UNIVERSITÄT DES SAARLANDES

SAARLAND UNIVERSITY
SAARBRÜCKEN GRADUATE SCHOOL OF COMPUTER SCIENCE

CBI CENTER FOR BIOINFORMATICS

Algorithmic Bioinformatics

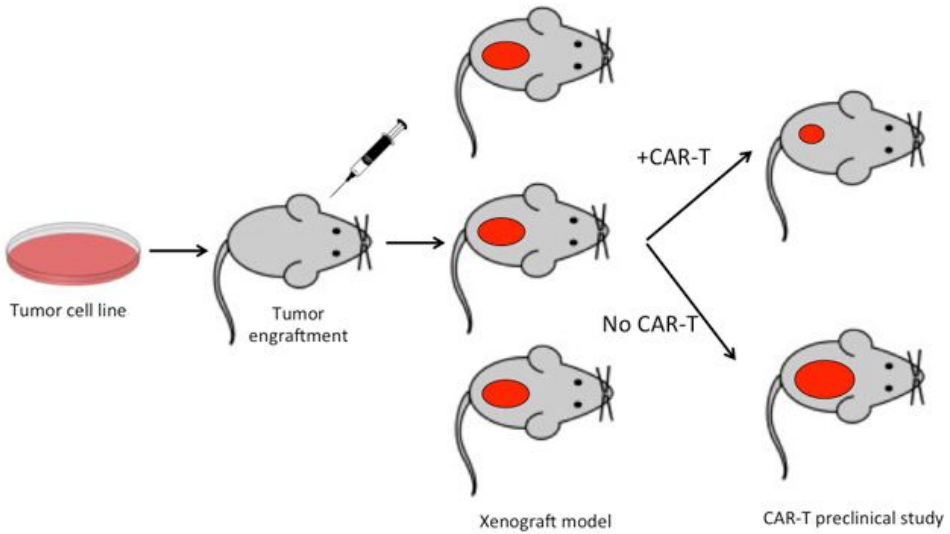
Xengsort2
Ultrafast accurate xenograft sorting

Jens Zentgraf and Sven Rahmann
Algorithmic Bioinformatics, Saarland University

German Conference on Bioinformatics (GCB)
2023, Hamburg

https://gitlab.com/genomeinformatics/xengsort

Xenograft data



- tumor cell lines
  or patient tumor samples
  implanted in mice
- study tumor heterogeneity, evolution
- sequencing of samples
- mixture of human+mouse DNA
- First task: separate/sort reads ("xenograft sorting"), or: extract graft (human) reads
- Problem: Human and mouse genomes are similar

Source: Creative AniModel,
https://www.creative-animodel.com/Featured-Service/Human-Tumor-Xenograft-Model.html

Xenograft sorting

- **Given**: Xenograft sample (Mixed reads from two species, host and graft)
- Sort reads into five categories: host, graft, both, neither, ambiguous

---

Alignment-based approach

- xenofilteR
- align each read against host and graft reference
- classify each read based on alignment %identity

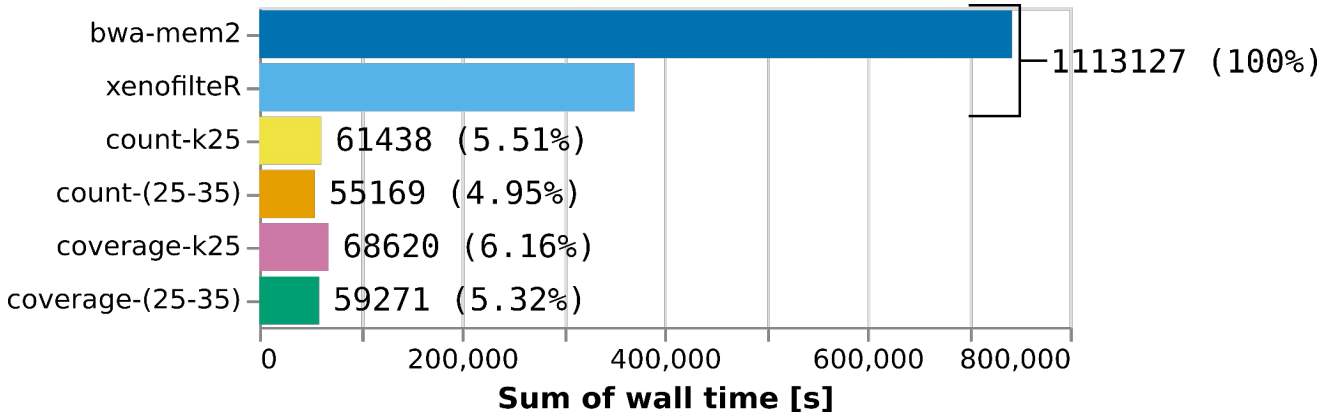
Alignment-free approach

- xengsort / xengsort2
- split each read into *k*-mers
- check how many *k*-mers belong to each species
- classify each read based on membership of *k*-mers to host and graft

Classification speed: Patient-derived xenograft RNA-seq data

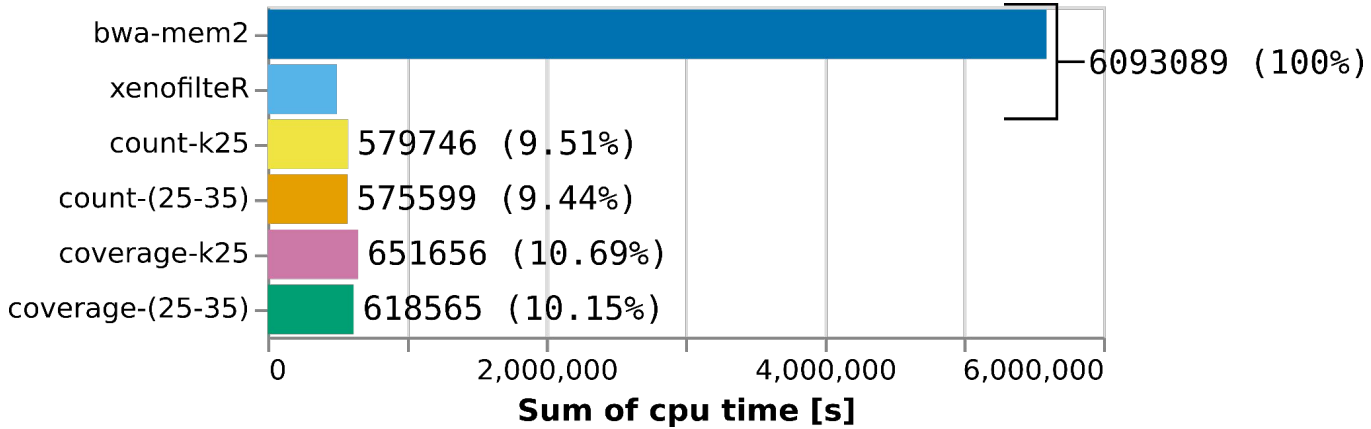
Classification time
(wall clock seconds)

bwa-mem: 8 threads
xenofilteR: 1 thread
xengsort: 8 threads



Classification time
(CPU seconds)

Evaluation on a "good gaming PC":
(AMD Ryzen 9 5950X,
16 cores / 32 threads,
128 GB DDR4 RAM,
16 TB HDD)



xengsort 2 improves many aspects of xengsort

Version 1

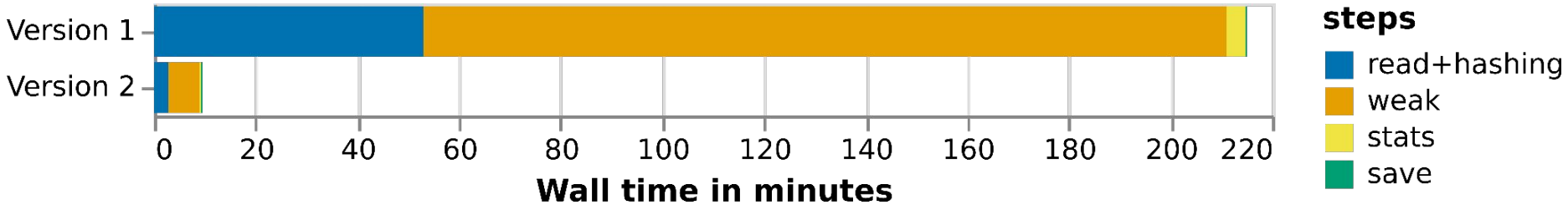
- **contiguous *k*-mers** only
- index: huge single table, **single-threaded** indexing
- identification of weak k-mers
- classification: **count-based** only
- **hard-coded** thresholds, optimized for **short reads** (100bp)
- compressed input (via shell), **uncompressed** FASTQ output
- 1000s of samples: each run loads hash table into memory **again**

Version 2

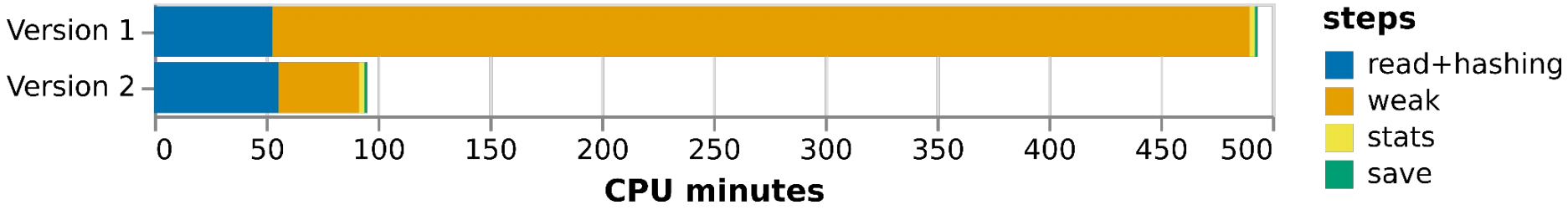
- contiguous and **gapped *k*-mers**
- index: many **subtables**, faster **parallelized** indexing
- **fast** identification of weak k-mers
- classification: count-based or **coverage-based**
- **config files** for different settings, support for **short and long reads**
- **compressed in- and output**, different formats (.gz, .bz2, .xz)
- 1000s of samples: load hash table **once** into **shared memory**

Speed: Index building (v1 vs. v2)

Indexing time (wall clock minutes)
with 15 subtables (v2)

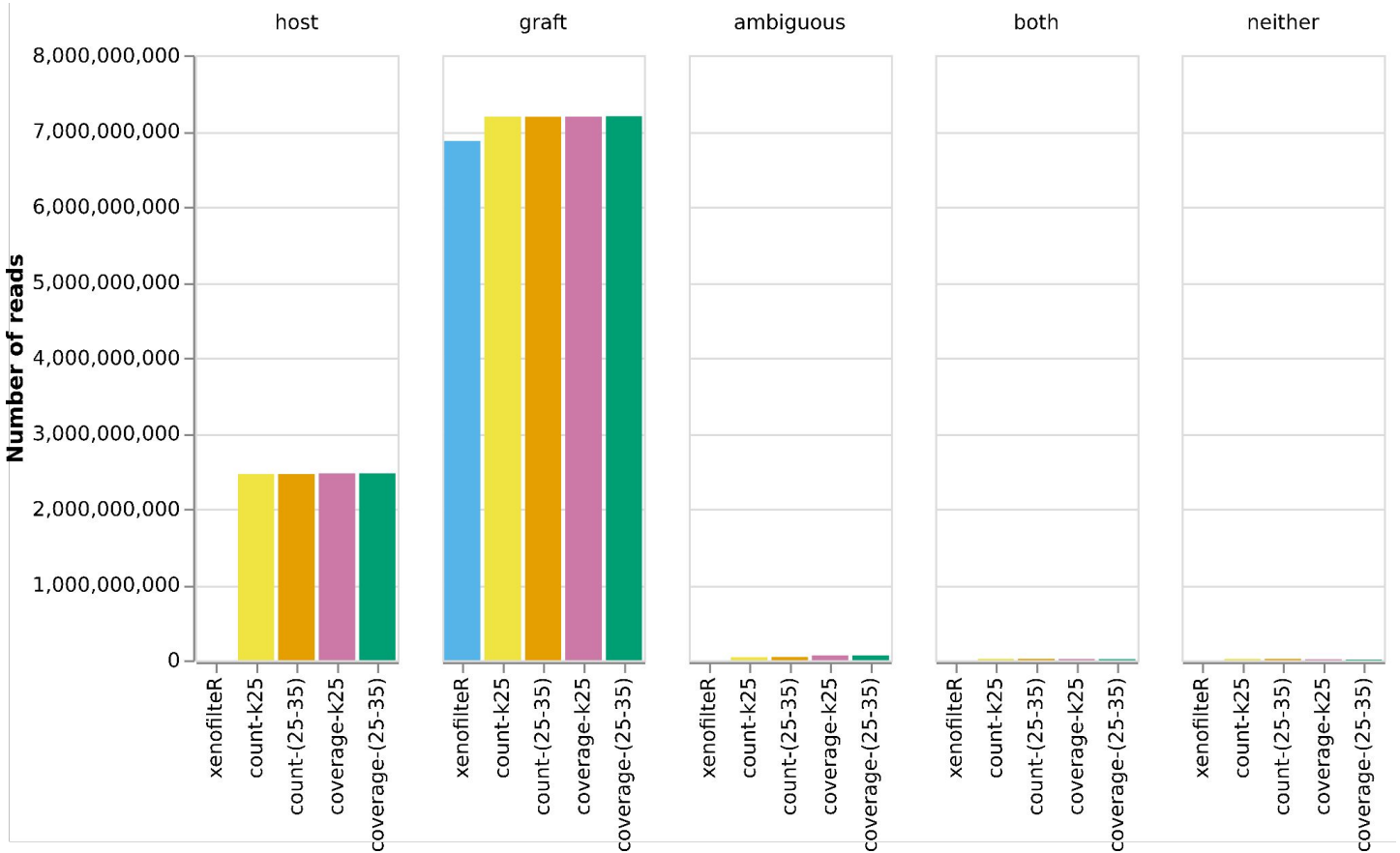


Indexing time (CPU minutes)



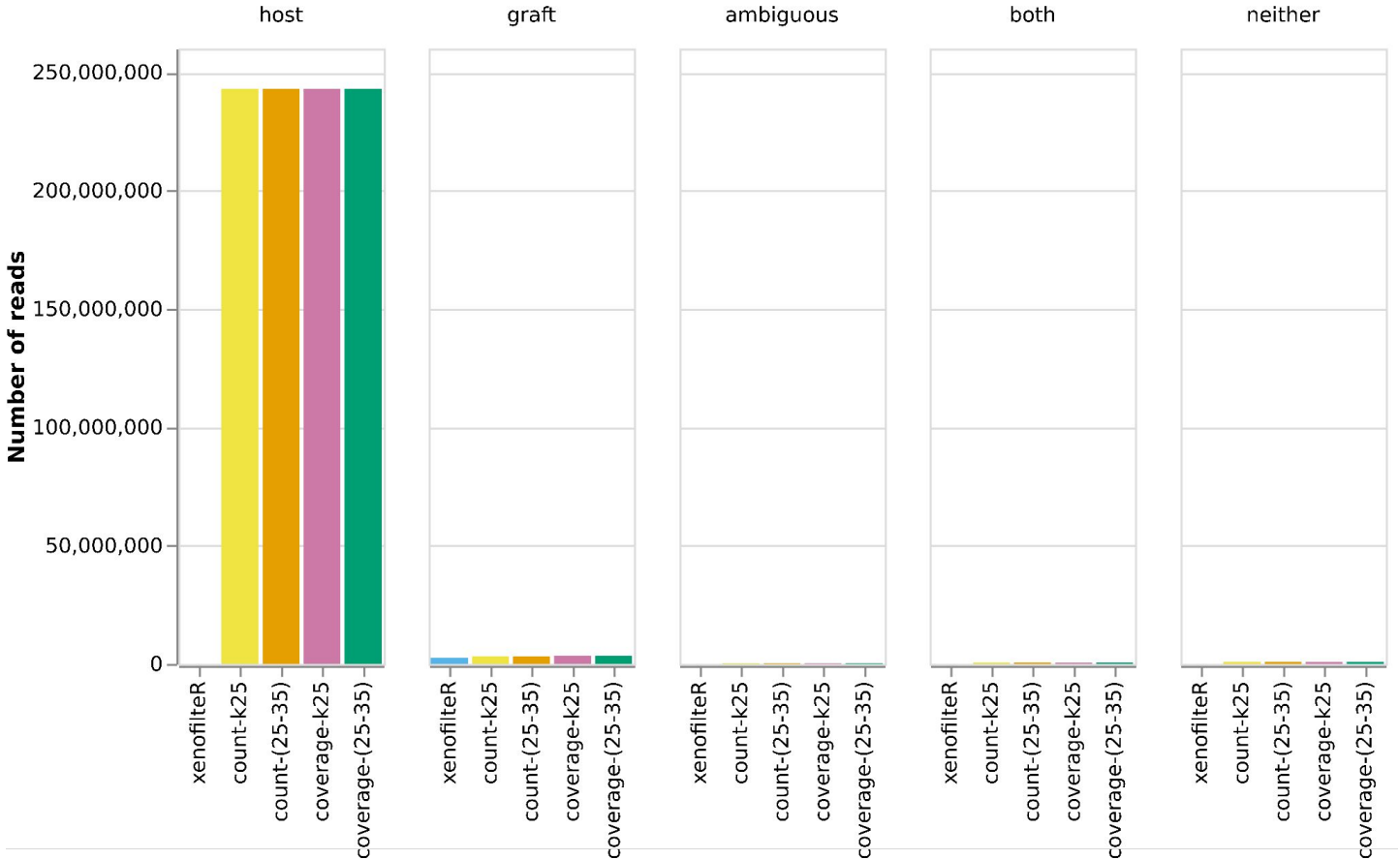
Accuracy: Patient-derived xenografts (graft, some host)

- 174 samples
- RNA-seq
- xenografts of pancreatic tumors
- total: 872 GB .fastq.gz
- almost 10 billion reads



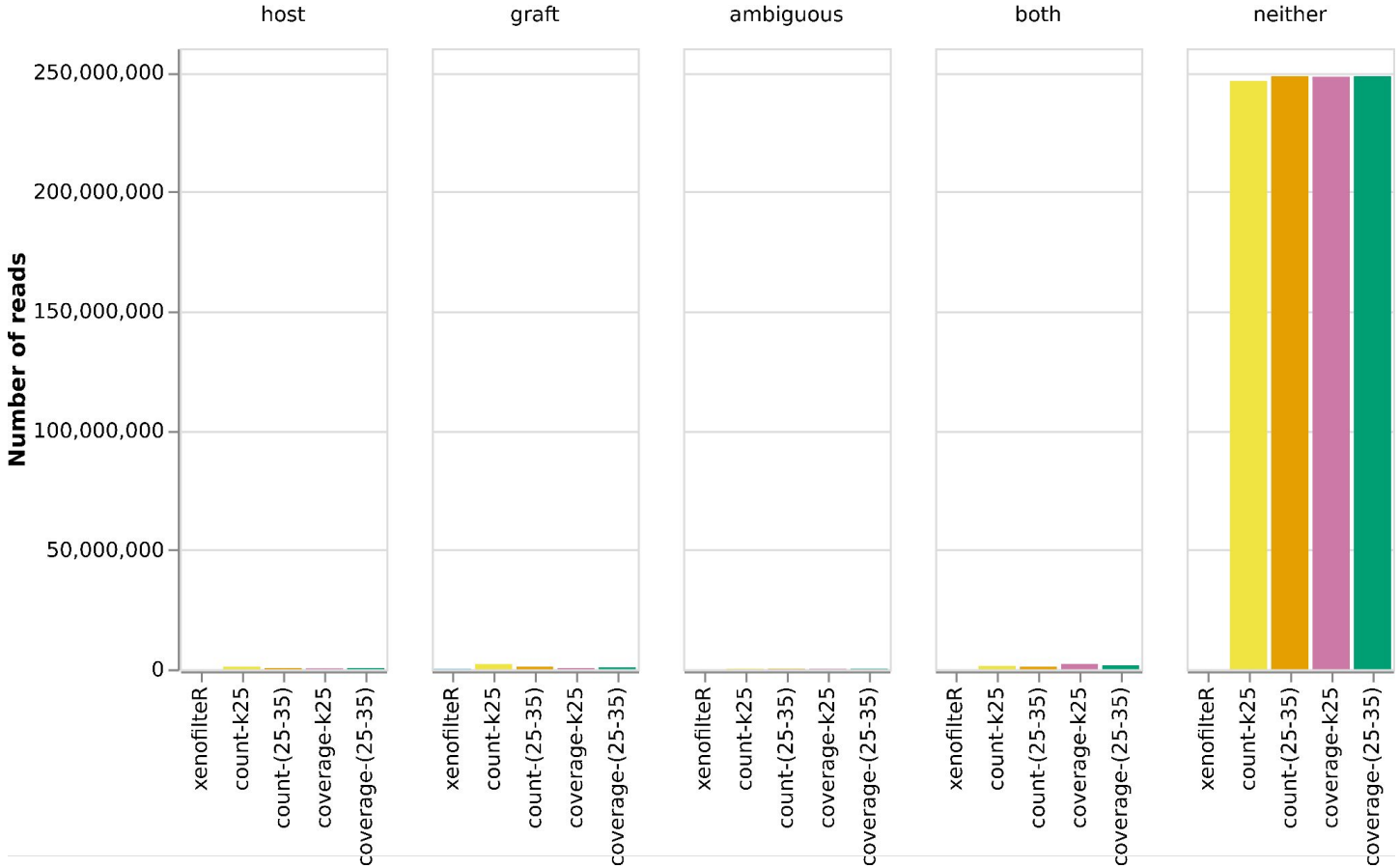
Accuracy: Human-captured mouse exomes (all host)

- Mouse exomes
- Human exome capture kit
- 4 samples
- paired-end reads
- total 35GB .fastq.gz



Accuracy: Chicken (all neither)

- 4 samples
- paired-end
- total 47 GB .fastq.gz



Selected method highlights

- gapped *k*-mers
- index: multi-way bucketed Cuckoo hash table, parallelized with subtables
- fast identification of weak k-mers
- classification: count-based or coverage-based
- config files supporting short and long reads
- compressed in- and output, different formats (.gz, .bz2, .xz)
- use shared memory for classification

Selected method highlights

- gapped *k*-mers
- index: multi-way bucketed Cuckoo hash table, parallelized with subtables
- fast identification of weak k-mers
- classification: count-based or coverage-based
- config files supporting short and long reads
- compressed in- and output, different formats (.gz, .bz2, .xz)
- use shared memory for classification

Contiguous *k*-mers

- Single parameter *k*
- Easy to handle
- One error changes
  k **consecutive** *k*-mers

```
###
TACAGATATA
TAC GAT
 ACA ATA
  CAG TAT
   AGA ATA
```

Contiguous *k*-mers

- Single parameter *k*
- Easy to handle
- One error changes
  k **consecutive** *k*-mers

```
###
TACAGATATA
TAC GAT
 ACA ATA
  CAG TAT
   AGA ATA
```

(*k*=3, *w*=5, symmetric)

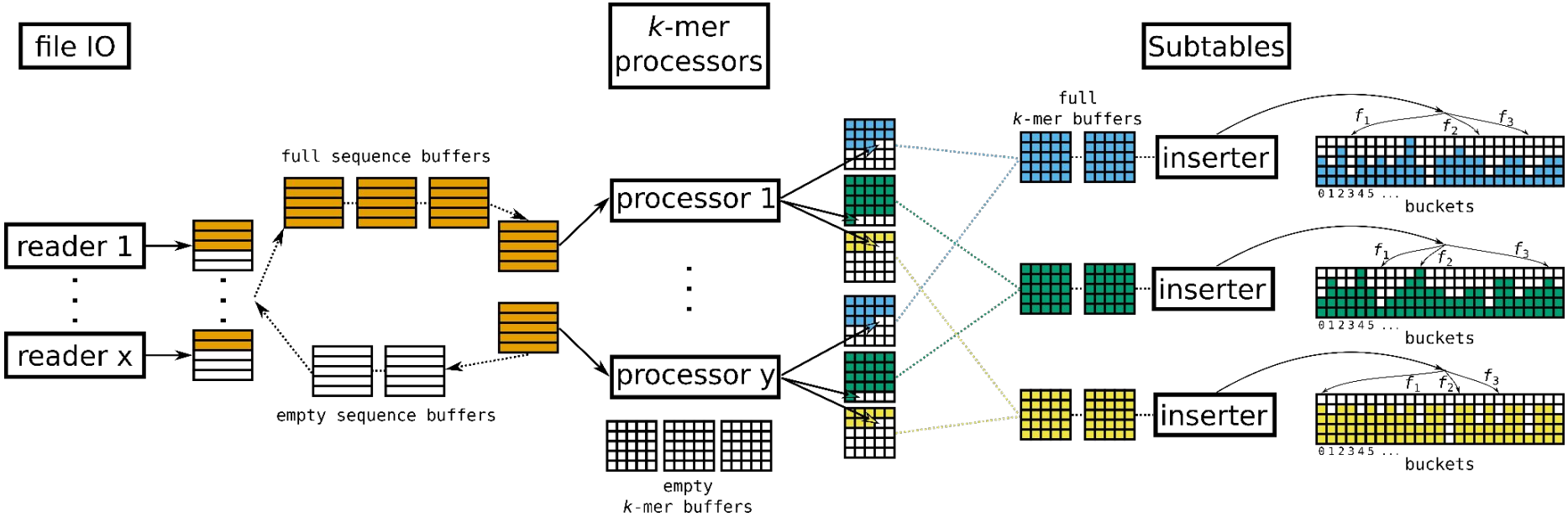
```
#_#_#
TACAGATATA
T_C_G
 A_A_A
  C_G_T
   A_A_A
    G_T_T
     A_A_A
```

Gapped *k*-mers

- Window size *w*
- Significant positions: *k* (#)
- Ignored gap positions: *w* - *k* (_)
- Mask or tuple of offsets
  - `Mask: #_#_#`
  - `Tuple: (0,2,4)`
- We use symmetric masks only.
- One error changes
  k **distributed** *k*-mers.
- This can lead to better
  error tolerance for the same *k*.

Parallelization with subtables

- Parallel file IO
- Producer: ***k*-mer Processor**
  - Extract *k*-mers of reads
  - Distribute *k*-mers to subtable
- Consumer: **inserter**
  - One inserter per subtable
  - Insert all *k*-mers in a subtable
  - Random walk



Xenograft classify count based

TTCAGAACAGGTTCTA…

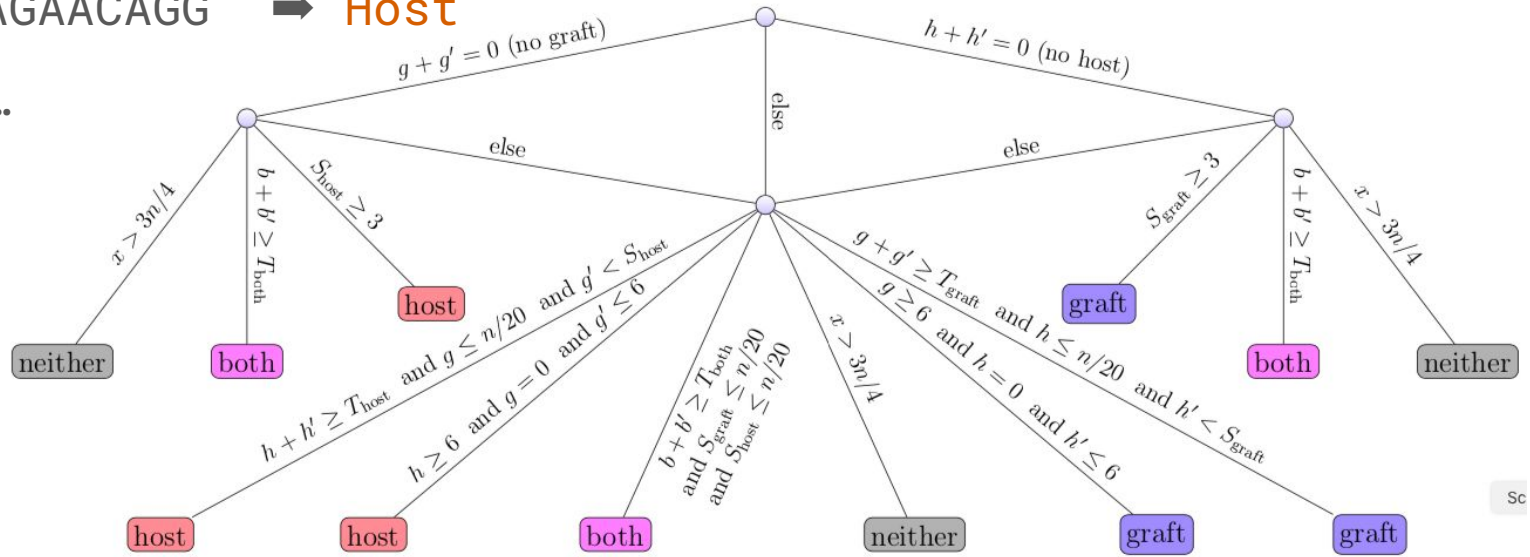
TTCAGAACA ➡ Host

TCAGAACAG ➡ Host

CAGAACAGG ➡ Host

…

host	weak host	graft	weak graft	both	neither
67	3	0	0	1	0



Disadvantage count based

TTCAGAACAGGTTCTACTACTGTCAAATGACCCCCCATACTTCCTCAAAGGCTGTGGTAAGTTTTGCACAGGTGAGGGCAGCAGAAAGGGGGTAGTTAC

TTCAGAACAGGTTCTACTACTGTCA

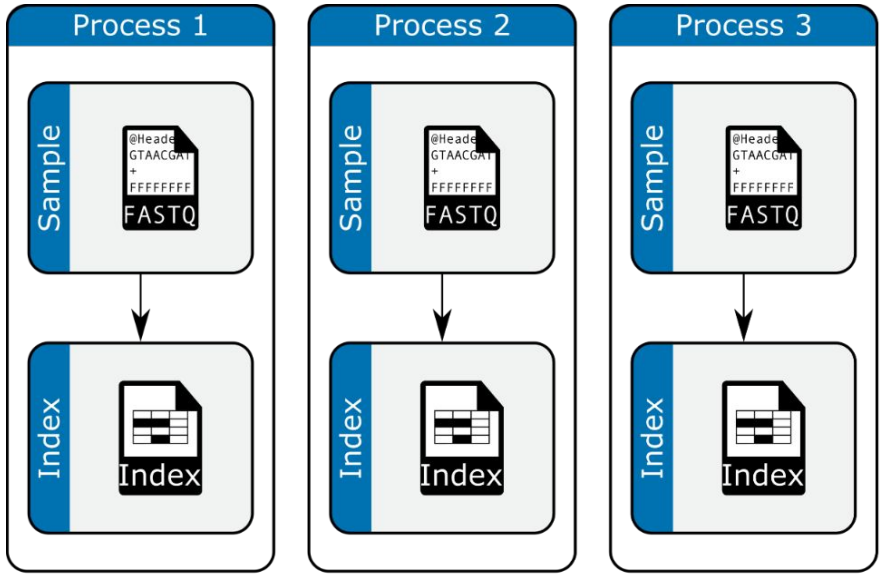
TCAGAACAGGTTCTACTACTGTCAA

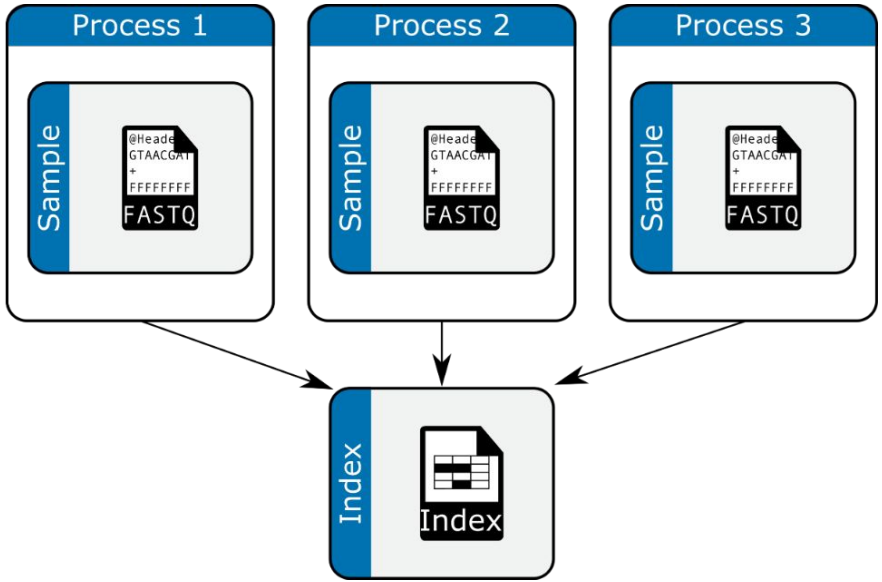
CAGAACAGGTTCTACTACTGTCAAA

TTCAGAACAGGTTCTACTACTGTCAATGACCCCCCATACTTCCTCAAAGGCTGTGGTAAGTTTTGCACAGGTGAGGGCAGCAGAAAGGGGGTAGTTAC

TTCAGAACAGGTTCTACTACTGTCA TACTTCCTCAAAGGCTGTGGTAAGT AGGGCAGCAGAAAGGGGGTAGTTAC

Shared memory





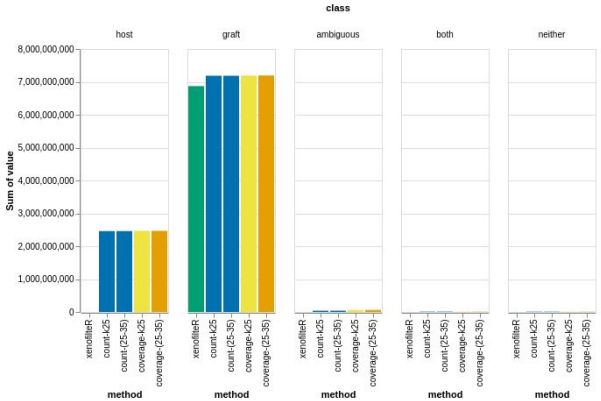
Summary

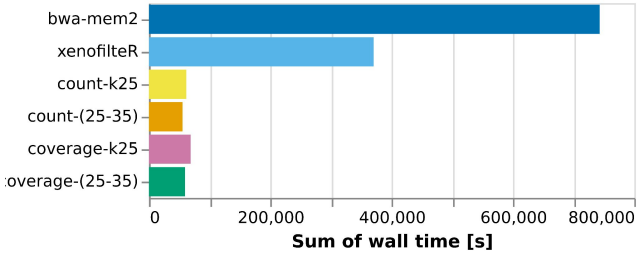
- **xengsort2**: ultrafast alignment-free xenograft sorting
- comparable classification results like alignment-based methods
- much faster (parallel index building even faster in v2)
- classification based on *k*-mer counts or covered bp
- support for parameter config files
- support for compressed input and output

Open PhD/postdoc position
Talk to me or to Sven Rahmann

python™ Package Index

BIOCONDA





https://gitlab.com/genomeinformatics/xengsort