BROAD INSTITUTE

Heterogeneous compute in the GATK

Mauricio Carneiro
GSA – Broad Institute

Intel Genomic Sequencing Pipeline Workshop
Mount Sinai
12/10/2013

This is the work of many…

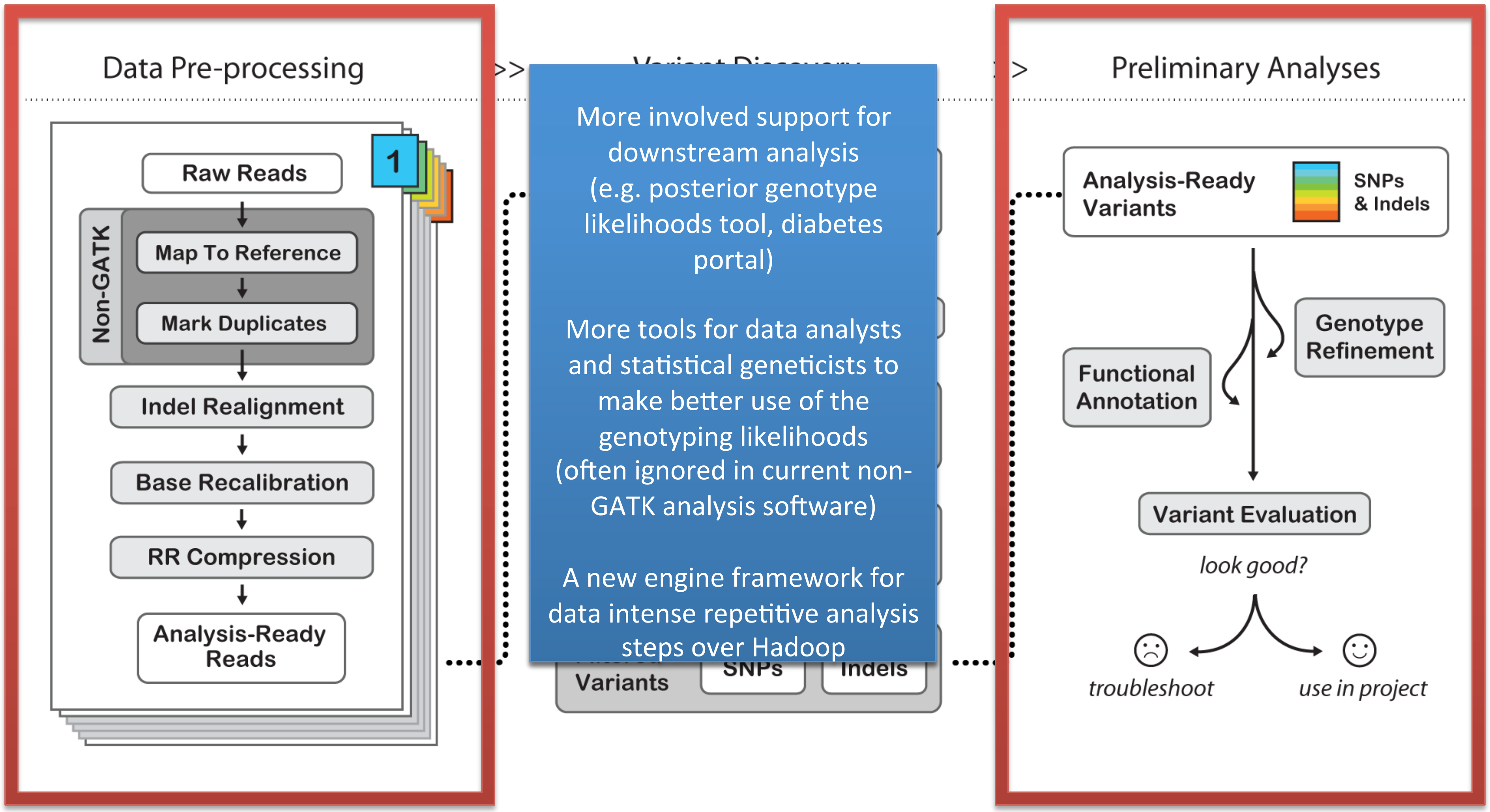
Mark DePristo

Eric Banks

Stacey Gabriel

David Altshuler

Scope and schema of the Best Practices



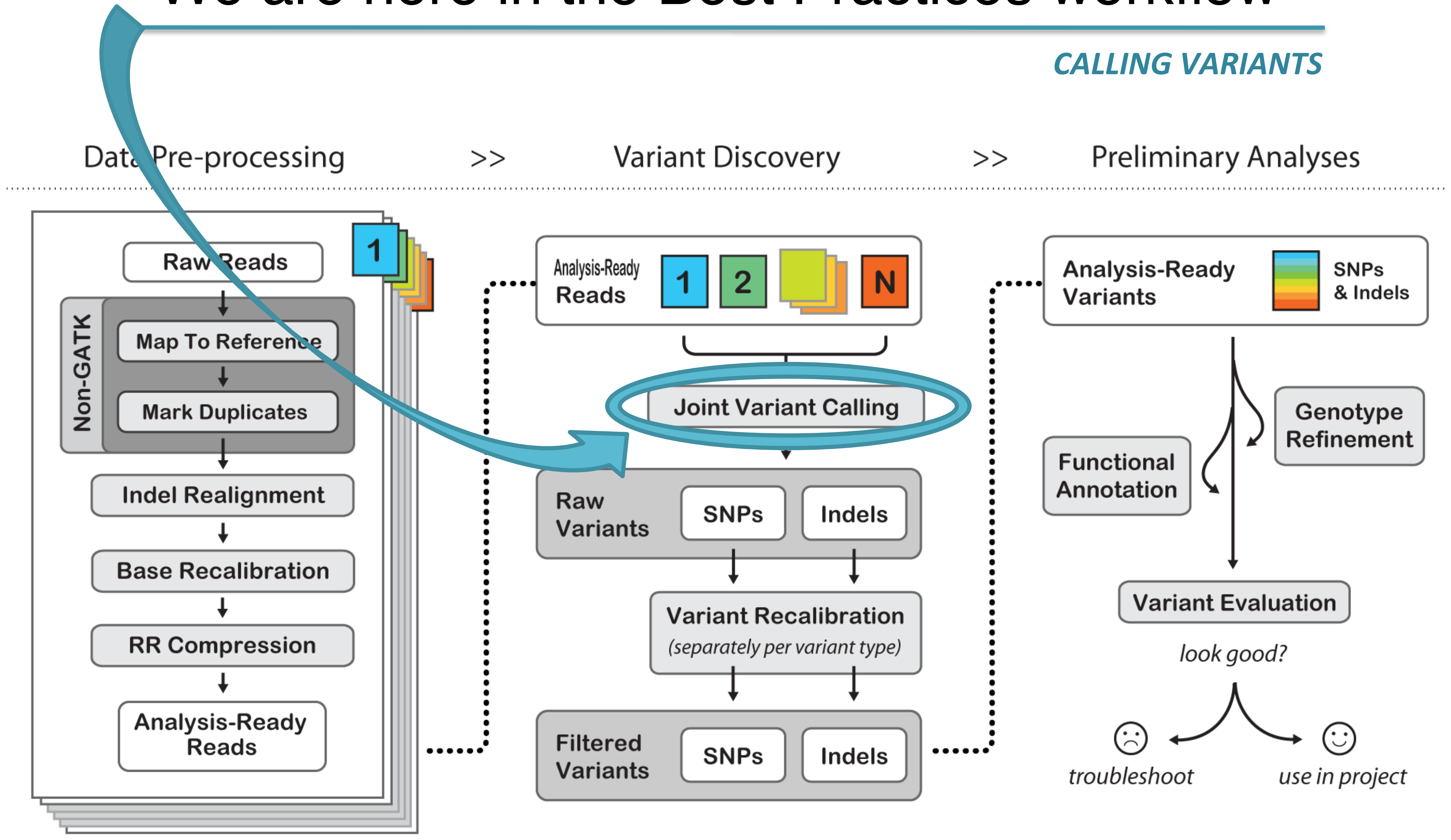
More involved support for downstream analysis (e.g. posterior genotype likelihoods tool, diabetes portal)

More tools for data analysts and statistical geneticists to make better use of the genotyping likelihoods (often ignored in current non-GATK analysis software)

A new engine framework for data intense repetitive analysis steps over Hadoop

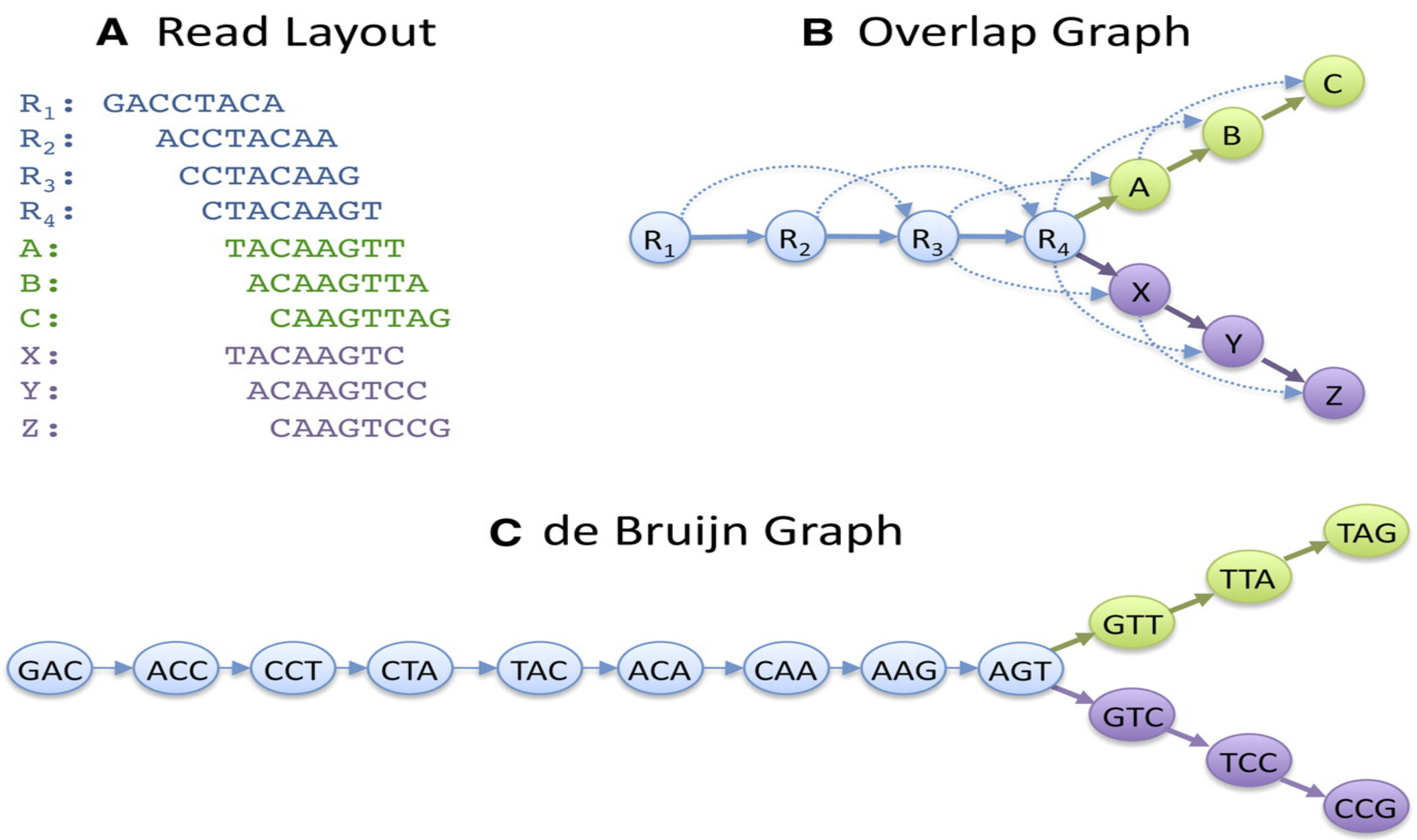
We are here in the Best Practices workflow

CALLING VARIANTS



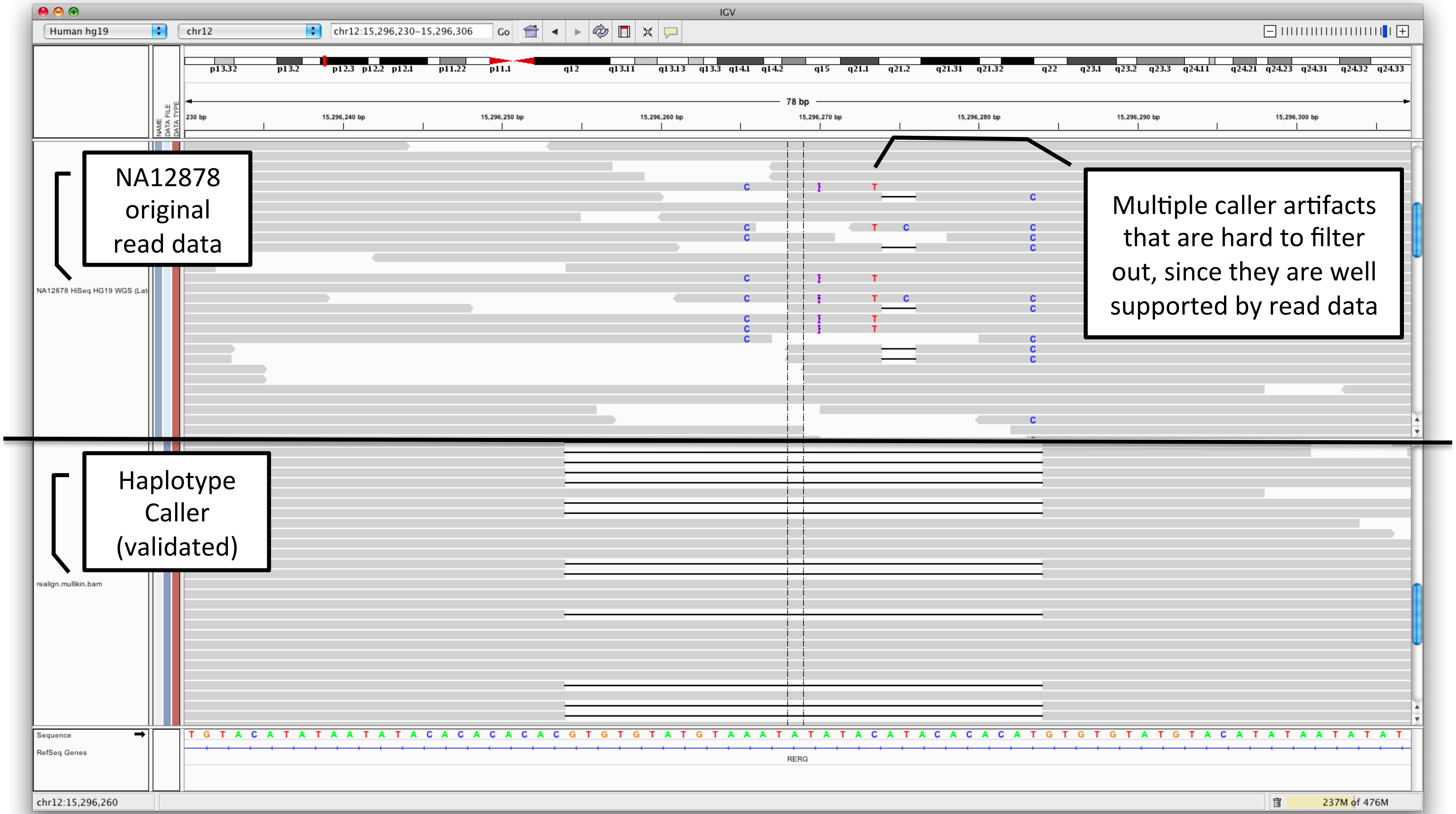
gatk

GATK's Haplotype Caller is replacing the seasoned Unified Genotyper



Haplotype Caller is ready today for a small number of samples, but needs to scale better

Artifact SNPs and small indels caused by large indel is only recovered by local assembly

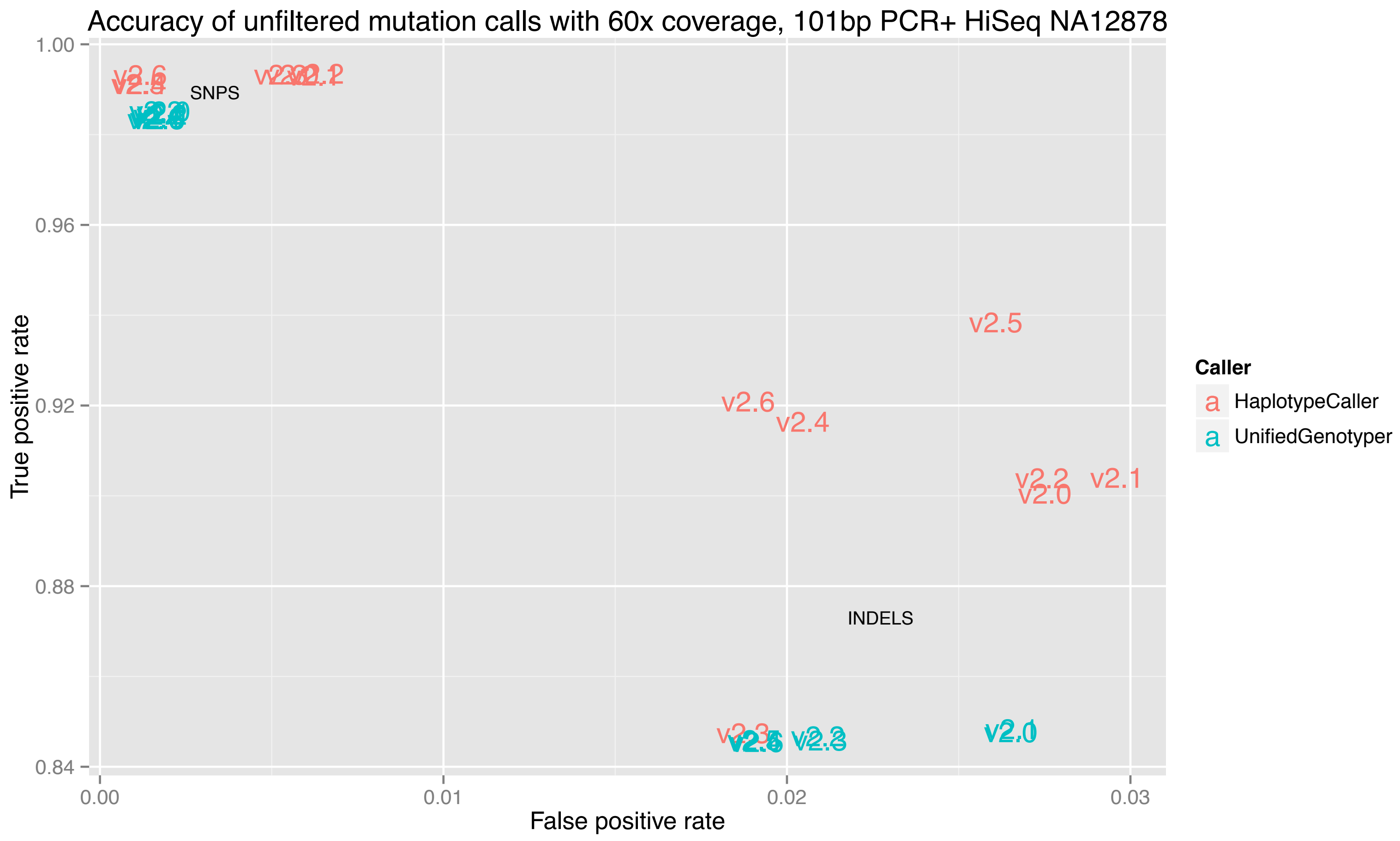

IGV
Human hg19
chr12
chr12:15,296,230-15,296,306
Go
p13.32 p13.2 p12.3 p12.2 p12.1 p11.22 p11.1 q12 q13.11 q13.13 q13.3 q14.1 q14.2 q15 q21.1 q21.2 q21.31 q21.32 q22 q23.1 q23.2 q23.3 q24.11 q24.21 q24.23 q24.31 q24.32 q24.33
78 bp
230 bp
15,296,240 bp
15,296,250 bp
15,296,260 bp
15,296,270 bp
15,296,280 bp
15,296,290 bp
15,296,300 bp
NAME
DATA FILE
DATA TYPE
NA12878 HiSeq HG19 WGS (Lat
realign.mullikin.bam
Sequence
RefSeq Genes
T G T A C A T A T A A T A T A T A C A C A C A C A C G T G T G T A T G T A A A T A T A T A C A T A C A C A C A T G T G T G T A T G T A C A T A T A A T A T A T
RERG
chr12:15,296,260
237M of 476M

NA12878 original read data

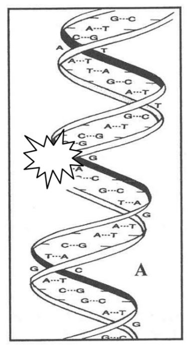
Multiple caller artifacts that are hard to filter out, since they are well supported by read data

Haplotype Caller (validated)

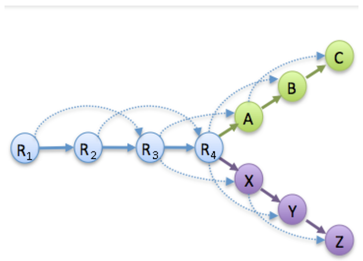
Haplotype Caller is more accurate than the Unified Genotyper



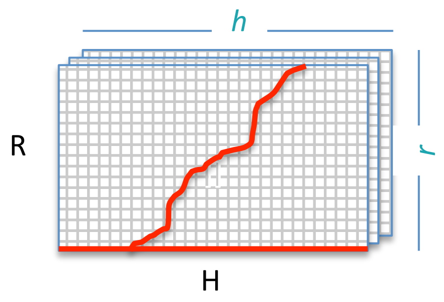
However the Haplotype Caller is more CPU intensive



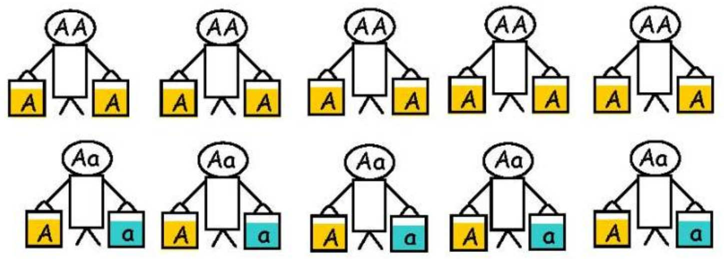
1. **Active region traversal** identifies the regions that need to be reassembled



2. **Local de-novo assembly** builds the most likely haplotypes for evaluation



3. **Pair-Hmm evaluation** of all reads against all haplotypes
(scales exponentially)



4. **Genotyping** using the exact model

Pair-HMM is the biggest culprit for the low performance

Stage	Time	Runtime %
Assembly	2,598s	13%
Pair-HMM	14,225s	70%
Traversal + Genotyping	3,379s	17%

NA12878 80xWGS chromosome 20 haplotype caller run
Chr20 time: 5.6 hours
WGS time: 7.6 days

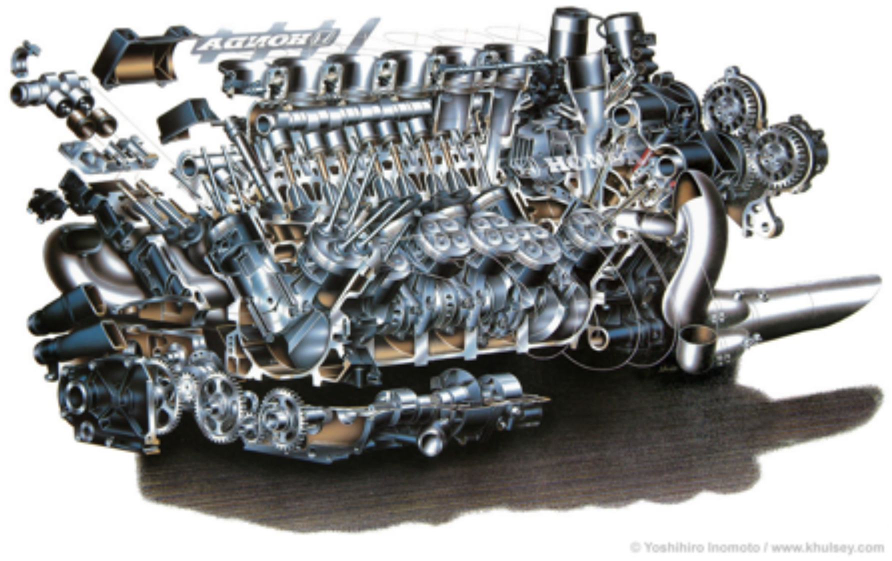
How we can improve performance?

1. Distributed parallelism: Queue/MapReduce
2. Alternative way to calculate likelihoods.
3. Heterogeneous parallel compute:
    - ❖ GPU, FPGA and Vectorization.
4. Joint-calling with incremental single sample discovery.

In memory parallelism + map/reduce parallelism with queue

PARALLELISM SUPPORT IN THE GATK TODAY

The GATK is actually two different beasts



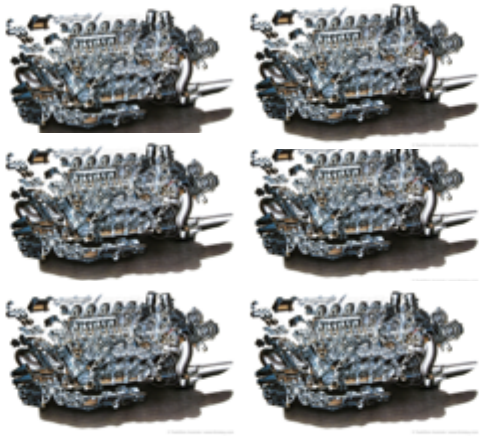
Engine

Takes care of the input and output. Preprocess and organizes a traversal system for the walkers

Walkers

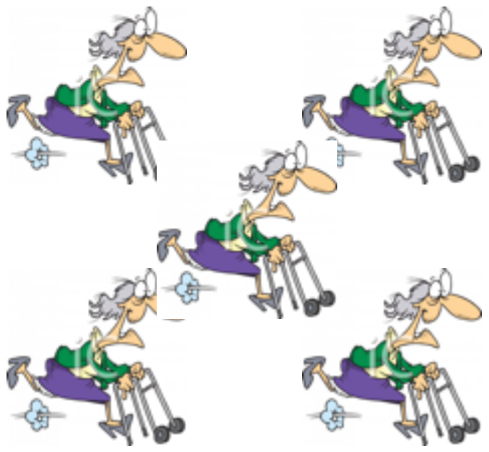
Sees the genome in an organized fashion and applies an algorithm to it

The three ways to parallelize the GATK



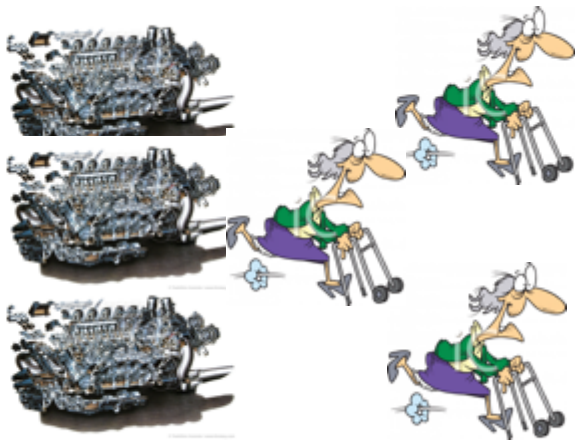
Parallelizing at the **engine** level to process different parts of the genome at the same time.

-nt



Parallelizing at the **walker** level to speed up the processing of each individual region of the genome.

-nct



Spawn many **instances** of the GATK to work on separate (arbitrary) parts of the genome at the same time.

Queue/MapReduce

This is not really a solution, but a last resort that we use routinely to make calls today

How we can improve performance?

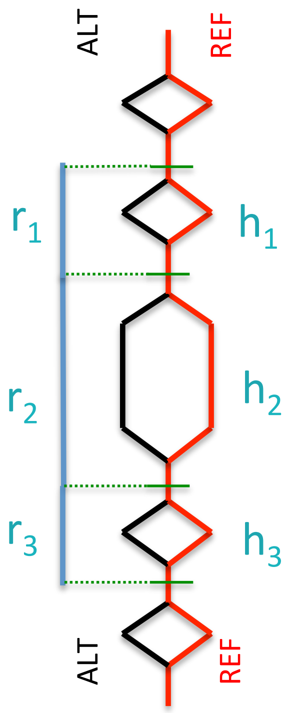
1. ~~Distributed parallelism: Queue/MapReduce~~
2. Alternative way to calculate likelihoods.
3. Heterogeneous parallel compute:
    - GPU, FPGA and Vectorization.
4. Joint-calling with incremental single sample discovery.

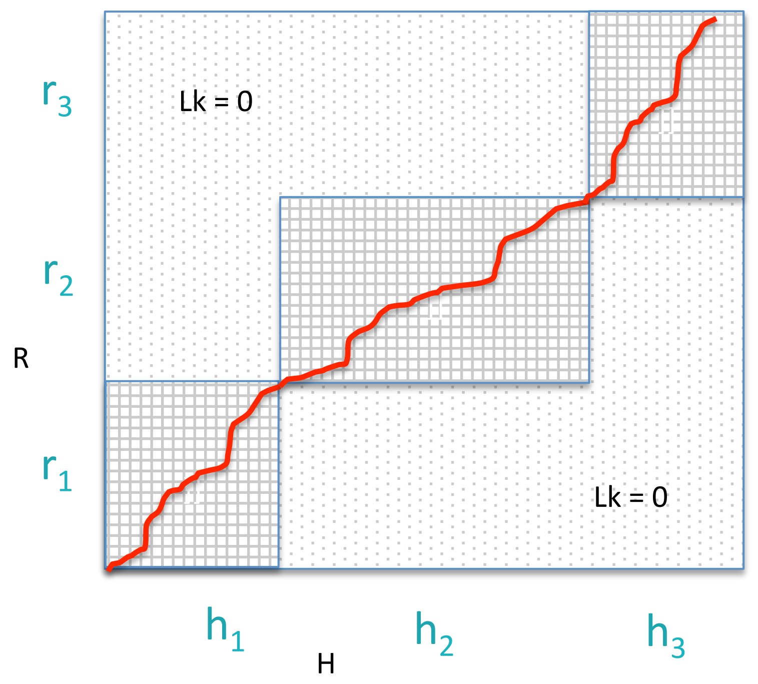
Reducing the number of times we need to run the pair-HMM

GRAPH BASED LIKELIHOODS

Calculating genotype likelihoods straight from the assembly graph reduces pair-HMM usage

Mapping each read to the haplotype assembly graph we can constrain the underlying pair-HMM to avoid quasi-zero likelihood unrealistic alignments.





The resulting sub-problems become modules that can be reused across haplotypes that share sub-paths in the graph.

Speed-up can be quite significant depending on data size and variation present.

Variation	*Civar*	PairHMM (ms)	GraphBased (ms)	Speed-up
1 SNP	**1T**	8139	493	16x
1 short ins.	**3I**	10785	485	22x
1 long ins.	**30I**	11249	522	21x
1 short del.	**3D**	10649	490	21x
1 long del.	**30D**	10212	546	18x
1 SNP 1 ins. close by	**1T*3=3I**	21552	584	36x
1 SNP 1 ins. far away	**1T*3I**	21420	626	34x
5 close by SNPs	**1T8=1T8=1T8=1T8=1T**	83035	3064	27x
5 far away SNPs	**1T*1T*1T*1T*1T**	57346	1332	43x

`Invariants: kmerSize = 10, readCount = 10000, readLength = 100, regionSize = 300`

First implementation just 4x speed-up but can (and will) be improved

How we can improve performance?

1. ~~Distributed parallelism: Queue/MapReduce~~
2. Alternative way to calculate likelihoods.
3. Heterogeneous parallel compute:
   - GPU, FPGA and Vectorization.
4. Joint-calling with incremental single sample discovery.

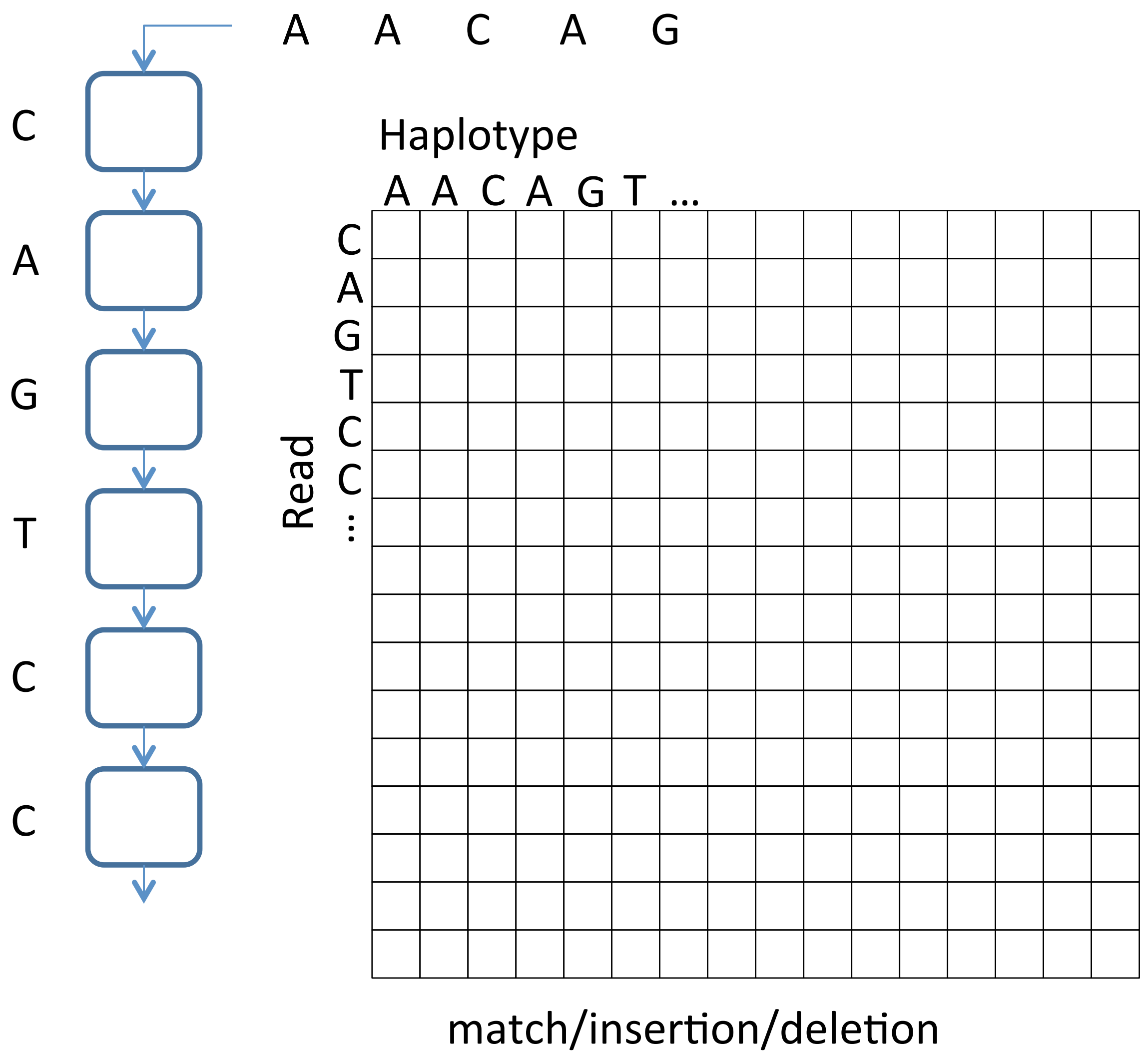
GPUs, FPGA and vectorized implementations of the pair-HMM

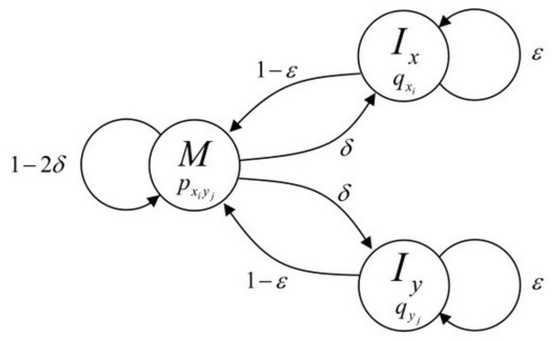
ALTERNATIVE PLATFORMS

Alternative implementations in the GATK

- **FPGA (Convey Computer)**
  Implemented by Scott Thibault from Green Mountain Computing Systems
  - Fast I/O with bi-directional data bus, somewhat large shared memory
  - Highly parallelizable (hundreds of processing elements)
- **GPU (NVidia CUDA)**
  Implemented in collaboration with Diego Nehab from IMPA-RJ
  - Fast I/O but limited memory with thousands of cores in one board
  - Especially fast float precision calculations
- **Vectorized (AVX)**
  Implemented in collaboration with Intel Corp
  - In processor, no extra I/O, already available in most computers
  - up to 32 512-bit registers per CPU

A parallelized version of the Pair-HMM





PairHMM Performance comparison

Data: NA12878 80xWGS chromosome 20

TECH	Hardware	Runtime (seconds)	Improvement (fold)
AVX	Intel Xeon 24-core	15	720x
GPU	NVidia Tesla K40	160	67x
GPU	NVidia GeForce GTX Titan	161	67x
GPU	NVidia GeForce GTX 480	190	56x
GPU	NVidia GeForce GTX 680	274	40x
GPU	NVidia GeForce GTX 670	288	38x
AVX	Intel Xeon 1-core	309	35x
FPGA	Convey Computers HC2	834	13x
-	C++ (baseline)	1,267	9x
-	Java (gatk)	10,800	-

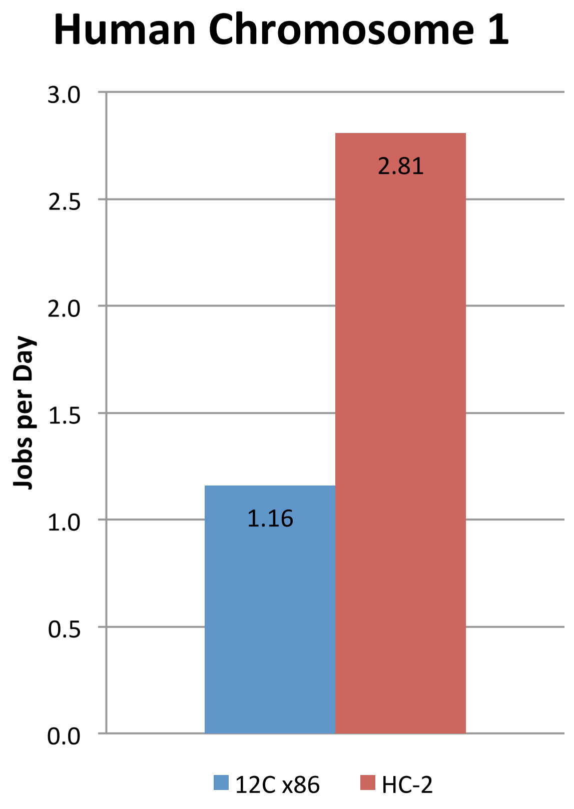
GATK engine is not ready to leverage the massive parallelism

Full Haplotype Caller run on 80xWGS PCR-Free NA12878:

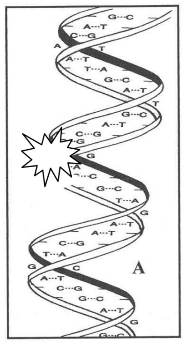
- 13 days on single CPU (java)
- 3.5 days on Convey HC-2

3.7x improvement instead of the expected 13x fold improvement

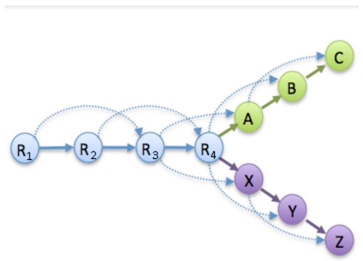
- GATK engine does not keep the "pipe full" for the alternative implementations of the pair-HMM to shine
- A different approach is necessary (GATK 3.x target release)



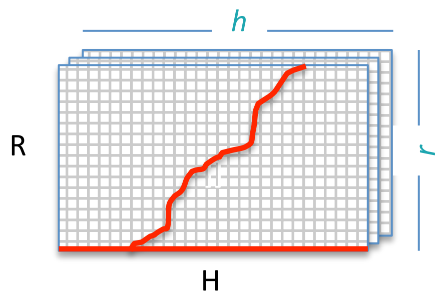
Synchronous traversal is to blame



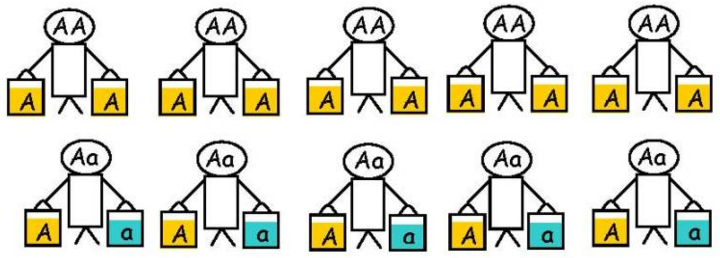
1. **Active region traversal** identifies the regions that need to be reassembled



2. **Local de-novo assembly** builds the most likely haplotypes for evaluation



3. **Pair-Hmm evaluation** of all reads against all haplotypes
(scales exponentially)



4. **Genotyping** using the exact model

Summary

The GSA team is continuously revising and updating the best practices for variant calling in order to enable the large scale of tomorrow's medical genetics needs.

Assembly genotyping will work in combination with any other accelerations made to the pair-HMM.

GPUs and AVX are the most promising platforms for large scale projects or pipelines in need of very fast turnaround (e.g. diagnostics)

The GATK will need an asynchronous engine to enable the full potential of these platforms

Available in the next major version release of the GATK

THE GATK TEAM NEEDS YOU

Talk to me for more information or email downing@broadinstitute.org