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@cfa1720](#) on August 28, 2024.

Authors

- **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. To enable similar enrichment of a pangenome reference, there is a dire need for tools and formats for pangenomic annotation. Simple text formats, like VCF or BED, have been widely adopted and helped this critical exchange of genomic information. 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. Here I review how some features of the `vg` ecosystem can already provide indexing, querying, and visualization capabilities for annotations represented as paths.

We developed efficient sorting, indexing and querying for GAF files. This approach can for example extract annotations overlapping a subgraph quickly. Alignments are currently sorted based on the covered node IDs, similar to the approach for sorting read alignments in the GAM format, a binary format used previously by the `vg` toolkit. To index the bgzipped GAF file, we extended HTSlib/tabix to work with the GAF format. Second, `vg annotate` was recently updated 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 the equivalent 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.

With the improved genome sequencing technologies, more high-quality genomes can be produced and compiled into pangenomes. A pangenome represents multiple genomes, often as a graph describing adjacency (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[1,**vg?**], variant calling[**vgsv?**], peak calling[**graphpeakcaller?**,**groza?**], or transcript expression quantification[**rpvg?**]. 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 visualization tools for pangenomes already exist but mostly focus on representing the graph topology with limited or specialized integration of additional layers of information. The sequenceTubeMap visualizes the pangenome and sequencing reads aligned to it, allowing the user to

query specific regions[**tubemap?**]. Bandage[**bandage?**] is an interactive assembly graph visualization tool which can scale to pangenomes up to `?` nodes. `review more tools`

To enable similar enrichment of a pangenome reference, there is a dire need for tools and formats for pangenomic annotation. Simple text formats, like VCF or BED, have been widely adopted and helped this critical exchange of genomic information. 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. Here we present new features of the `vg` ecosystem that provide indexing, querying, and visualization capabilities for annotations represented as paths.

GraphAligner, vg giraffe[1].

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[**minigraph?**] (`doublecheck`), minigraph-cactus[**minigraph_cactus?**], or PGGB[**pggb?**]. Otherwise, the pangenome graph can be *sorted*, i.e. changing the node IDs, using `vg` or `odgi` [**odgi?**]. If the node IDs are sorted integers, a short path through the graph should only traverse nodes with IDs contained in a small interval. 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.

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 using `bgzip` (`short explanation of bgzip?`).

HTSlib[**htslib?**] was then modified HTSlib to index bgzipped GAF files. Similar than for 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. 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 ??. We mapped them using giraffe[1] on a personnalized pangenome[**hapsamp?**] 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 commands and details for this benchmarking is available at `GITHUB_REPO/analysis/??`.

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* GFF field `doublecheck`.

We test this approach by projecting the gene annotation, repeat annotations, and segmental duplication annotation for each of the ?? assembled haplotypes in the draft human pangenome from the HPRC (v1.1). The gene annotations from CAT[**cat?**] (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 position, repeat class, and repeat family `check/update`. The projection of those annotations for each haplotype was automated with a Snakemake workflow available at `GITHUB_REPO/analysis/??`.

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[**libbdsg?**] to query the pangenome. It is made available in the public repository of this study at `GITHUB_REPO/analysis/encode`.

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[**libbdsg?**] internally to get node size information necessary to write proper GAF records.

We test this approach on the GWAS catalog (REF??) and GTEX expression QTLs (REF??). The GWAS catalog was downloaded from the UCSC Genome Browser `??` table. The eQTLs from GTEX were downloaded from ??. 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`improve`.

The scripts and pipeline to annotate variants is available in the public repository of this study at `GITHUB_REPO/analysis/variants`.

Visualization in the sequenceTubeMap

The sequenceTubeMap was develop to interactively explore a pangenome graph, haplotypes traversing it, and reads mapping to it[**tubemap?**]. 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[**bandage?**], called BandageNG[**bandageng?**], 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 `GITHUB_REPO/analysis/??`.

Results

Sorting and indexing short sequencing reads

Sort 30x Illumina short-read dataset (~300M paired-end reads) GAM File size: 69G sorted Sorting: ~9h and ~2.5Gb mem, GAF: File size: 24 Gb sorted Sorting: ~6h and ~3.5 Gb mem. GAM to GAF conversion: ~2h.

Annotation of a human pangenome

Human Pangenome Reference Consotium[2]

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 ?? variants from the GWAS catalog(REF??) and ?? variants expression QTLs from the GTEx catalog(REF??). On average, ??% variants were found in the HPRC pangenome. The variants files in GAF take only ??Gb 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([**igv?**]).

Coverage of ?? functional datasets from ENCODE

We aligned ?ATAC-seq? datasets from ?? cell types to the draft human pangenome to produce coverage tracks as indexed GAF files. On average, there were about ?? paths representing high read coverage which were ?? nodes (?? bases) long. Thousands of paths with high ATAC-seq read coverage traversed more than three nodes, i.e. regions of the pangenome with variation (see 1).

Table 1: High coverage tracks from ?? functional datasets on the HPRC pangenome.

Dataset	paths	average bases	average nodes	traversing >2 nodes
Exp1 Cell typeA	??			?? (??%)
Exp1 Cell typeA	??			
Exp1 Cell typeA	??			
Exp1 Cell typeA	??			
Exp1 Cell typeA	??			

It took on average ?? cpu.hours to map the reads to the pangenome using VG Giraffe, and ?? cpu.hours to produce the coverage tracks. Sorting, compressing and indexing them took only ?? cpu.hours, on average. Table 2 compiles the runtimes and memory used for each step across all samples.

`example of what we could look for and describe` Fig 1 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?.
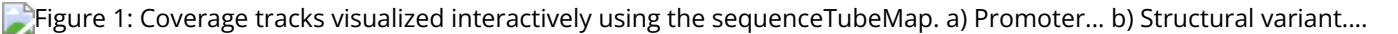

Figure 1: Coverage tracks visualized interactively using the sequenceTubeMap. a) Promoter… b) Structural variant….

Figure 1: 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

After this overview of the current "`vg`" options, it is clear that more needs to be done to make it a useful solution for the community. For example, we are assuming 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, except through reanalysis/reannotation of each haplotype. Another limitation is that the annotation information is currently reduced to a single label. For many annotations, it would be useful to keep the metadata organized, so that the user can access/use it

within visualization tools. Overall, we are also in need for visualization tools that can efficiently layout and organize many *paths* through a pangenome.

- No metadata recorded, all in one path name.
- Simplistic handling of clipped paths.
- Optimized for short paths/ranges.
- Requires ordered integer node IDs for best performance

Code and data availability

- vg https://github.com/vgteam/vg ("gafidx" branch)
  - docker: quay.io/jmonlong/vg:gafidx
- Bandage NG v2022.09 https://github.com/asl/BandageNG
- sequenceTubeMap https://github.com/vgteam/sequenceTubemap
  - docker: quay.io/jmonlong/sequencetubemap:

References

1. **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

2. **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 2: 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
Exp1 Cell typeA	??	?? cpu.hours (?? Gb)		
Exp1 Cell typeA	??	?? cpu.hours (?? Gb)		
Exp1 Cell typeA	??	?? cpu.hours (?? Gb)		
Exp1 Cell typeA	??	?? cpu.hours (?? Gb)		
Exp1 Cell typeA	??	?? cpu.hours (?? Gb)		
Exp1 Cell typeA	??	?? cpu.hours (?? Gb)		
Exp1 Cell typeA	??	?? cpu.hours (?? Gb)		
Exp1 Cell typeA	??	?? cpu.hours (?? Gb)		
Exp1 Cell typeA	??	?? cpu.hours (?? Gb)		