1/2)

- Exact ccdBG representation
  - k-mers are stored by using SSHash
    - Unitigs are explicitly stored in the given input order (order-preserving property)
  - Colors are lists of reference ids
  - Mapping k-mers to their colors is done by a simple bit-vector

Fulgor (2/2)

- Bit-vector B (the mapping) has negligible space in practice
- Almost all space is for storing unitigs and colors



Approximate cddBG representations

Two types of approximations:

1. Error-prone
2. Membership-oblivious

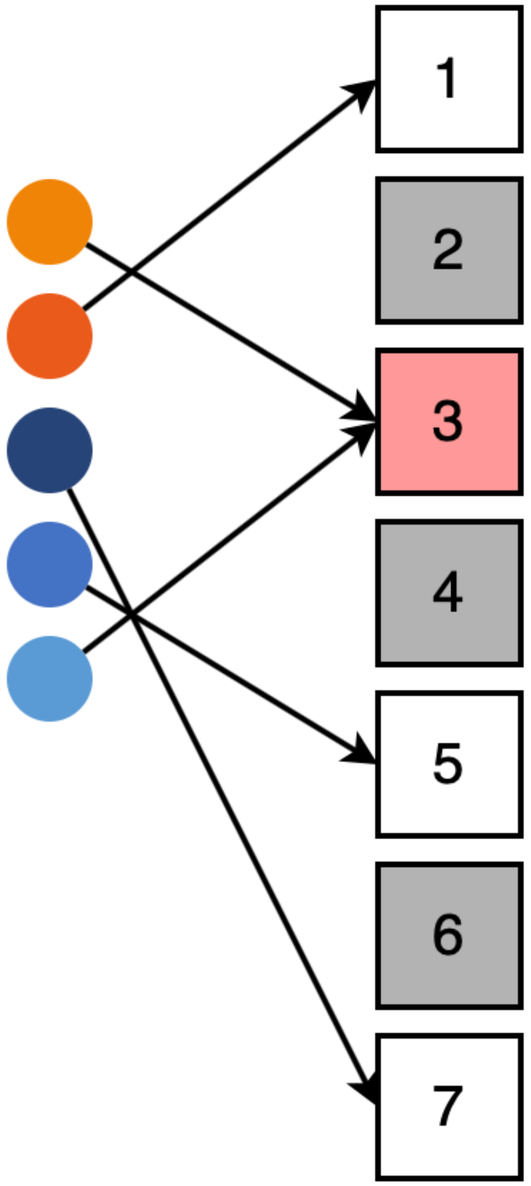
Error-prone

- False positives/negatives
    - A k-mer can appear to be present/absent even if it’s not
- Query errors
    - Some k-mers can be assigned the wrong color id
        - The actual inverted list pointed to the color id can be “almost” correct
- Usually implemented with Bloom Filters or similar approximate data structures

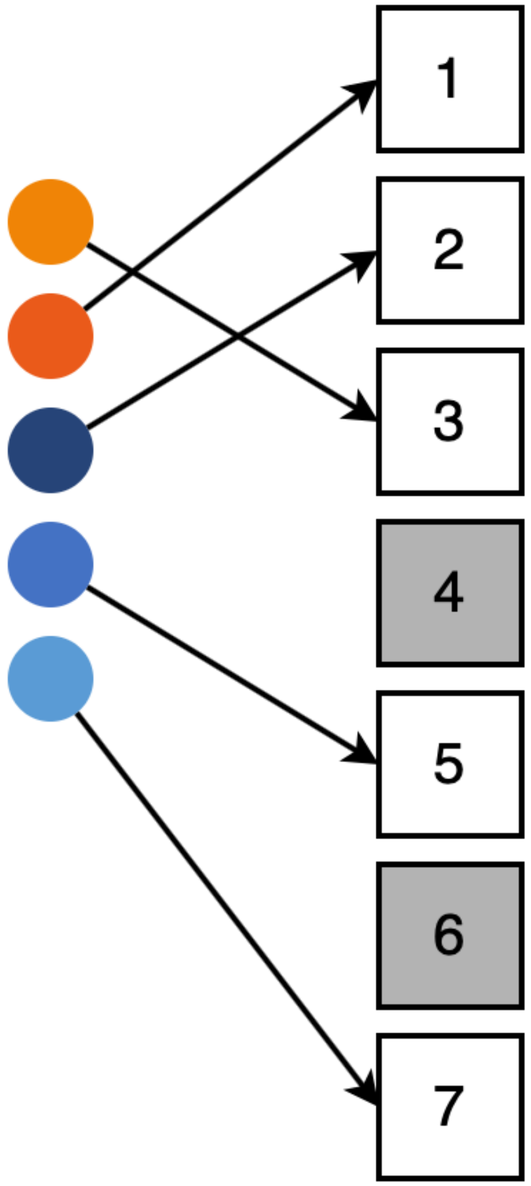
Membership-oblivious

- Given a k-mer K,
    - if $K \in \mathscr{R}$ then return its color id
    - Otherwise return a random answer
- Usually implemented with Minimal Perfect Hash Functions (MPHFs)
    - Locality-preserving MPHFs have been recently proposed

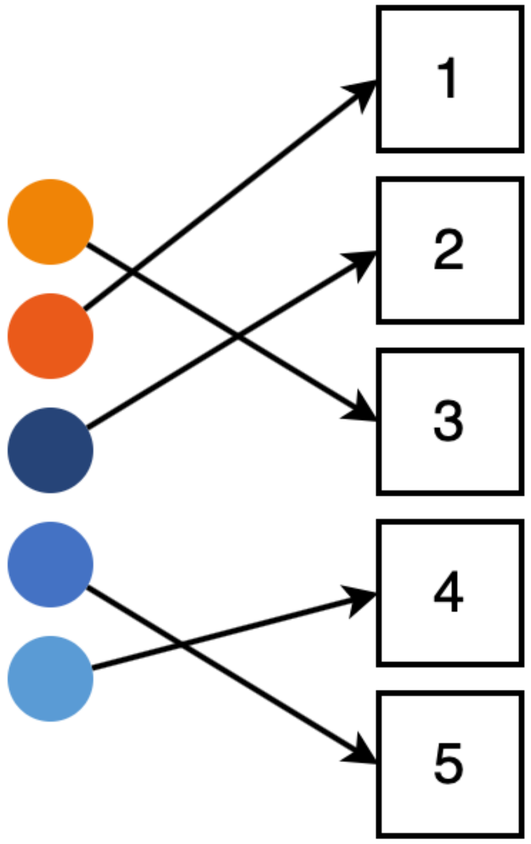
Universal



Perfect

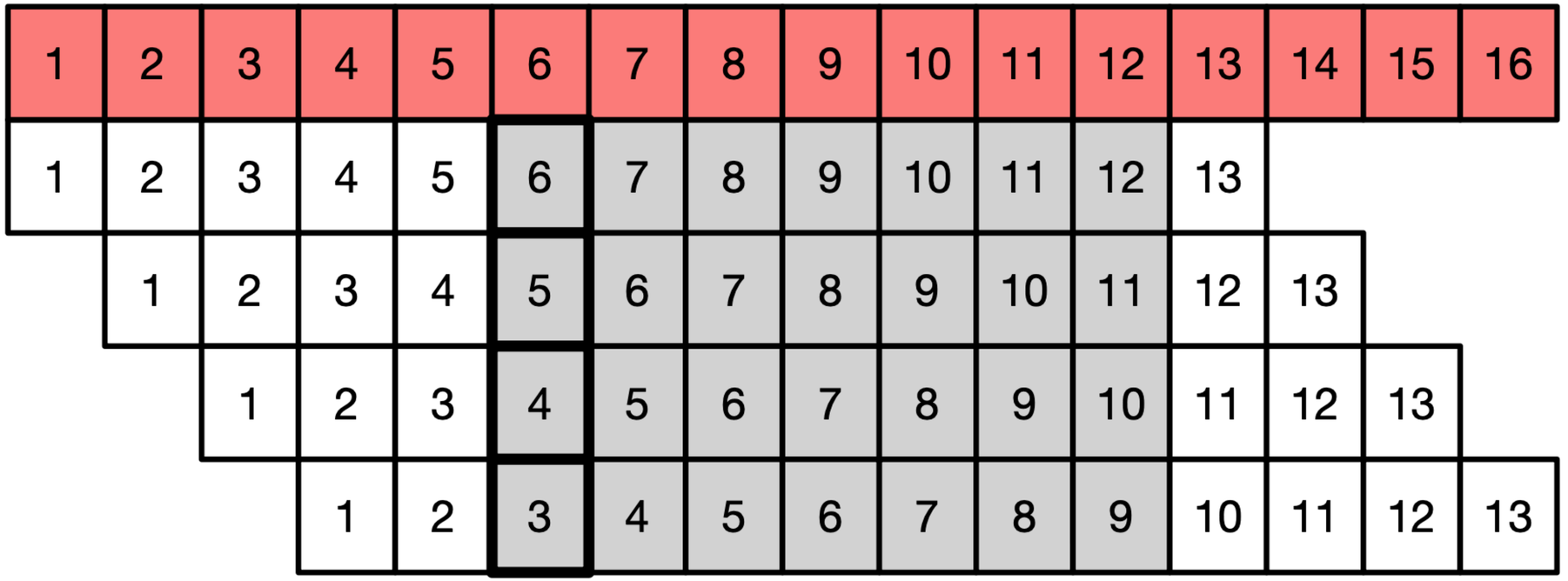


Minimal Perfect



LPHash

- Assigns consecutive hash values to consecutive k-mers in the same super-k-mer by looking at the minimizer position
- For suitable values of k and m, space < 1.44 (theoretical lower-bound of g.p. MPHFs)

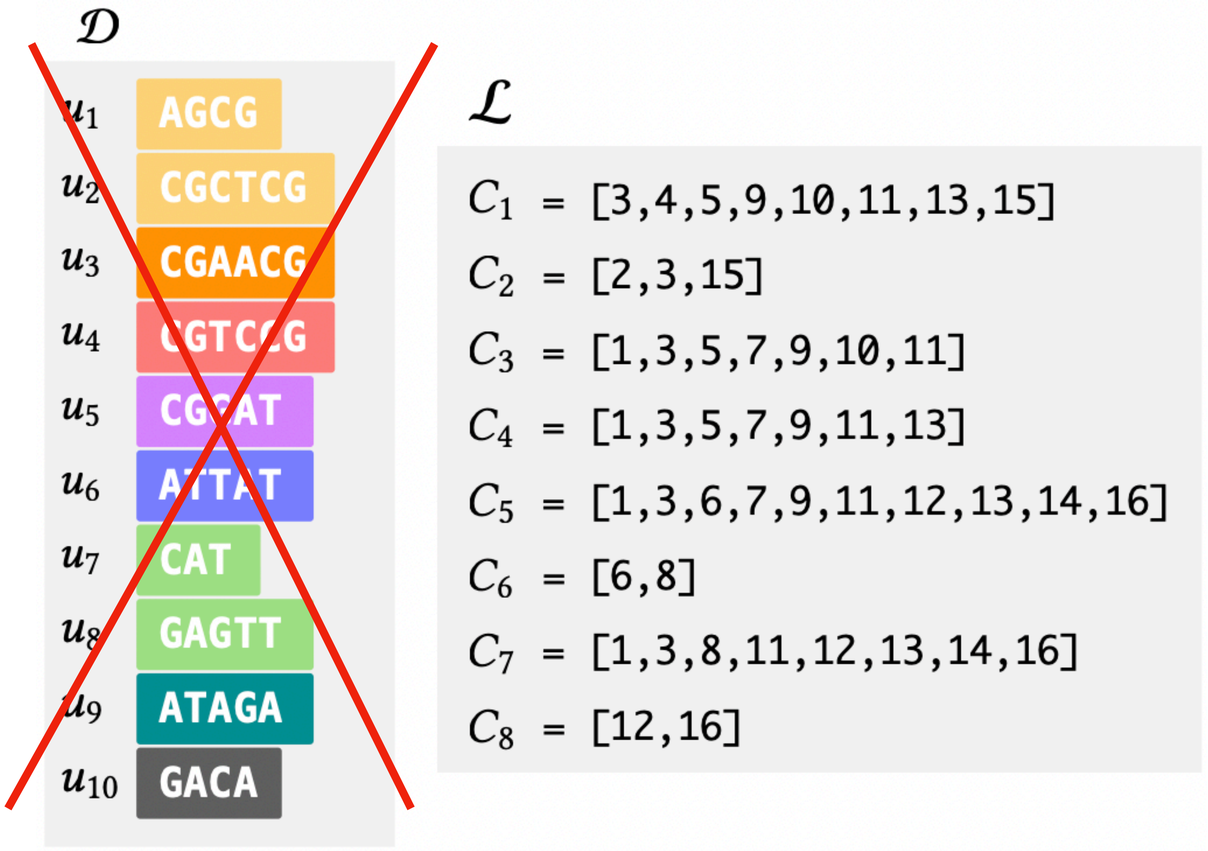


Main question

Can we achieve further space reduction in ccdBG representations if not interested in membership queries ?

- Neighboring k-mers tend to have the same color
    - High compression if locality is properly exploited
- No need to save the actual unitigs

雷 Kaminari



Replace with LPHash

- Same thing as Fulgor but with LPHash instead of SSHash
- RLE array maps hash values to color ids
  - Values are stored in $\log_2(\text{cid}_{\max})$bits
  - Lengths are compressed with Elias-Fano
- The color data structure is the same

Datasets

- [Salmonella] 4546 Salmonella enterica genomes
    - Genomes are similar to each other
    - Basically a pangenome

- [HumGut] 30691 (human) gut bacteria genomes
    - More heterogeneous dataset
    - Bloom-filter-based approaches work well

Results

Dataset	k	M	Fulgor size [GB]	Kaminari size [GB]
Salmonella	31	20	0.266	0.261
HumGut (whole)	31	17	34	Error
HumGut (r = 6)	31	17	6.2	11
HumGut (r = 6)	63	31	Max k=32	7.3
HumGut(r = 60)	31	17	0.764	0.837

Why?

- Currently, LPHash cannot deal with ambiguous minimizers (minimizers which appear in multiple super-k-mers)
    - On HumGut (whole) for example, ~40% of minimizers are ambiguous
- On simpler cases (e.g. Salmonella), Fulgor is still better than Kaminari
    - LPHash while being locality-preserving still breaks colors into small blocks which are hashed at different offsets
        - SSHash's order-preserving property is on the entire input set
    - The RLE mapping in Kaminari takes most of the space (Fulgor's map is negligible in size)

Future directions

- Improve LPHash
    - Deal with ambiguous minimizers in a smarter way (other than building a classic fallback MPHF)
- Better compress the mapping k-mers—colors?
- Others?

Open question

- Are exact representations closing the gap to their approximate counterparts?

	Genomes	Mac-dBG			Fulgor			Themisto			MetaGraph			COBS
		dBG	Colors	Total	dBG	Colors	Total	dBG	Colors	Total	dBG	Colors	Total	Total
EC	3,682	0.29	0.52	0.81	0.29	1.36	1.65	0.22	1.85	2.08	0.10	0.23	0.33	7.53
SE	5,000	0.16	0.16	0.32	0.16	0.59	0.75	0.14	1.29	1.43	0.07	0.19	0.26	9.11
	10,000	0.35	0.33	0.68	0.35	1.66	2.01	0.32	3.50	3.81	0.13	0.38	0.51	18.68
	50,000	1.26	2.14	3.40	1.26	17.03	18.30	1.07	32.42	33.48	0.36	1.95	2.31	88.61
	100,000	1.72	3.83	5.55	1.72	40.70	42.44	1.35	75.94	77.28	0.45	3.50	3.95	173.58
	150,000	2.03	5.37	7.40	2.03	68.60	70.66	1.58	125.16	126.74	—	—	—	265.49
GB	30,691	21.31	7.85	29.16	21.31	15.45	36.85	18.33	30.88	49.21	5.23	4.77	10.00	21.23

- Can a membership-oblivious data structure (no false positives) beat COBS and the other Bloom filter based algorithms ?

Taken from: Meta-colored compacted de Bruijn Graphs, G.E. Pìbiri, J. Fan, R. Patro, 2023, bioRxiv


That's all Folks!