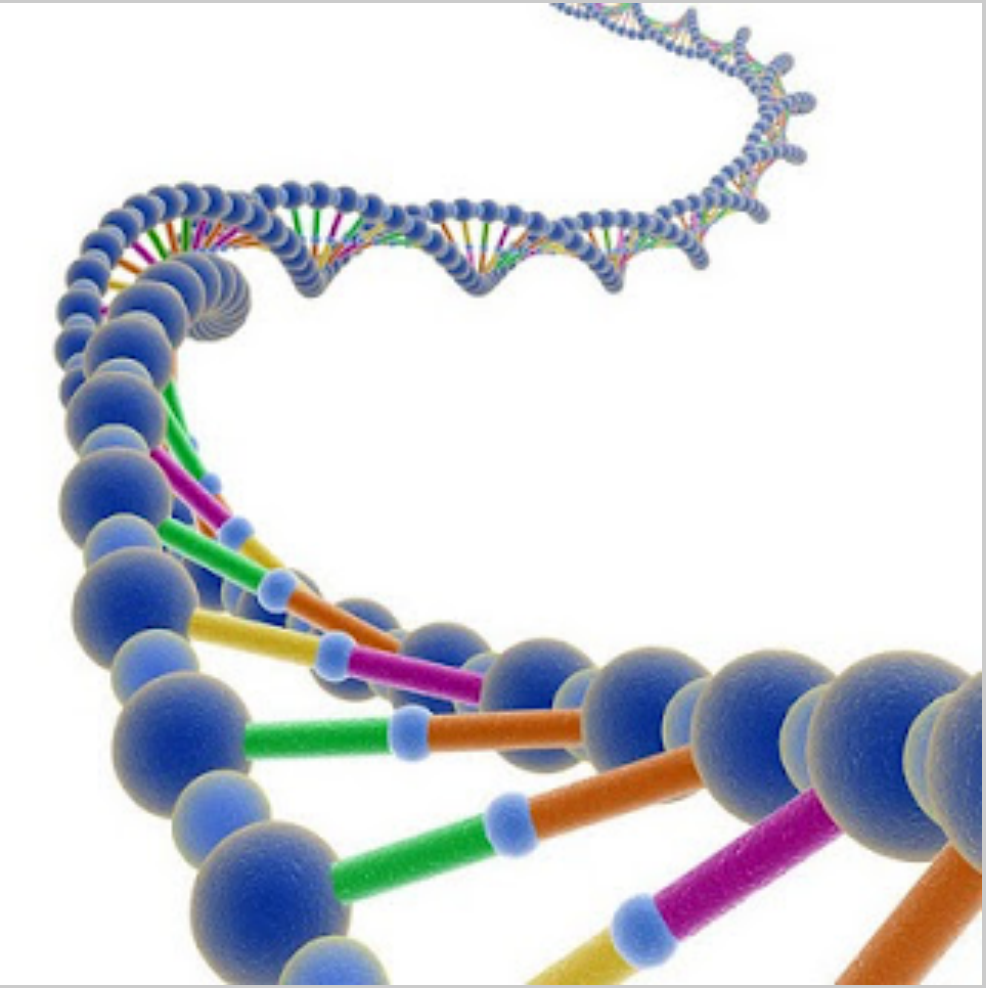
Biocaml
The OCaml Bioinformatics Library

Ashish Agarwal, Sebastien Mondet, Philippe Veber,
Christophe Troestler, Francois Berenger

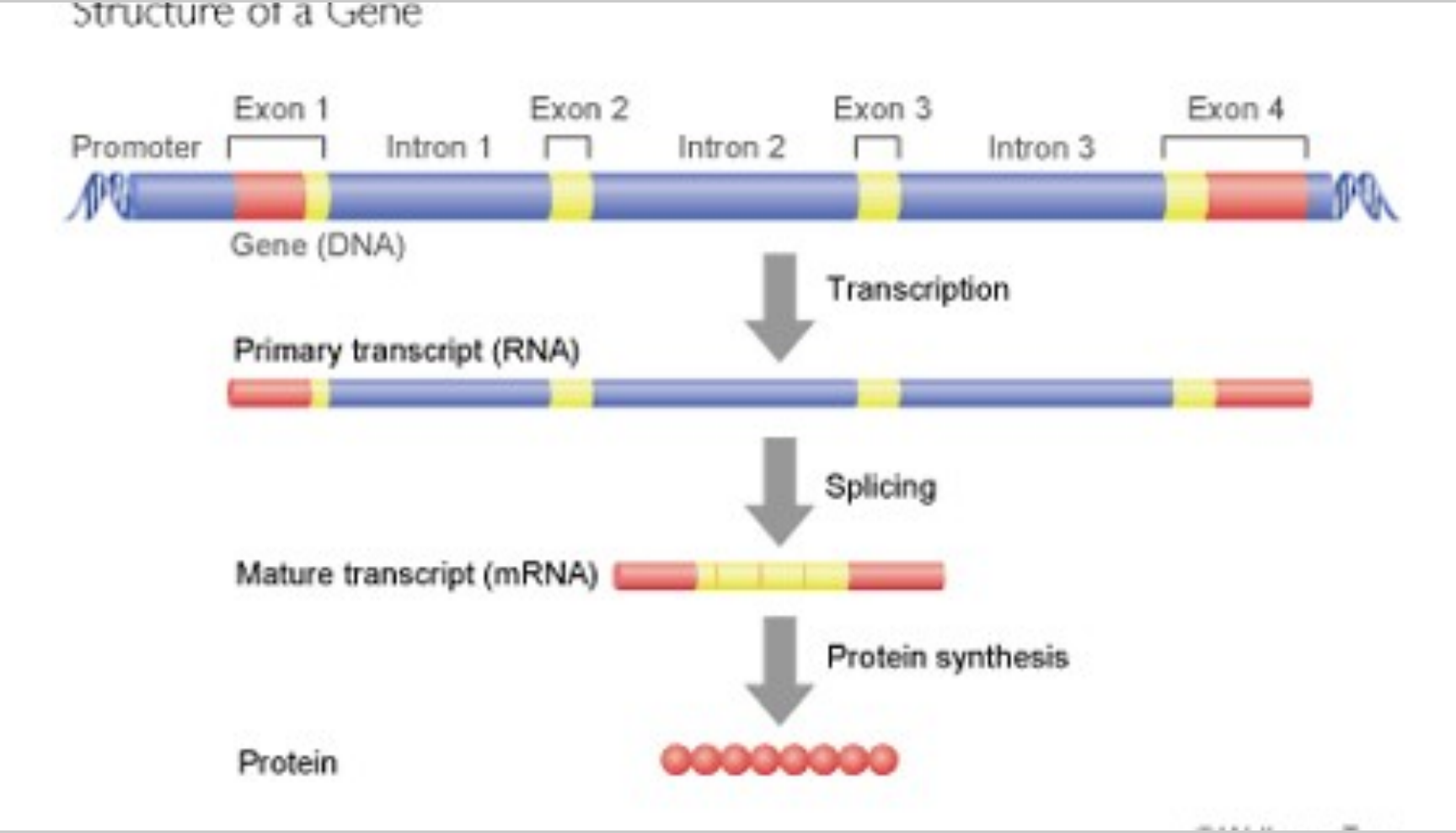
OCaml Users and Developers Meeting
Copenhagen, Denmark
Sep 14, 2012

DNA – The Code of Life



DNA Unravelled

DNA Sub-structrues



Biocaml: Main Features

- File Formats
- Data structures
- Public data repositories
- … Algorithms

File Formats

- Currently supported file formats
  - bar, bed, bpmap, cel, fasta, fastq, gff, sam, bam, sbml, sgr, ucsc tracks, wig, tsv/csv with column names

FASTA

```
>I
gcctaagcctaagcctaagcctaagcctaagcctaagcctaagcctaagc
ctaagcctaagcctaagcctaagcctaagcctaagcctaagcctaagcct
aagcctaagcctaagcctaagcctaagcctaagcctaagcctaagcctaa
gcctaagcctaagcctaagcctaagcctaagcctaagcctaagcctaagc
ctaagcctaagcctaagcctaagcctaagcctaagcctaagcctaagcct
aagcctaagcctaagcctaagcctaagcctaagcctaagcctaagcctaa
gcctaagcctaagcctaagcctaagcctaagcctaagcctaagcctaagc
ctaagcctaagcctaagcctaagcctaagcctaagcctaagcctaagcct
aagcctaagcctaagcctaagcctaagcctaaaaaattgagataagaaaa
cattttactttttcaaaattgttttcatgctaaattcaaaacgtttttt
tttagtgaagcttctagatatttggcgggtacctctaattttgcctgcct
gccaacctatatgctcctgtgtttaggcctaatactaagcctaagcctaa
gcctaatactaagcctaagcctaagactaagcctaatactaagcctaagc
ctaagactaagcctaagactaagcctaagactaagcctaatactaagcct
aagcctaagactaagcctaagcctaatactaagcctaagcctaagactaa
```

WIG

```
track type=wiggle_0 description="L2-1 normalized signal" name="L2-1 norm"
I       454     479     17.3333333333
I       494     519     -40.0
I       521     546     35.6666666667
I       545     570     41.3333333333
I       575     600     745.666666667
I       690     715     2299.33333333
I       807     832     1431.66666667
I       833     858     21.3333333333
I       859     884     52.3333333333
I       881     906     22.6666666667
I       909     934     41.3333333333
I       931     956     -7.33333333333
I       960     985     -339.0
I       981     1006    32.3333333333
I       1009    1034    -5.66666666667
I       1035    1060    -2.66666666667
I       1066    1091    -79.0
I       1094    1119    13.3333333333
I       1118    1143    2.66666666667
I       1172    1197    20.6666666667
I       1221    1246    54.6666666667
```

GFF

```
I       Coding_transcript       Transcript      11495   16793   .       +       .       \
 Transcript "Y74C9A.2.3";WormPep "WP:CE24660";Note "nlp-40";Prediction_status "Confirme\
d";Gene "WBGene00022276";CDS "Y74C9A.2"
I       Coding_transcript       Transcript      11499   16790   .       +       .       \
 Transcript "Y74C9A.2.4";WormPep "WP:CE24660";Note "nlp-40";Prediction_status "Confirme\
d";Gene "WBGene00022276";CDS "Y74C9A.2"
I       Coding_transcript       Transcript      11499   16828   .       +       .       \
 Transcript "Y74C9A.2.1";WormPep "WP:CE24660";Note "nlp-40";Prediction_status "Confirme\
d";Gene "WBGene00022276";CDS "Y74C9A.2"
I       Coding_transcript       Transcript      11505   16790   .       +       .       \
 Transcript "Y74C9A.2.5";WormPep "WP:CE24660";Note "nlp-40";Prediction_status "Confirme\
d";Gene "WBGene00022276";CDS "Y74C9A.2"
I       Coding_transcript       exon    11495   11561   .       +       .       Transcr\
ipt "Y74C9A.2.3"
I       Coding_transcript       exon    11499   11557   .       +       .       Transcr\
ipt "Y74C9A.2.4"
I       Coding_transcript       exon    11499   11561   .       +       .       Transcr\
ipt "Y74C9A.2.1"
```

File Formats: General Features

- Streaming – for big data
- Partial parsing – for speed
- Non-blocking
- Error handling
  - explicit in return types
  - exceptionful
- Comprehensive documentation
- g(un)zip-able

Fasta: Types

- type `a item = {
  header : string;
  sequence : `a }

- type `a raw_item = [
  | `comment of string
  | `header of string
  | `partial_sequence of `a]

Speed ups up to 35%.

Polymorphic Variants for Errors

- type **string_to_raw_item** = [
  | `empty_line of Pos.t
  | `incomplete_input of Pos.t * string list * string option
  | `malformed_partial_sequence of Pos.t * string]
- type **raw_item_to_item** = [
  | `unnamed_char_seq of char_seq
  | `unnamed_int_seq of int_seq]
- type **t** = [
  | string_to_raw_item
  | raw_item_to_item]

Precise yet easy-to-provide error information.

Error Handling

- Strongly typed interface:

```
in_channel_to_char_seq_item_stream :
  in_channel ->
  (char_seq, Error.t) Result.t Stream.t
```

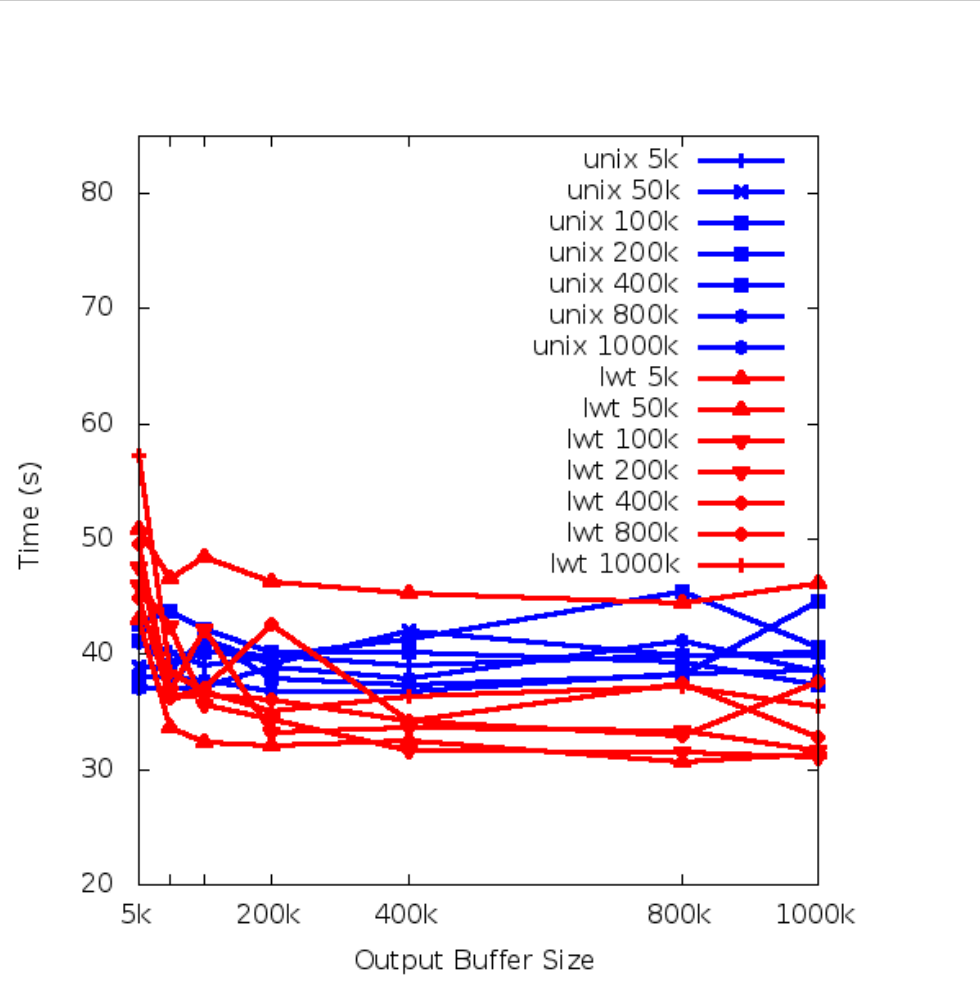
- Exceptionful interface for scripting:

```
in_channel_to_char_seq_item_stream :
  in_channel ->
  char_seq Stream.t
```

Non-blocking IO

- Lwt or Async? And standard IO
- Our solution:
- Buffered Transforms: (`a, `b) t
- val feed : ('a, 'b) t -> `a unit
- val next : ('a, 'b) t ->
  [`end_of_stream | `not_ready | `output of 'b]

Affect of Buffer Size



Legos

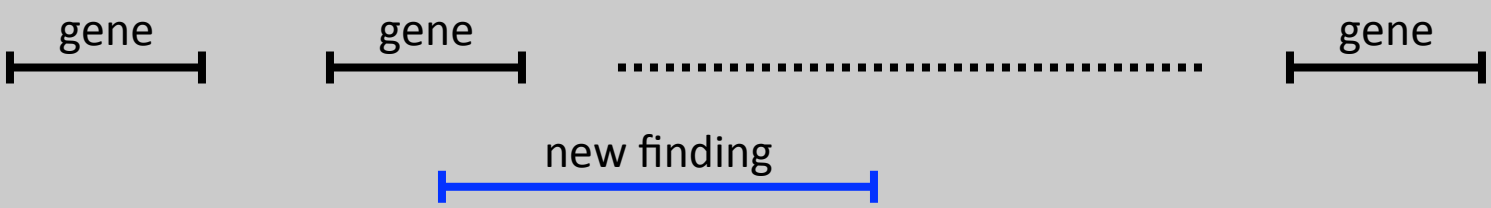
- let (|-) = Transform.compose
- let parser =

```
    Zip.unzip ~format:`gzip
  |- Fasta.string_to_char_seq_raw_item
  |- Fasta.char_seq_raw_item_to_item
```

Data Structures

- Data structures
  - integer interval trees
  - sparse integer sets
  - maps from integer intervals to `a
  - efficient polymorphic histograms

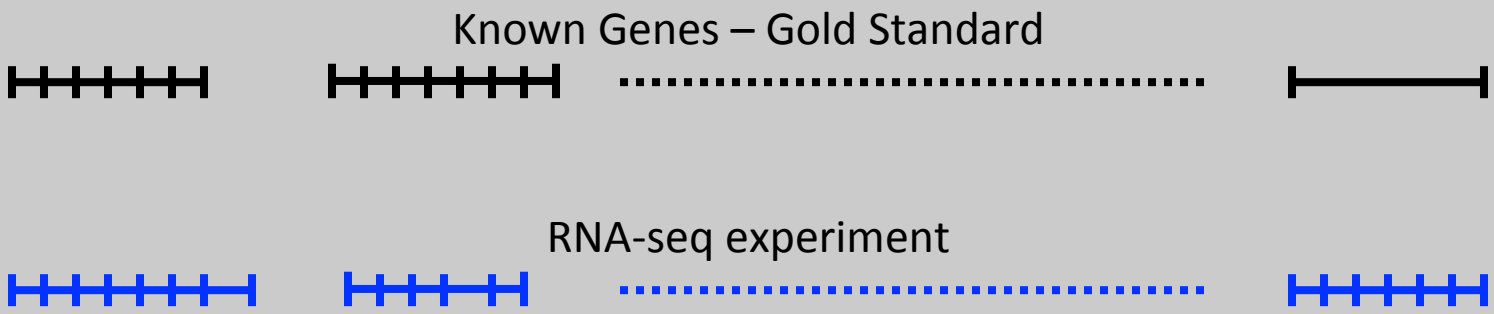
Overlap Query



Read Counting

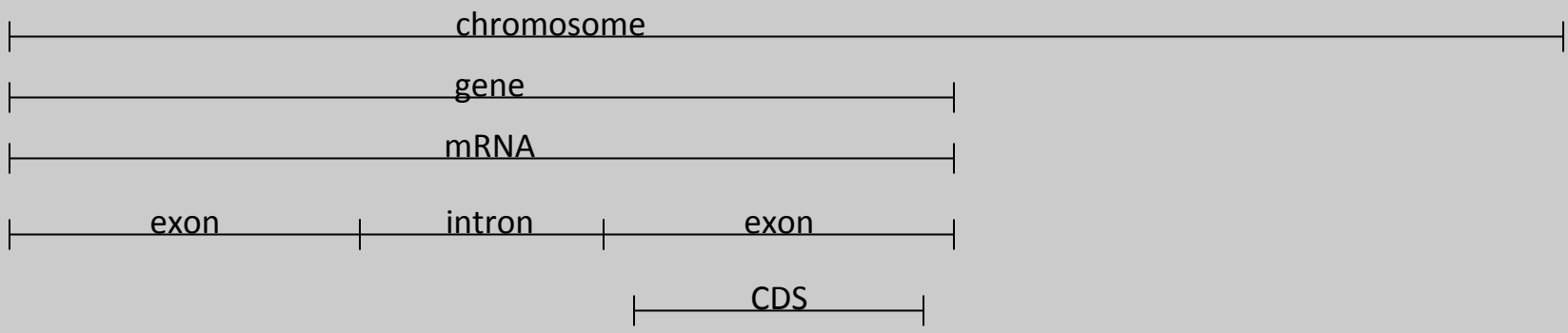
- Given aligned reads, compute read count at each genomic position

ROC Curve Statistics



- false positives = bp's in new experiment minus those in annotation
- true positives = bp's in new experiment and in annotation
- …

- Annotations are hierarchical

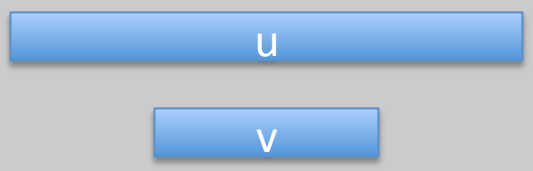


Two Partial Orders on Integer Intervals

- Positional
  - intervals are to the *left* or *right* of each other
  - Example 1

  - Example 2

- Containment
  - intervals *contain* or are *contained by* each other
  - Example



Sparse Integer Sets (DIET Sets)

- Desired set of integers:
  {3, 4, 5, 6, 7, 8, 9, 10, 22, 23, 24, 25, 26}
- Internal representation
  [(3,10), (22, 26)]

- Example: intersect
  - set1 = [(3,10), (22, 26)]
  - set2 = [(8,12), (30, 42)]
  - Result: [(8,10)]

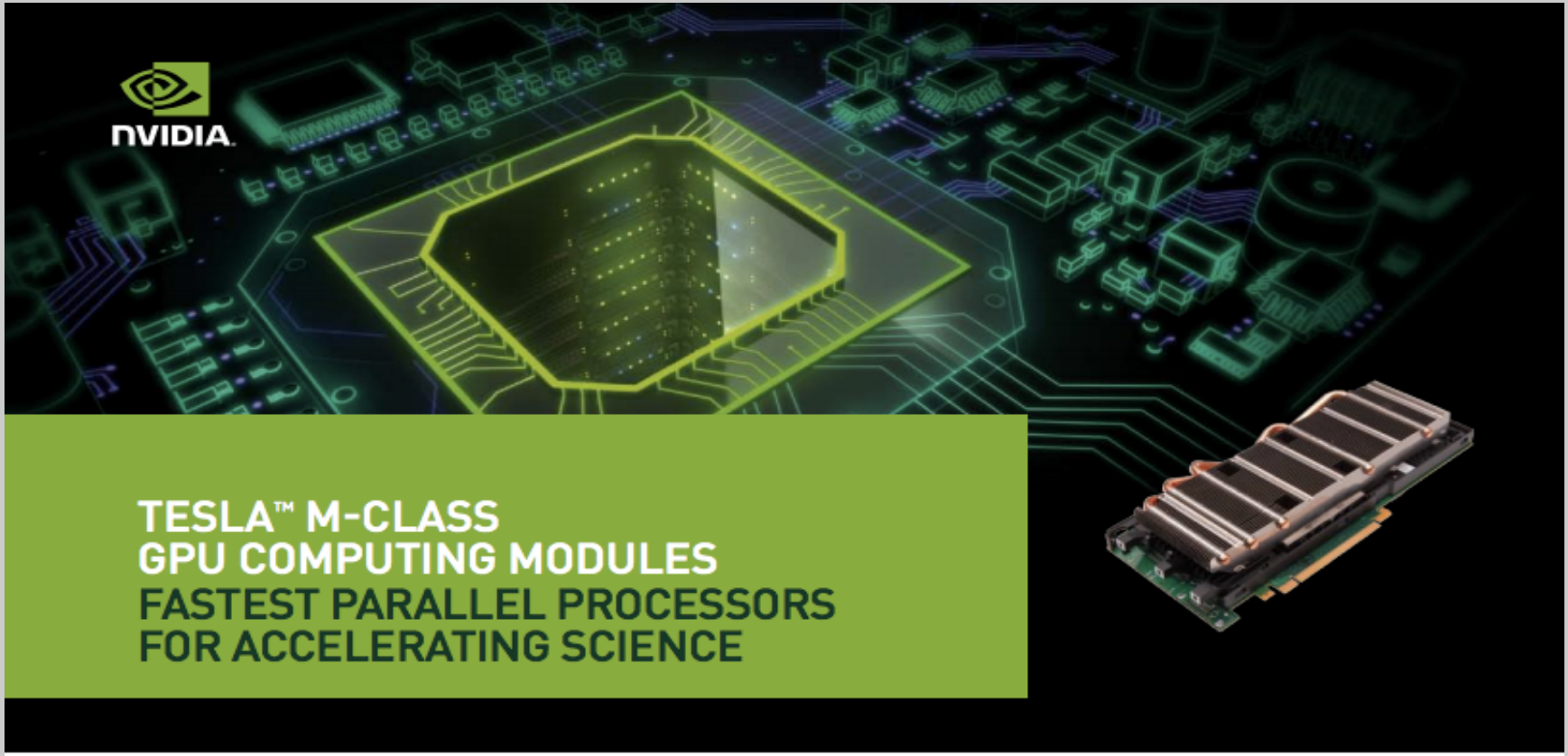
Read Counting

- If input reads are positionally sorted:
  - low memory solution possible
  - print count for position i when lower bound of current interval > i
- Else:
  - need an interval tree with nodes carrying counts
  - insert requires merging/splitting nodes

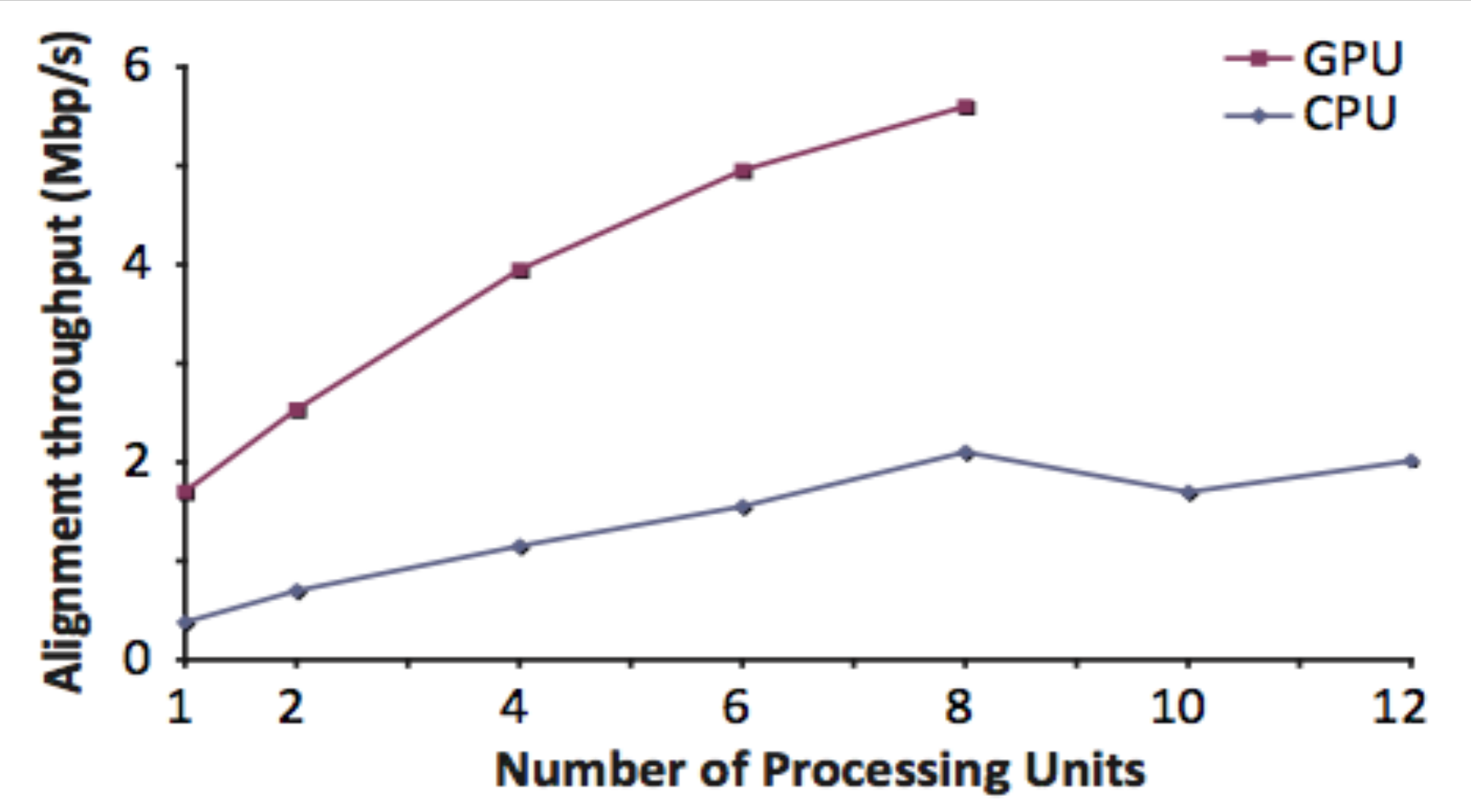
Public Data Repositories

- Essential to all Biologists
- Submission to public repositories a requirement of publication
- Entrez, GEO, SRA, ...
  and hundreds more

GPU Tesla M2070 nodes



BarraCUDA: Multiple GPUs vs Multiple CPUs



Conclusions

- All aspects of CS applicable to Bio
- USA: health care costs = 18% of GDP
- Biocaml
  - just starting
  - your contributions are welcome
  - open source

http://biocaml.org