AXE

The Rapid and Accurate Demultiplexer

Kevin Murray
@kdmurray91
kevin@kdmurray.id.au

Borevitz Lab, ANU

December 5, 2014

About Me

- Bioinformatician @ Borevitz Lab
  - TraitCapture Project Developer
  - Genomics (Low level Sequence Analysis)
  - Phenomics (Image analysis)
  - Sample tracking, data standards, HPC
- Starting PhD in Bioinformatics next year (Borevitz/Forêt)

In big data, binary goes up to 2. #eyenary

The High Throughput Problem

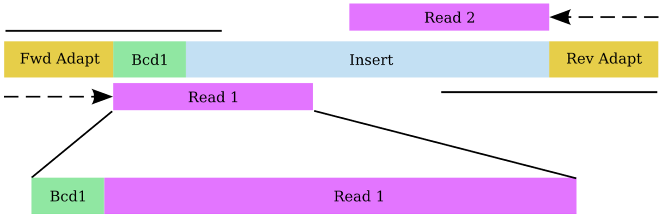
- Sequencing == fire hose of data
- Need to put > 1 sample / lane
- Give BRF only one tube
- ∴ we need multiplexing
- ∴ de-multiplexing

The High Throughput Problem

- Sequencing == fire hose of data
- Need to put > 1 sample / lane
- Give BRF only one tube
- ∴ we need multiplexing
- ∴ de-multiplexing

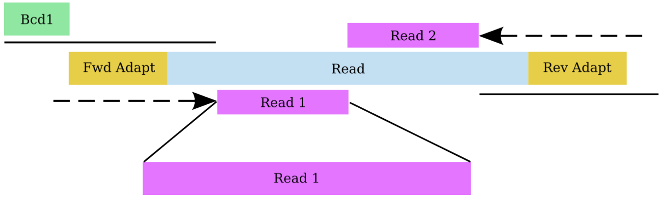
DNA/Molecular Barcoding

- DNA fragment contains per-sample unique seq.
- Sequenced, and "attached" to a read
- Must be balanced
- Hamming distance >2



So you need a demulitplexer?

- Some studies use kit protocols
- Most kits use index reads
- Illumina pipeline automatically demuxes these

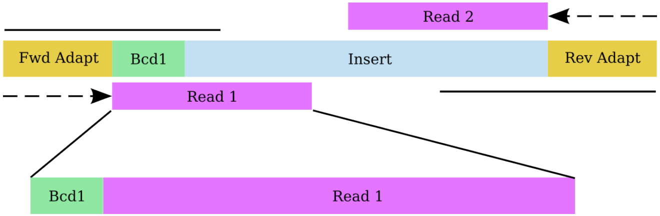


plant energy biology
ARC CENTRE OF EXCELLENCE

So you need a demulitplexer?

- Some studies use kit protocols
- Most kits use index reads
- Illumina pipeline automatically demuxes these

- Many advanced/homebrew protocols use "in-read"
- Must be demultiplexed by user



De-multiplex with `AXE`

- Barcoding scheme requires advanced de-multiplexing
- Trie-based lookup algorithm
- Fast (PE lane in 5-10 mins)
- Implemented in C
- CLI and `libaxe.so` + `axe.h`
- GNU GPL v3+

ACGTCATCGTAGCTGATCATGCTAGCTGACTGATCGATGCTAGCACAC
CACCGATCGATCGATGCTAGCTAGTCGATGCTAGTCGATGCTAGTCCA

Hamming distance

- Distance between two strings of equal length

  ```
  ACTGTG
  ..x.x.  = 2
  ACAGCG
  ```

Hamming distance

- Distance between two strings of equal length

  ```
  ACTGTG
  ..x.x.  = 2
  ACAGCG
  ```

- Is a poor measure of seq. divergence:

  ```
  ACTGTG
  xxxxxx = 6
  CTGTGA
  ```

Hamming distance

- Distance between two strings of equal length

  ```
  ACTGTG
  ..x.x.  = 2
  ACAGCG
  ```

- Is a poor measure of seq. divergence:

  ```
  ACTGTG
  x.....  = 1
  -CTGTG
  ```

Hamming distance

- Distance between two strings of equal length

  ```
  ACTGTG
  ..x.x. = 2
  ACAGCG
  ```

- Is a poor measure of seq. divergence:

  ```
  ACTGTG
  x..... = 1
  -CTGTG
  ```

- `sum([0 if s1[i] == s2[i] else 1 for i in range(l)])`

Hamming distance

- Distance between two strings of equal length

  ```
  ACTGTG
  ..x.x. = 2
  ACAGCG
  ```

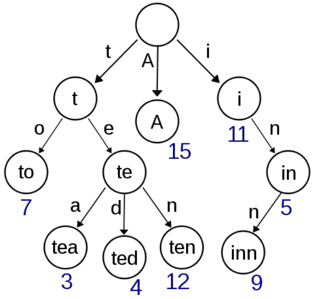
- Is a poor measure of seq. divergence:

  ```
  ACTGTG
  x..... = 1
  -CTGTG
  ```

- `sum([0 if s1[i] == s2[i] else 1 for i in range(l)])`
- Conservative, Good Enough™

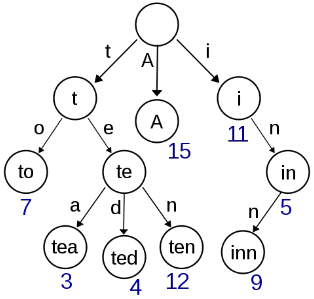
A Forest of Tries?

- Trie is k-ary tree for k letter alphabet



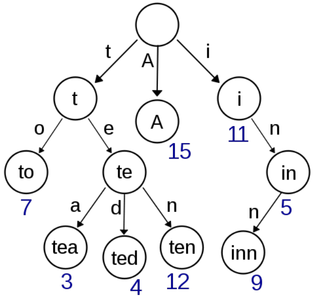
A Forest of Tries?

- Trie is k-ary tree for k letter alphabet
- Lookups are $\mathcal{O}(l)$ for keys of l



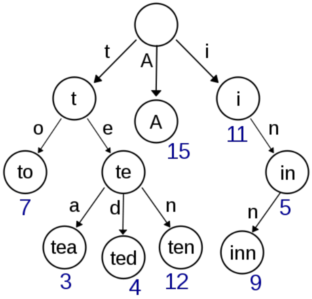
A Forest of Tries?

- Trie is k-ary tree for k letter alphabet
- Lookups are $\mathcal{O}(l)$ for keys of l
- i.e., $\mathcal{O}(1)$ WRT number of items, *a la* hash-tables



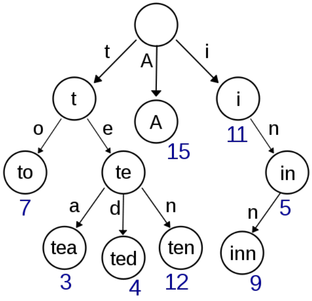
A Forest of Tries?

- Trie is k-ary tree for k letter alphabet
- Lookups are $\mathcal{O}(l)$ for keys of l
- i.e., $\mathcal{O}(1)$ WRT number of items, *a la* hash-tables
- Memory efficient (prefixes collapsed)



A Forest of Tries?

- Trie is k-ary tree for k letter alphabet
- Lookups are $\mathcal{O}(l)$ for keys of l
- i.e., $\mathcal{O}(1)$ WRT number of items, *a la* hash-tables
- Memory efficient (prefixes collapsed)
- Prefix lookups very fast (re**trie**val)



Hamming Mismatch Trie

- Pre-calculate all strings with Hamming dist *d* from target
- Make trie, do lookups
- Essentially a FSM

```
AAAA
CAAA
GAAA
TAAA
ACAA
AGAA
ATAA
AACA
AAGA
AATA
AAAC
AAAG
AAAT
```

Hamming Mismatch Trie

- Pre-calculate all strings with Hamming dist *d* from target
- Make trie, do lookups
- Essentially a FSM

```
 /-A-A-A-A
 | | | \-C
 | | |  -G
 | | |  -T
 | | \-C-A
 | |  -G-A
-| |  -T-A
 | \-C-A-A
 |  -G-A-A
 |  -T-A-A
 |-C-A-A-A
 |-G-A-A-A
 \-T-A-A-A
```

The Algorithm

- Take read
- Walk trie w/ start of read
- Mark full-length matches
- Take longest match

```
AAAA
AAAC
AAAG
AAAT
AACA
AAGA
AATA
ACAA
AGAA
ATAA
CAAA
GAAA
TAAA

AAAACACACGTGG
```

The Algorithm

- Take read
- Walk trie w/ start of read
- Mark full-length matches
- Take longest match

```
AAAA
AAAC
AAAG
AAAT
AACA
AAGA
AATA
ACAA
AGAA
ATAA
CAAA
GAAA
TAAA

AAAACACACGTGG
```

The Algorithm

- Take read
- Walk trie w/ start of read
- Mark full-length matches
- Take longest match

```
AAAA
AAAC
AAAG
AAAT
AACA
AAGA
AATA
ACAA
AGAA
ATAA
CAAA
GAAA
TAAA

AAAACACACGTGG
```

plant energy biology
ARC CENTRE OF EXCELLENCE

The Algorithm

- Take read
- Walk trie w/ start of read
- Mark full-length matches
- Take longest match

```
AAAA
AAAC
AAAG
AAAT
AACA
AAGA
AATA
ACAA
AGAA
ATAA
CAAA
GAAA
TAAA

AAAACACACGTGG
```

The Algorithm

- Take read
- Walk trie w/ start of read
- Mark full-length matches
- Take longest match

```
AAAA
AAAC
AAAG
AAAT
AACA
AAGA
AATA
ACAA
AGAA
ATAA
CAAA
GAAA
TAAA

AAAACACACGTGG
```

The Algorithm

- Take read
- Walk trie w/ start of read
- Mark full-length matches
- Take longest match

```
AAAA
AAAC
AAAG
AAAT
AACA
AAGA
AATA
ACAA
AGAA
ATAA
CAAA
GAAA
TAAA

ACAACACACGTGG
```

The Algorithm

- Take read
- Walk trie w/ start of read
- Mark full-length matches
- Take longest match

```
AAAA
AAAC
AAAG
AAAT
AACA
AAGA
AATA
ACAA
AGAA
ATAA
CAAA
GAAA
TAAA

ACAACACACGTGG
```

The Algorithm

- Take read
- Walk trie w/ start of read
- Mark full-length matches
- Take longest match

```
AAAA
AAAC
AAAG
AAAT
AACA
AAGA
AATA
ACAA
AGAA
ATAA
CAAA
GAAA
TAAA

ACAACACACGTGG
```

The Algorithm

- Take read
- Walk trie w/ start of read
- Mark full-length matches
- Take longest match

```
AAAA
AAAC
AAAG
AAAT
AACA
AAGA
AATA
ACAA
AGAA
ATAA
CAAA
GAAA
TAAA

ACAACACACGTGG
```

The Algorithm

- Take read
- Walk trie w/ start of read
- Mark full-length matches
- Take longest match

```
AAAA
AAAC
AAAG
AAAT
AACA
AAGA
AATA
ACAA
AGAA
ATAA
CAAA
GAAA
TAAA

CCCAACATACAGC
```

The Algorithm

- Take read
- Walk trie w/ start of read
- Mark full-length matches
- Take longest match

```
AAAA
AAAC
AAAG
AAAT
AACA
AAGA
AATA
ACAA
AGAA
ATAA
CAAA
GAAA
TAAA

CCCAACATACAGC
```

plant energy biology
ARC CENTRE OF EXCELLENCE

The Algorithm

- Take read
- Walk trie w/ start of read
- Mark full-length matches
- Take longest match

```
AAAA
AAAC
AAAG
AAAT
AACA
AAGA
AATA
ACAA
AGAA
ATAA
CAAA
GAAA
TAAA

CCCAACATACAGC
```

The Algorithm

- Take read
- Walk trie w/ start of read
- Mark full-length matches
- Take longest match

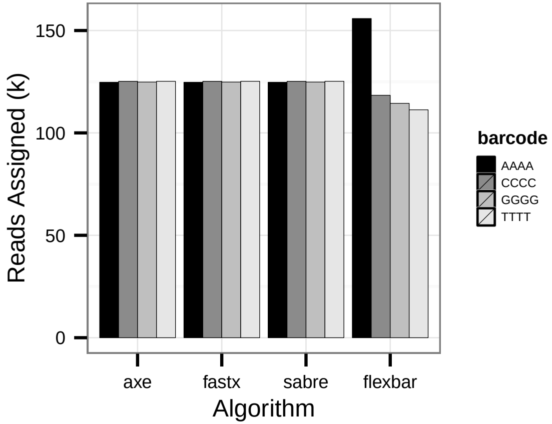
```
AAAA
AAAC
AAAG
AAAT
AACA
AAGA
AATA
ACAA
AGAA
ATAA
CAAA
GAAA
TAAA

CCCAACATACAGC
```

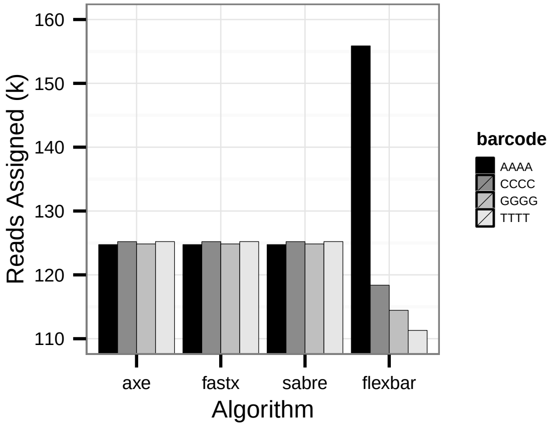
Benchmarking

- Contrived example:
  - `wgsim` 500k reads from *A. thaliana* ChrC
  - Add `AAAA`, `CCCC`, `GGGG`, `TTTT` to 5′
  - Insert 1 mismatch
- "GBS-like" example:
  - Different length barcodes
  - RE site in reads
  - `AAAA`, `CCCC`, `GGGG`, `TTTT`, `AAAAAAAA`, `CCCCCCCC`, `GGGGGGGG`, `TTTTTTTT`
  - Mismatches in RE and barcode
- Measure reads assigned, USER + SYS time, reads/sec
- Cross-compare to `sabre`, `fastx`, `flexbar`
- I'll show you how it's done if we have time

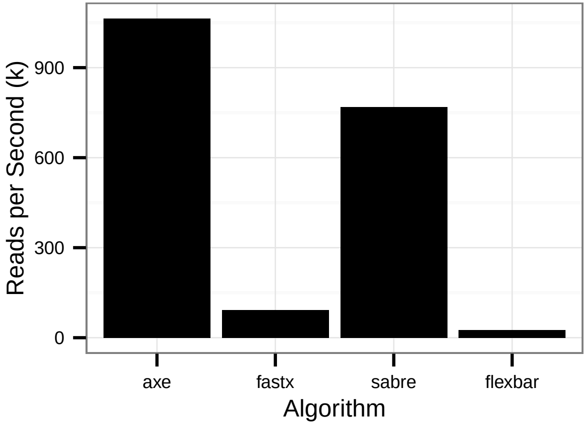
Benchmarking Results



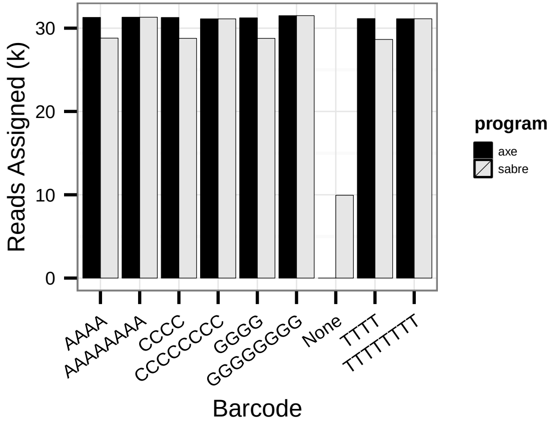
Benchmarking Results



Benchmarking Results



Benchmarking Results



Testing Axe

- If we have time, let's look at the IPython Notebook, it contains a lot more benchmarking.
- Or, I'll show you how to use it on the CLI

Conclusions

- Made a fast, accurate demultiplexer
  - More barcoding modes supported
  - Faster than all others I've seen
  - Just as or more accurate as slower algorithms
- Future work
  - Adaptor removal: load adaptors in, do it backwards?
  - Levenshtein distance? NDFSM?
  - Integrate w/ sample tracking (w/ Aaron, Cam @ GDU)
  - Code audit & tests

Thanks

- Justin Borevitz & Norman Warthmann
- Sylvain Forêt
- Aaron Chuah
- Cam Jack
- Jared Streich & Collin Ahrens (β-testers)

Thanks

- Justin Borevitz & Norman Warthmann
- Sylvain Forêt
- Aaron Chuah
- Cam Jack
- Jared Streich & Collin Ahrens (β-testers)