SPARK SUMMIT 2017

(More!) Tools and Algorithms for Genomic Analysis on Spark

Ryan Williams

6/6/2017

Previously, at Spark Summit East…

- Guacamole: somatic variant caller on Spark
- magic-rdds: collections algorithms on RDDs
- slides, video

This episode

- coverage-depth analysis tool
- cluster bake-off: in-house hadoop vs. gcloud
- hadoop-bam: parable of a legacy genomics file format in a distributed world
- bonus: suffix-arrays

Hammer Lab

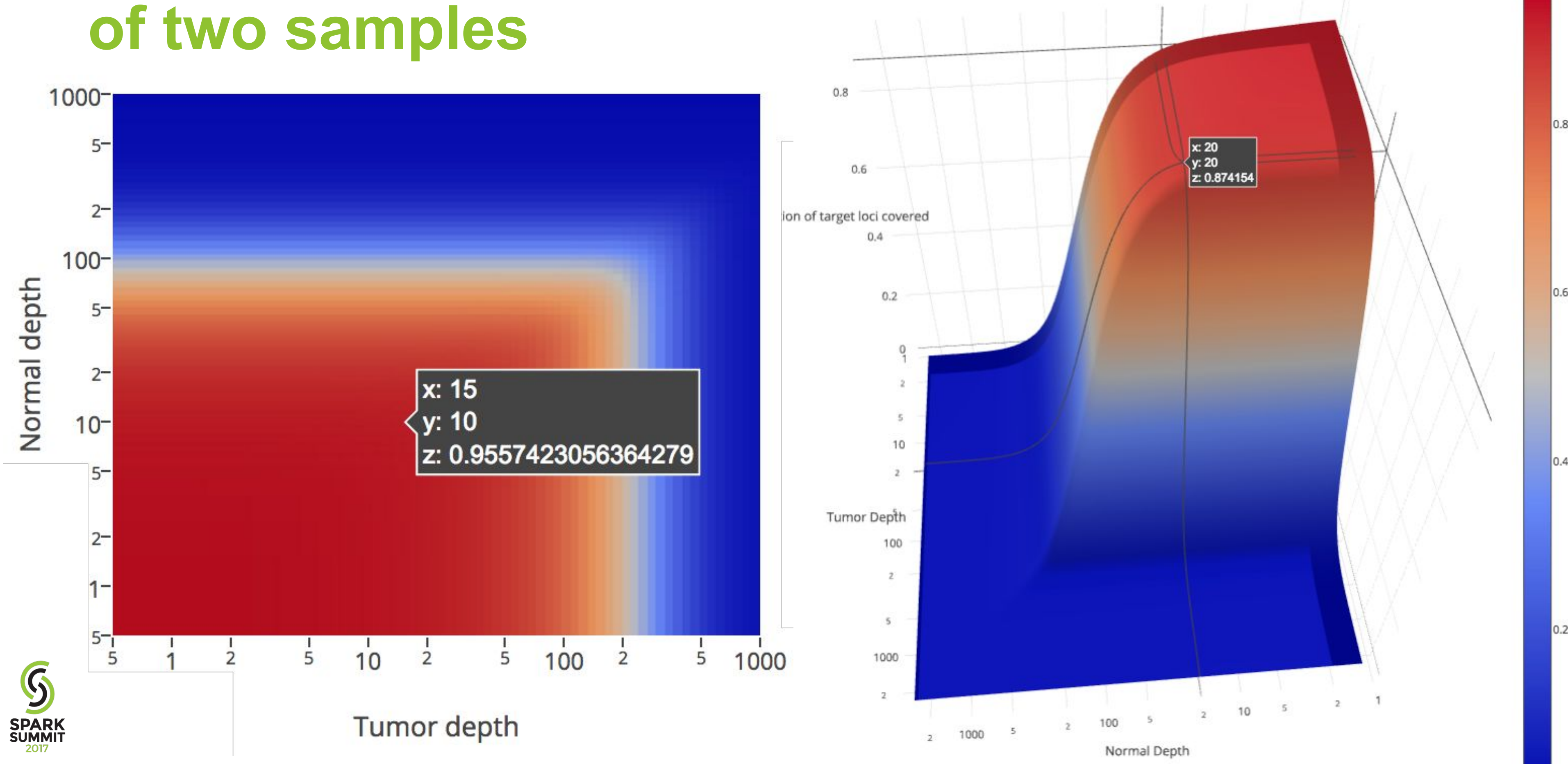
- Mt. Sinai School of Medicine, Parker Institute for Cancer Immunotherapy
- 12 people, mostly computational + _____
- personal genome vaccine trial(s) underway
- misc clinical data analysis
- long-running background thread porting biofx tools to Spark

Slides: http://bit.ly/ss17-ryan

Spark-based Genomic Analysis tools/platforms

- Broad Institute
  - GATK4 - next generation of GATK suite of tools
  - Hail - variant analysis at scale
- AMP Lab: bigdatagenomics
  - ADAM - QC / variant-calling / viz tools
  - bdg-formats - avro schemas for genomic record-types
- Hammer Lab: pageant
  - coverage-depth: QC analyses
  - guacamole: somatic variant caller

coverage-depth - joint histogram of distribution of two samples



coverage-depth: progress and WIP

- running on google cloud and local hadoop cluster
- WIP: multi-plot.ly web-based report
- real-world use:
  - “Contribution of systemic and somatic factors to clinical response and resistance to PD-L1 blockade in urothelial cancer: An exploratory multi-omic analysis”, Snyder et al. 2017
  - upcoming lung-cancer study
  - normalizing mutation counts by # exonic loci with depth ≥ cutoff

In-house Hadoop cluster vs. Google Cloud Dataproc

- Demeter: 100-node, 2400-core cluster
  - $500k circa 2013…
    - ≈ half now?
    - + X% sysadmin allocation
- Google Cloud Dataproc:
  - pre-emptible nodes: $0.02/cpu/hr
    - non-pre-emptible nodes: $0.06/cpu/hr
  - 1 Demeter’s worth of cores for 4 years: $1.7MM
  - utilization break-even range: 10-25% 👋👋👋👋

Recent analysis: coverage-depth of TCGA lung cancer BAMs

- 1060 BAMs (LUAD + LUSC): 14TB
- filter to ensembl exons + by minimum depth
  - goal: normalize each sample's mutation-count by its number of exonic loci with sufficient depth
- 1 ephemeral cluster per app?
- or: 1 big cluster w/ many apps simultaneously

⇒ 10 dataproc clusters of 77 4-core nodes (308 cores)
  - 10mins per sample, 2 samples on a cluster at a time
- 6hrs, $400

SPARK SUMMIT 2017

Slides: http://bit.ly/ss17-ryan

Recent analysis: coverage-depth of TCGA lung cancer BAMs

- Twist: 2 (of 1060) BAMs consistently failed:

  "`MRNM should not be set for unpaired read.`"

- BAMs seemed ok in samtools

  … debugging

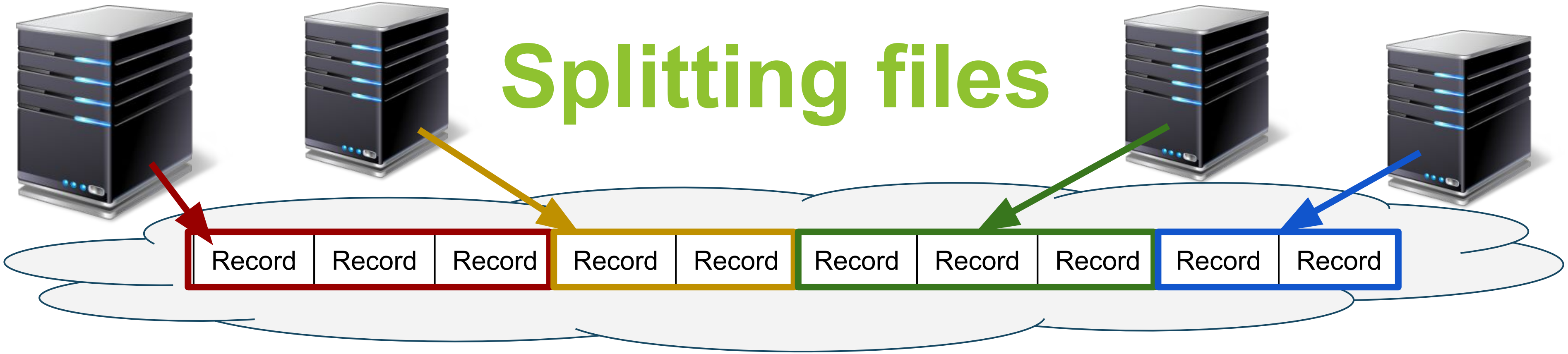
⟹ Bad splits!

SPARK SUMMIT 2017

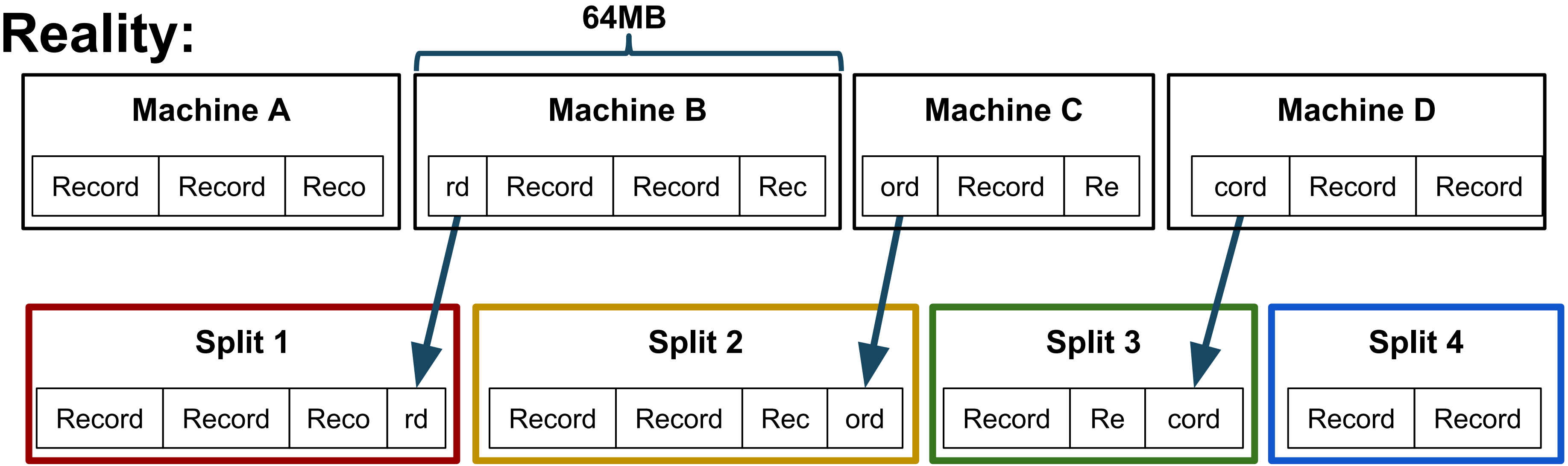
Slides: http://bit.ly/ss17-ryan

Splitting BAM files

Splitting files



Reality:



64MB

Machine A

Record	Record	Reco

Machine B

rd	Record	Record	Rec

Machine C

ord	Record	Re

Machine D

cord	Record	Record

Split 1

Record	Record	Reco	rd

Split 2

Record	Record	Rec	ord

Split 3

Record	Re	cord

Split 4

Record	Record

hadoop-bam

- Implementation of Hadoop `File{In,Out}putFormat`
- Original implementation circa 2010
- Semi-abandoned but critical library underneath Hammer Lab, BDG, and Broad efforts
- Main goal: “split” BAM files

~~BAM~~ SAM format

- Sequence Alignment/Map

```
        @HD   VN:1.4    GO:none   SO:coordinate
Header  @SQ   SN:1 LN:249250621
        @SQ   SN:2 LN:243199373
        …
        HWI-ST807:8592:79724   163   1   10001    0       101M   =   10009    109   TAACCCTAACC…
        HWI-ST807:8592:79724    83   1   10009    0       101M   =   10001   -109   ACCCTAACCCT…
Reads   HWI-ST807:9505:89866   163   1   10048   29   20M1D81M   =   10368    374   CCAACCCTAAC…
        HWI-ST807:6431:65669   163   1   10335   29    1S90M2D   =   10458    224   CAACCCTAACC…
        …
```

- Probably splittable (on newlines)?

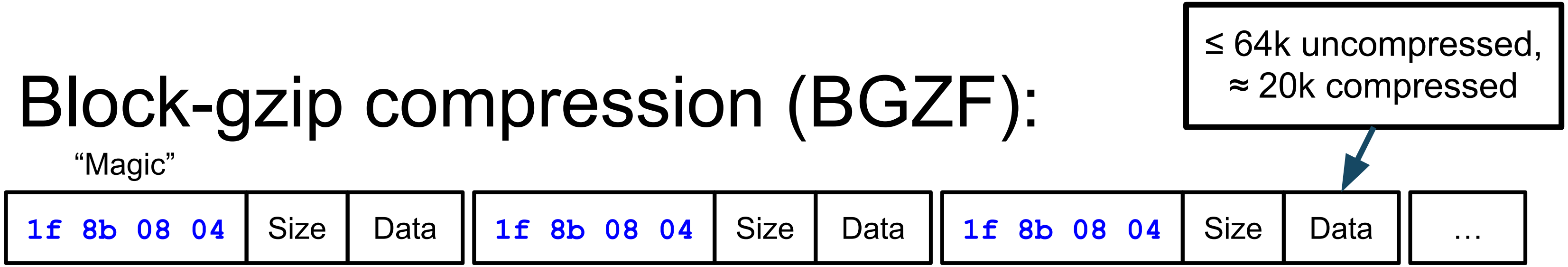
BAM format

→ SAM format

+ Binary record codec:

#bytes	contig	start	mapq	len(name)	name	len(cigar)	flags	len(seq)	cigar	seq	quals	tags

+ Block-gzip compression (BGZF):



≤ 64k uncompressed,
≈ 20k compressed

“Magic”

1f 8b 08 04	Size	Data	1f 8b 08 04	Size	Data	1f 8b 08 04	Size	Data	…

Splitting BAMs

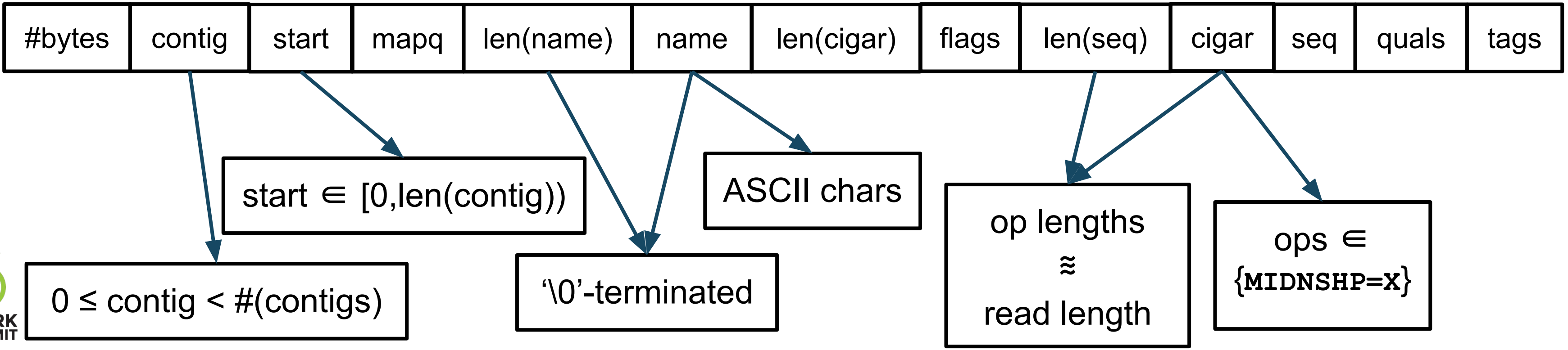
- scan (≤ 64k) until magic `0x1f8b0804`
- optional: skip ahead “size” bytes, verify “magic” again
- certainty: (2^32)^(N blocks)

- BGZF: 🤔 ✅

“Magic”

`1f 8b 08 04`	Size	Data	`1f 8b 08 04`	Size	Data	`1f 8b 08 04`	Size	Data	…

- Binary records: 🤔 🚫



#bytes	contig	start	mapq	len(name)	name	len(cigar)	flags	len(seq)	cigar	seq	quals	tags

start ∈ [0,len(contig))

0 ≤ contig < #(contigs)

‘\0’-terminated

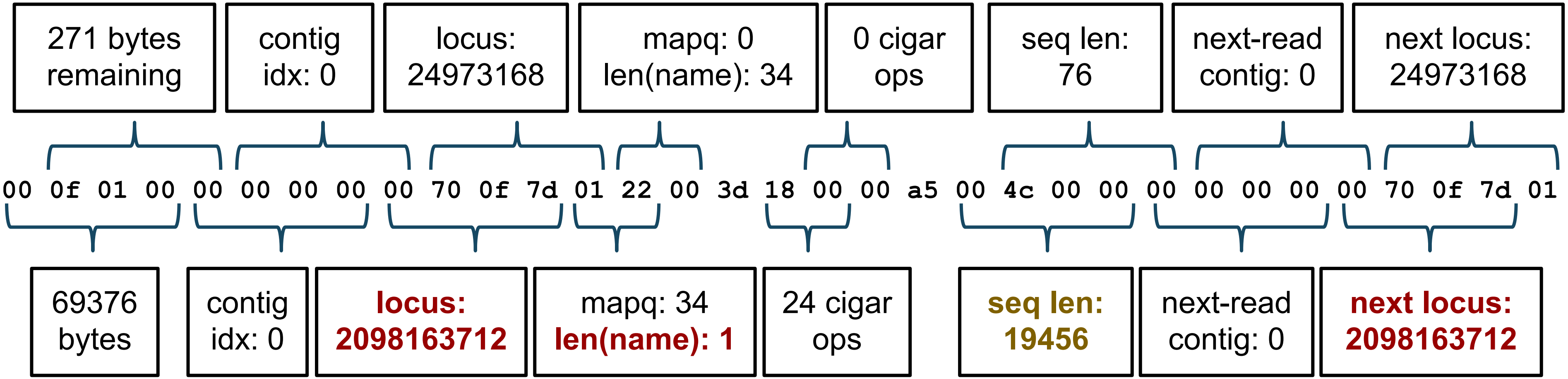
ASCII chars

op lengths ≋ read length

ops ∈ `{MIDNSHP=X}`

Case Study: BAM-splitting false positive

- TCGA 19155553-8199-4c4d-a35d-9a2f94dd2e7d, offset 268458108:115



271 bytes remaining

contig idx: 0

locus: 24973168

mapq: 0 len(name): 34

0 cigar ops

seq len: 76

next-read contig: 0

next locus: 24973168

```
00 0f 01 00 00 00 00 00 00 70 0f 7d 01 22 00 3d 18 00 00 a5 00 4c 00 00 00 00 00 00 00 70 0f 7d 01
```

69376 bytes

contig idx: 0

locus: 2098163712

mapq: 34 **len(name): 1**

24 cigar ops

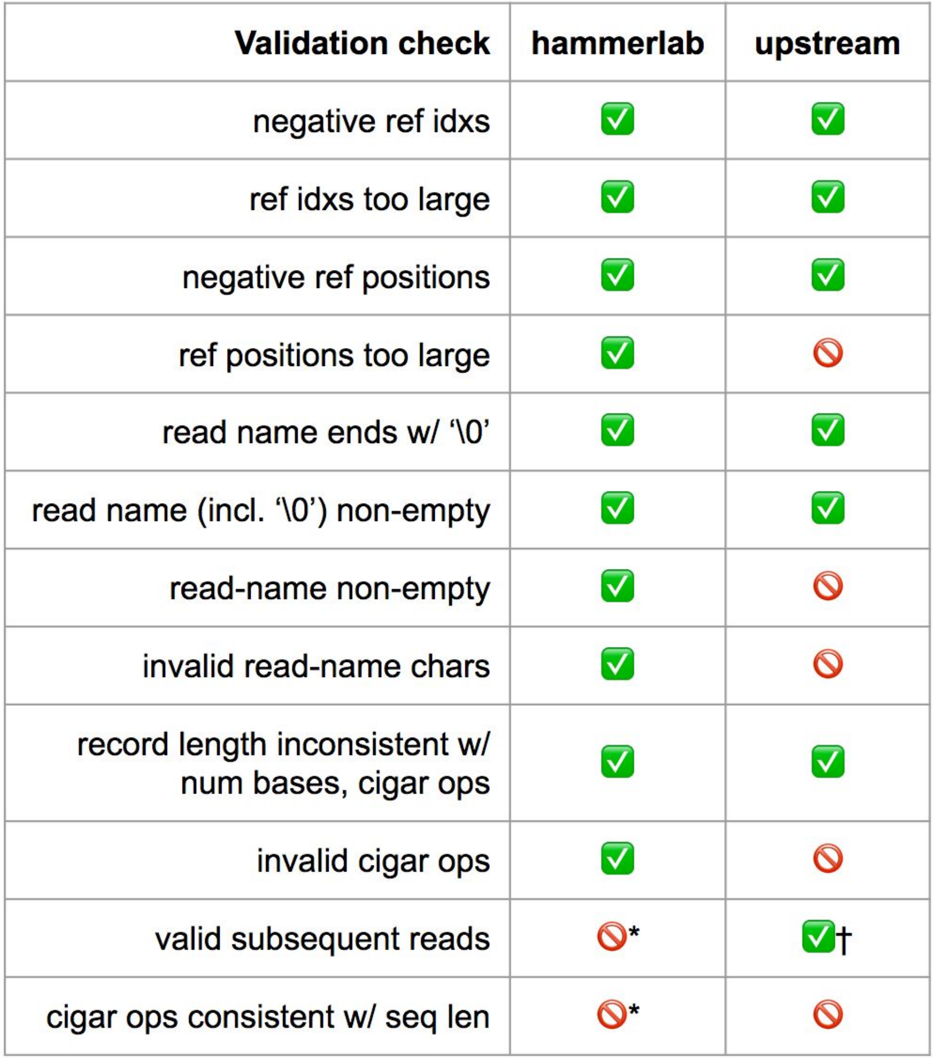
seq len: 19456

next-read contig: 0

next locus: 2098163712

hammerlab/hadoop-bam

- “fork” of upstream hadoop-bam
- additional checks avoid known false-positives



Validation check	hammerlab	upstream
negative ref idxs	✅	✅
ref idxs too large	✅	✅
negative ref positions	✅	✅
ref positions too large	✅	🚫
read name ends w/ ‘\0’	✅	✅
read name (incl. ‘\0’) non-empty	✅	✅
read-name non-empty	✅	🚫
invalid read-name chars	✅	🚫
record length inconsistent w/ num bases, cigar ops	✅	✅
invalid cigar ops	✅	🚫
valid subsequent reads	🚫*	✅†
cigar ops consistent w/ seq len	🚫*	🚫

* easy to add, seemingly unnecessary thus far

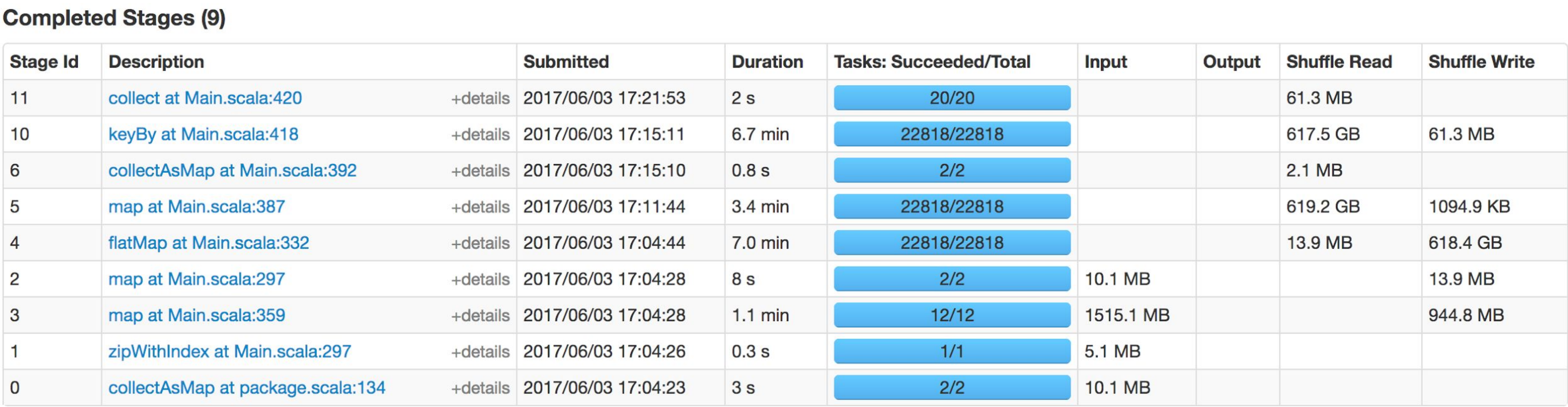
† partial credit; only 1 random check performed on subsequent reads

hammerlab/hadoop-bam

- “check” mode evaluates every position in BAM ⟶
- also: positions where ≤ 2 checks supported (true) “negative” call

```
               invalidCigarOp:    28661374692
           tooLargeNextReadIdx:    27924049452
               tooLargeReadIdx:    27924049452
     nonNullTerminatedReadName:    24885666031
tooFewRemainingBytesImplied:    23071387740
              nonASCIIReadName:     2367016056
                    noReadName:     2271887125
          negativeNextReadIdx:     1582430053
              negativeReadIdx:     1582430053
              negativeReadPos:     1582430053
          negativeNextReadPos:     1582430053
                emptyReadName:      232401822
          tooLargeNextReadPos:       43095171
              tooLargeReadPos:       43095171
        tooFewBytesForReadName:             73
         tooFewFixedBlockBytes:             35
        tooFewBytesForCigarOps:             16
```

“Full” Checker - Spark History



Completed Stages (9)

Stage Id	Description		Submitted	Duration	Tasks: Succeeded/Total	Input	Output	Shuffle Read	Shuffle Write
11	collect at Main.scala:420	+details	2017/06/03 17:21:53	2 s	20/20			61.3 MB	
10	keyBy at Main.scala:418	+details	2017/06/03 17:15:11	6.7 min	22818/22818			617.5 GB	61.3 MB
6	collectAsMap at Main.scala:392	+details	2017/06/03 17:15:10	0.8 s	2/2			2.1 MB	
5	map at Main.scala:387	+details	2017/06/03 17:11:44	3.4 min	22818/22818			619.2 GB	1094.9 KB
4	flatMap at Main.scala:332	+details	2017/06/03 17:04:44	7.0 min	22818/22818			13.9 MB	618.4 GB
2	map at Main.scala:297	+details	2017/06/03 17:04:28	8 s	2/2	10.1 MB			13.9 MB
3	map at Main.scala:359	+details	2017/06/03 17:04:28	1.1 min	12/12	1515.1 MB			944.8 MB
1	zipWithIndex at Main.scala:297	+details	2017/06/03 17:04:26	0.3 s	1/1	5.1 MB			
0	collectAsMap at package.scala:134	+details	2017/06/03 17:04:23	3 s	2/2	10.1 MB			

- 10GB BAM, 30BN uncompressed positions, 94MM reads
- 100% checker accuracy
- Largest shuffle: 600+ GB

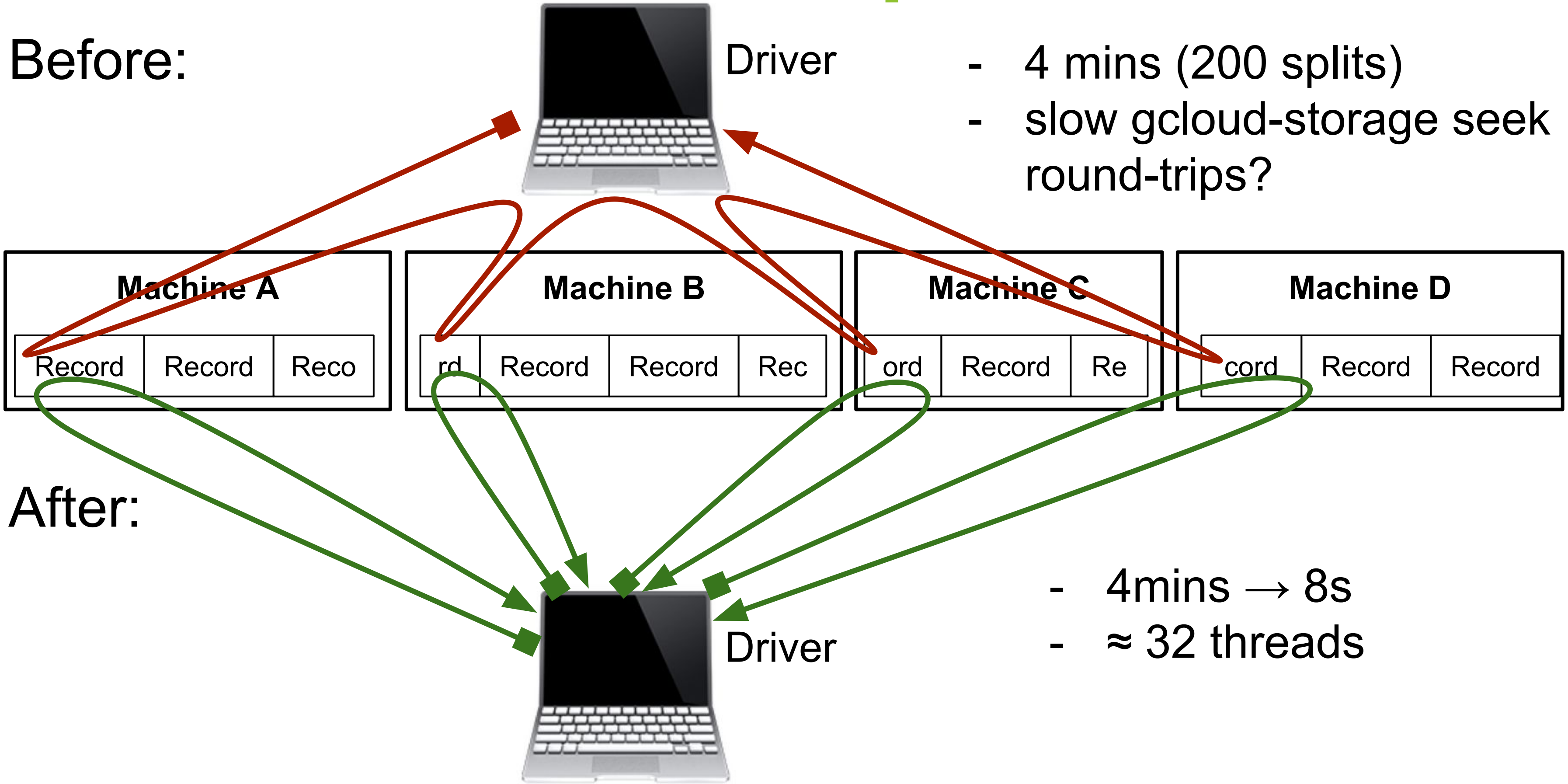
  ⇒ 20 bytes / position (compressed)

Slides: http://bit.ly/ss17-ryan

Parallelize split computation

Before:

- 4 mins (200 splits)
- slow gcloud-storage seek round-trips?

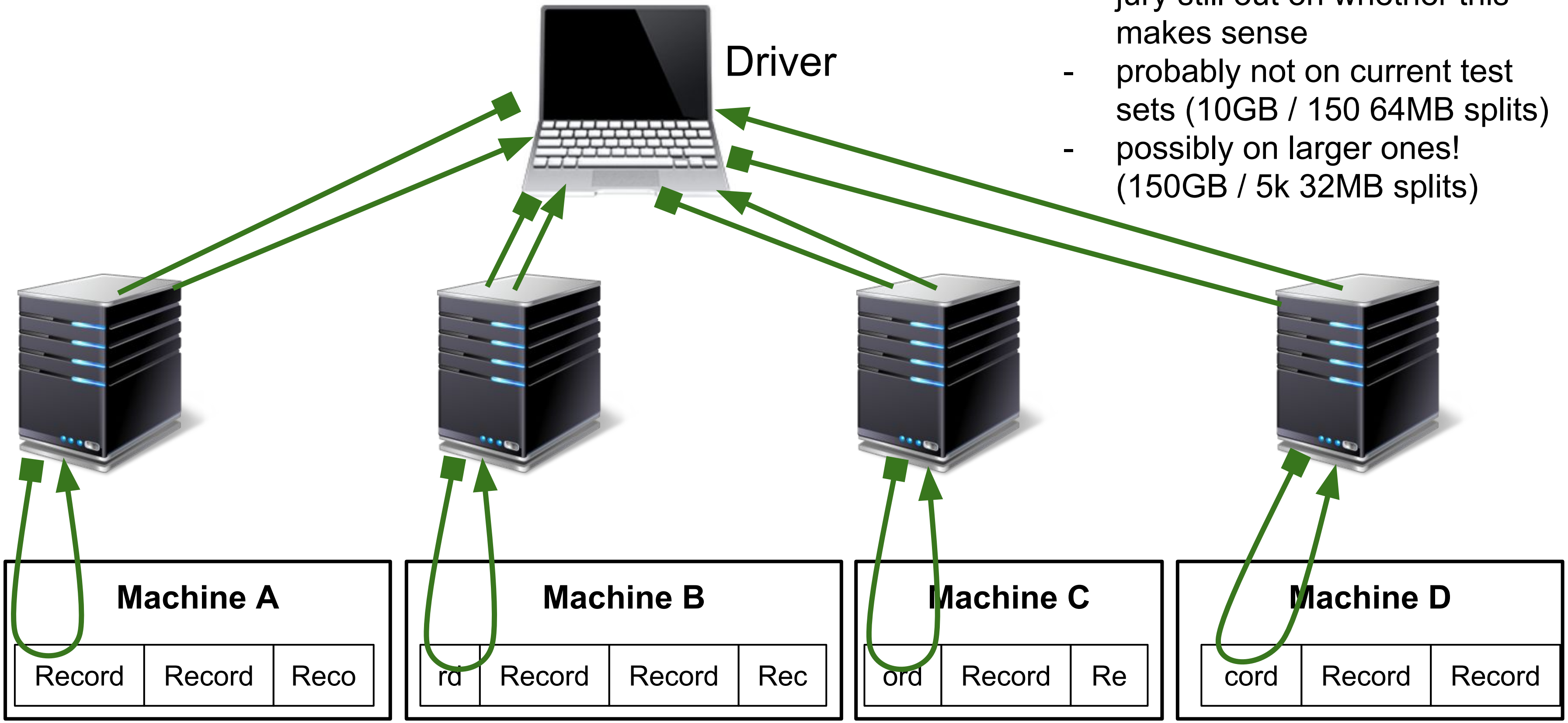


After:

- 4mins → 8s
- ≈ 32 threads

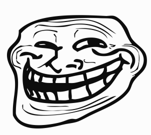
Parallelize split computation, pt 2

- jury still out on whether this makes sense
- probably not on current test sets (10GB / 150 64MB splits)
- possibly on larger ones! (150GB / 5k 32MB splits)



Do we have to use BAMs?

- VCFs being deprecated (at least culturally)
- BAMs seem like they're sticking around
    - Long reads may incentivize dropping BAM
- Aligners output BAMs

⟹ Someone should write a distributed aligner 

hammerlab/suffix-arrays

- Distributed construction of suffix arrays and FM-Indices
- WIP
- Open q's
    - how to use them in distributed env
    - output binary-compatible indices that other tools would generate?

Ongoing/Future Work

- release / publish Pageant suite of tools
    - top of stack: guacamole (somatic variant caller)
- long reads?

Questions?