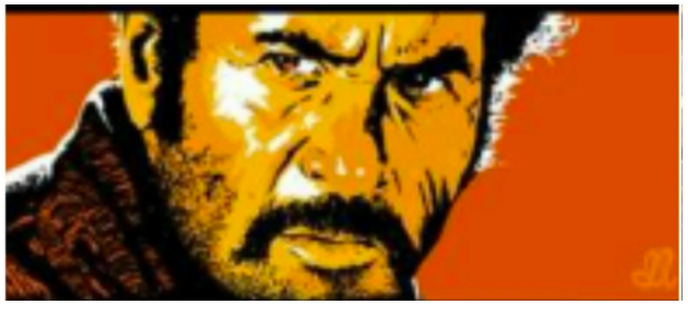
BUS format

Caltech Bioinformatics Symposium
February 14, 2019

The wild west of single-cell RNA-seq



- **The good (biology)**
  - New insights into cell types and heterogeneous tissue
  - Avoid Simpson's paradox
- **The bad (analysis)**
  - Sampling is sparse and non-uniform
  - Geometry and statistics in high dimension
- **The ugly (informatics)**
  - Complex protocols -> complicated bioinformatics
  - Technology is in flux, software is a mess

The informatics challenges

- Many technologies: 10x Genomics, Cel-seq2, Drop-seq, inDrops, SureCell, SCRB-seq, etc.
- Technologies are changing rapidly: 10x v1, v2, v3 chemistry, Cel-seq v1, v2, etc.
- Increasingly complex assays: cite-seq/REAP-seq, multiplexing, etc.
- Workflows require numerous software programs: CellRanger, STAR, Seurat, velocyto, etc.
- Numerous languages involved: C/C++, R, python, etc.
- Datasets are large: one single-cell RNA-seq dataset is about 10 times larger than a bulk RNA-seq dataset

Existing workflows

barcode error correction
- correct sequencing errors in barcodes

read alignment
- align reads to a reference genome

UMI error correction
- correct sequencing errors in UMIs

cell assignment
- decide which reads are associated with which barcodes

cell filtering
- remove barcodes that correspond to failed cells

technology choice
- determine how to extract information from reads

gene counting
- produce cell x gene matrix of read counts

Existing workflows

technology choice

↓

cell filtering

↓

read alignment

↓

barcode error correction

↓

cell assignment

↓

UMI error correction

↓

gene counting

Existing workflows

technology choice

↓

cell filtering

↓

read alignment

↓

barcode error correction

↓

cell assignment

↓

UMI error correction

↓

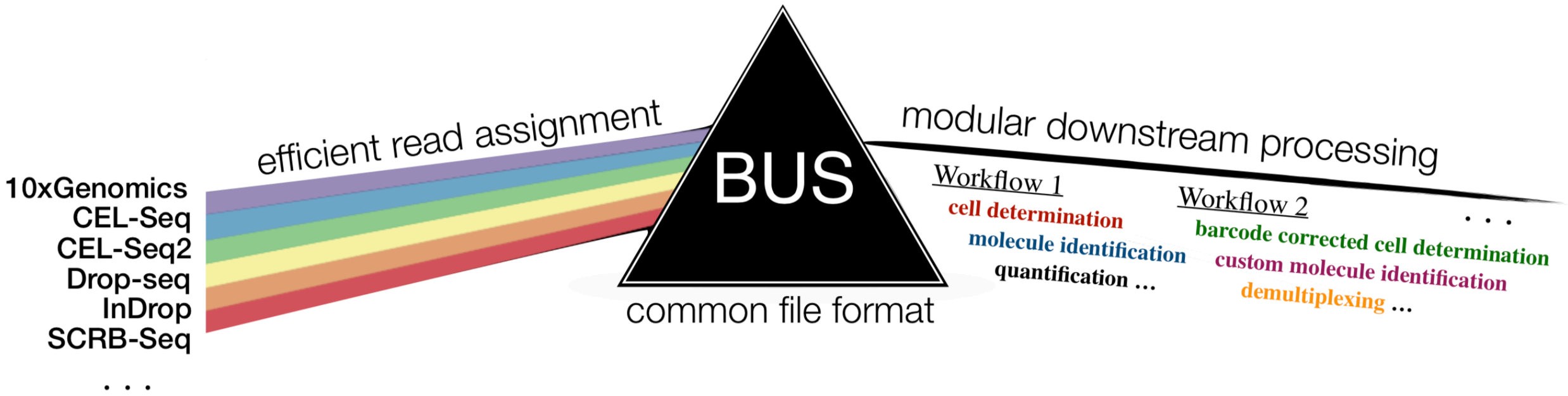
gene counting

Godot: **g**o **o**ut (and) **d**ance **o**ur **t**rash (takes forever)

- **Godot** requires a ton of memory
- **Godot** will take a day to run
- **Godot** requires a server
- Want to analyze a different kind of experiment? LOL!
- BUT….
- **Godot** is open source!!

Proposal

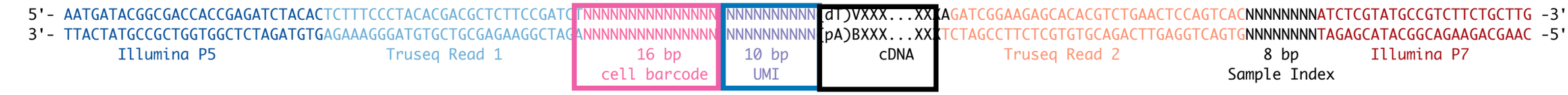
- A new format which decouples technology dependencies from algorithm choices.



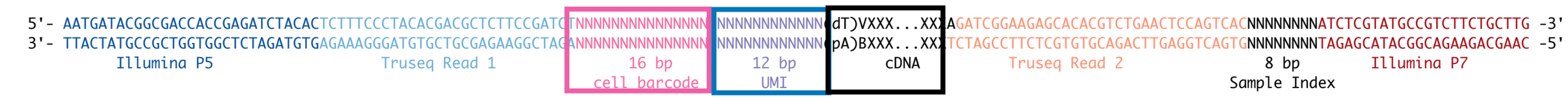
- We call this format **B**arcode, **U**MI, **S**et (BUS) format.

Common structure to data

V2 library:



V3 library:



10x v2 and v3

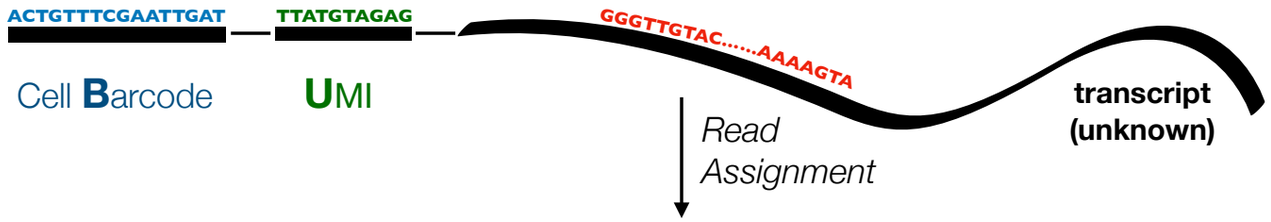
SureCell

Drop-seq

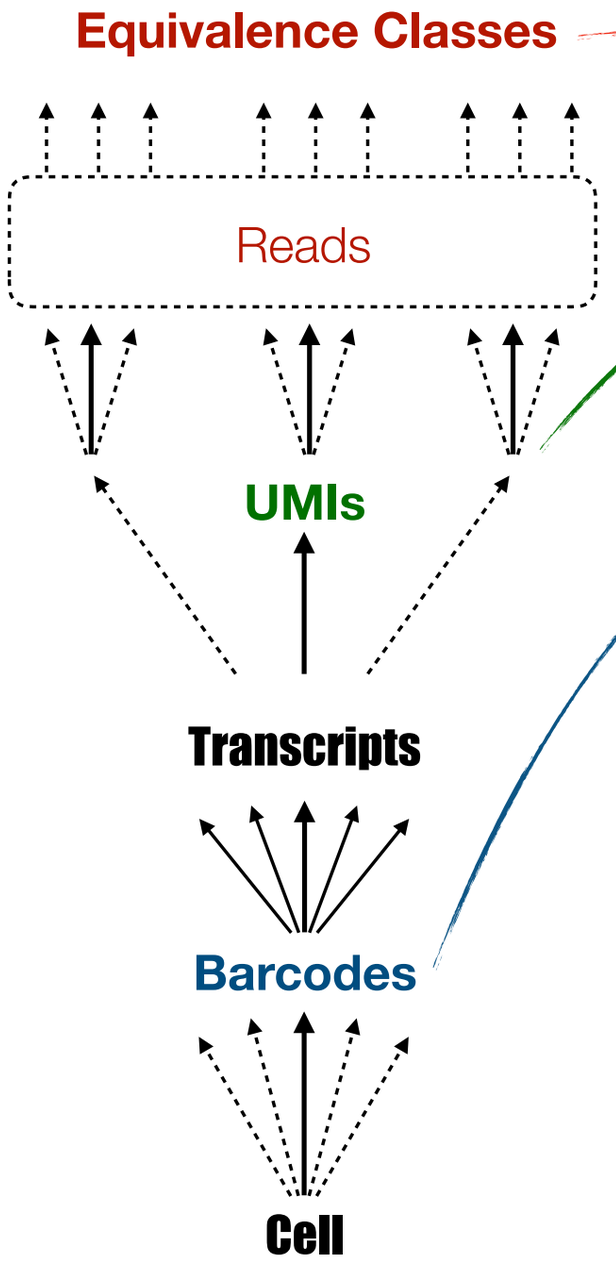
InDrops

https://teichlab.github.io/scg_lib_structs/

The Barcode, UMI, Set file format:



Set of possible transcripts of origin
(equivalence class)



BUStools

BUS file

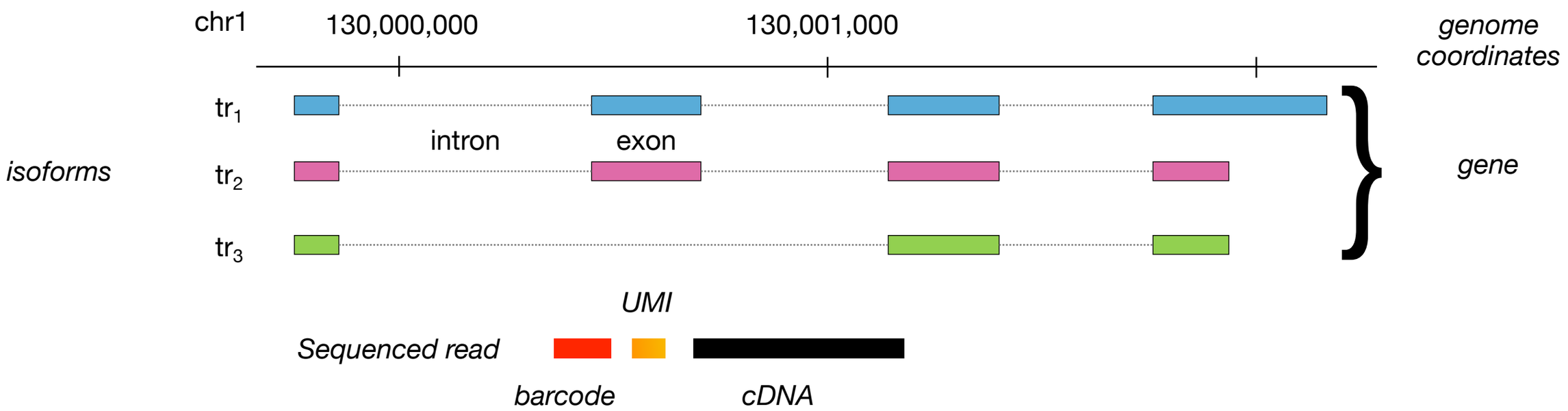
2-bit encoding

<32bp	<32bp	32bit	32bit	32bit
barcode	umi	eq. class id	count	flags
01001101001…	110111…	000001101001…	000100111001…	0000…
01001101001…	110111…	110010010010…	111011001100…	0000…
01001101001…	110111…	000001101001…	000100111001…	0000…
01001101001…	010010…	000001101001…	000100111001…	0000…
01001101001…	110111…	000001101001…	111010011111…	0000…
01001101001…	110111…	100100100010…	000100111001…	0000…
11010001001…	110111…	000001101001…	000100111001…	0000…
11010001001…	110111…	000001101001…	001110010101…	0000…
11010001001…	110111…	000001101001…	001110010101…	0000…
11010001001…	110111…	000001101001…	001110010101…	0000…

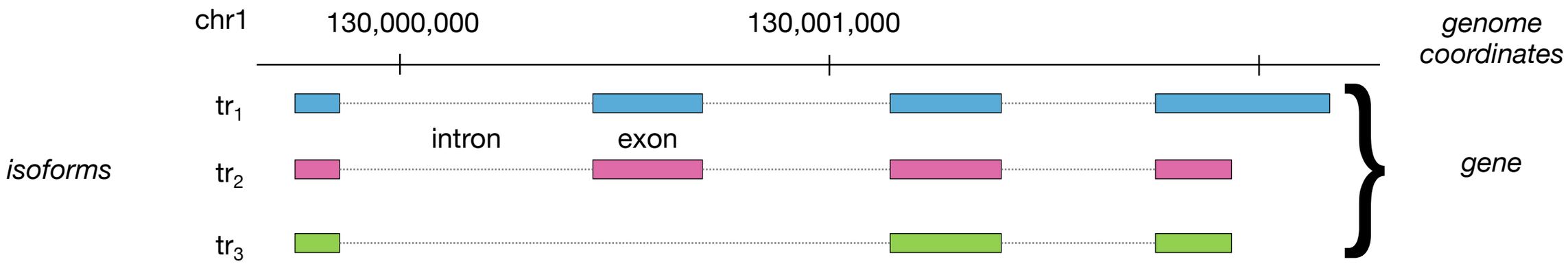
BUS centered workflow

- BUS can be generated with **kallisto** (Bray et al. 2016)
    - kallisto is fast: no sorting or alignment is required
    - kallisto streams bus records directly to disk, no memory overhead
        - Easy to process all technologies. kallisto already supports 10x v1,v2 and v3 chemistry, Drop-seq, inDrops, SureCell, etc.
- **BUStools** can be used for generic processing of BUS files
- Downstream processing **notebooks** in Python and R

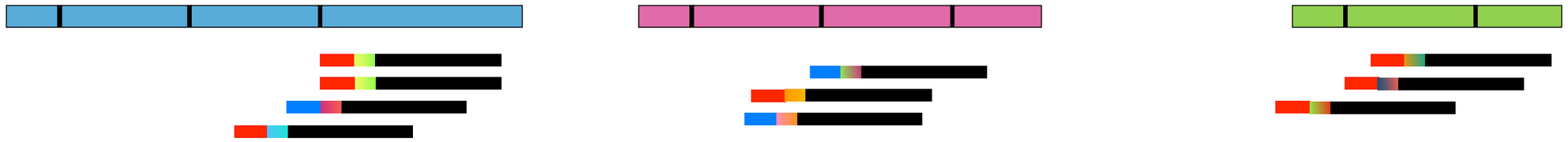
Example


chr1
130,000,000
130,001,000
genome coordinates
isoforms
tr1
tr2
tr3
intron
exon
gene
UMI
Sequenced read
barcode
cDNA

Example

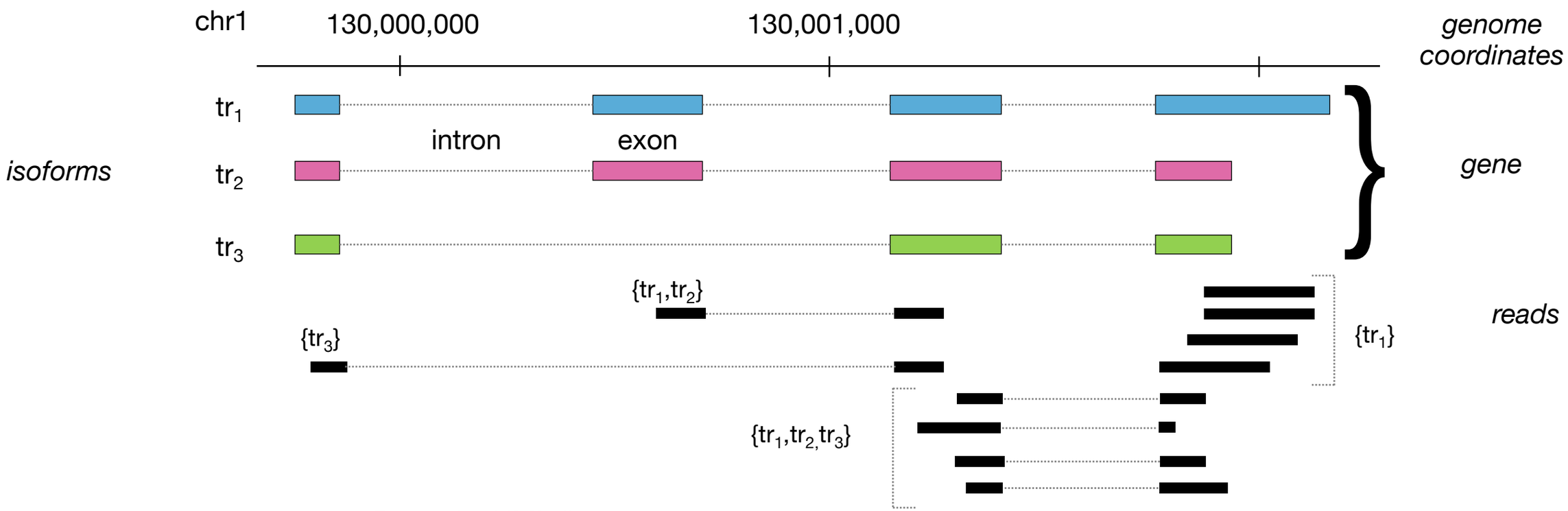


transcripts



reads

Example



Equivalence classes

$\{tr_1\}$	1
$\{tr_2\}$	2
$\{tr_3\}$	3
$\{tr_1,tr_2\}$	4
$\{tr_1,tr_2,tr_3\}$	5

Barcode, UMI, Set

	1
	1
	3
	4
	5
	5
	4
	4
	4
	4

Identical BUS records are merged

Downstream analysis in notebooks

BUS notebook review

- Download data
- Download reference transcriptome
- Build kallisto index
- Run kallisto bus
- Sort the bus file and convert to text
- Parse bus file in python
- Collate counts to make cell x gene counts matrix
- Analyze data…

BUS notebook review

- Run kallisto bus

```
kallisto bus -i index -o output R1.fastq R2.fastq
```

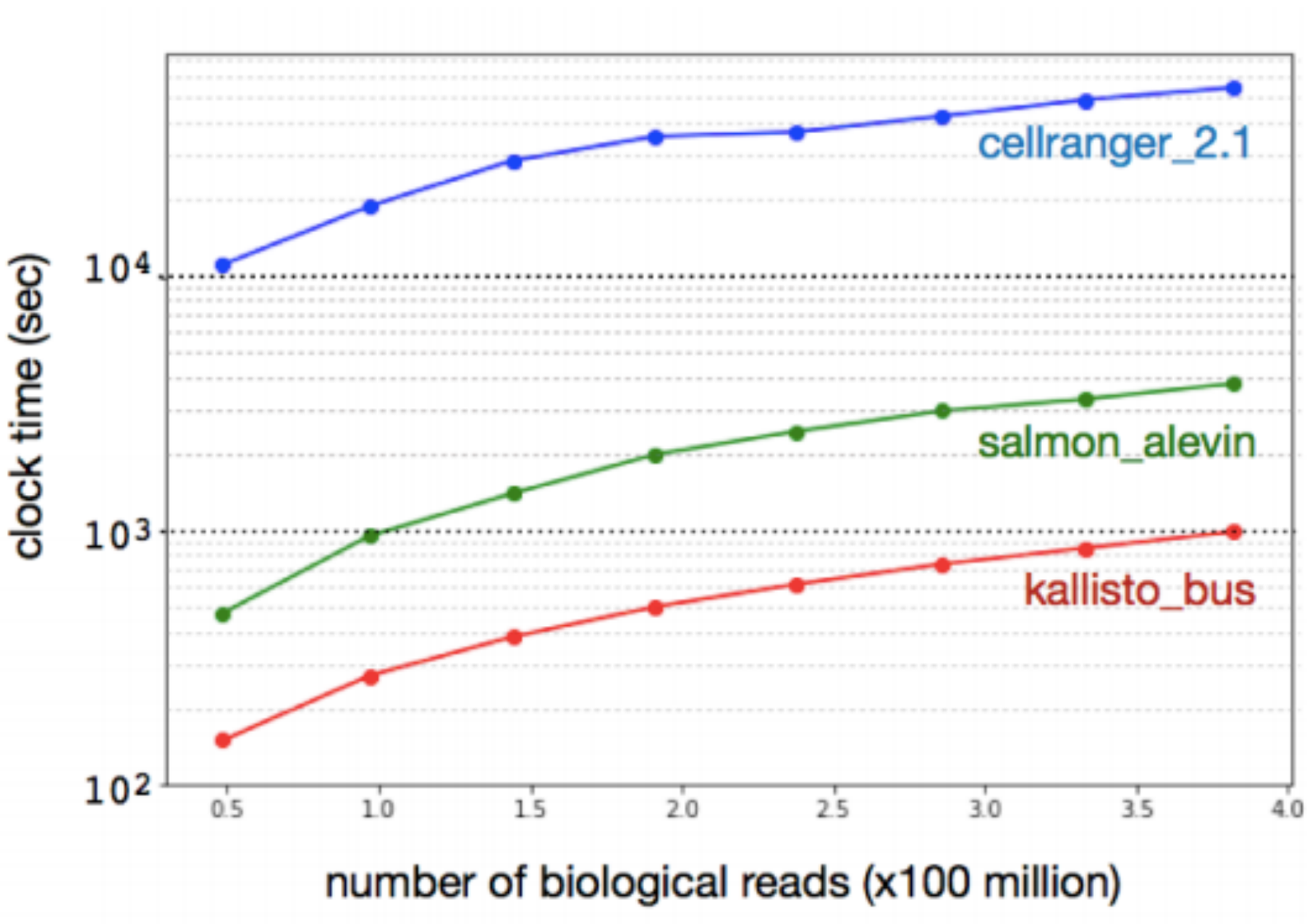
- Sort the bus file and convert to text

```
bustools sort -o output.sorted.bus output/outpus.bus
bustools text -o output.sorted.txt output.sorted.bus
```

- Parse bus file in python
- Collate counts to make cell x gene counts matrix
- Analyze data…
  - Provided in notebooks

In practice…

- Running time for 350M reads, 8 threads: 10 minutes of **kallisto**



Current and future work

- Better algorithms for barcode and UMI correction
- Standardized workflows for popular technologies and assays
- RNA velocity workflows
- Compression of BUS format
- Large-scale processing of publicly available single-cell RNA-seq