Introducing the Practical Haplotype Graph Version 2: A Streamlined and Simple Pangenome System

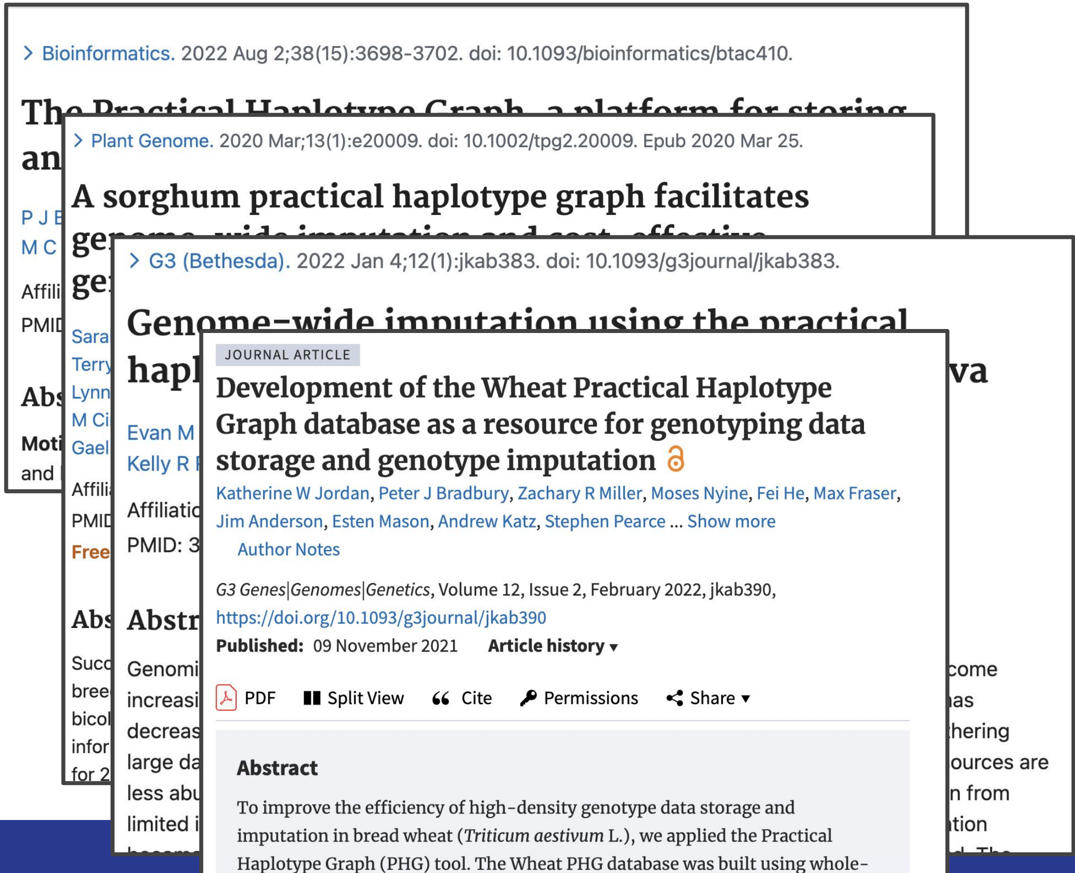
Zachary Miller
Institute for Genomic Diversity
Cornell University

Introduction

- Lower cost of sequencing - but still cost prohibitive to sequence entire plant breeding populations at high depths
- Reference Quality Assemblies are becoming widely available for a number of staple food crops with more on the way
- Can we use the diversity captured by a collection of assemblies (pangenome) to impute low cost short read genotype data better than traditional reference alignment techniques?

Introducing the Practical Haplotype Graph(PHG)

- Started development in 2017
- Initial success in building PHGs
    - Imputation
    - Genomic Selection
    - General data storage
- But had some issues
    - Utilized a custom Postgres DB
    - Certain components slow
    - Overly Parameterized
    - User Interface hard to use
    - Poor/Out of date documentation



> Bioinformatics. 2022 Aug 2;38(15):3698-3702. doi: 10.1093/bioinformatics/btac410.

The Practical Haplotype Graph, a platform for storing

an

P J E
M C

Affili

PMID

Abs

Moti
and

> Plant Genome. 2020 Mar;13(1):e20009. doi: 10.1002/tpg2.20009. Epub 2020 Mar 25.

A sorghum practical haplotype graph facilitates
genome-wide imputation and cost-effective
ge

Sara
Terry
Lynn
M Ci
Gael

Affili
PMID
Free

Abs

Succ
bree
bicol
infor
for 2

> G3 (Bethesda). 2022 Jan 4;12(1):jkab383. doi: 10.1093/g3journal/jkab383.

Genome-wide imputation using the practical
hapl ... va

Evan M
Kelly R

Affiliatio
PMID: 3

Abstr

Genomi ... come
increasi ... has
decreas ... hering
large da ... ources are
less abu ... n from
limited i ... tion

JOURNAL ARTICLE

Development of the Wheat Practical Haplotype Graph database as a resource for genotyping data storage and genotype imputation

Katherine W Jordan, Peter J Bradbury, Zachary R Miller, Moses Nyine, Fei He, Max Fraser, Jim Anderson, Esten Mason, Andrew Katz, Stephen Pearce ... Show more

Author Notes

G3 Genes|Genomes|Genetics, Volume 12, Issue 2, February 2022, jkab390, https://doi.org/10.1093/g3journal/jkab390

Published: 09 November 2021 **Article history ▾**

PDF Split View Cite Permissions Share ▾

Abstract

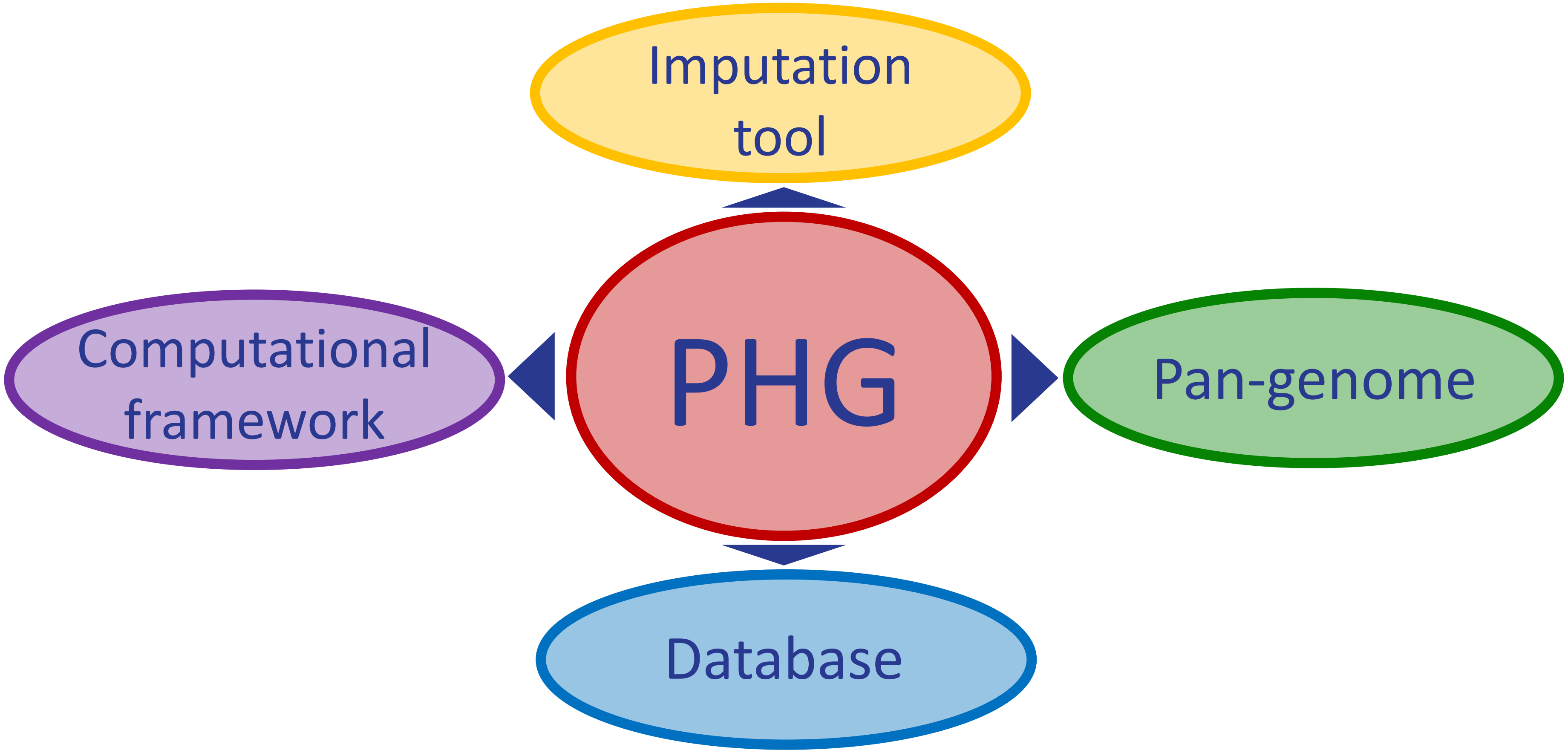
To improve the efficiency of high-density genotype data storage and imputation in bread wheat (*Triticum aestivum* L.), we applied the Practical Haplotype Graph (PHG) tool. The Wheat PHG database was built using whole-

PHGv2 - Works to address these issues

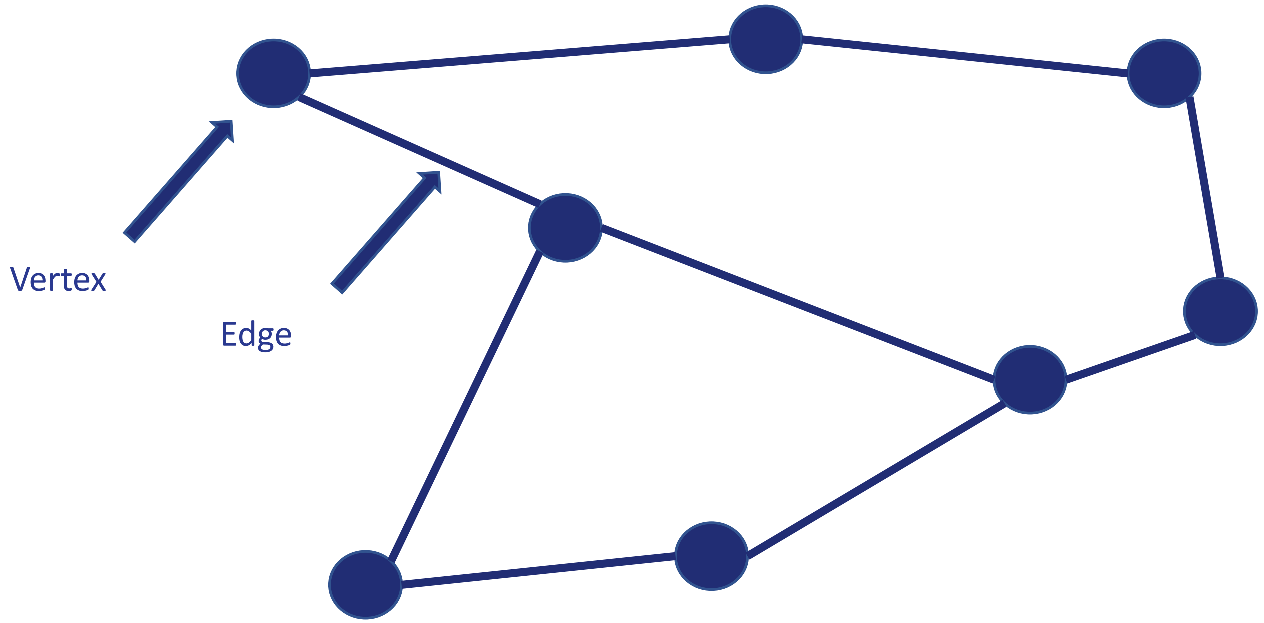
- Fast
- Easy to use
- Clear, Concise, and Up-to-date Documentation
- Integrate standard software development practices
  - Continuous Integration - Code is tested early, often and automatically. >80% of code is covered by unit tests
  - Continuous Delivery - Once code is reviewed and merged in a new build and release of the package happens automatically
- Utilize state of the art community tools as much as possible
  - Anchorwave - aligner
  - tileDB - Genotype(VCF) storage
  - Assembled Genomes Compressor(AGC) - Sequence Storage

What do we mean by Practical Haplotype Graph?

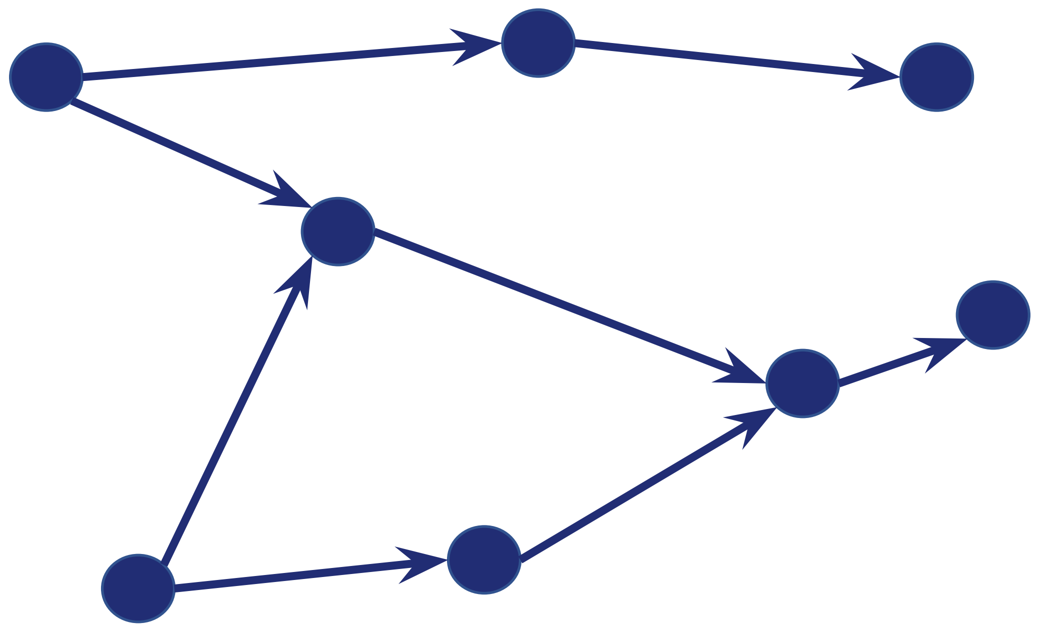
- Practical
  - Keep it simple!
  - Biology produces genomes with consistent patterns
  - Somewhat Conserved Genes + Intergenic Regions with tremendous variation
  - Slice the genome at these conserved boundaries -> Reference Ranges
    - Simplifies the pangenome representation
- Haplotype
  - A set of DNA variations, or polymorphisms, that tend to be inherited together
  - Store both sequence and Variants
- Graph


Imputation tool
Computational framework
PHG
Pan-genome
Database

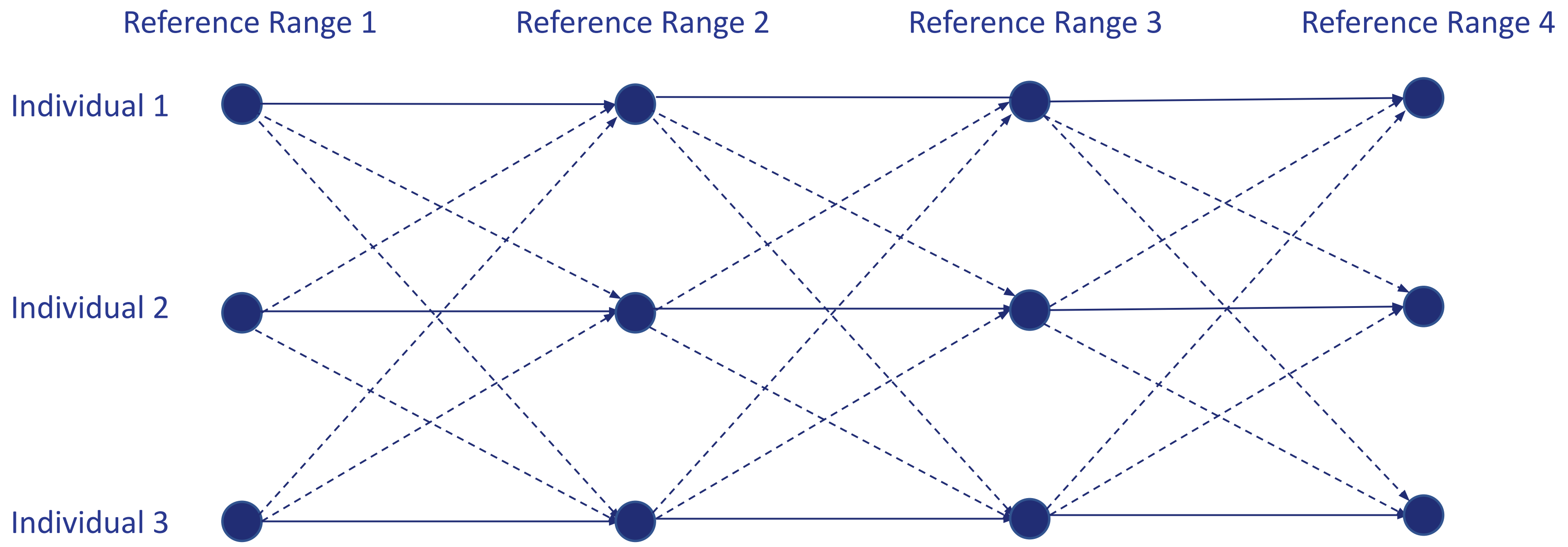
What is a graph?



A directed acyclic graph (DAG)



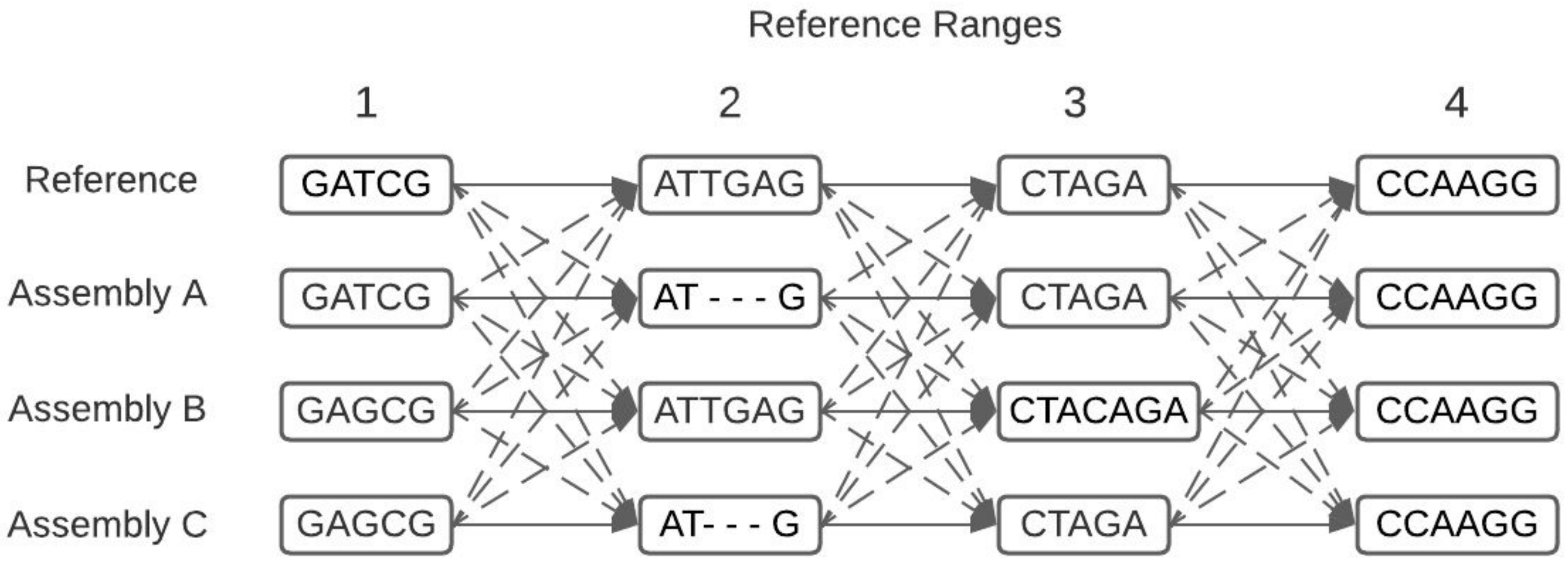
PHG - Trellis Graph

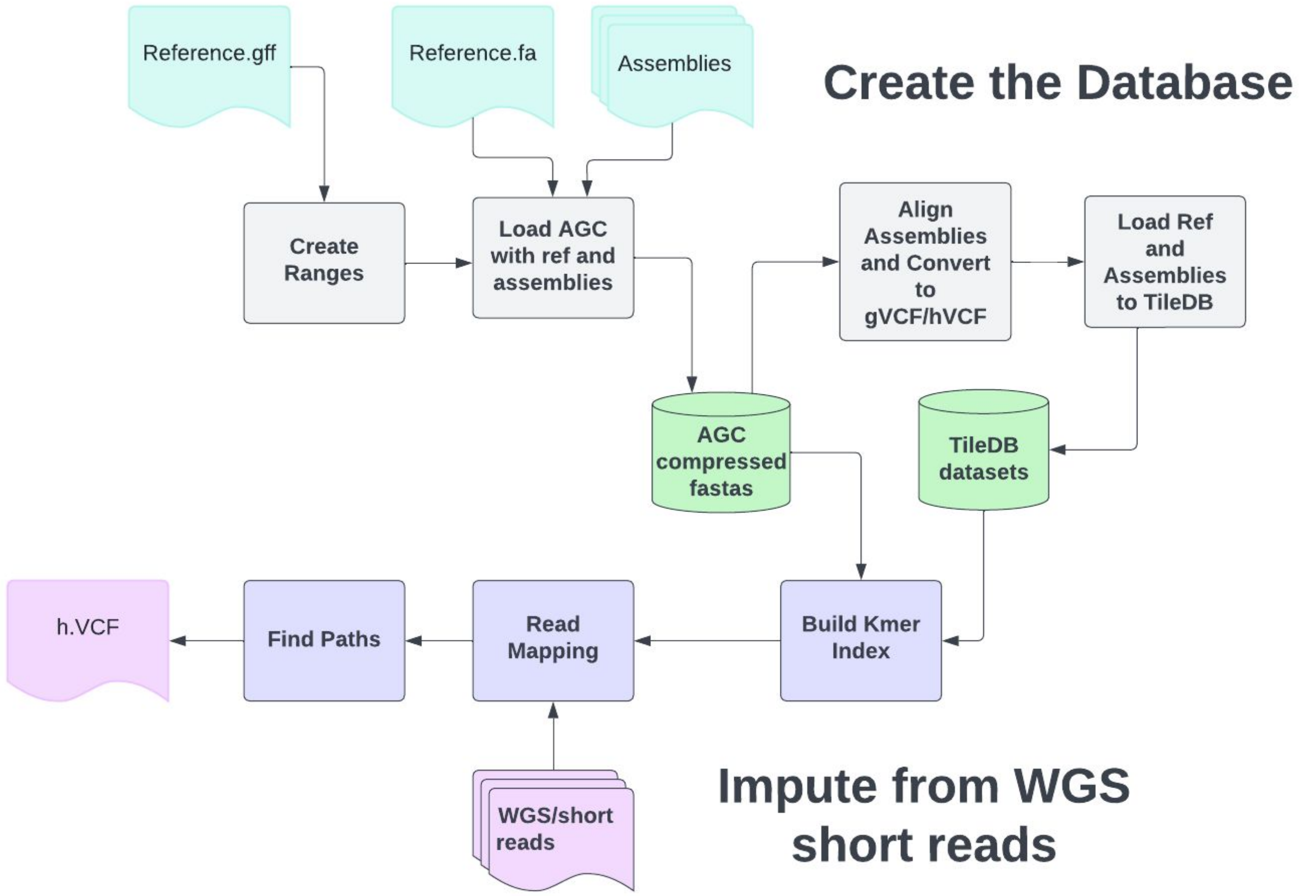


Terms

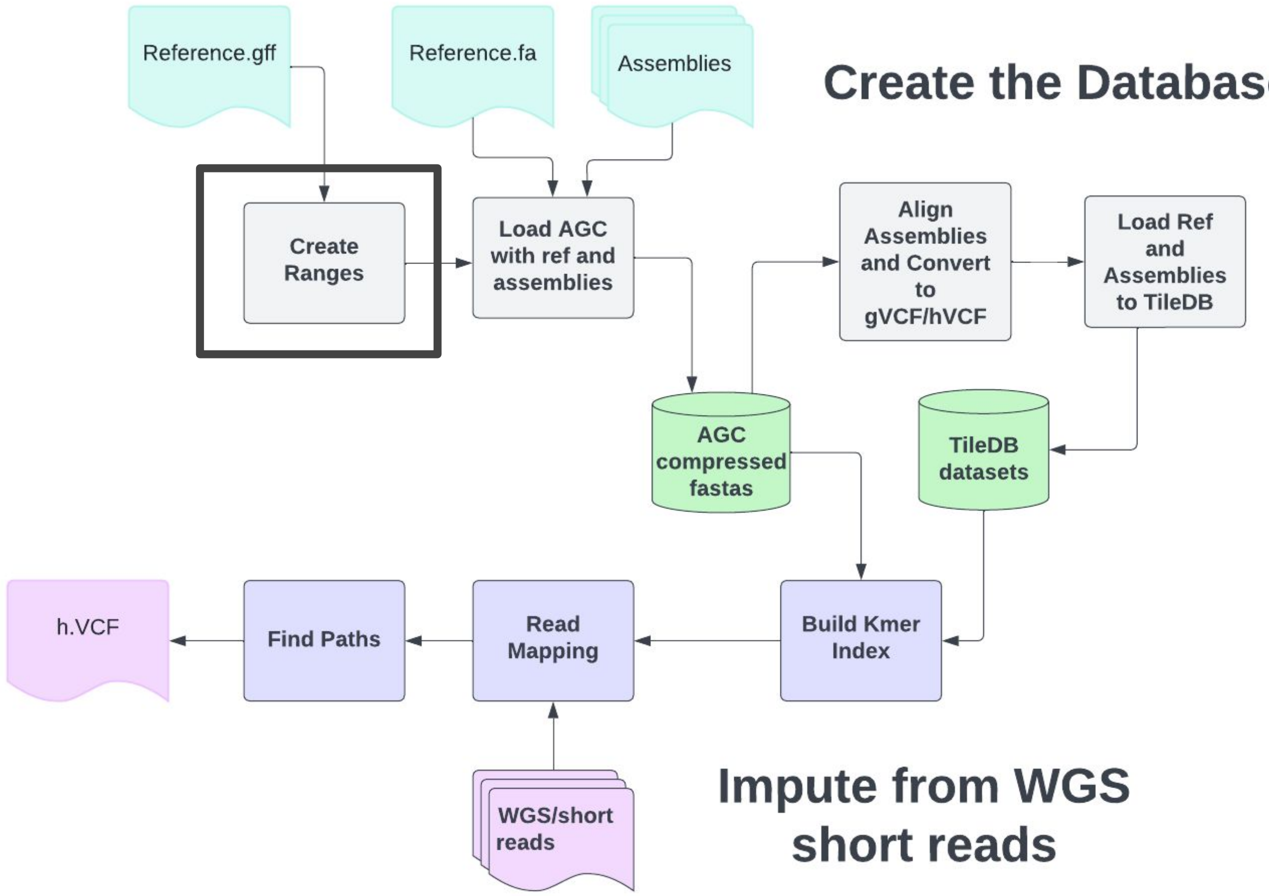
- Reference Range
    - A Segment of the Reference Genome
    - Typically recommended that range Start and End are conserved
    - Common way to define these are genic boundaries
- Haplotype
    - A set of DNA variations, or polymorphisms, that tend to be inherited together
    - The PHG holds the following information for each haplotype
        - Variants - in gVCF file
        - Nucleotide Sequence - aligned to the reference for a specific Reference Range

PHG - Trellis Graph



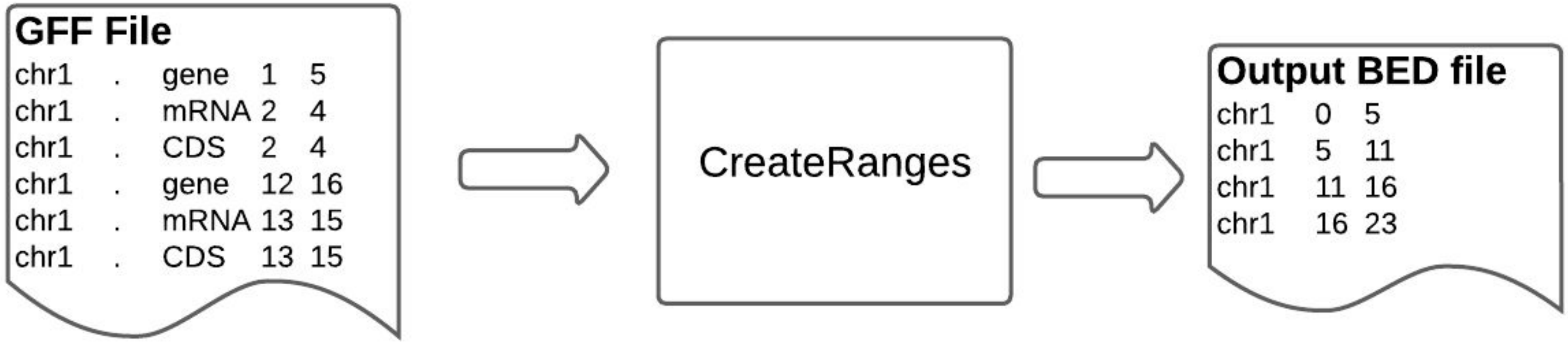

Reference.gff
Reference.fa
Assemblies
Create the Database
Create Ranges
Load AGC with ref and assemblies
Align Assemblies and Convert to gVCF/hVCF
Load Ref and Assemblies to TileDB
AGC compressed fastas
TileDB datasets
h.VCF
Find Paths
Read Mapping
Build Kmer Index
WGS/short reads
Impute from WGS short reads

Create the Database



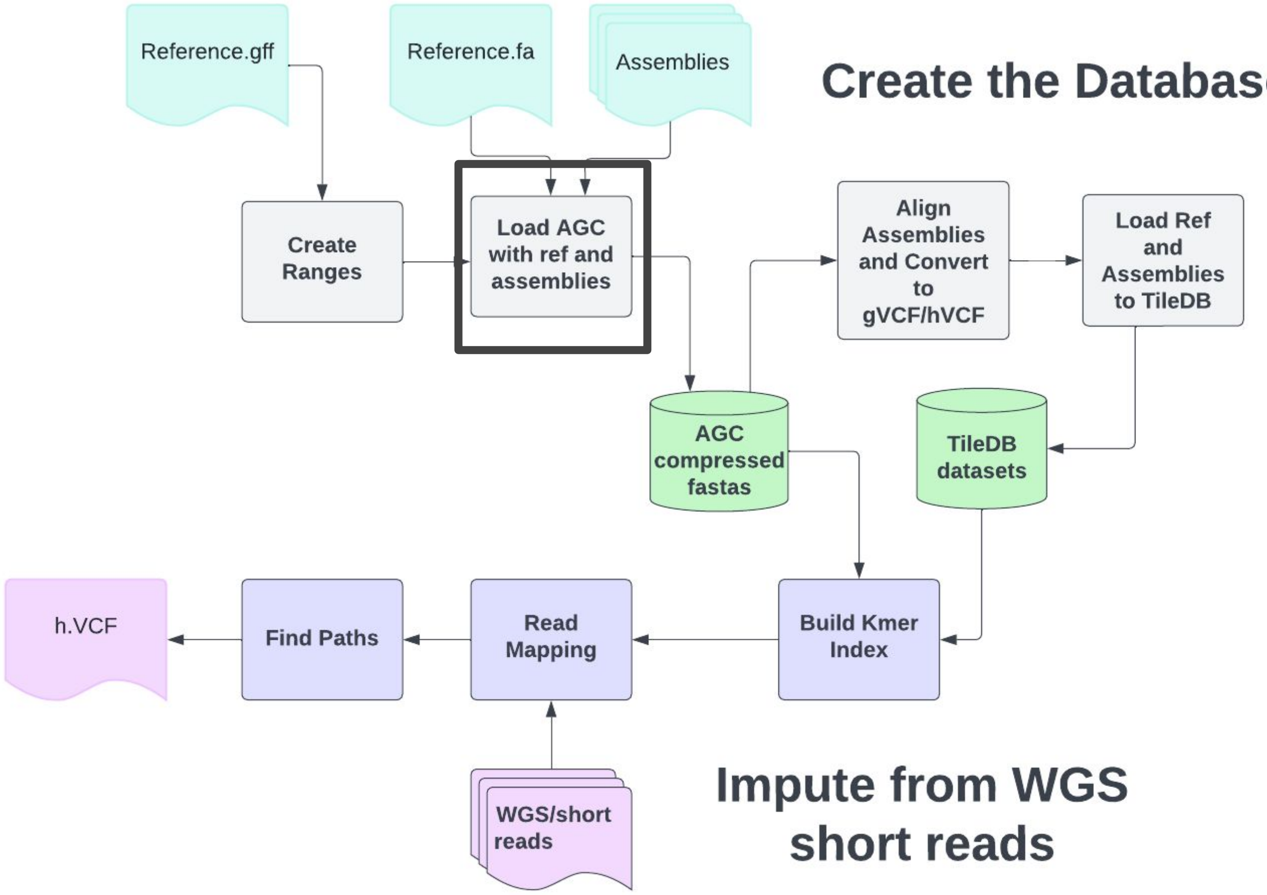
Impute from WGS short reads

Build a PHG - Create Ranges



```
>phg create-ranges --gff reference.gff --reference-file Reference.fasta
                   --output refRanges.bed
```

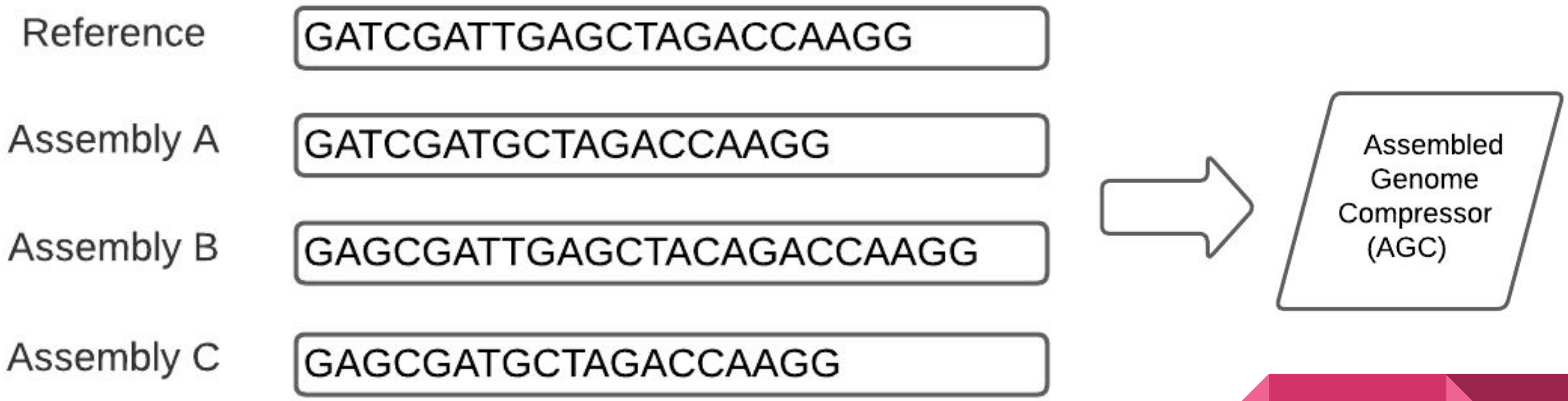
Create the Database



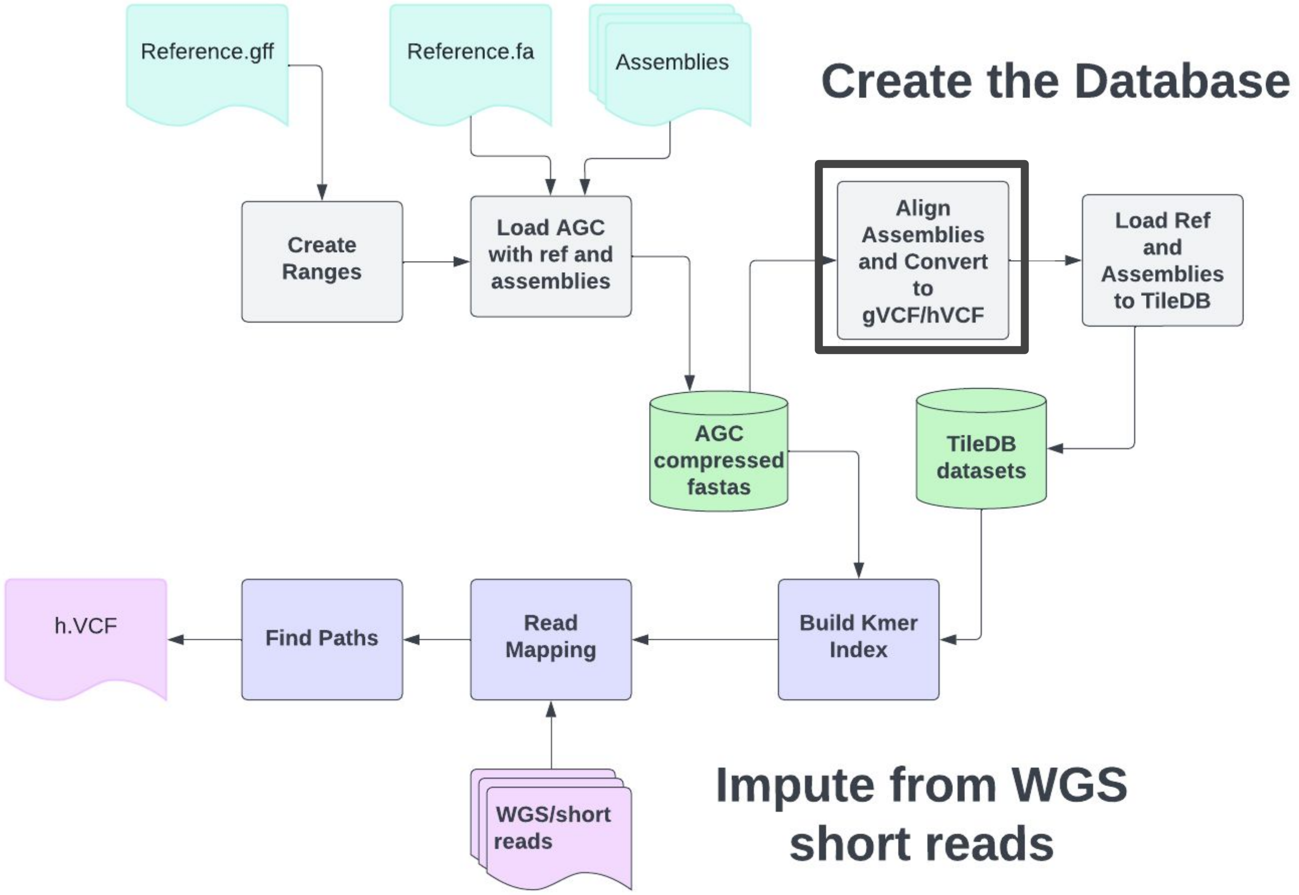
Impute from WGS short reads

Build a PHG

- We have a Reference and 3 Assemblies



```
>phg agc-compress --reference-file Ref.fa --fasta-list listOfFastas.txt
```

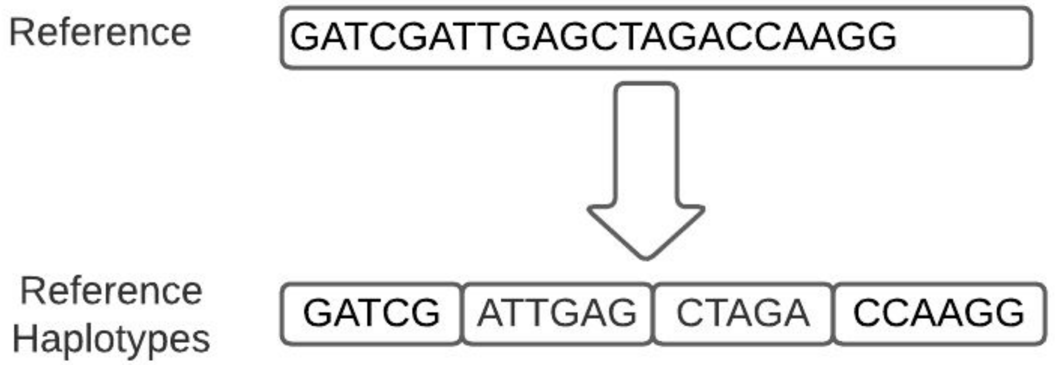


Create the Database

Impute from WGS short reads

Build a PHG: Build Reference Haplotypes

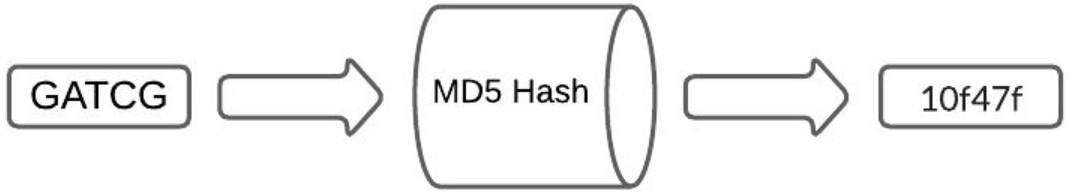
- Use Conserved Base Pairs to slice the genome
  - Can use genic boundaries from the GFF annotation



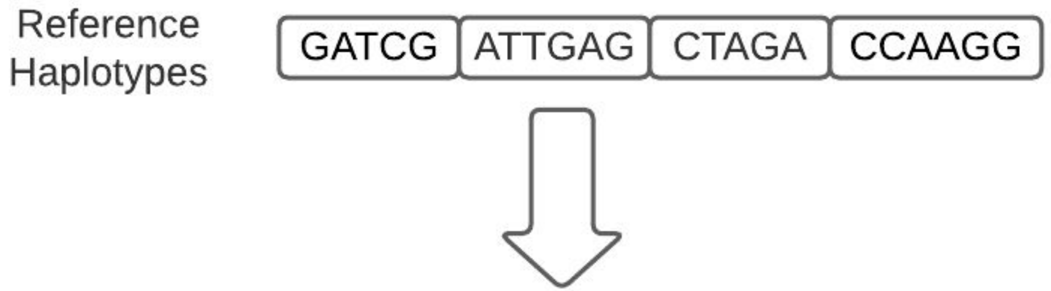
```
>phg create-ref-vcf --bed /my/bed/file.bed --reference-file Ref.fa
                         --reference-name Reference --output-dir /path/to/vcfs
```

hVCF - Simple Haplotype Storage Format

- We can store a set of haplotypes in a VCF based file format
- Key idea: hash the sequence to have a unique identifier
    - These identifiers can be stored as Symbolic alleles in a VCF file



hVCF - Simple Haplotype Storage Format



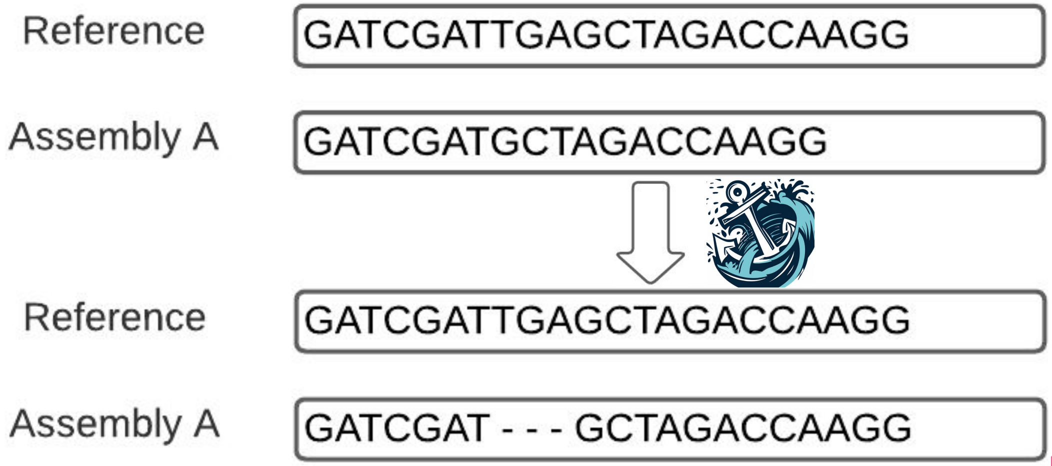
```
##ALT=<ID=10f47f,SampleName="Reference",Regions=chr1:1-5...
##ALT=<ID=7d046e,SampleName="Reference",Regions=chr1:6-11...
##ALT=<ID=9fb476,SampleName="Reference",Regions=chr1:12-16...
##ALT=<ID=1d471f,SampleName="Reference",Regions=chr1:17-22...
...
chr1  1     .    G    <10f47f>  END=5    GT   1/1
chr1  6     .    A    <7d046e>  END=11   GT   1/1
chr1  12    .    C    <9fb476>  END=16   GT   1/1
chr1  17    .    C    <1d471f>  END=22   GT   1/1
```

hVCF Benefits

- Small # of ‘variants’
  - Only number of Reference ranges ~100k
- VCF based
  - Community Standard
  - Easy to understand
  - Lots of tools out there to process and analyze the data
- Works with small and large genomes
  - Supports .csi indexing so big genomes like wheat(15-17 Gbp) work just fine
- Sequences can be reconstituted based on haplotype metadata
  - Verify by checking the ID against the hash
- Can load into TileDB

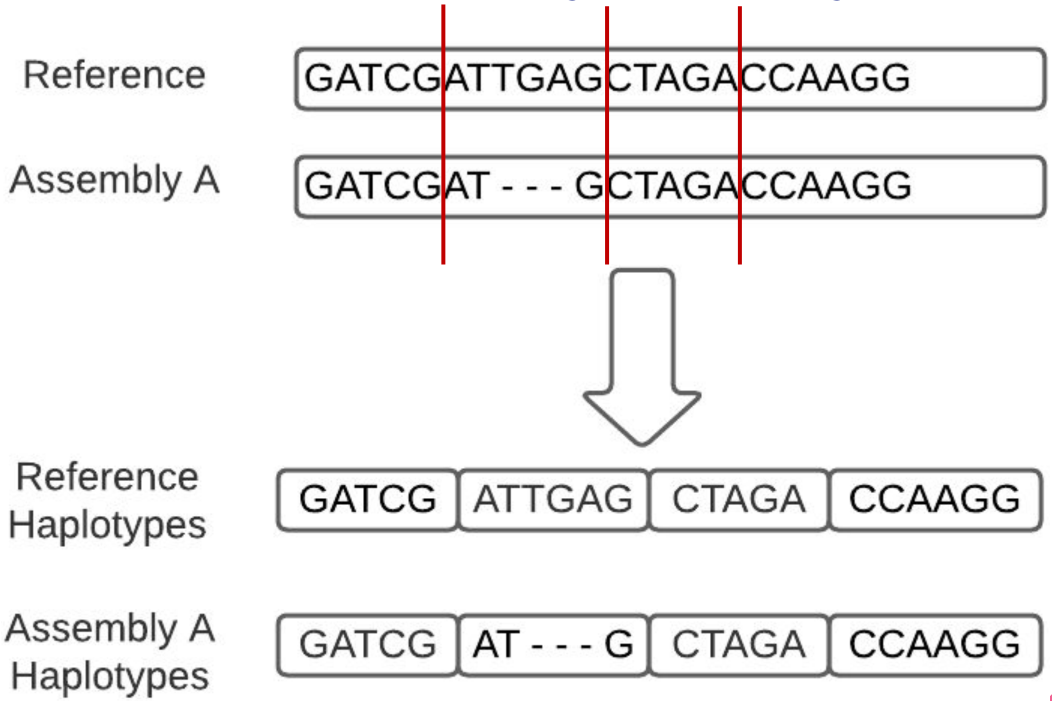
Build a PHG: Align Assemblies to Reference

- We wrap the Anchorwave aligner to do this accurately and efficiently



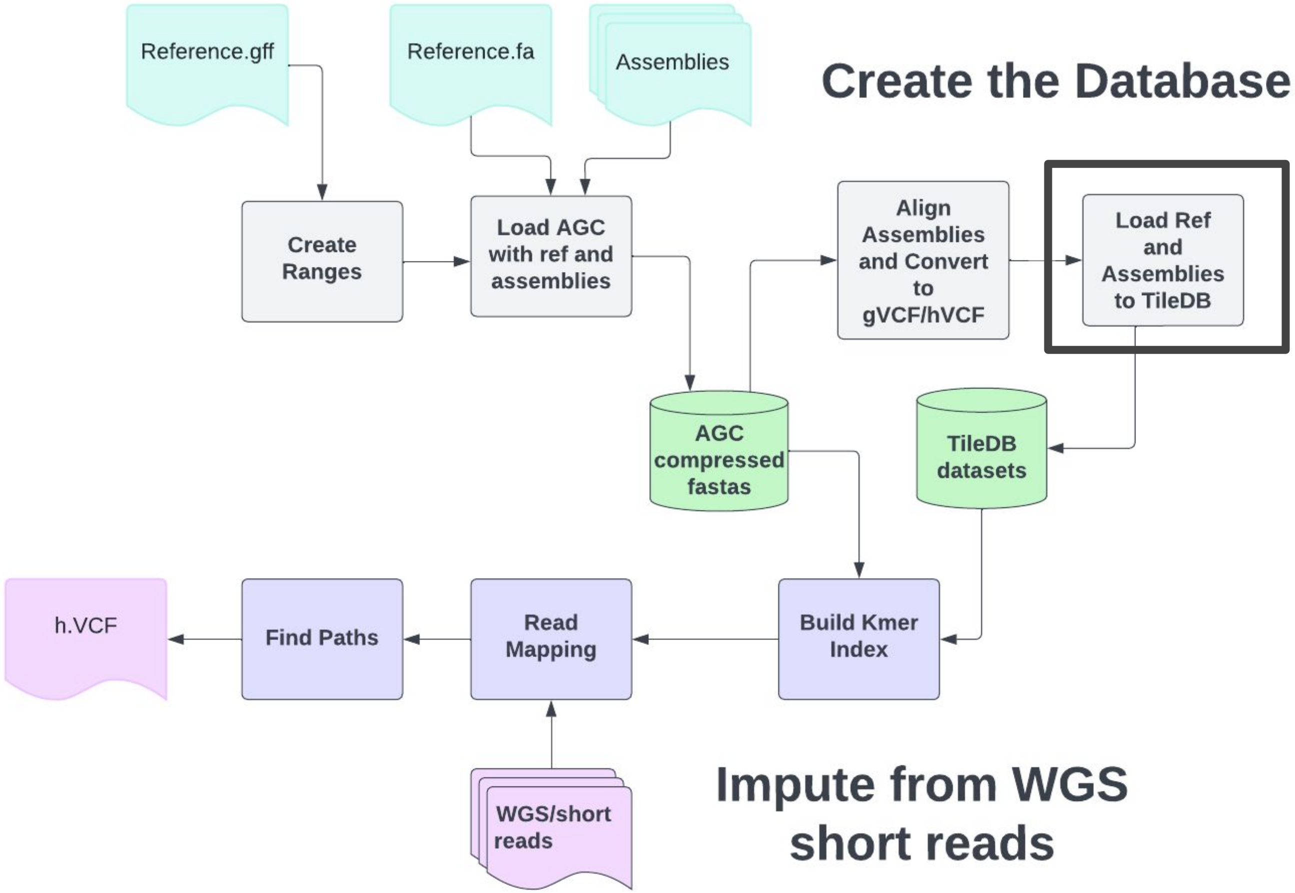
```
>phg align-assemblies --gff anchors.gff --reference-file Ref.fa
                          --a assembliesList.txt -o /path/for/alignment/files
```

Build a PHG: Build Assembly Haplotypes



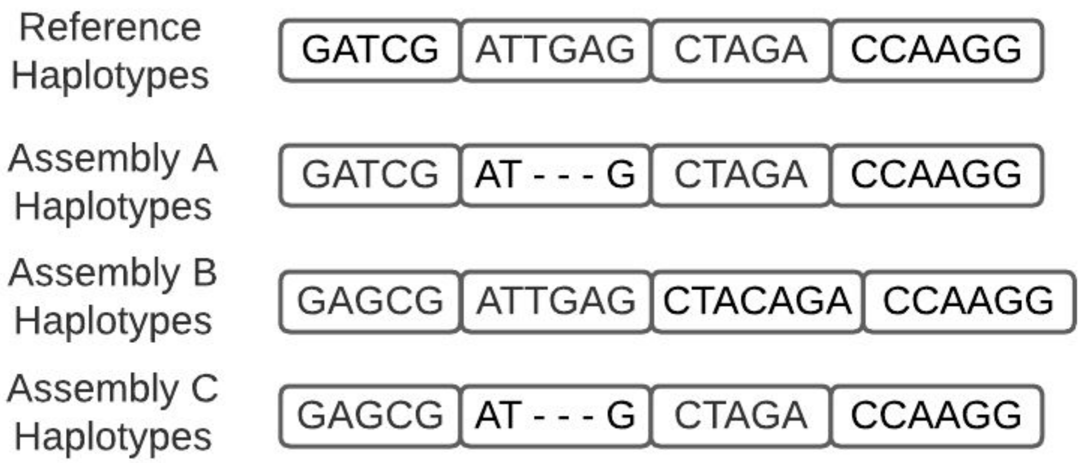
```
>phg create-maf-vcf --bed anchors.bed --reference-file Ref.fa
                  --maf-dir /path/for/alignment/files -o path/to/vcfs
```

Create the Database



Impute from WGS short reads

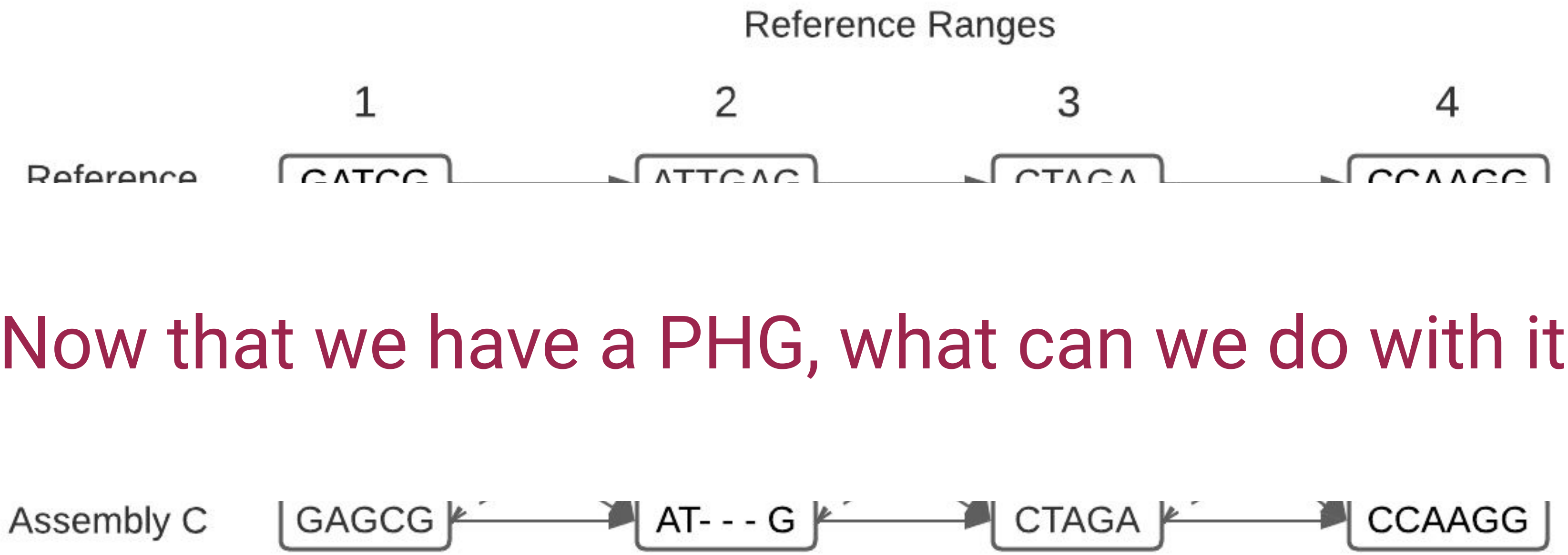
Build a PHG: Do all Assemblies + Load



[tile]DB

```
>phg load-vcf --vcf-dir /path/to/vcfs
```

PHG - Built!

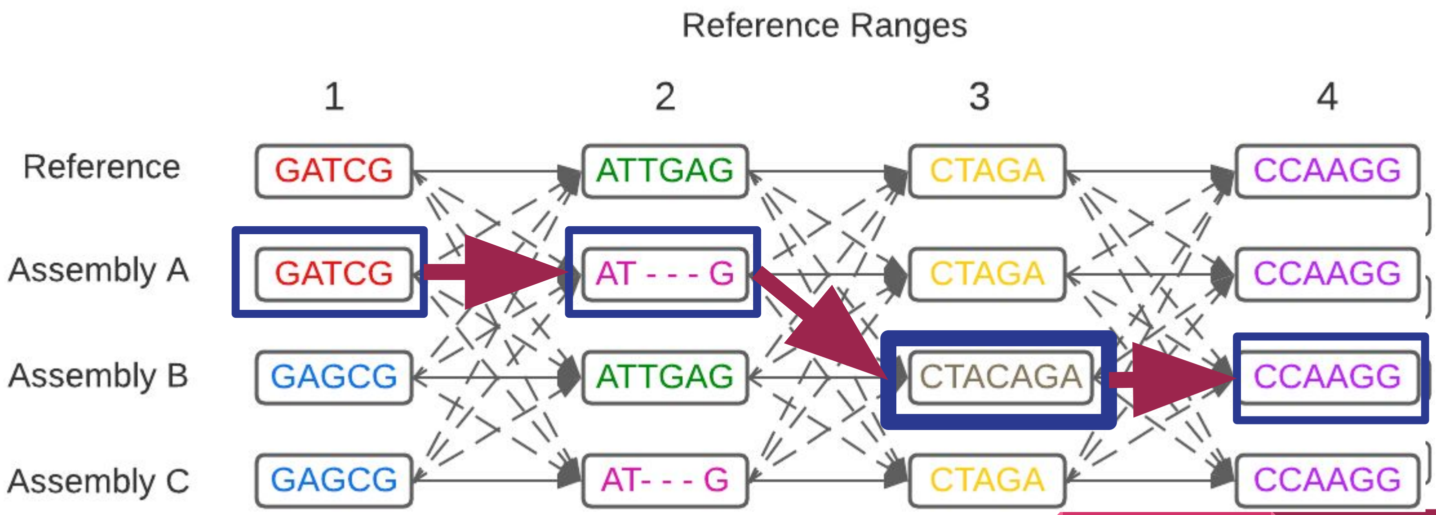


Now that we have a PHG, what can we do with it?

- Directed edges connect each haplotype with all haplotypes in next reference range
- Stronger weights are set for consecutive haplotypes of a given assembly

Imputation

>read1
ATC
>read2
ATG
>read3
CTA
>read4
CAG
>read5
AAG



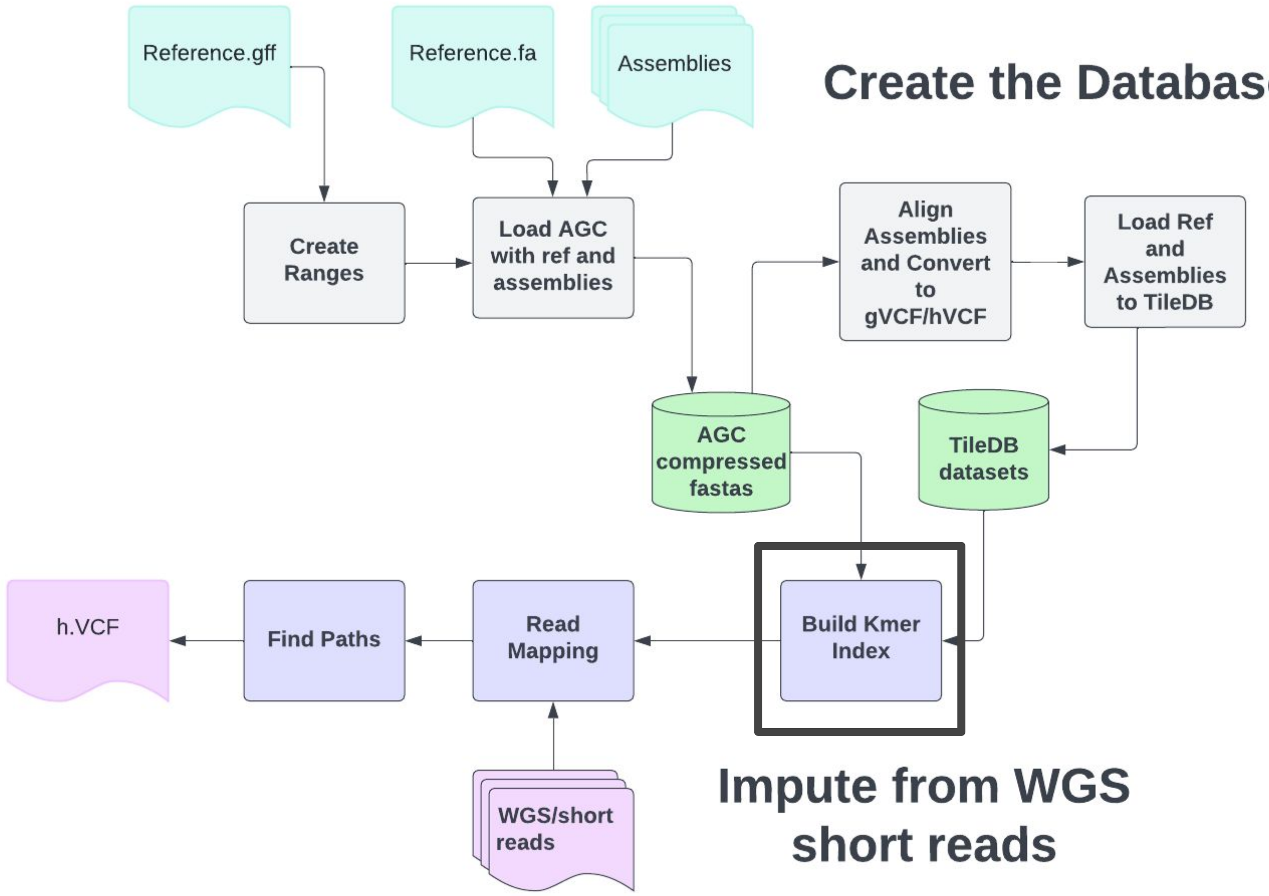
Imputation

- We can have 1000s of samples sequenced with low depth short reads
  - GBS, DaRTSeq, Skim Sequence, anything in a fastq
- 2 Step Process - per sample
  - Read Mapping
  - Path Finding
- Goal is to have this process take minutes

Imputation: Read Mapping

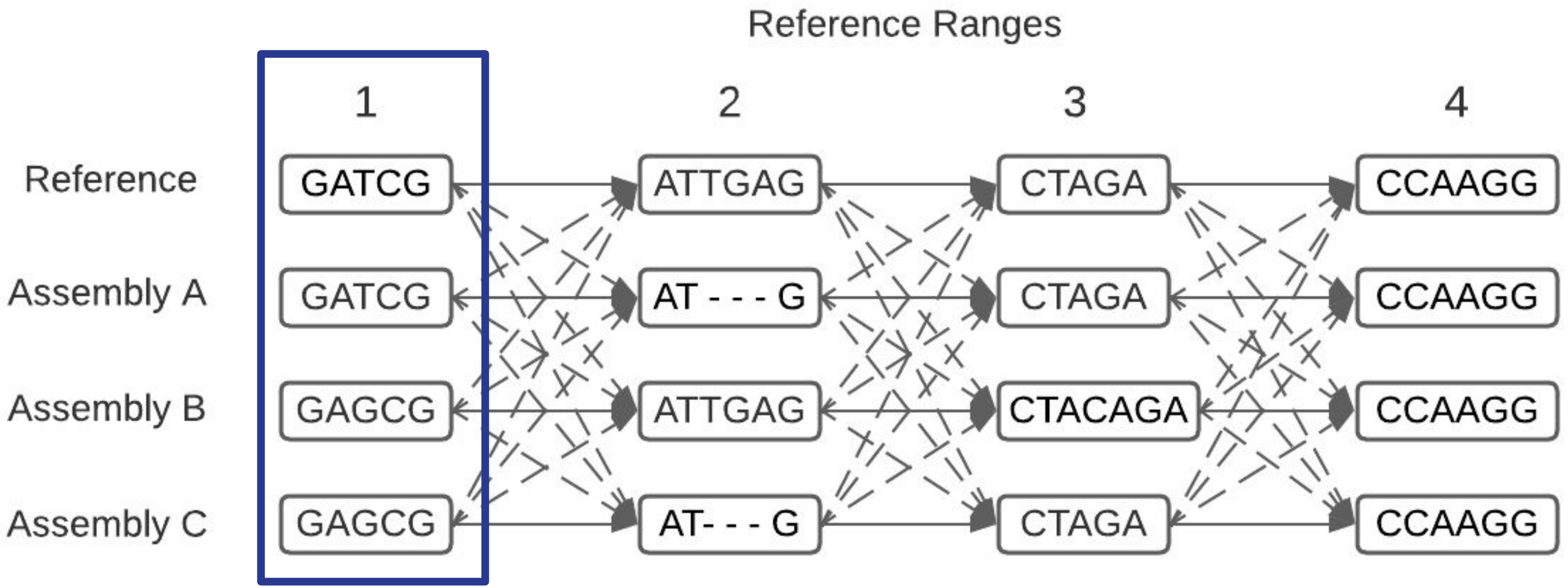
- Traditional Alignment tools like Minimap2 do work, but can we be more efficient?
- Aligning against the pangenome substantially increases resource requirements
- For this reason, we developed a Kmer based read mapping approach

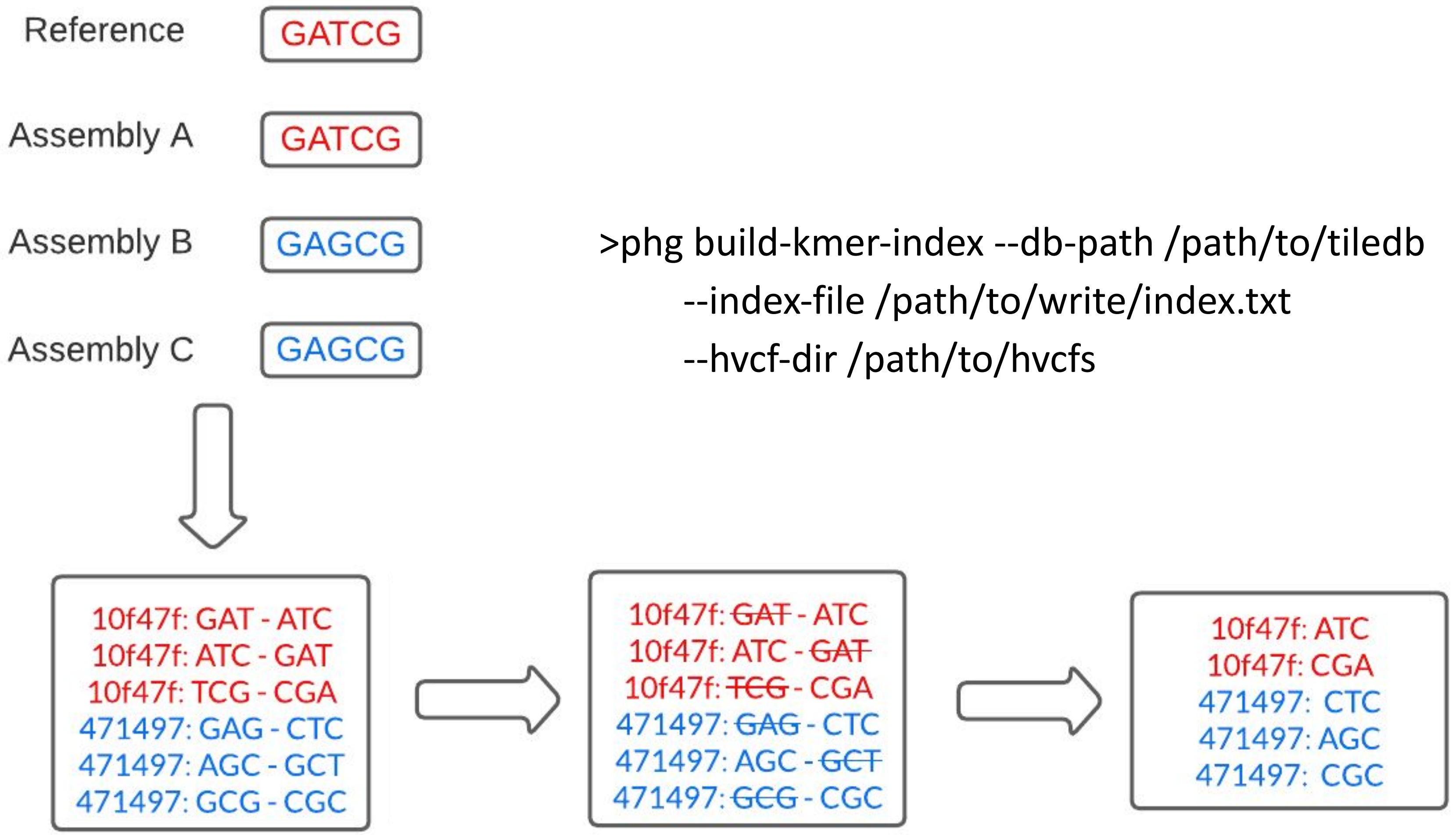
Create the Database



Impute from WGS short reads

Imputation: Index the Graph

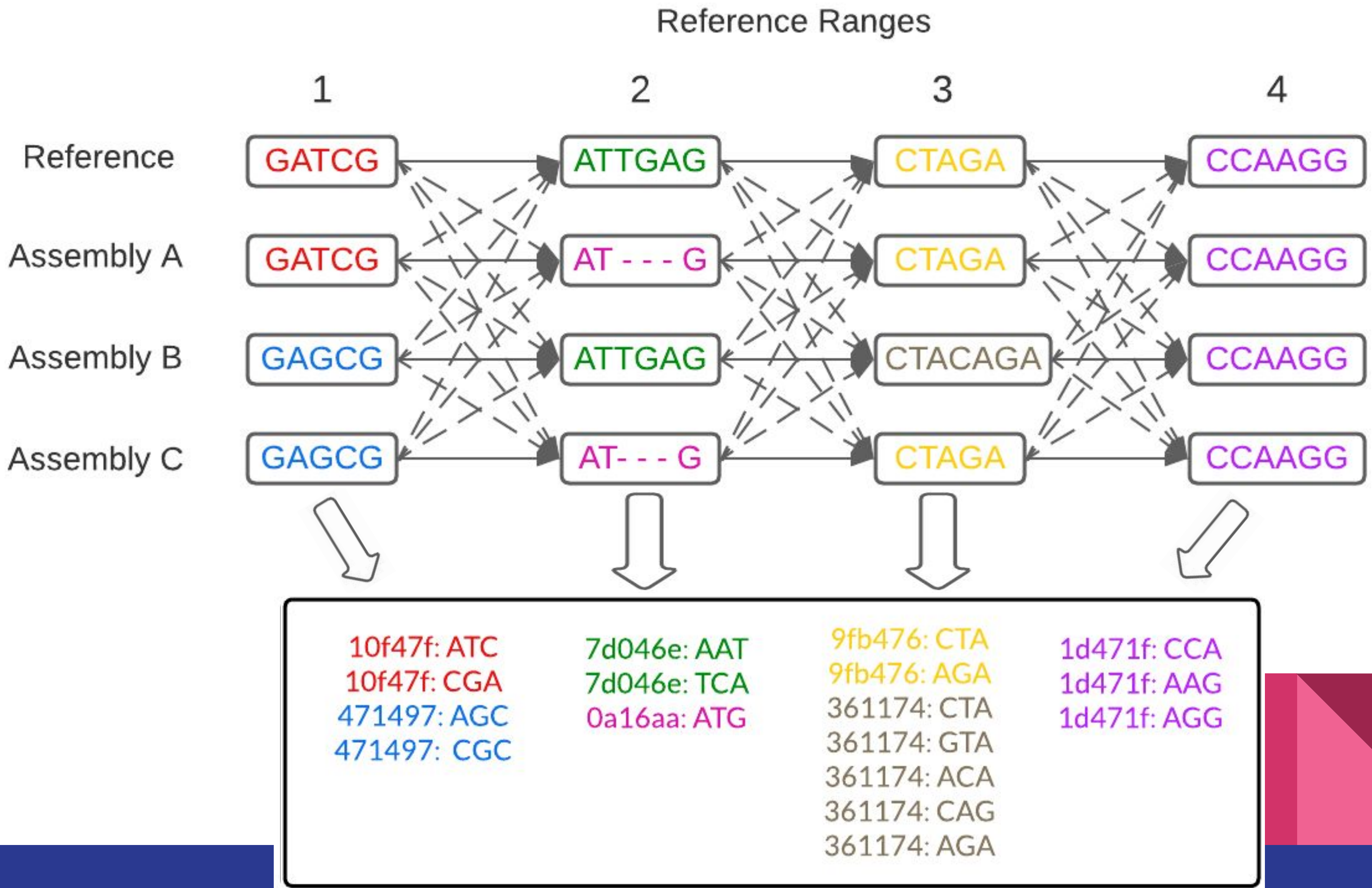




```
>phg build-kmer-index --db-path /path/to/tiledb
        --index-file /path/to/write/index.txt
        --hvcf-dir /path/to/hvcfs
```

Keep Lowest

Remove Duplicates

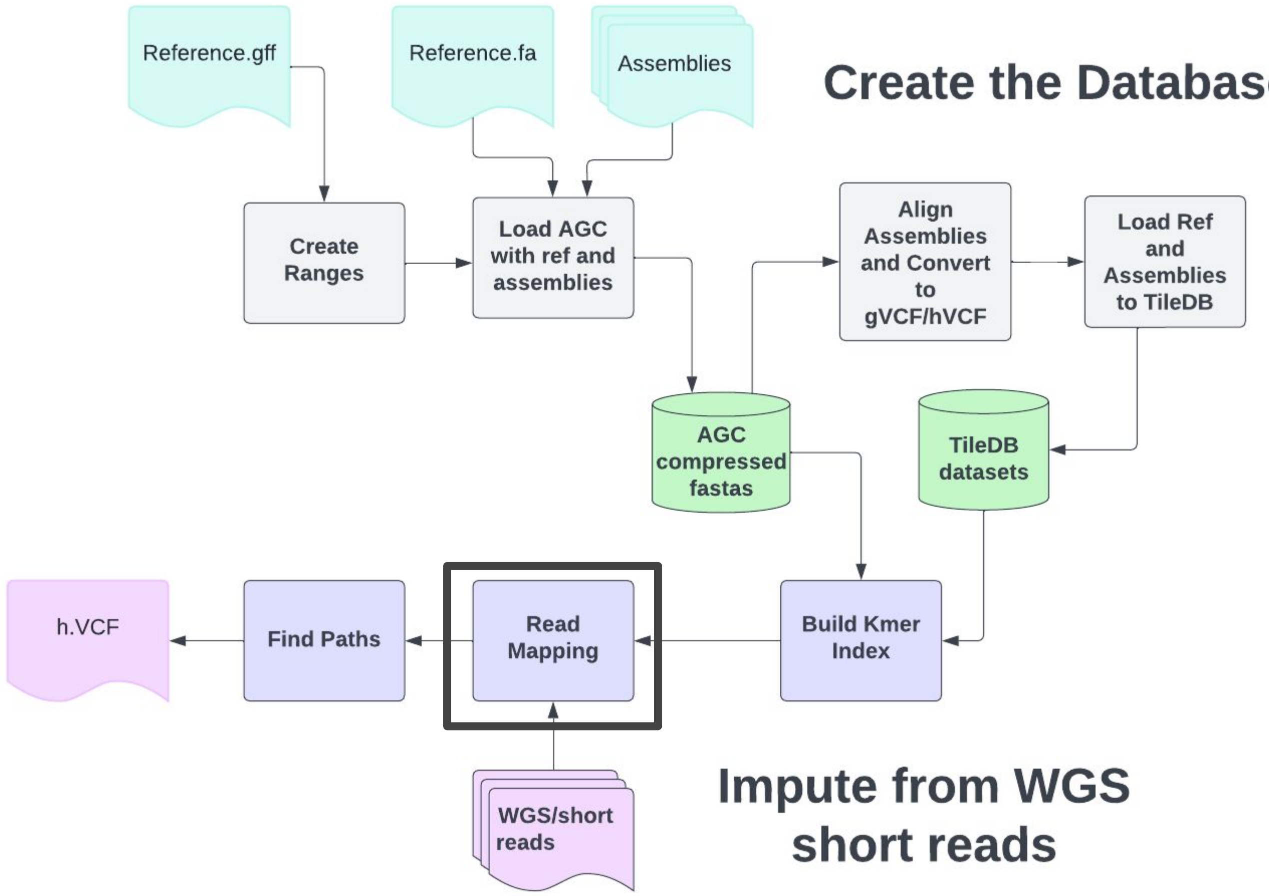


10f47f: ATC
10f47f: CGA
471497: AGC
471497: CGC

7d046e: AAT
7d046e: TCA
0a16aa: ATG

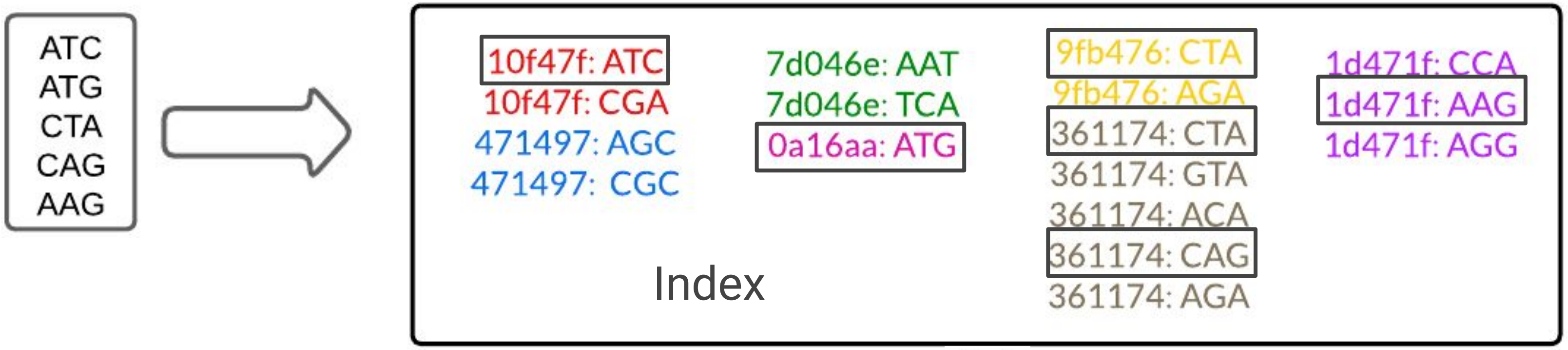
9fb476: CTA
9fb476: AGA
361174: CTA
361174: GTA
361174: ACA
361174: CAG
361174: AGA

1d471f: CCA
1d471f: AAG
1d471f: AGG



Create the Database

Impute from WGS short reads



Map Reads

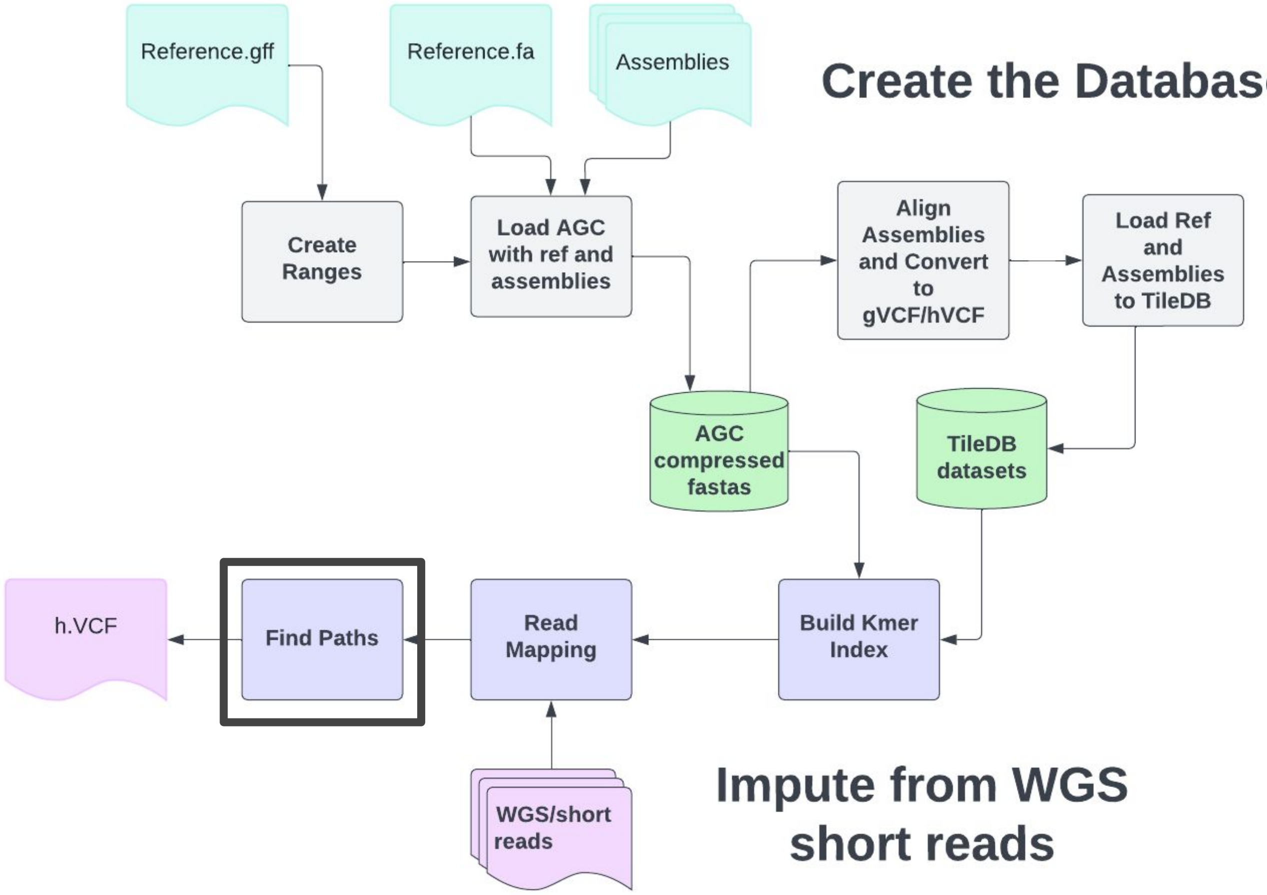
```
>phg map-kmers
        --hvcf-dir /path/to/hvcfs/
        --kmer-index /path/to/index
        --read-files file1.fq, file2.fq
        --output-dir /path/to/output
```

HapId Set	Counts
10f47f	1
0a16aa	1
9fb476, 361174	1
361174	1
1d471f	1

Kmer Read Mapper Performance

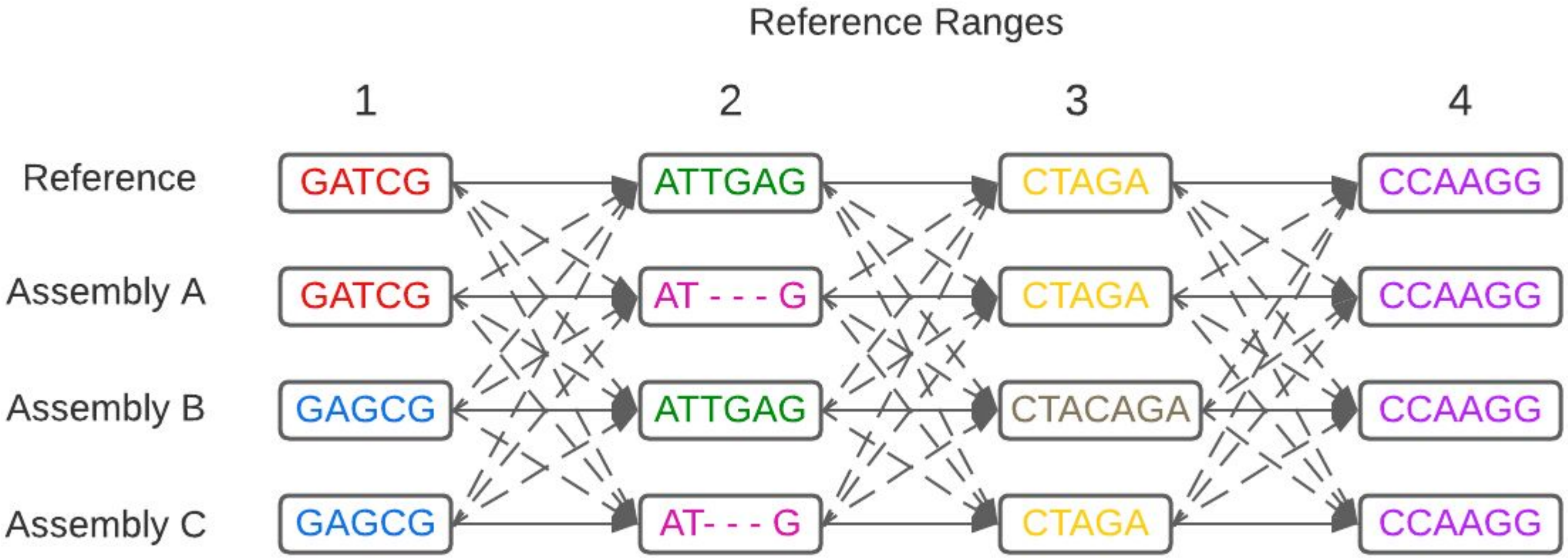
- Tested with a Maize PHG made with ~80 assemblies
  - Paired end 150 bp WGS reads
- Because of the low footprint for mapping, once an index is built, read mappings can be generated on just about any linux machine

	Minimap2(PHGv1)	Kmer (PHGv2)
Index Graph - 1 time	30 min(100GB RAM)	60 min(80GB RAM)
Map 5x WGS Paired End	90 min(100GB RAM)	50 sec(<10GB RAM)



Create the Database

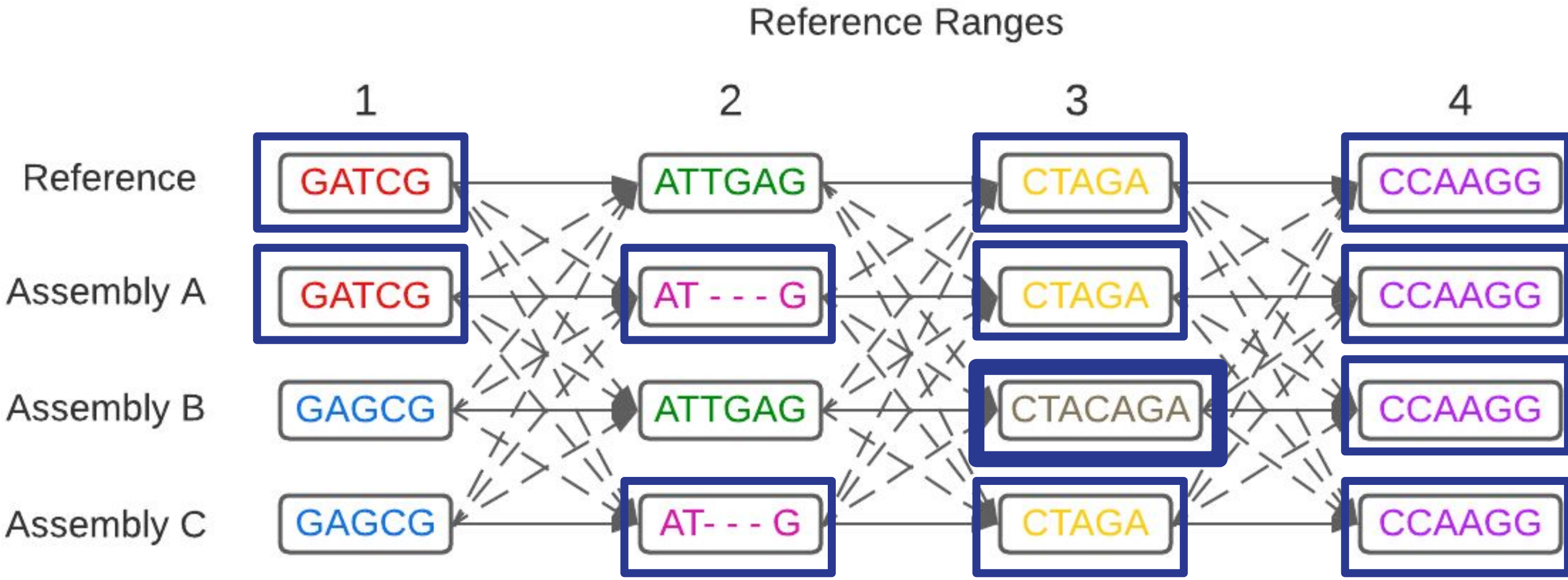
Impute from WGS short reads



HapId Set	Counts
10f47f	1
0a16aa	1
9fb476, 361174	1
361174	1
1d471f	1

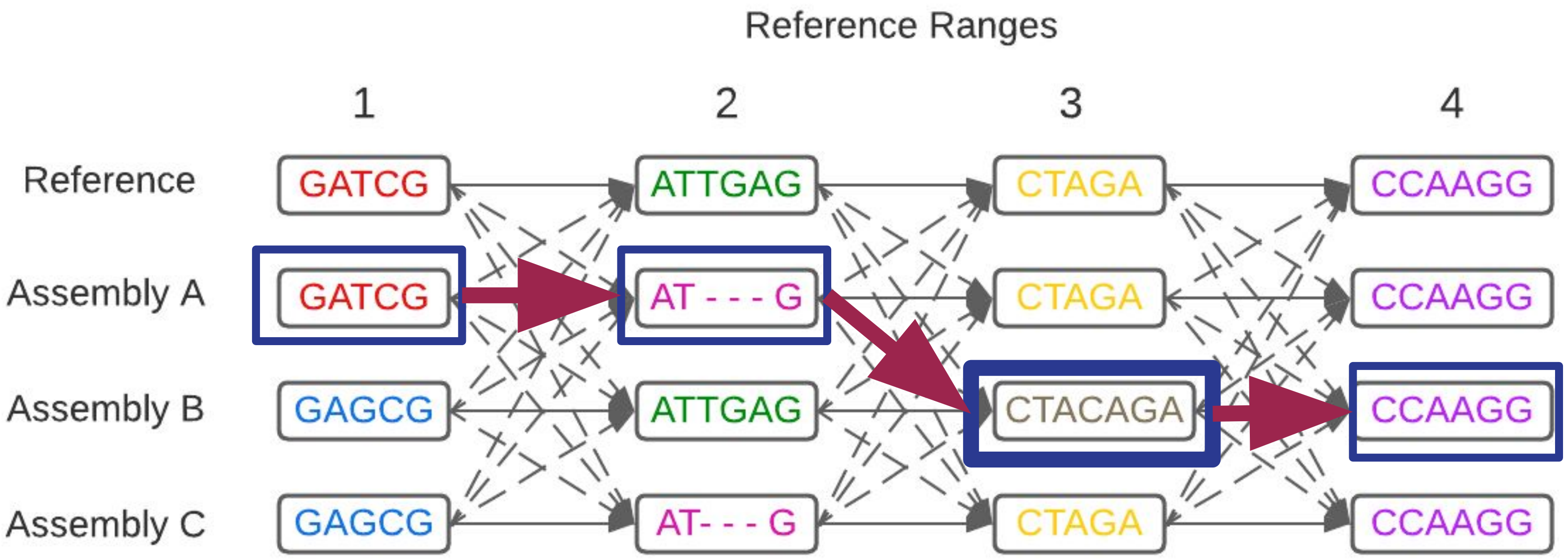
Imputation: Path Finding

Imputation: Path Finding



HapId Set	Counts
10f47f	1
0a16aa	1
9fb476, 361174	1
361174	1
1d471f	1

Apply Hidden Markov Model



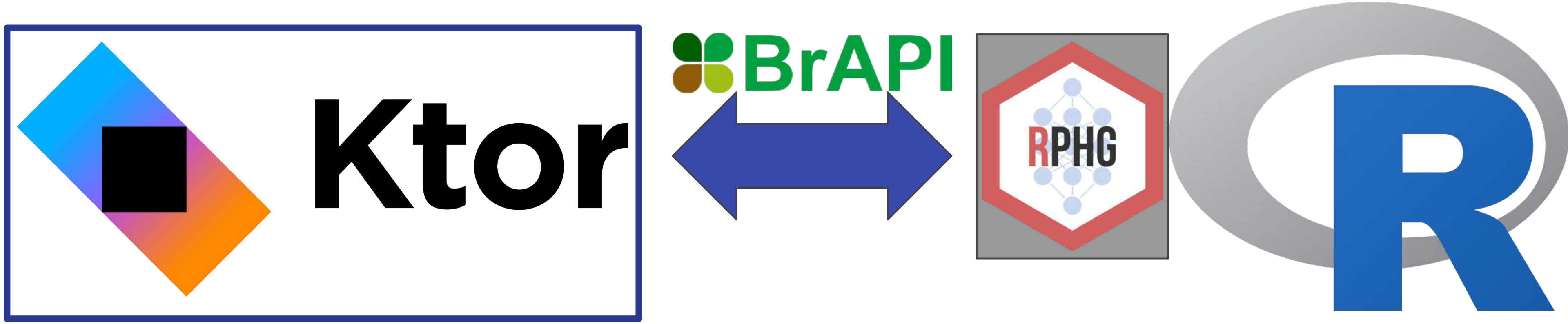
```
>phg find-paths --path-key-file key-file.txt --hvcf-dir /path/to/hvcfs/
        --output-dir /path/to/hvcf/output --path-type haploid
```

Imputation: Path Finding

- The find-paths command will output a hVCF for each sample
- Allows the PHG to associate SNPs from assemblies with a new sample
- Can do both haploid and diploid path finding
- Option to use likely-parents to improve imputation

How Can You Actually Use the Graph?

- phg_v2 comes bundled with a simple BrAPI compliant ktor server



```
>phg start-server
```

Types of Analysis You Can Do!

- Genomic Selection / Genome Wide Association Studies
  - Can use imputed variants or imputed haplotypes
- Link and Visualize Haplotype information with metadata
- Generate Kinship/Distance Matrices
- Subset regions of the paths that you would like to focus on
  - This gene is interesting, what is surrounding it?
- Any type of analysis you can do with a VCF you can try with an hVCF

Whats next? - Summer Plans

- QC Reports
  - Each step will report back useful QC metrics for easy pipeline debugging
- Rare Allele pipeline
  - Using imputed paths can we find rare alleles within samples?
  - Can we use this to add additional diversity to the graph?
- AI Driven imputation
  - Can we train a model to give better imputation results?
- Support more Species
  - Right now ~5 species PHGv2s are being built
    - Maize, Sorghum, Cassava, Wheat, Cotton and more on the way
    - What are stress points?
    - What is confusing?

Check out our Github!

- We have great documentation
- Daily builds and releases
- We welcome code contributions!
- Issues can be submitted through Github or the ‘phg’ Biostars tag

Acknowledgements

Core Development Team:

- Ana Berthel
- Brandon Monier
- Lynn Johnson
- Peter Bradbury
- Terry Casstevens

Biology QC/Alpha Test:

- Bethany Econopouly
- Cinta Romay
- Qi Sun

IGD/Buckler Lab Leadership

- Ed Buckler
- Sara Miller

USDA U.S. DEPARTMENT OF AGRICULTURE

BILL & MELINDA GATES foundation

National Science Foundation WHERE DISCOVERIES BEGIN

This research is funded by NSF Plant Genome Project (NSF #1822330), Bill and Melinda Gates Foundation, and USDA-ARS.

Questions?