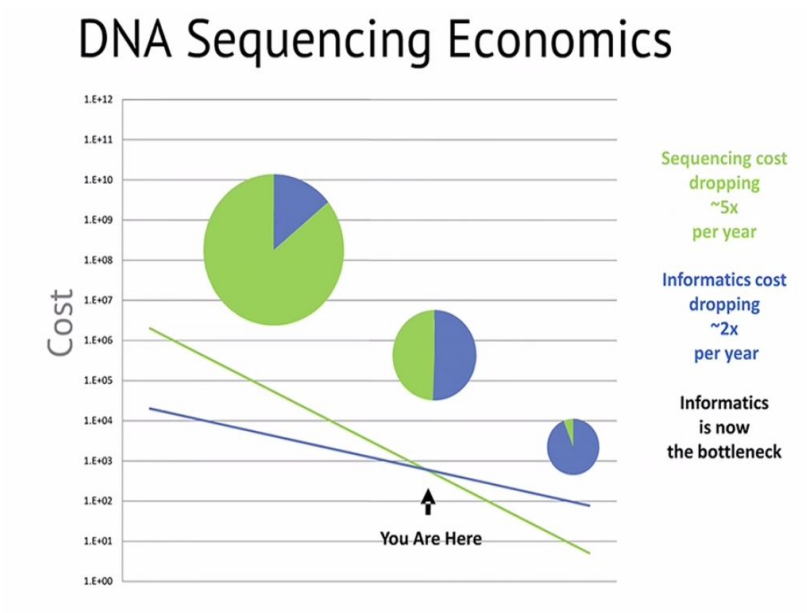
Hummingbird: Efficient Performance Prediction for Executing Genomics Applications in the Cloud

Utsab Ray*, Vandhana Krishnan*, Amir Bahmani*,
Cuiping Pan, Keith Bettinger,
Frank Mueller, Philip Tsao, Michael Snyder

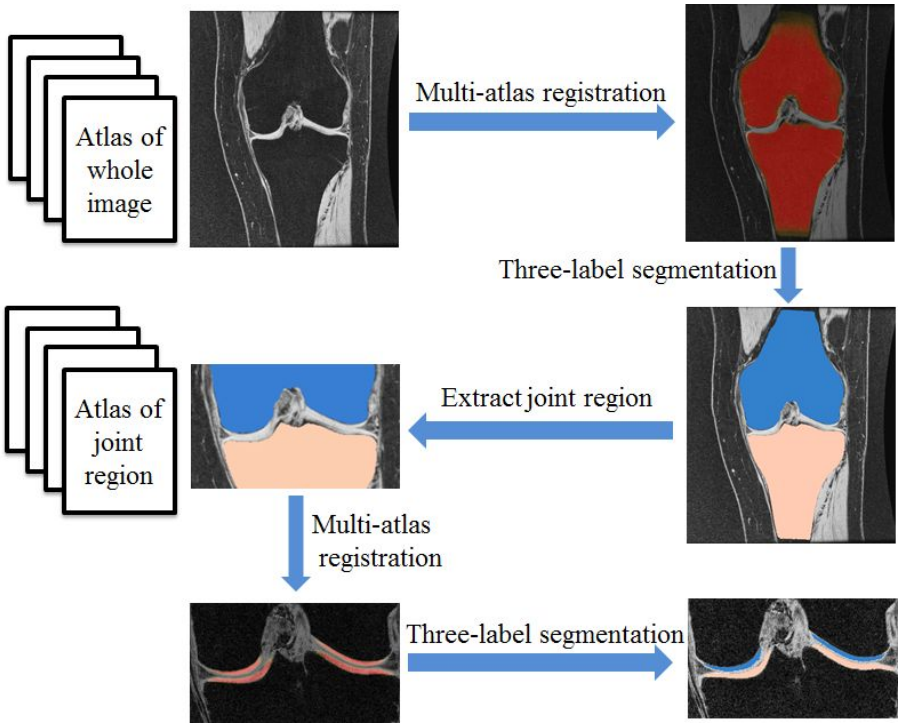
Introduction

- Advances in medical computing result in pipelines being executed on cloud
- Medical pipelines cost significant money to run on cloud
- Lack of frameworks to identify best way to run pipeline
- Objective: High performance and low cost
  - How to choose best configuration?



What are Medical Pipelines?

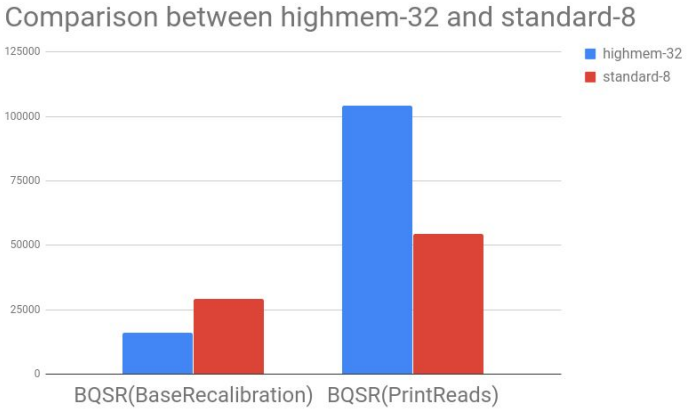
- Multiple stages
- Output of one stage -> input of next stage
- High cost to execute pipeline on cloud
- One whole run of GATK pipeline -> at least $35
- Image on right shows flowchart of overall cartilage segmentation pipeline



Motivation

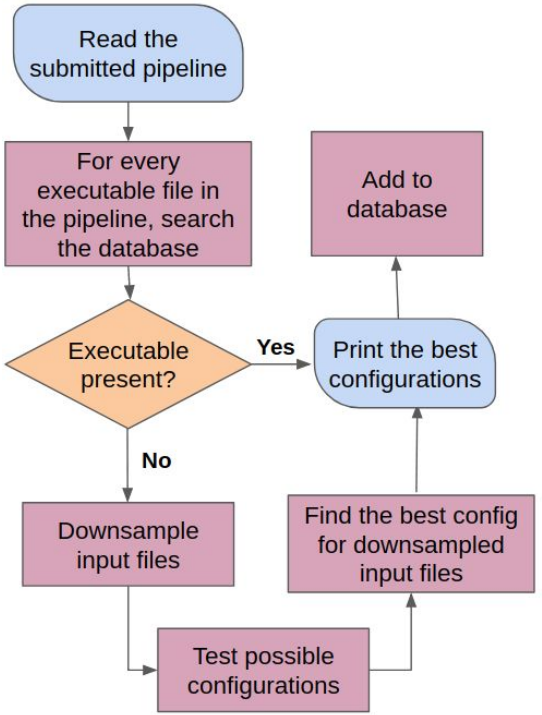
- Medical pipelines have many stages
- Different configuration best for different stage
  - Using same config for all stages
    - More cost and/or exec time
  - Config: highmem-32, standard-8
    - Instance types on Google cloud
- Our contribution: Hummingbird
  - Recommend best configuration for each stage

Google Cloud Execution



Hummingbird Overview

- Execute pipeline in reduced time
  - Use downsampling
- Provide user with best configurations
  - Each stage has different recommendation
- Helps in getting best way to execute pipeline in minimal time



Sample Configuration File

- Image shows information required to launch dsub job
  - "min-ram", "min-cores" and "t" can be varied
- Varying min-ram and min-cores helps in execution on different instance types

```
dsub \
--project gbsc-gcp-project-hummingbird \
--zones "us-east1-*" \
--logging gs://gbsc-sgc-hummingbird-uray/logging-1mil-res
--input-recursive INPUT_PATH=gs://gbsc-sgc-hummingbird-ur
--output OUTPUT_FILE=gs://gbsc-sgc-hummingbird-uray/test_
--min-ram 33 \
--min-cores 22 \
--image vandhanak/broadcloudgatk36gatk37-cgs:v1 \
--command '/usr/gitc/bwa mem -v 3 -t 32 -M -R "@RG\tID:0\
 "${INPUT_PATH}"/GRCh37-lite.fa "${INPUT_PATH}"/one_mil.f
 | /usr/local/bin/samtools view -b -S -h - -o "$(dirname
--command '/usr/local/bin/samtools view -H "${INPUT_PATH}
 | /usr/local/bin/samtools reheader - "${INPUT_PATH}"/ura
/uray-bwa-bam-reheadered.bam'

Input_File_1="one_mil.fastq.gz"
Input_File_2="two_mil.fastq.gz"

Multi-threaded=-t,NO
```

Downsampling

- Major contribution of Hummingbird
- Run pipeline on small fraction of input
  - Challenge: Not all pipelines can execute on small fraction of input
- Helps in fast prediction
- Hummingbird downsamples two commonly used data formats in genomics: *FASTQ* and *BAM*(Binary Alignment Map)
  - In fastq files only the first n lines comprise the downsampled file
  - BAM files downsampled using DownsampleSam tool
    - Retains a subset of reads according to probability argument

Downsampling(contd.)

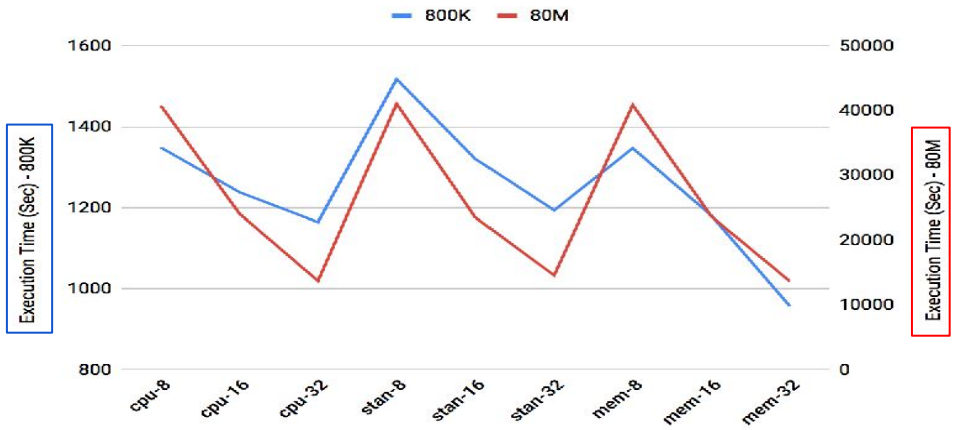
```
@ERR194158.1 HSQ1008:176:D0UYCACXX:2:2303:13853:63128/1
CTCGCAATGCTATCGCGTGCATCCCCCCCCAGACGAAAATACCAAATGCATGGAGAGCTCCCGTGAGTCGTAAATAGGGGGATAGACCTGTGATCCATCGCG
+
@@?D?<0AFFBAFB;AF)<CEC9?FDAEF:;(5'@':'7=;)7)(6((.((6;2?B<55></,,(2((&+(+8:@::(&&0-(((+:A3(+43>++4(8(&
@ERR194158.2 HSQ1008:176:D0UYCACXX:2:2312:18606:26079/1
GATCACAGGTCTATCACCCTATTAACCACTCACGGGAGCTCTCCATGCATTTGGTATTTTCGTCTGGGGGGGATGCACGCGATAGCATTGCGAGACGCTGG
+
CCCFFFFFHHHHHJJJJJJJJJJJIJJIJJJJJJJJJJIJJJJJIJJJJJJJJJJBGIJJJJJJJJHHGHDDDBDDBDDDDDDDDDDDEDDDDBDBDDBDDD
```

- Above fastq file has 2 sequences
- Downsampling results in below fastq file with 1 sequence
- Pipeline can execute on downsampled file because sequences independent of each other

```
@ERR194158.1 HSQ1008:176:D0UYCACXX:2:2303:13853:63128/1
CTCGCAATGCTATCGCGTGCATCCCCCCCCAGACGAAAATACCAAATGCATGGAGAGCTCCCGTGAGTCGTAAATAGGGGGATAGACCTGTGATCCATCGCG
+
@@?D?<0AFFBAFB;AF)<CEC9?FDAEF:;(5'@':'7=;)7)(6((.((6;2?B<55></,,(2((&+(+8:@::(&&0-(((+:A3(+43>++4(8(&
```

Does Downsampling work?(contd.)

- Figure shows downsampling for MuTect
  - Left axis execution time for 800K lines(downsampled file)
  - Right axis execution time for 80M lines(original file)



- Executing MuTect on both files results in similar trends
- Inference: Best configuration for downsampled will also be best configuration for whole input

Experimental Framework

- Pipelines
  - GATK(Genome Analysis Toolkit) 3.7
  - MuTect2
- Datasets
  - GATK 3.7: Open source Illumina Platinum Genomes
  - MuTect2: BAM files provided by Texas Cancer Research Biobank

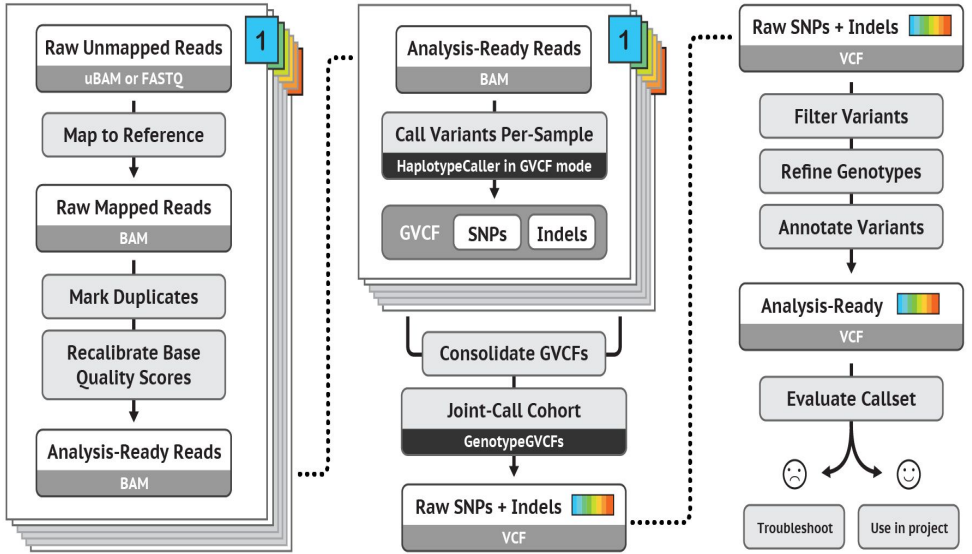
Experimental Framework(contd.)

- dsub
  - Command line tool to submit jobs on Google cloud
- Google Cloud
  - Three main categories of instances: high-cpu, standard and high-mem
  - CPU platform same across all categories
  - Table shows amount of RAM differs

Number of VCPUs	High-CPU (GB)	Standard (GB)	High-Mem (GB)
8	7.2	30	52
16	14.4	60	104
32	28.8	120	208

GATK HaplotypeCaller Pipeline

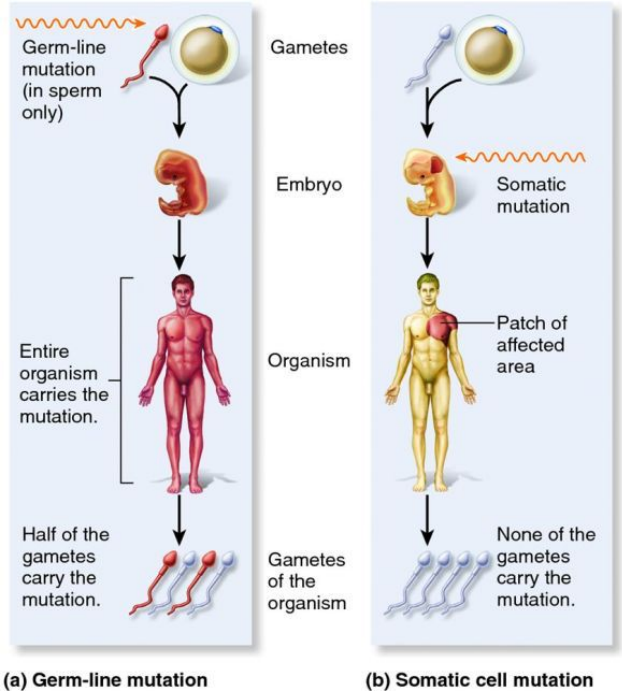
- GATK best practices provided by Broad Institute
- HaplotypeCaller used to call germline SNPs (single nucleotide polymorphisms) and indels (insertions-deletions)



SOURCE: https://software.broadinstitute.org/gatk/best-practices/workflow?id=11145

MuTect Pipeline

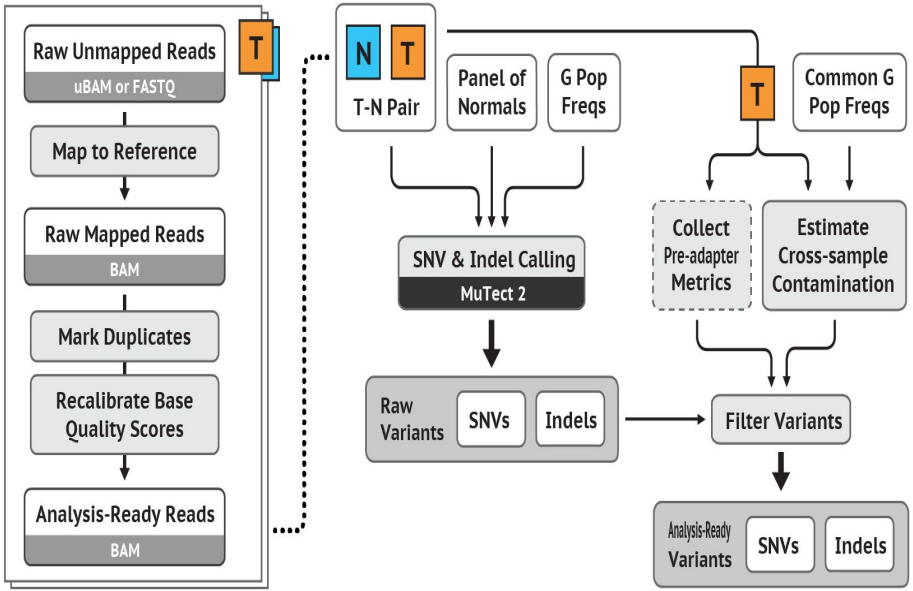
- MuTect2 is a ***somatic*** SNP and indel caller
  - Fuses somatic genotyping engine with aspects of GATK HaplotypeCaller algorithm.
  - Method for reliable and accurate identification of somatic point mutations in next generation sequencing data of cancer genomes



SOURCE: https://macscience.wordpress.com/level-2-biology/genetics/somatic-vs-germline-mutations/

MuTect Pipeline (cont'd)

- GATK pipeline that employs MuTect2 overlaps significantly with one that uses HaplotypeCaller
- MuTect2 discovers variants in tumor BAM files
  - HaploType caller discovers variants in normal BAM files



SOURCE: https://software.broadinstitute.org/gatk/best-practices/workflow?id=11146

Decision Table

- Table shows calculations which Hummingbird uses to recommend different configurations

Instance	Ideal Speedup	Execution Time	Real Speedup	Normalized Speedup	Cost
T_8(base)	1	E_8	$S_8=E_8/E_8$	1	C_8
T_{16}	2	E_{16}	$S_{16}=E_8/E_{16}$	S_{16}	C_{16}
T_{32}	4	E_{32}	$S_{32}=E_8/E_{32}$	$S_{32}/2$	C_{32}

Decision Table(cont'd.)

- For each category of instance Hummingbird makes 1 table
- Results in total of 3 tables for high-cpu, standard and high-mem
- Normalized speedup column shows scalability of algorithm
- Cost = Execution Time * Cost of instance per second
- Per second cost obtained from Google website
- For recommendation, Hummingbird looks at all 3 tables to calculate best configurations

Recommended Configurations

- Hummingbird recommends three different configurations to the user
  - Cheapest: Least cost among 3 tables
  - Fastest: Least execution time among 3 tables
  - Fast and Cheap
    - From each table select instance with highest normalized speedup
    - Amongst those 3 select instance with minimum cost

Sample Decision Table

- Table shows High-CPU decision table for MuTect2

High-CPU

	Instance	Ideal Speedup	Execution Time(sec)	Real Speedup	Normalized Speedup	Cost ($)
Cheapest →	T_8(base)	1	E_8=1348.591	1	1	0.11
	T_{16}	2	E_{16}=1238.294	1.09	1.09	0.19
Fastest →	T_{32}	4	E_{32}=1164.396	1.16	0.58	0.37

Results and Analysis (BWA)

- Recommends highcpu-32 to be fastest
  - Actual fastest is standard-32, which is 3.1% faster than highcpu-32

	Hummingbird		Whole Input	
	Config	Exec Time (s)	Config	Exec Time (s)
Cheapest	standard-8	26.075	standard-8	92,124.358
Fastest	highcpu-32	15.757	standard-32	30,286.578
Fast & Cheap	standard-16	19.094	standard-8	92,124.358

- For fast-and-cheap Hummingbird results in 5.1% higher cost and 47% faster by predicting incorrectly
- Hummingbird executes in 3 orders of magnitude faster than whole input

Results and Analysis (BWA)

- Recommends highcpu-32 to be fastest
  - Actual fastest is standard-32, which is 3.1% faster than highcpu-32

	Hummingbird		Whole Input	
	Config	Exec Time (s)	Config	Exec Time (s)
Cheapest	standard-8	26.075	standard-8	92,124.358
Fastest	highcpu-32	15.757	standard-32	30,286.578
Fast & Cheap	standard-16	19.094	standard-8	92,124.358

- For fast-and-cheap Hummingbird results in **5.1**% higher cost and **47**% faster by predicting incorrectly
- Hummingbird executes in 3 orders of magnitude faster than whole input

Results and Analysis (MuTect)

- Table compares Hummingbird's recommendation for MuTect2 with actual recommendation obtained by executing whole input
- Hummingbird able to accurately predict cheapest, fastest and fast-and-cheap configuration

	Hummingbird		Whole Input	
	Config	Exec Time (s)	Config	Exec Time (s)
Cheapest	highcpu-8	1,348.591	highcpu-8	40,754.48
Fastest	highmem-32	957.078	highmem-32	13,674.71
Fast & Cheap	highcpu-16	1,164.396	highcpu-16	24,087.63

Cost & Execution Time Comparison

- Execution of Hummingbird recommendation for cheapest on each stage of GATK vs Execution of Standard-16 on entire GATK pipeline

Hummingbird cheapest vs Standard-16

	Hummingbird Cheapest	Standard -16
Execution Time(hours)	92.5	77.9
Cost($)	35.1	59.2

Cost & Execution Time Comparison

- Execution of Hummingbird recommendation for fastest on each stage of GATK vs Execution of Highmem-32 on entire GATK pipeline

Hummingbird fastest vs Highmem-32

	Hummingbird Fastest	Highmem -32
Execution Time(hours)	66.9	75.5
Cost($)	80.8	143.1

Cost & Execution Time Comparison

- Execution of Hummingbird recommendation for fast-and-cheap on each stage of GATK vs Execution of Highmem-32 on entire GATK pipeline

Hummingbird fast-and-cheap vs Highmem-32

	Hummingbird Fast-&-Cheap	Highmem -32
Execution Time(hours)	77.5	75.5
Cost($)	38.7	143.1

Conclusion

- Hummingbird: Efficient way of predicting best to execute genomics application in Google cloud
- Minimal training cost
- Prediction performed in short time by using downsampling
- User provided with three different configurations

Acknowledgement

- I am extremely grateful to Stanford Center for Genomics and Personalized Medicine, where a big chunk of this work was done
- The contributions of my co-authors Vandhana, Amir, and Keith, along with my advisor Dr. Frank Mueller were also invaluable

Questions?

Future Work

- Tune the prediction model for Hummingbird to ensure better predictions
- Extend the downsampling concept to fields other than genomics
- Introduce auto-tuning so Hummingbird can provide user with an optimal set of parameters for execution
- Make the framework cloud agnostic
- Predict time taken for execution of pipeline on whole input

Hummingbird Overview

- Parse user provided configuration file
- For each stage of the pipeline
  - See if configurations to execute that stage present in database
  - If present
    - Provide user with best configuration
  - Else
    - Downsample
    - Execute stage on downsampled files
    - According to execution time calculate cheapest, fastest and fast-and-cheap configuration
    - Provide user with best configuration
    - Add those configurations to the database