Structures de données pour les grands ensembles de k-mer

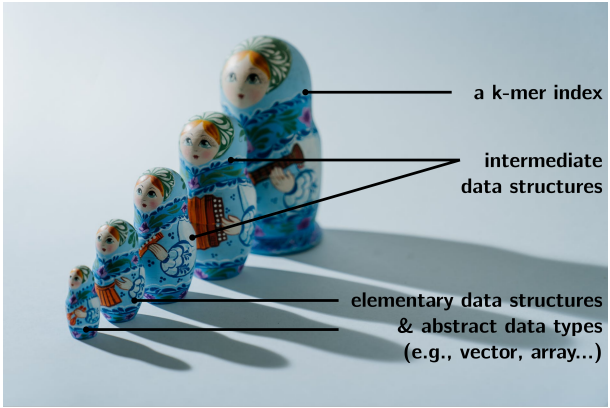
Séminaire MIAT
28 janvier 2022

Camille Marchet
CNRS, CRIStAL Lille, France

camille.marchet@univ-lille.fr
@CamilleMrcht

Introduction - Data structures

- Russian dolls
  - Legos or "building blocks"
  - *Abstract data types*: set, multi-set, list...
- Existence (or not) of an implementation



adapted from copyright free, @cottonbro on Pexel

Introduction - Data structures

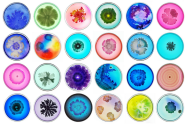
Data structures are purposeless without **operations**

- Test for emptiness
- Add/delete elements
- Check membership of an element
- Go over all elements

Operations go hand in hand with notions of cost and complexity

- Computation time
- Size in memory

Introduction - When it comes to *k*-mers



	human	*Pinus taeda*	*Ambystoma mexicanum*	*Paris japonica*	metagenomics
# 31-mers	3.2G	10.5G	18.5G	~150G	...

In practice :

- *k* sizes : 11-15 (long reads), 21-51 (short reads)
- Billions (of distinct *k*-mers) easily reached in experiments
- The notion of cost becomes central

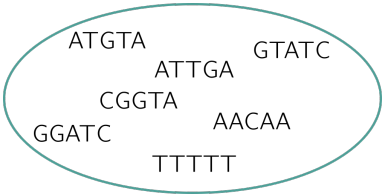
Introduction - k-mer data structures

These data structures are foundations of many applications:

- alignment methods
- alignement-free methods (pseudo-alignement, quantification, taxonomic assignation, ecological distances. . .)
- quality analysis, read correction
- representation and usage of graphs (assembly, variant calling, . . .)

Data structures for k-mers sets

Representation of k-mer sets



k-mer sets - Burrows Wheeler transform?[1]

text

```
row_row_row_your_boat
row_row_row_your_boat
row_row_row_your_boat$
```

Burrows Wheeler transform (BWT)

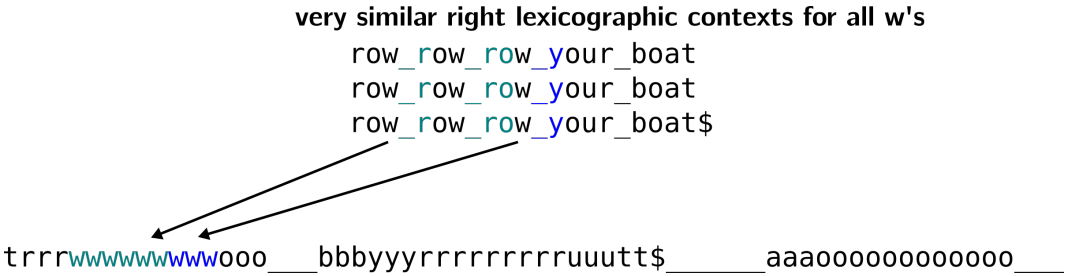
```
trrrwwwwwwwwooo___bbbyyyrrrrrrrrruuutt$_____aaaooooooooooooo___
```

Compression through run length encoding

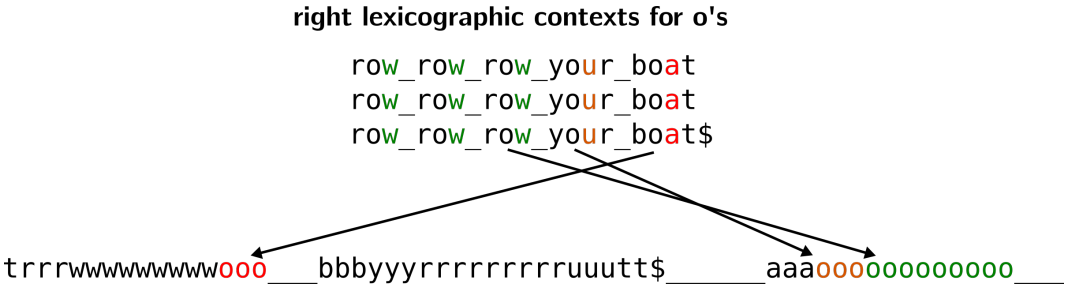
```
(t,1)(r,3)(w,9)(o,3) ... (_,3)
```

[1] Adapted from Ben Langmead's course

k-mer sets - Right contexts of w's



k-mer sets - Right contexts of o's



k-mer sets - BWT for *k*-mer sets?

- BWTs can be queried with strings of arbitrary length: answer to *k*-mer membership
- *k*-mer sets are not texts (while genomes are): BWTs will index read sets or genomes
- Which other structure can we choose?

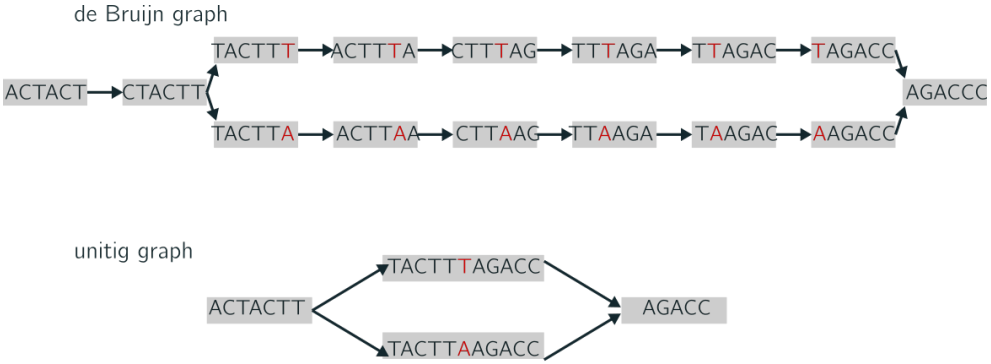
k-mer sets - Representation using a de Bruijn graph[2]

k-mer set

ACTACT TACTTA
CTACTT ACTTAA
TACTTT CTTAAG
ACTTTA TTAAGA
CTTTAG TAAGAC
TTTAGA AAGACC
TTAGAC TAGACC
AGACCC

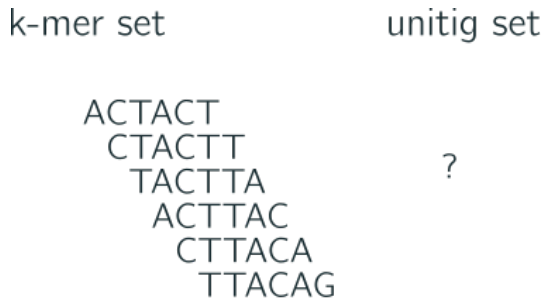
[2] I use the *node-centric* definition of a DBG

k-mer sets - Representation using unitigs

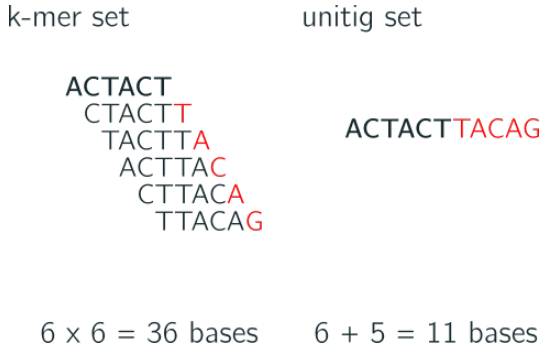


- Compacted de Bruijn graph

k-mer sets - Representation using unitigs

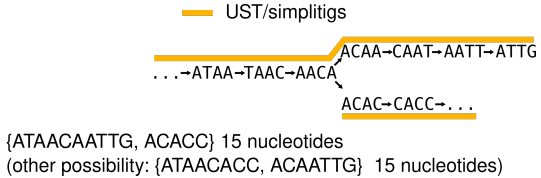


k-mer sets - Representation using unitigs



k-mer sets - Spectrum preserving string sets

- Other sequences extracted from the DBG are now used
- Spectrum preserving string sets (SPSS)[3]



- [Rahman et al. 2020, Brinda et al. 2020] Greedy algorithm, nearly optimal
- Applications: de Bruijn graph implementation, alignment
- Open question: constraints on SPSS

[3] A brief description of several SPSS: `https://kamimrcht.github.io/webpage/tigs.html`

k-mer sets - Two paradigms

- Full-text methods (BWT) rely on the **lexicographic context** of nucleotides to provide a compressed representation
  - Needs a text such as a genome as an input

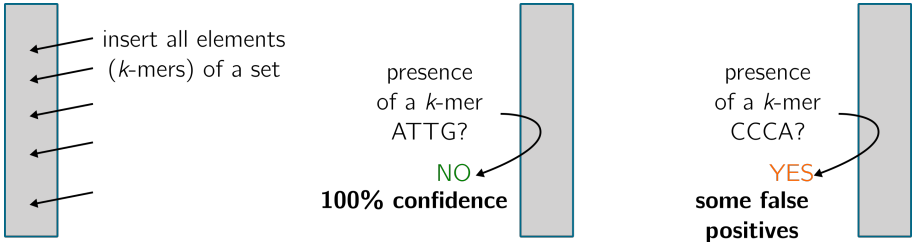
- SPSS-based methods rely on the **genomic context** ("assemblability") of nucleotides to provide a compacted representation
  - Will structure a *k*-mer set according to the underlying genome

de Bruijn graphs:

- Can be seen as objects to:
  1. assemble sequences
  2. represent a *k*-mer set and to structure the redundancy of datasets in some way
- Interesting feature: facilitate error correction/filtering → impact on performances

k-mers sets - A third way: probabilistic representation

Bloom filters [Bloom 1970]



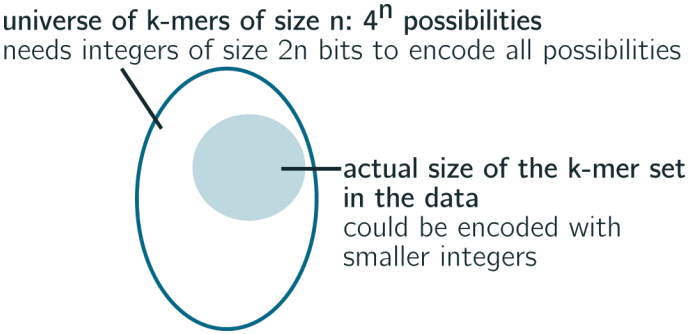
- Extreme simplicity in terms of implementation: a bit array + hash functions

k-mers sets - A third way: probabilistic representation

base encoding	k-mers, k=2	binary representation	
A 00	AC	0001	
C 01	AA	0000	4 bits
G 10	TG	1110	for 2 bases
T 11	...		

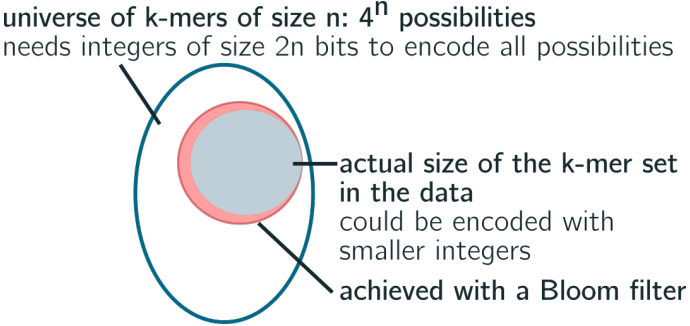
in practice, 64 bits integers for *k*-mers of size up to 32

k-mers sets - A third way: probabilistic representation



- Example: $\sim 3.10^9$ 31-mers in the human genome vs $\sim 4.10^{18}$ possible 31-mers

k-mers sets - A third way: probabilistic representation



- Example: $\sim n = 3.10^9$ human 31-mers, BF size of $10n$ bits
- $\sim 30.10^9$ bits or $\sim$ 4GB and a false positive rate of $\sim \frac{1}{10^f}$
- Applications: assembly (Minia, Abyss, Hifi-asm), *k*-mer counting (Jellyfish), alignment (Minimap)

Data structures for k-mers sets

Associative indexes for k-mer sets

ATGTA : 6
GTATC : 127
ATTGA : 2
CGGTA : 53
AACAA : 55
TTTTT : 272

k-mers sets index - full-text methods

Full-text methods based on the BTW, with the same limitations as previously stated:

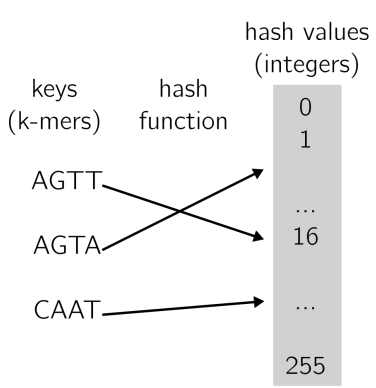
- FM-index [Ferragina & Manzini 2004], r-index [Gagie et al. 2017] (improves on space complexity)

Use the paths of the de Bruijn graph as a text, then index the *k*-mers:

- BOSS [Bowe et al. 2012]: a FM-index specialized for *k*-mers
- Applications: indexing large collections of bacterial datasets [Muggli et al. 2017,2019], implementing de Bruijn graphs [Boucher et al. 2015; Karasikov et al. 2021]
- Main downsides:
  - Hypothesis on the paths lengths
  - Slower query in comparison to other approaches

k-mers sets index - associate (key,value) pairs with a hash table

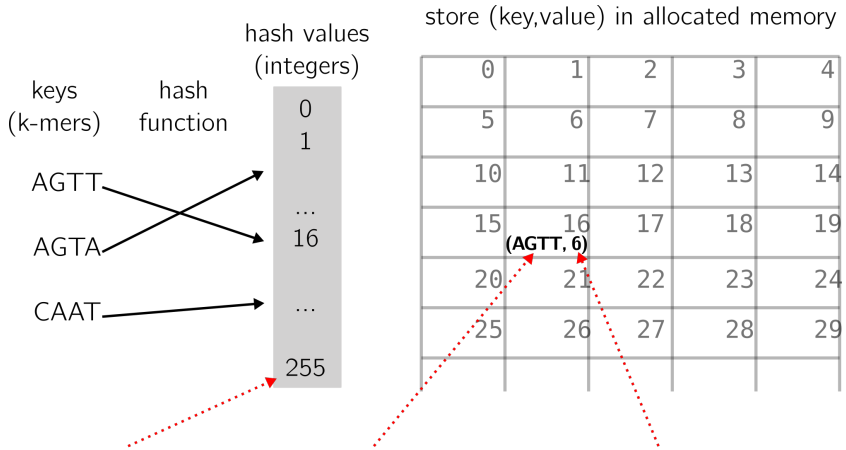
associate k-mers to their abundances
AGTT:6, AGTA:100, CAAT:4



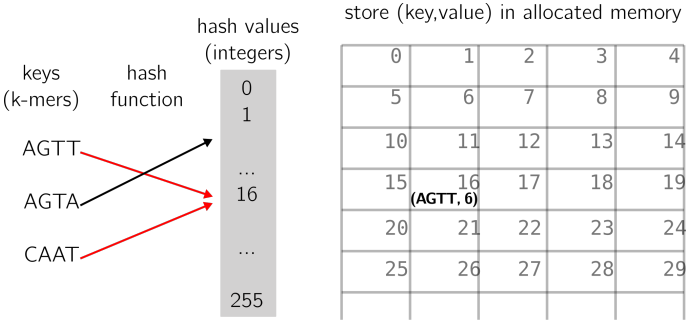
store (key,value) in allocated memory

0	1	2	3	4
5	6	7	8	9
10	11	12	13	14
15	16 **(AGTT, 6)**	17	18	19
20	21	22	23	24
25	26	27	28	29

k-mers sets index - hash table



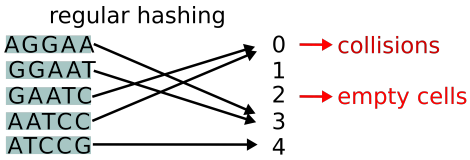
k-mers sets index - collisions: a big issue

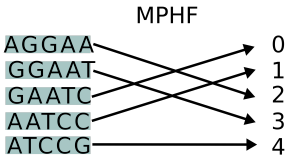


Solutions:

- Pointers: decrease space performances
- Open addressing: decreases time performances (loss of locality)
- A third way...

k-mer sets index - Minimal perfect hash functions (MPHF)





Implementations used in bioinformatics:

- BBHASH [Limasset et al. 2014]
- PTHash [Pibiri et al. 2021]

- ⚠MPHFs are static
- ⚠MPHFs are **only** hash **functions**, in order to build a hash table they need a representation of the keys to deal with alien keys

k-mer sets index - Specialized hash tables

Efficient *k*-mer hash tables:

- Pufferfish: MPHF+unitigs [Almodaresi et al. 2018]
- BLight: MPHF+partitioning+SPSS [Marchet et al. 2019][4]
- Counting quotient filters [Pandey et al. 2017]: another hashing strategy

Applications:

- Example of achievement:
  index the 31-mers of the human genome in RAM in <8GB (BLight)
- Counting *k*-mers [Pandey et al. 2018]
- Large scale quantification [Marchet et al. 2020]
- Read alignment [Almodaresi et al. 2021]

[4] and a promising recent preprint by Pibiri in 2022

Summary on indexing k-mer sets

- Full-text (BWT-based) or hashing+SPSS appear to be the two major ways for indexing k-mers
  - Tradeoff: BWT-based has more expressivity (order preserving) but lower performances in practice (notably query)
- Indexing de Bruijn graphs (+navigational operations, dynamicity) is a field on its own [5]

A survey about all these data structures: Chikhi R, Holub J, Medvedev P. 2019. Data structures to represent a set of k-long DNA sequences. *ACM Computing Surveys*

[5] see for instance `http://rayan.chikhi.name/pdf/2021-july-9-cie.pdf`

Data structures for k-mers sets

Collections of k-mer sets

Collections of *k*-mer sets

a set of datasets $\{d_1, d_2, \ldots d_n\}$
(reads multisets)

query sequence

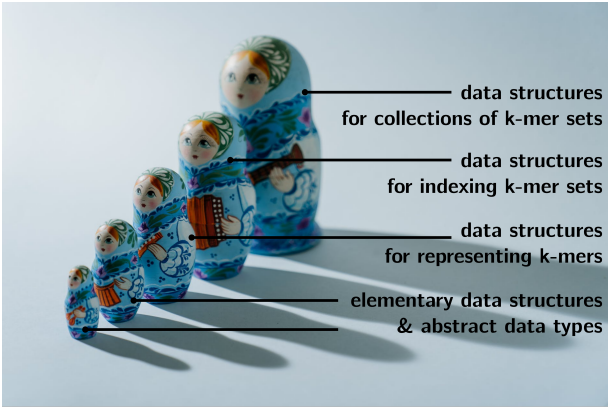
...ATTACGTAGTA...

return all d_i's where the query occurs

- Each dataset (and the query) are seen as sets of *k*-mers
- The query is "present" in a dataset if *enough* of its *k*-mers are found

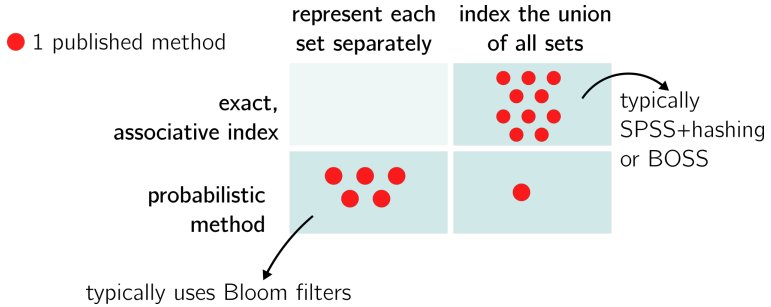
Collections of k-mer sets

We need to handle **multiple** sets of k-mers and query the presence/absence of a sequence



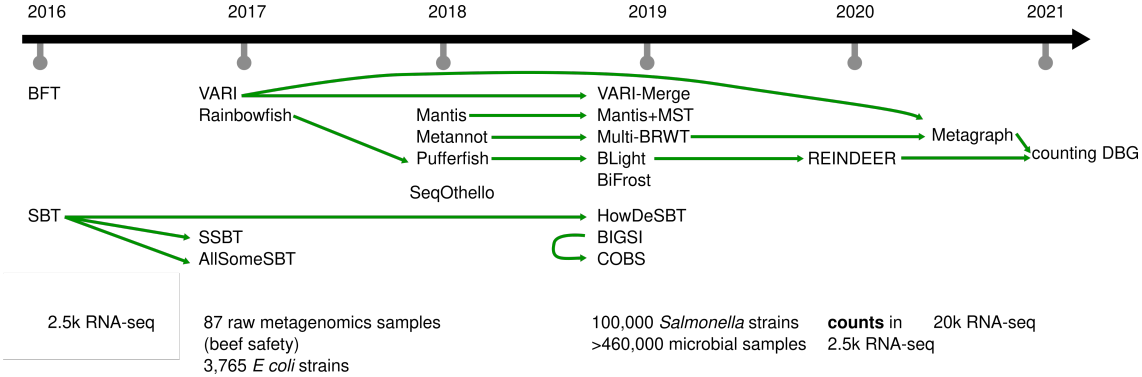
adapted from copyright free, @cottonbro on Pexel

Collections of k-mer sets - State of the art



- Exact methods: for precise, short queries or when colored de Bruijn graphs are needed
- Probabilistic methods: better scalability if false positives are acceptable

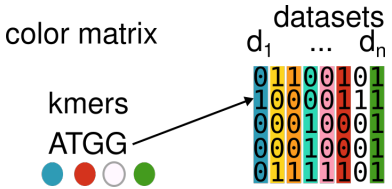
Collections of *k*-mer sets - State of the art



- Different optimizations/features: construction time, space, query speed, dynamicity
- Examples of queries: search of a mutation, alternative splicing, ...

Collections of *k*-mer sets - Exact methods

Associate *k*-mers to color matrix:

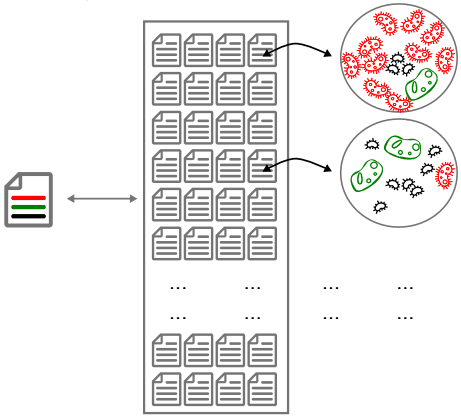


- *k*-mers in SPSS or BWT
- Hash-table or FM-index (BOSS)
- Compression of colored matrix
- Some methods support de Bruijn graph operations

- Examples of exact methods: VARI [Muggli et al. 2017], Mantis [Pandey et al. 2018], BiFrost [Holley & Melsted 2019]

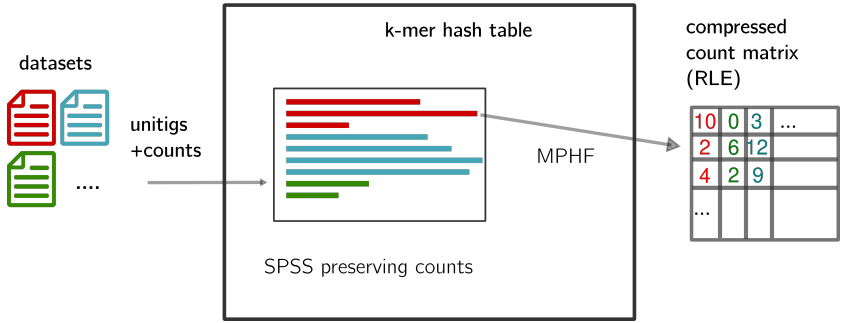
Collections of *k*-mer sets - Quantification

Associate *k*-mers with ~~presence/absence~~ **abundances** in datasets



Collections of *k*-mer sets - large scale abundance index

REINDEER [Marchet et al. 2020][6]



[6] and very recently, counting de Bruijn graphs [Karasikov et al. 2021]

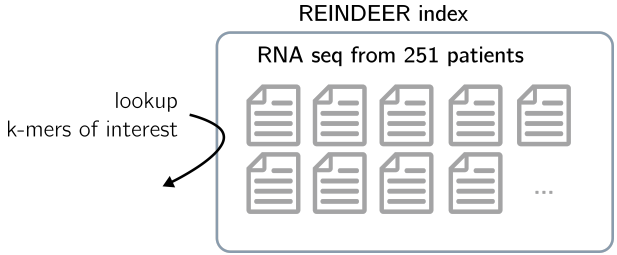
Collections of *k*-mer sets - large scale abundance index

~ 2500 human RNA-seq datasets (~ 4 billions 31-mers)

	Tool	Counts	Time (h)	Peak RAM (GB)	Index Size (GB)
k-mer aggregative	SBT	No	55	25	200
	HowDeSBT	No	10	N/A	15
	BIGSI	No	N/A	N/A	145
color aggregative	Mantis	No	20	N/A	30
	SeqOthello	No	2	15	20
	REINDEER - presence/absence	No	40	27	36
	REINDEER - counts	**Yes**	**45**	**56**	**52**

Collections of *k*-mer sets - example of application

- Stéphane Pyronnet's team @CRCT Toulouse
- Acute myeloid leukemia (AML)
- An (anonymized) WHO gene is a good prognosis indicator for survival. But why?



Conclusion - large scale *k*-mer data structures

- Alignment-free, large scale discovery/search of biological markers
- A survey on set collections data structures: Marchet C, et al 2019. Data structures based on k-mers for querying large collections of sequencing data sets.

Acknowledgments:

- Céline Brouard, Sandra Plancade @MIAT
- Team R'n Blood Toulouse:
  Marina Bousquet, Eulalie Corre, Stéphane Pyronnet
- Team Bio2M Montpellier:
  Chloé Bessières, Anthony Boureux, Benoit Guibert, Thérèse Commes
- Team SeqBio Pasteur Paris:
  Rayan Chikhi
- Team Bonsai Lille:
  Mikaël Salson, Antoine Limasset, Maël Kerbiriou
- Elsewhere on Earth:
  Paul Medvedev, Zamin Iqbal, Christina Boucher