Current options to index, represent, and visualize annotations in a pangenome with the vg toolkit

This manuscript (permalink) was automatically generated from jmonlong/manu-vggafannot@727fe07 on September 25, 2024.

Authors

- **Adam M. Novak**
  0000-0001-5828-047X · adamnovak
  UC Santa Cruz Genomics Institute, University of California, Santa Cruz, Santa Cruz, CA, USA

- **Dickson Chung**
  UC Santa Cruz Genomics Institute, University of California, Santa Cruz, Santa Cruz, CA, USA

- **Glenn Hickey**
  · glennhickey
  UC Santa Cruz Genomics Institute, University of California, Santa Cruz, Santa Cruz, CA, USA

- **Sarah Djebali**
  0000-0002-0599-1267 · sdjebali
  IRSD - Digestive Health Research Institute, University of Toulouse, INSERM, INRAE, ENVT, UPS, Toulouse, France

- **Toshiyuki T. Yokoyama**
  Department of Computational Biology and Medical Sciences, Graduate School of Frontier Sciences, The University of Tokyo, Chiba, Japan

- **Erik Garrison**
  0000-0003-3821-631X · ekg
  Department of Genetics, Genomics and Informatics, University of Tennessee Health Science Center, Memphis, TN, USA

- **Benedict Paten**
  0000-0001-8863-3539 · benedictpaten
  UC Santa Cruz Genomics Institute, University of California, Santa Cruz, Santa Cruz, CA, USA

- **Jean Monlong** ✉
  0000-0002-9737-5516 · jmonlong
  IRSD - Digestive Health Research Institute, University of Toulouse, INSERM, INRAE, ENVT, UPS, Toulouse, France

✉ — Correspondence possible via email to Jean Monlong <jean.monlong@inserm.fr>.

Abstract

The current reference genome is the backbone of diverse and rich annotations. Simple text formats, like VCF or BED, have been widely adopted and helped this critical exchange of genomic information. To enable similar enrichment of a pangenome reference, there is a dire need for tools and formats for pangenomic annotation. The Graph Alignment Format (GAF) is a text format, tab-delimitted like BED/VCF files, which was proposed to represent alignments While tt could be used to represent any type of annotation in a pangenome graph, there is no tools to query them efficiently.

Here, we present extension to vg and HTSlib that provide efficient sorting, indexing and querying for GAF files. With this approach, annotations overlapping a subgraph can be extracted quickly. Paths are sorted based on the IDs of traversed nodes, compressed with BGZIP, and indexed with HTSlib/tabix that we extended to work with the GAF format. In addition, we updated `vg annotate` to better produce graph annotations as paths, starting from annotation files relative to linear references. More precisely, it can to take annotations in BED or GFF3 files, written relative to reference paths or haplotypes, and produce GAF files representing their paths through the pangenome.

To showcase these commands, we projected annotations for all haplotypes in the latest draft human pangenome (HPRC v1.1 GRCh38-based Minigraph-Cactus pangenome). This included genes, segmental duplications, tandem repeats and repeats annotations. `vg annotate` can annotate ~4M gene annotations in ~16 mins, and ~5.5M repeats from RepeatMasker in ~9 mins on a single-threaded machine. Finally, these rich annotations can then be quickly queried with `vg` and visualized using existing tools like the sequenceTubeMap or Bandage.

Introduction

The current reference genome is the backbone of diverse and rich annotations. It is used as a reference to map sequencing reads. The coordinate system that it provides is used by large annotation databases gathering functional elements, known variant information, genomic elements. Over the past decades, the organization and visualization of these annotations has been central for understanding and sharing results from genomic studies. In practice, annotations are typically saved in additional files following a text format that can be compressed and indexed for fast query (e.g. VCF, BED, GFF). Under the hood, the HTSlib library supports the indexing of the most used file formats[1]. These annotation files are easy to write and to load in software like IGV[2] or website like the UCSC Genome Browser[3].

With the improved genome sequencing technologies, more high-quality genomes can be produced and combined into pangenomes. A pangenome represents multiple genomes, often as a graph describing adjancency (edges) of the sequences (nodes) in the genomes. Tools to work with such pangenomes are still in their infancy although there are now several options that shows improved performance over traditional approaches. In particular, sequencing data analysis benefits from using pangenomes as a reference for read mapping[4,5], variant calling[6], peak calling[7,8], or transcript expression quantification[9]. This means that we are now manipulating genomic objects like sequencing reads, epigenetic regions, genomic variants, in the pangenome graph space. Currently, those results are typically projected to the linear reference genome. As for linear genome reference, it will be essential to be able to organize and visualize genomic annotation in the pangenome. A lack of user-friendly querying and visualization options would hamper the adoption of the pangenomic paradigm.

Several interactive visualization tools for pangenomes already exist but mostly focus on representing the graph topology with specialized integration of additional information layers. Bandage is an interactive assembly graph visualization tool which can scale to pangenomes up to hundreds of thousands of nodes on modern computers[10]. GfaViz is another interactive sequence graph visualization tool which supports the GFA 2 format and its new features[11]. In particular, it can represent the different paths annotated in the GFA file. GfaViz cannot load additional annotation files: the annotations, written for example as GFA *paths*, must be included in the main pangenome file to be accessible during exploration. The sequenceTubeMap visualizes the pangenome and sequencing reads aligned to it, allowing the user to query specific regions[12]. MoMI-G, or Modular Multi-scale Integrated Genome Graph Browser, focuses on the visualization of structural variants[13]. Variants are interactively selected and represented in a pangenome view a custom sequenceTubeMap representation. Supporting reads and genome annotations can also be included in that representation. Currently, only the reference genome path can be annotated from files in BED/GFF/WIG format. Panache is another pangenomic tool specializing in gene-centric visualization of a linearized view of the pangenome, where blocks of homologous sequences are represented side by side[14]. The blocks are interactively explored in a web-based application to explore their gene content and their absence/presence in the different genimes. Panache is notably used in the Banana Genome Hub[15]. In summary, some visualization tools exist but it is not yet clear how to best provide additional annotation layers to their graph representation.

A few options also exist to display static representations of a pangenome graph (or subgraph). The `vg` toolkit can output a pangenome and alignments in the DOT format. Paths can be displayed too, although the image can become hard to read when many paths are included or in large grahs. The `odgi` toolkit offers visualizations that scale better to larger graphs and annotations[16]. The pangenome is, for example, linearized using an 1D layout[17] and annotations displayed on top of the graph as an array/heatmap. Of note, `odgi` implemented two options to add annotations from external BED files: one to convert annotations to a CSV file to help color nodes in Bandage, another by injecting new paths into a graph before visualization.

There is a dire need for a format supporting annotations in the pangenome and that can is easy to create, index and query. As demonstrated by the success of the BED, GFF or VCF formats, it could help the critical exchange of (pan)genomic information, and allow pangenomic tools access additional information in the pangenomic space. The Graph Alignment Format (GAF) text format, which was proposed to represent alignments, could be used to represent any type of annotation in a pangenome graph. It's the lack of techniques to compress, index and query it that limits its adoption at a larger scale. Here, we present new features of the `vg` and HTSlib ecosystem that provide efficient indexing and querying for pangenomic annotations represented as paths in the GAF format. We illustrate its values in several applications: projecting and visualizing gene and repeat annotations, summarizing open-chromatin from epigenomics sequencing datasets, or positioning known variants in the pangenome.

Methods

Indexing paths in GAF files

The sorting and indexing algorithm is most efficient and makes sense when node IDs are integer sorted based on the topology of the pangenome graph. This is the case for pangenomes constructed by minigraph-cactus[18], PGGB[19], or `vg construct` [4]. Otherwise, the pangenome graph can be *sorted*, i.e. changing the node IDs, using `vg` or `odgi` [16]. If the node IDs are sorted integers, a short path through the graph should only traverse nodes with IDs contained in a small interval (see Fig 1). The main approach of the GAF sorting and indexing approach is to work with those intervals. Hence, to sort a GAF file, each path is first parsed to extract the minimum and maximum node IDs. The paths are then sorted first by their minimum ID, and then by their maximum ID. This is similar to the approach used to sort BED or VCF files on linear genomes: they are sorted by sequence name, then start position, then end position.

Figure 1: Path sorting and indexing using vg and HTSlib/tabix A. A region of pangenome is represented with nodes (containing sequences) in yellow and edges in black. The node IDs are topologically sorted integers, ranging here from 23 to 36. Three paths are highlighted in red, blue and green. B. The three paths are written with the GAF syntax, by specifying the orientations (</>) and IDs of the traversed nodes. For each path, the node range n-N, between the minimum and maximum node IDs, is used for sorting the path. C. Overview of the workflow to sort a GAF file using vg gamsort, compress it with bgzip and index using tabix. D. The small .tbi index file helps query slices of the GAF file quickly. For example using tabix, or vg subcommands like find or chunk.

Figure 1: Path sorting and indexing using vg and HTSlib/tabix A. A region of pangenome is represented with nodes (containing sequences) in yellow and edges in black. The node IDs are topologically sorted integers, ranging here from 23 to 36. Three paths are highlighted in red, blue and green. **B.** The three paths are written with the GAF syntax, by specifying the orientations (`<` / `>`) and IDs of the traversed nodes. For each path, the node range *n-N*, between the minimum and maximum node IDs, is used for sorting the path. **C.** Overview of the workflow to sort a GAF file using `vg gamsort`, compress it with `bgzip` and index using `tabix`. **D.** The small *.tbi* index file helps query slices of the GAF file quickly. For example using `tabix`, or `vg` subcommands like `find` or `chunk`.

A GAF sorting feature has been added to the `vg` toolkit, within the `gamsort` subcommand. It first sorts chunks of GAF records (e.g. reads), to avoid having to hold the entire set in memory. The sorted chunks are then merged into the final sorted GAF file. This GAF sorting implementation was included in `vg` in version 1.56. A sorted GAF file can be compressed in the BGZF format using `bgzip`. The BGZF format is an implementation of the standard gzip format that compresses a file by block to facilitate their random access. It is used to compress among others, VCF, BED, or BAM files with HTSlib[1].

HTSlib[1] was then modified to index bgzipped GAF files. Similar to other tab-separated file like VCF or BED, a *gaf* preset was added to `tabix`. For BED or VCF, `tabix` extract the interval information from the columns with the sequence names and genomic position[20]. In the new *gaf* preset, it instead parses the path information in the GAF file to extract the minimum and maximum node IDs. The indexing is then based on this interval, the same as used for the sorting described above.

We tested the GAF sorting and indexing performance on 30X coverage Illumina short reads from HG002. The reads were downloaded from `https://s3-us-west-2.amazonaws.com/human-pangenomics/NHGRI_UCSC_panel/HG002/hpp_HG002_NA24385_son_v1/ILMN/downsampled/`. We mapped them using giraffe[5] on a personnalized pangenome[21] from the HPRC v1.1 Minigraph-Cactus pangenome. The reads were outputted in GAM first to compare the file size and sorting runtimes. The GAM file was sorted with `vg gamsort`. It was then converted to a GAF file using `vg convert` and sorted using `vg gamsort` with the new GAF sorting mode described above. We compared the file sizes and sorting runtimes between both approach. The query time was also measured on ten thousand regions of the pangenome defined as node ranges. Those node ranges were defined by picking a random starting node position the reference path and walking 50 steps along that path. The commands and details for this benchmarking is available in the `analysis/readsorting` folder of this paper's repository[**repo?**].

Querying GAF files

Instead of indexing on a sequence name and genomic position, we can query on a node interval. In HTSlib, `tabix` was modified to disregard the sequence name when querying intervals for a GAF file. The interval is defined by the values typically used for the *start* and *end* position of genomic coordinates.

Commands to query slices of the pangenome in `vg` were also updated. The `find` and `chunk` subcommands use the updated HTSlib library to extract the appropriate paths from the nodes selected. Internally, those commands identify a first subgraph, for example corresponding to a genomic interval on the reference path provided by the user. This subgraph is then extended with more *context* to include non-reference regions of the graph. The amount of context is also controlled by the user. Finally, the subgraph or the paths (usually reads) overlapping these nodes are extracted. This last step was updated to be able to extract paths in an indexed GAF file using HTSlib. As for sorted GAM files, it is now possible to extract a slice of a indexed GAF file based on node intervals, coordinates on a reference path, multiple coordinates in a BED file, a provided subgraph (see User Guide at ??).

Projecting annotations onto a pangenome

A pangenome represents multiple genomes for which annotations might be available. We describe an approach to project annotations relative to a genome onto a pangenome. We recently updated the `annotate` subcommand from the `vg` toolkit to project regions represented in the BED or GFF files onto the latest pangenomes from the HPRC. Currently, the genome to annotate must be a *reference* path in the pangenome. The `gbwt` subcommand can making a specific path into a *reference* path in about ?? minutes. Once a genome or haplotype is a *reference* path, a pangenome can be queried using coordinates on this path. Internally, `vg annotate` looks for the location of a path in pangenome graph for each input region. The path, represented as an *alignment* record, is then written either in GAM or GAF formats. Of note, a path can be broken in multiple disjointed parts. It happens in the recent human draft pangenome when some regions are clipped out, for example across centromeres or when creating suspiciously large structural variants. Projected annotations are hence also broken up if needed when they overlap with breakpoints. The name of the annotated path in the output GAF is picked from the BED file's 4th column, or the *Name* and *ID* GFF field.

We test this approach by projecting the gene annotation, repeat annotations, and segmental duplication annotation for each of the 88 assembled haplotypes in the draft human pangenome from the HPRC (v1.1). The gene annotations from CAT[22] (GFF files) were downloaded from ??. The repeat annotations from RepeatMasker (BED files) were downloaded from ??. The predicted segmental duplications from ?? (BED files) were downloaded from ??. A helper script was implemented to prepare the BED files with informative names. For example, for repeats from the RepeatMasker annotation, we name each element by their repeat class and repeat family. The projection of those annotations for each haplotype was automated with a Snakemake workflow available in the `analysis/annotate` folder of this paper's repository[**repo?**].

Coverage track from mapped reads

Functional genomics datasets are often visualized as a coverage track. High coverage in a region might suggest a strong transcription factor binding site or regulatory region.

We implemented an approach to summarize the coverage of reads across the pangenome into paths with similar coverage. The coverage in every node is first binned into a few coverage classes, for example representing low, medium and high coverage. By default we use 1, 5, and 30 reads as coverage breakpoints to save three bins: 1-5, 5-30, 30+. Regions with no coverage are not saved. Once the coverage is binned, we extend greedily to connected nodes and bins if in the same coverage class. This extension step produces path through the pangenome with consistent coverage. The paths are written in the GAF format, recording the coverage class and the average coverage across the path.

`cartoon to explain this in supplement?`

This algorithm was implemented in Python and uses the *libbdsg* module[23] to query the pangenome. It is made available in the public repository of this study in the `analysis/encode` folder of this paper's repository[**repo?**].

Annotating known variants

Variants from public databases were annotated when matching a variant site in the pangenome. We implemented an approach to look for input variants in the pangenome and write a GAF file with the path followed by those variants in the pangenome. The variants are matched with the VCF representation of the HPRC Minigraph-Cactus v1.1 pangenome, produced by running `vg deconstruct` on the pangenome file. This VCF file contains the all variants and their paths through the pangenome in the *AT INFO* field. We look for the variants of interest in this VCF and extract the appropriate path using the *AT* field. The annotation path follows either the alternate allele of the variant if known and present in the pangenome, or the reference allele path if the alternate allele is unknown but there is a variant at this position in the pangenome. Variant that are not matched in the pangenome are skipped. This algorithm was implemented in Python and uses the *libbdsg* module[23] internally to get node size information necessary to write proper GAF records.

We test this approach on the GWAS catalog[24] and GTEX expression QTLs[25]. The GWAS catalog was downloaded from the UCSC Genome Browser `gwasCatalog` table[3]. The eQTLs from GTEX v8 were downloaded from `https://storage.googleapis.com/adult-gtex/bulk-qtl/v8/single-tissue-cis-qtl/GTEx_Analysis_v8_eQTL.tar`. The files defining associations between variant/gene pairs were parsed for each tissue. As before, the output annotation in GAF were bgzipped, sorted and indexed.

We implemented another, more straightforward approach, to annotate variants that were genotyped using the pangenome. Here we simply convert a VCF that contain allele traversal information (*AT* field) to a GAF file representing the alternate allele. We test this approach by genotyping HG002 from short-reads Illumina Novaseq data. `more details`

The scripts and pipeline to annotate variants is available in the public repository of this study in the `analysis/variants` folder of this paper's repository[**repo?**].

Visualization in the sequenceTubeMap

The sequenceTubeMap was develop to interactively explore a pangenome graph, haplotypes traversing it, and reads mapping to it[12]. It internally calls *vg* to extract the relevant slice of the pangenome graph and reads. To extract reads, it was only accepting GAM file that had been sorted and indexed *vg*. We have updated it to accept GAF files that have been sorted, compressed and indexed as explained above.

The new version of the sequenceTubeMap can also display multiple layers of haplotypes or reads. As *reads*, the user can now add layers of annotations represented and prepared as indexed GAF files. A different color or color palette can be assigned to each layer to facilitate the visualization of different datasets in the same local pangenome region. For example, one could visualize the coding regions of genes and coverage tracks for different cell types from the ENCODE project (Fig ??).

Visualization in Bandage

A fork of Bandage[10], called BandageNG[26], can visualize paths of the input graph by coloring the nodes. We implemented a wrapper script to facilitate the preprocessing of a subgraph to visualize with BandageNG. It starts by extracting the subgraph of a full pangenome for a region of interest. It also extract annotations from indexed files on that same region. The annotated paths are then added to the subgraph using `vg augment`. The new pangenome subgraph is converted to GFA and contains both the original paths (e.g. haplotypes) and the newly integrated annotations (e.g. gene location, repeats). The output of this script can be opened with BandageNG for interactive exploration. In particular, the *Path search* feature can find nodes corresponding to specific paths, which are either haplotypes or annotations in this pre-processed GFA file. The user can select a *path*, color the nodes and label the path `check/update`. The helper script and a tutorial are available at the `analysis/??` folder of this paper's repository[**repo?**].

Results

Sorting and indexing short sequencing reads

Read sorting was tested on Illumina HiSeq paired-end short reads for the HG002 sample, each 150 bp long and a genome coverage of about 30X. The gzipped GAF file with about 682 million reads was sorted in 6h32 using a single thread and about 2Gb of memory (Table 1). Indexing the sorted GAF with `tabix` took 18 mins. We compared that approach with the current implementation in `vg` which uses files in the GAM format. GAM is a protobuf alignemnt format introduced in Garrison et al.[4]. Sorting a GAM with the same reads took 11h47 using a single thread and about 6 Gb of memory.

In addition to being about twice as fast to sort, reads written in the GAF format (and bgzipped) also take about half the disk space (52G vs 108G). The main reason for this reduced space is that GAF doesn't save the full read sequence, only the path through the pangenome and edits to reconstruct it. GAM files produced by `vg giraffe` can also save additional information as annotations, like the mapping time of each read, that are currently not kept when converting to the GAF format. Overall, the bgzipped GAF files are half as small and twice as fast to sort for short sequencing reads.

Once indexed, extracting a slice of the pangenome is as efficient as extracting a slice of an indexed BAM, VCF, or BED file in a genomic regions, as it uses the same approach. For example, extracting reads for ten thousand random regions in the pangenome took about 0.07 second per region to retrieve an average of 1707 reads. For comparison, the same extraction took on average XX second using the GAM format`TODO`.

Table 1: Resources used to sort short sequencing reads for a 30x coverage Illumina human dataset.

Format	Time (H:M:S)	Max. memory used (Kb)	File size (Gb)
GAM	11:46:58	6,236.60	108
GAF	6:50:28	1,904.83	52

Annotation of a human pangenome

Human Pangenome Reference Consotium[27]

To showcase these commands, we projected annotations for all haplotypes in the latest draft human pangenome (HPRC v1.1 GRCh38-based Minigraph-Cactus pangenome). This included genes, segmental duplications, tandem repeats and repeats annotations. `vg annotate` can annotate ~4M gene annotations in ~16 mins, and ~5.5M repeats from RepeatMasker in ~9 mins on a single-threaded machine. Finally, these rich annotations can then be quickly queried with `vg` and visualized using existing tools like the sequenceTubeMap or Bandage.

We also matched and annotated more than 660 thousand variants from the GWAS catalog[24] and expression QTLs from the GTEx catalog[25] across 49 tissues (on average 1.45 million variants per tissue). On average, 94% variants were found in the HPRC pangenome. The variants files in GAF take only 907Mb of space and can be queried fast for visualization in the sequenceTubeMap or Bandage. This annotation is showcased in example ?? described in more detail below (see *Example??* section). Of note, it is also straightforward to convert genotypes for variants from the pangenome to annotated paths. This could be the case when the pangenome is used as the backbone of the genotyping of new samples, for example with Pangenie or *vg call*. To illustrate this use case, we genotyped HG002 using short Illumina reads aligned on the draft human pangenome with vg giraffe and vg call. The predicted genotypes were converted to GAF and indexed. They could be visualized using the sequenceTubeMap, for example, along with the aligned reads. This type of annotation can help a developer explore variant calls and aligned reads. A example is shown in figure ?? and described in more details below. On the traditional reference genome, the equivalent would be to load both reads and variants in IGV[2].

Figure 2: sequenceTubeMap examples In progress... A. black: Reference path (GRCh38), pale colors: other human haplotypes, reds: GWAS catalog, blues: eQTLs across tissues (GTEx). B. yellow/green: annotation paths from vg call genotypes. reds/blues: short sequencing reads. C. Compressed view (no sequence shown, node size not to scale). Reference paths CHM13/GRCh38 (black/grey) and ATAC-seq coverage track for different tissues (colors). Opacity represents coverage level.

Figure 2: sequenceTubeMap examples `In progress...` **A.** *black*: Reference path (GRCh38), *pale colors*: other human haplotypes, *reds*: GWAS catalog, *blues*: eQTLs across tissues (GTEx). **B.** *yellow/green*: annotation paths from `vg call` genotypes. *reds/blues*: short sequencing reads. **C.** Compressed view (no sequence shown, node size not to scale). Reference paths CHM13/GRCh38 (*black/grey*) and ATAC-seq coverage track for different tissues (*colors*). Opacity represents coverage level.

Figure 3: Bandage example In progress... Node color: blue for the reference path, orange for the AluYa5 transposon.

Figure 3: Bandage example `In progress...` Node color: *blue* for the reference path, *orange* for the AluYa5 transposon.

Coverage of seven functional datasets from ENCODE

We aligned ATAC-seq datasets from 7 cell types to the draft human pangenome to produce coverage tracks as indexed GAF files. On average, there were about 488 thousand paths representing high read coverage which were 2.5 nodes (101.9 bases) long. On average, 65 thousand paths with high ATAC-seq read coverage traversed more than three nodes, i.e. regions of the pangenome with variation (see Table 2).

Table 2: High coverage tracks from seven functional datasets on the HPRC pangenome. For each sample, the table shows how many paths had a mean coverage of at least 10 reads, and how long they were.

Dataset	Paths	Average bases	Average nodes	Traversing >2 nodes
Breast epithelium	587,869	106.45	2.59	67,050 (11.4%)
Gastrocnemius medialis	349,293	92.39	2.38	54,953 (15.7%)
Gastroesophageal sphincter	572,482	112.12	2.64	74,849 (13.1%)
Peyer's patch	278,081	94.93	2.54	44,852 (16.1%)
Sigmoid colon	697,030	111.76	2.61	85,362 (12.2%)
Spleen	546,530	102.00	2.59	68,880 (12.6%)
Thyroid gland	383,558	93.70	2.36	58,042 (15.1%)

It took on average 7 cpu.hours to map the reads to the pangenome using VG Giraffe, and 2.7 cpu.hours to produce the coverage tracks. Sorting, compressing and indexing them took only 0.16

cpu.hours, on average. Table 3 compiles the runtimes and memory used for each step across all samples.

`example of what we could look for and describe` Fig 4 shows examples of those tracks visualized using the sequenceTubeMap. In Fig ??a, the promoter of the *??* gene is seen to be open in the ?? cell type. Thanks to the pangenomic view, we see differential coverage of the functional tracks across the variants in the region. For instance, we notice a small insertion-deletion (indel) where the alternate allele is only covered by 2 reads, while the reference allele is covered by more than 30 reads. Fig ??b highlights a structural variant, a ??bp insertion, that is highly covered by ATAC-seq in several cell types. The RepeatMasker annotation in this region, also extracted from an indexed GAF file, flags this insertion as a ?? transposable element. ?? can indeed attract TF?? that lead to open chromatin ?REF?.

Figure 4: Coverage tracks visualized interactively using the sequenceTubeMap. a) Promoter... b) Structural variant....

Figure 4: Coverage tracks visualized interactively using the sequenceTubeMap. a) Promoter... b) Structural variant....

Examples

Using sequenceTubeMap, haplotypes, read alignments and paths can be visualized interactively. Hovering on a path displays its name, here the ID of a coding region of the BOLA2B gene.

> Haplotypes: CHM13 (purple), HG00621 (greys). Annotated CDS for HG00621 hap 1 (reds) and 2 (blues).

Using BandageNG, a fork that can import paths in GAF files, paths were searched and colored to illustrate a mobile element insertion.

> Yellow: AluYa5 element annotated in the haplotype 1 of HG00438. Blue: CHM13 reference path.

Discussion

The tools and applications described here present an option to streamline the production of annotations in the GAF format, their indexing and efficient querying. First, these techniques will help integrate additional information into existing or future pangenomic tools. They will simplify the integration of information like gene or repeat annotations, known variants, or more generally other functional annotations, in pangenomic analysis. These analysis will be able to work with those annotations in light of underlying genomic variation recapitulated in the pangenome. Visualization is a specific example where the users typically want to include several layer of annotations, including some custom-made. Tools to visualize pangenomes will greatly benefit from a simple pipeline to create or load pangenomic annotations. Second, this work is critical to allow the community to make, share and reuse annotations in the pangenome space. The GAF format is already used by multiple independent tools, although mostly read mappers. Thanks to our work, GAF files can now be indexed and queried efficiently, like BED, VCF, or GFF on a linear reference genome. We also showcase how annotations on a linear genome can be converted to a path in the pangenome in the GAF format. For these reasons, we believe it could become the de facto format to represent annotations in the pangenome and accelerate the adoption of the new pangenomic paradigm by the broader genomics field. Although informative annotations could be analyzed already, the current approach has some limitations.

The indexing scheme relies on integer node IDs, compacted relative to their position in the pangenome. If node IDs are not integer, it adds a layer of complexity for the user as they will need to convert them to integers and keep track of the translation between original pangenome and new pangenome. Both `vg` and `odgi` offer ways to convert pangenome to a new pangenome with compact integer node IDs.

The current implementation is designed with short paths in mind, where the user wants to extract the full annotated paths in a region. It is convenient when working with short reads, gene annotations, and most genomic repeats, for example. Larger genomic regions, like chromosome bands, large assembled contigs, or large segmental duplications, can still be represented and manipulated, but might be less practical to work with. With the current implementation, the full annotated region will be extracted when querying an overlapping region. The extracted annotations are not clipped to the specified range which could be inconvenient to a user who zooming into a small region but would still like to know about the much larger annotated path traversing this subgraph. To address this use case, we plan to implement an extraction mode where output annotations are trimmed to keep only the queried subgraph.

The naive metadata integration is another limitation. Indeed, the annotation information is currently reduced to a single label. In our applications, those label would contain for example a gene name and haplotype names, or a read coverage bin and average coverage value, all in one label. This was easy to implement and sufficient for now because there are no tools that handle more advanced metadata organization. For many annotations, it would be useful to keep the metadata better organized, so that the user can access/use it within visualization tools. The different metadata could be saved using optional tags that can be added at the end of each GAF record.

Our approach to convert annotations from a linear genome to the pangenome assumes that we have annotations of the different haplotypes in the pangenome. There is still no clear solution to lift annotations from one reference/haplotype to other haplotypes in the pangenome, except through reanalysis/reannotation of each haplotype. The homology information embedded in the pangenome could potentially be used to propagate annotations from one haplotype to others more easily. This strategy can already be used by annotation tools like CAT[22], as a additional source of gene annotation evidence. In the future, these techniques might help propagate other types of annotations across the pangenome more efficiently than by reanalyzing the raw data from scratch on each haplotype. `something about odgi paths/untangle maybe`

It also highlights the limitations of the existing tools to integrate these files. Some tools, like Bandage or GfaViz, require manual pre-processing, for example extracting a subgraph and integrating the annotations as embedded paths. The sequenceTubeMap can now handle indexed bgzipped GAF files, but the query time for large pangenome remains long in practice. Defining and integrating annotation metadata into its interface will also require a significant amount of development. Overall, we stress the need for visualization tools that can efficiently layout and organize many paths through a pangenome.

We showed that the GAF format, thanks to new tools for their efficient manipulation, could offer a path for the future of annotations in pangenome graphs. While it provides an important building block, it is clear that more needs to be done to make it a useful solution for the community.

Code and data availability

`vg` is available at https://github.com/vgteam/vg. The fork of Bandage that allow for path coloring is available at https://github.com/asl/BandageNG. The sequenceTubeMap is hosted at https://github.com/vgteam/sequenceTubemap

The analysis presented in this manuscript is documented in the https://github.com/jmonlong/manu-vggafannot. It contains scripts used to prepare the annotation files, commands and automated pipelines used to annotate them in the pangenome, and helper scripts to summarize the output files.

The annotations of the HPRC v1.1 pangenome were deposited on Zenodo at `??ZENODO_LINK??`. This includes gene annotations, repeats from RepeatMasker, simple repeats and segmental duplications. Coverage tracks for the `??` ENCODE ATAC-seq samples are also available in this repository.

References

1. **HTSlib: C library for reading/writing high-throughput sequencing data**
James K Bonfield, John Marshall, Petr Danecek, Heng Li, Valeriu Ohan, Andrew Whitwham, Thomas Keane, Robert M Davies
GigaScience (2021-01-29) https://doi.org/gt98
DOI: 10.1093/gigascience/giab007 · PMID: 33594436 · PMCID: PMC7931820

2. **Integrative genomics viewer**
James T Robinson, Helga Thorvaldsdóttir, Wendy Winckler, Mitchell Guttman, Eric S Lander, Gad Getz, Jill P Mesirov
Nature Biotechnology (2011-01) https://doi.org/cpmj8s
DOI: 10.1038/nbt.1754 · PMID: 21221095 · PMCID: PMC3346182

3. **The UCSC Genome Browser database: 2023 update**
Luis R Nassar, Galt P Barber, Anna Benet-Pagès, Jonathan Casper, Hiram Clawson, Mark Diekhans, Clay Fischer, Jairo Navarro Gonzalez, Angie S Hinrichs, Brian T Lee, … W James Kent
Nucleic Acids Research (2022-11-24) https://doi.org/gxpftd
DOI: 10.1093/nar/gkac1072 · PMID: 36420891 · PMCID: PMC9825520

4. **Variation graph toolkit improves read mapping by representing genetic variation in the reference**
Erik Garrison, Jouni Sirén, Adam M Novak, Glenn Hickey, Jordan M Eizenga, Eric T Dawson, William Jones, Shilpa Garg, Charles Markello, Michael F Lin, … Richard Durbin
Nature Biotechnology (2018-10) https://doi.org/gd2zqs
DOI: 10.1038/nbt.4227 · PMID: 30125266 · PMCID: PMC6126949

5. **Pangenomics enables genotyping of known structural variants in 5202 diverse genomes**
Jouni Sirén, Jean Monlong, Xian Chang, Adam M Novak, Jordan M Eizenga, Charles Markello, Jonas A Sibbesen, Glenn Hickey, Pi-Chuan Chang, Andrew Carroll, … Benedict Paten
Science (2021-12-17) https://doi.org/gns2wr
DOI: 10.1126/science.abg8871 · PMID: 34914532 · PMCID: PMC9365333

6. **Genotyping structural variants in pangenome graphs using the vg toolkit**
Glenn Hickey, David Heller, Jean Monlong, Jonas A Sibbesen, Jouni Sirén, Jordan Eizenga, Eric T Dawson, Erik Garrison, Adam M Novak, Benedict Paten
Genome Biology (2020-02-12) https://doi.org/ghh78w
DOI: 10.1186/s13059-020-1941-7 · PMID: 32051000 · PMCID: PMC7017486

7. **Graph Peak Caller: Calling ChIP-seq peaks on graph-based reference genomes**
Ivar Grytten, Knut D Rand, Alexander J Nederbragt, Geir O Storvik, Ingrid K Glad, Geir K Sandve
PLOS Computational Biology (2019-02-19) https://doi.org/ghwpwv
DOI: 10.1371/journal.pcbi.1006731 · PMID: 30779737 · PMCID: PMC6396939

8. **Personalized and graph genomes reveal missing signal in epigenomic data**
Cristian Groza, Tony Kwan, Nicole Soranzo, Tomi Pastinen, Guillaume Bourque
Genome Biology (2020-05-25) https://doi.org/ggzxbs
DOI: 10.1186/s13059-020-02038-8 · PMID: 32450900 · PMCID: PMC7249353

9. **Haplotype-aware pantranscriptome analyses using spliced pangenome graphs**
Jonas A Sibbesen, Jordan M Eizenga, Adam M Novak, Jouni Sirén, Xian Chang, Erik Garrison, Benedict Paten
Nature Methods (2023-01-16) https://doi.org/gt9fj7
DOI: 10.1038/s41592-022-01731-9 · PMID: 36646895

10. **Bandage: interactive visualization of <i>de novo</i> genome assemblies**
Ryan R Wick, Mark B Schultz, Justin Zobel, Kathryn E Holt
Bioinformatics (2015-06-22) https://doi.org/gb5g64
DOI: 10.1093/bioinformatics/btv383 · PMID: 26099265 · PMCID: PMC4595904

11. **<i>GfaViz</i>: flexible and interactive visualization of GFA sequence graphs**
Giorgio Gonnella, Niklas Niehus, Stefan Kurtz
Bioinformatics (2018-12-31) https://doi.org/gf9pj6
DOI: 10.1093/bioinformatics/bty1046 · PMID: 30596893

12. **Sequence tube maps: making graph genomes intuitive to commuters**
Wolfgang Beyer, Adam M Novak, Glenn Hickey, Jeffrey Chan, Vanessa Tan, Benedict Paten, Daniel R Zerbino
Bioinformatics (2019-08-01) https://doi.org/gghmj8
DOI: 10.1093/bioinformatics/btz597 · PMID: 31368484 · PMCID: PMC6954646

13. **MoMI-G: modular multi-scale integrated genome graph browser**
Toshiyuki T Yokoyama, Yoshitaka Sakamoto, Masahide Seki, Yutaka Suzuki, Masahiro Kasahara
BMC Bioinformatics (2019-11-05) https://doi.org/gghcjh
DOI: 10.1186/s12859-019-3145-2 · PMID: 31690272 · PMCID: PMC6833150

14. **Panache: a web browser-based viewer for linearized pangenomes**
Éloi Durant, François Sabot, Matthieu Conte, Mathieu Rouard
Bioinformatics (2021-10-02) https://doi.org/gxn9ks
DOI: 10.1093/bioinformatics/btab688 · PMID: 34601567 · PMCID: PMC8652104

15. **The Banana Genome Hub**
Gaëtan Droc, Delphine Larivière, Valentin Guignon, Nabila Yahiaoui, Dominique This, Olivier Garsmeur, Alexis Dereeper, Chantal Hamelin, Xavier Argout, Jean-François Dufayard, … Stéphanie Bocs
Database (2013-01-01) https://doi.org/gxn9mc
DOI: 10.1093/database/bat035 · PMID: 23707967 · PMCID: PMC3662865

16. **ODGI: understanding pangenome graphs**
Andrea Guarracino, Simon Heumos, Sven Nahnsen, Pjotr Prins, Erik Garrison
Bioinformatics (2022-05-13) https://doi.org/gt9fj8
DOI: 10.1093/bioinformatics/btac308 · PMID: 35552372 · PMCID: PMC9237687

17. **Pangenome graph layout by Path-Guided Stochastic Gradient Descent**
Simon Heumos, Andrea Guarracino, Jan-Niklas M Schmelzle, Jiajie Li, Zhiru Zhang, Jörg Hagmann, Sven Nahnsen, Pjotr Prins, Erik Garrison
Bioinformatics (2024-07-01) https://doi.org/g5gwzs
DOI: 10.1093/bioinformatics/btae363 · PMID: 38960860 · PMCID: PMC11227364

18. **Pangenome graph construction from genome alignments with Minigraph-Cactus**
Glenn Hickey, Jean Monlong, Jana Ebler, Adam M Novak, Jordan M Eizenga, Yan Gao, Haley J Abel, Lucinda L Antonacci-Fulton, Mobin Asri, … Benedict Paten
Nature Biotechnology (2023-05-10) https://doi.org/gsfwq7
DOI: 10.1038/s41587-023-01793-w · PMID: 37165083 · PMCID: PMC10638906

19. **Building pangenome graphs**
Erik Garrison, Andrea Guarracino, Simon Heumos, Flavia Villani, Zhigui Bao, Lorenzo Tattini, Jörg Hagmann, Sebastian Vorbrugg, Santiago Marco-Sola, Christian Kubica, … Pjotr Prins
Cold Spring Harbor Laboratory (2023-04-06) https://doi.org/gt9fj9
DOI: 10.1101/2023.04.05.535718 · PMID: 37066137 · PMCID: PMC10104075

20. **Tabix: fast retrieval of sequence features from generic TAB-delimited files**
Heng Li
Bioinformatics (2011-01-05) https://doi.org/bb9
DOI: 10.1093/bioinformatics/btq671 · PMID: 21208982 · PMCID: PMC3042176

21. **Personalized Pangenome References**
Jouni Sirén, Parsa Eskandar, Matteo Tommaso Ungaro, Glenn Hickey, Jordan M Eizenga, Adam M Novak, Xian Chang, Pi-Chuan Chang, Mikhail Kolmogorov, Andrew Carroll, … Benedict Paten
Cold Spring Harbor Laboratory (2023-12-14) https://doi.org/gt9fkb
DOI: 10.1101/2023.12.13.571553 · PMID: 38168361 · PMCID: PMC10760139

22. **Comparative Annotation Toolkit (CAT)—simultaneous clade and personal genome annotation**
Ian T Fiddes, Joel Armstrong, Mark Diekhans, Stefanie Nachtweide, Zev N Kronenberg, Jason G Underwood, David Gordon, Dent Earl, Thomas Keane, Evan E Eichler, … Benedict Paten
Genome Research (2018-06-08) https://doi.org/gdpg7n
DOI: 10.1101/gr.233460.117 · PMID: 29884752 · PMCID: PMC6028123

23. **Efficient dynamic variation graphs**
Jordan M Eizenga, Adam M Novak, Emily Kobayashi, Flavia Villani, Cecilia Cisar, Simon Heumos, Glenn Hickey, Vincenza Colonna, Benedict Paten, Erik Garrison
Bioinformatics (2020-07-16) https://doi.org/ghqdzw
DOI: 10.1093/bioinformatics/btaa640 · PMID: 33040146 · PMCID: PMC7850124

24. **The NHGRI-EBI GWAS Catalog: knowledgebase and deposition resource**
Elliot Sollis, Abayomi Mosaku, Ala Abid, Annalisa Buniello, Maria Cerezo, Laurent Gil, Tudor Groza, Osman Güneş, Peggy Hall, James Hayhurst, … Laura W Harris
Nucleic Acids Research (2022-11-09) https://doi.org/gq73r5
DOI: 10.1093/nar/gkac1010 · PMID: 36350656 · PMCID: PMC9825413

25. **The Genotype-Tissue Expression (GTEx) project**
John Lonsdale, Jeffrey Thomas, Mike Salvatore, Rebecca Phillips, Edmund Lo, Saboor Shad, Richard Hasz, Gary Walters, Fernando Garcia, Nancy Young, … Helen F Moore
Nature Genetics (2013-05-29) https://doi.org/gd5z68
DOI: 10.1038/ng.2653 · PMID: 23715323 · PMCID: PMC4010069

26. **asl/BandageNG**
Anton Korobeynikov
(2024-09-06) https://github.com/asl/BandageNG

27. **A draft human pangenome reference**
Wen-Wei Liao, Mobin Asri, Jana Ebler, Daniel Doerr, Marina Haukness, Glenn Hickey, Shuangjia Lu, Julian K Lucas, Jean Monlong, Haley J Abel, … Benedict Paten
Nature (2023-05-10) https://doi.org/gr8b6s
DOI: 10.1038/s41586-023-05896-x · PMID: 37165242 · PMCID: PMC10172123

Supplementary material

Table 3: update Compute resources used for the analysis of the functional datasets and production of the indexed coverage tracks.

Dataset	**reads (M)**	**read mapping**	**coverage track**	**sorting + compressing + indexing**
Breast epithelium	193.6	8.9 cpu.h (54 Gb)	3.1 cpu.h (108 Gb)	0.2 cpu.h (1 Gb)
Gastrocnemius medialis	98.8	4.8 cpu.h (54 Gb)	2.4 cpu.h (100 Gb)	0.1 cpu.h (1 Gb)
Gastroesophageal sphincter	168.5	7.3 cpu.h (54 Gb)	2.7 cpu.h (107 Gb)	0.2 cpu.h (1 Gb)
Peyer's patch	145.3	8 cpu.h (54 Gb)	2.6 cpu.h (104 Gb)	0.2 cpu.h (1 Gb)
Sigmoid colon	173.5	8.2 cpu.h (54 Gb)	2.7 cpu.h (104 Gb)	0.2 cpu.h (1 Gb)
Spleen	157.2	7.6 cpu.h (54 Gb)	2.8 cpu.h (105 Gb)	0.2 cpu.h (1 Gb)
Thyroid gland	91.4	4.6 cpu.h (54 Gb)	2.3 cpu.h (94 Gb)	0.1 cpu.h (1 Gb)