**Motivating example**

In a pangenome graph, each vertex represents a DNA sequence. One sample is represented as a walk through the graph, from a start vertex to an end vertex, and its DNA sequence is the concatenation of those vertices. Genetic variants are ways that walks may differ. Most regions of the human genome have a simple graphical structure, such that genetic variants form a linear sequence of separate “bubbles”, which can be described as differences with a linear reference genome (Figure). In such regions, a walk can be described as a sequence of independent alleles.

In other regions, the graph contains interlocking and nested structure. Some walks may skip some branches, bypassing certain variants entirely; others may visit a branch repeatedly, even having both alleles of a variant in series; sequences may be visited in a reverse order due to inversions. In Figure 1a, an example is given of a pangenome graph with 10 nodes and five walks. The linear reference (Figure 1b) is the walk (1,2,3,4). However, some individuals have an insertion at node 2, involving nodes 5, 6, 7 and 8. One possible walk (Figure 1c) is (1,2,5,6,7,8,2,3,4). However, there is nested structure within the insertion: another walk (Figure 1d) skips part of the insertion (node 5) and repeats a different part (nodes 6,7,8). A fourth walk (Figure 1e) skips node 2 and the possible insertion entirely. Finally, walk five (Figure 1f) skips node 2 initially, but visits a part of the insertion, and node 2, after it has visited nodes 3 and 4.

One possible approach is to consider the entire region (nodes 1 through 8) as a single variant with many alleles - likely, a different allele for every sample. This is the approach of reference [], which defines genetic variants as non-overlapping “superbubbles”. A limitation of this approach is that it misses similarities between different alleles: for example, walks 4 and 5 both skip node 2 (and the possible insertion), even though walk 5 returns to it later. Walks 2 and 3 both visit the insertion, even though walk 2 visits each inserted node once while walk 3 skips some and visits others twice. Even with a limited number of nodes and edges, the number of possible walks can be very large, and assigning a new allele for every unique walk is unparsimonious..

We define genetic variants using a *reference tree*, which is a spanning tree of the pangenome graph (Figure 1g). This tree includes all of the nodes of the pangenome graph, and a subset of its edges. Every edge which is not in the reference tree is a *variant edge* (Figure 1h). Variant edges are sufficient to reconstruct any walk: given the number of times a walk visits each variant edge, it is possible to calculate the number of times that it visits every edge. Moreover, no smaller set of biallelic variants has this property.

**A diagram of a diagram

Description automatically generated with medium confidence**

**Figure 1. A visual representation of walks in a pangenome graph.** **(a)** Pangenome graph with 10 nodes. **(b)** The linear reference genome. **(c)** Walk 2 introduces an insertion at node 2. **(d)** Walk 3 skips part of the insertion at node 5, visiting nodes 6, 7, and 8 twice. **(e)** Walk 4 bypasses node 2. **(f)** Walk 5 skips node 2 initially, but revisits part of the insertion before completing the sequence. **(g)** The reference tree constructed using depth-first search. **(h)** Variant edges.

The linear reference (Figure 1b) is the only walk, beginning and ending at the start and end vertices, that does not visit a variant edge. The only possible walk that visits variant edges (8, 7) and (8, 2) is the second walk (Figure 1c). The third walk visits (2, 6) , (8, 6), and (8, 2) once and the variant edge (6, 7) twice; again, it is the only walk that does so (Figure 1.d). Enumerating the variant edges as {(1,3), (2,6), (6,7), (8, 2), (8, 6)}, the genotype vectors of the first tree walks are (0,0,0,0,0), (0,0,1,0,1,0), and (0,1,1,1,2), respectively.

The set of variant edges is akin to a choice of basis for a vector space. They are sufficient to uniquely identify any walk, and they are independent, meaning that each of them provides non-redundant information. The choice of reference tree is logically parallel to the choice of reference allele in a linear genome reference: different choices yield different reference vs. alternative alleles, but the variants themselves remain constant.

**Pangenomes as bidirected graphs**

Pangenome graphs are *bidirected* due to the strandedness of DNA. A bidirected graph is an ordered pair , where is the set of vertices, is a set of directed edges, and is a complement function denoted with the properties that:

1. , and for all
2. For all , .

In a bidirected graph, a *binode* is the pair ; a biedge is . A *walk* between binodes and is a pair of sequences of nodes, , such that for . A walk is called a *path* if it has no repeated binodes: .

**Reference tree and variant edges**

A bidirected graph is a *rooted tree* with root if for all other binodes , there exists a unique path between and . Such a bidirected graph is inversion-free: its nodes can be partitioned into two components with no edges between them, and , such that if then . Before finding the reference tree, we pre-specify the direction of each node (see below). A bidirected graph is a *subgraph* of a bidirected graph if is a bidirected graph and and , and called a *spanning* subgraph if . A *rooted spanning tree* of a bidirected graph is a spanning subgraph that is a rooted tree.

Given the pangenome graph and a rooted spanning tree, called a *reference tree*, we define *variant edges* as the biedges of that are not in . The genotype of an individual, represented as a walk through the graph, is the number of times that that walk visits each variant edge. We use the notation for the number of times that walk visits biedge .

Given the sequence of variant edges visited by walk , as well as its starting and ending nodes, can be reconstructed uniquely because there is at most path between any two nodes in the reference tree, in particular from the ending point of one variant edge to the starting point of the next one. Moreover, given the genotype of a walk – which tracks variant edge visit counts, but not their order – it is possible to reconstruct the number of times that it visits every biedge.

**Theorem 1:** *Given a bidirected pangenome graph , its reference tree , and a bidirected walk between two binodes, the genotype determines up to the order of cycles.*

**Proof:** [To do; see PangenomeGraph.count\_edge\_visits]

**The position of variants along the linear reference**

It is important in practice that variants have a well-defined position with respect to the linear reference genome. We assign positions to variants by making an appropriate choice of reference tree. In particular, we choose a depth-first search (DFS) tree whose first branch is the linear reference genome. If a node is not on the linear reference, it is assigned the largest position of any node on the linear reference that is able to reach in . Equivalently, this is the largest position of any node on the linear reference that is able to reach in without visiting a back-edge with respect to .

The position of a binode is defined as the linear reference position of (or ) if either one lies on . Otherwise, it is the position of vertex on the that serves as the ancestor of (or ) in . The position of a variant edge in , denoted by , is defined as the interval , where is the position of and is the position of .

**Theorem 2:** *If position of is then is determined by the variant edges whose position overlaps :*

**Proof**: [To do]

Theorem 2 shows that our definition of position not only has the linear nature of genome but also allows us to work locally.

**Direction of nodes within each binode**

The reference is constructed via depth-first search on the positive-direction subgraph of the bidirected pangenome graph. Before doing so, we assign directions to each node of the graph.

A subgraph of a bidirected graph is *a partition subgraph* if it is a spanning subgraph and there is no connected component of the underlying undirected graph of containing both and for any . Such a graph induces a non-unique partition of into two components, labeled and , with no directed edges between them, such that if , then .

A partition subgraph of a bidirected graph is called *maximally connected* if it has the fewest connected components out of all partition subgraphs.

**Proposition 1.** Any spanning forest of the underlying undirected graph of a directed graph induces a maximally connected partition subgraph of .

Using Proposition 1, we perform a DFS traversal and partition into and such that the number of inversion edges is minimized, meaning that during DFS traversal, any switch in orientation increases the number of inversion edges.

**Reference and Alternate Alleles**

A variant edge , in conjunction with the reference tree , introduces two unique, internally disjoint paths from the lowest common ancestor of and in , called the *branch point,* toand . Let be the path from to , excluding both and , and let be the path from to , excluding but including . The reference sequence associated with variant edge is defined as the sequence derived from and the alternate sequence is the sequence derived from .

**Algorithms and complexity**

*Step 1. Directing the Binodes.*The algorithm begins by directing the nodes of the bidirected graph using depth-first search (DFS) traversal (**Figure 2.a** and **2.b**). The search is initialized at the reference path (GRCh38), whose nodes are assigned the positive direction; their complements are assigned the negative direction. When visiting a node, both that node and its complement are marked as visited, and never visited again. The node that is visited is assigned a positive direction; its complement is negative. After performing the search, every binode comprises a positive-direction node and a negative-direction node.

Edges cannot be oriented in the same way because for an inversion going from a positive to a negative node, its complement also goes from a positive to a negative node. We designate the *representative edge* of each biedge to be whichever edge is contained in the .gfa file.

*Step 2. Adding terminal nodes.*This step involves modifying the graph to ensure that a spanning DFS tree can be constructed. We add two artificial binodes, called the positive terminus and the negative terminus. The positive node of the positive terminus has an out-edge to every positive node having in-degree zero. The positive node of the negative terminus has an in-edge from every positive node having out-degree zero. This way, the only source and sink nodes are the termini. In addition, the starting node and ending node of every walk is connected to the appropriate terminus. This allows the start and end edges of a walk to be encoded as variant edges. We call these special edges terminal edges and exclude them from most analyses because they are artificial.

*Step 3. Constructing the reference tree.* We perform depth-first search on the positive-direction subgraph. Again, the DFS traversal is initialized at the linear reference (GRCh38). All biedges that do not contain an edge in the DFS tree are defined as variant edges. We record the representative edge of each such biedge.

*Step 4. Positions.*To identify the position of a variant edge we need to find the ancestors of and within . To do this task efficiently, we walk up by reverse the edge direction of and find the shortest path between (and ) to . Walking up is efficient since is a DFS tree and each node has a unique parent.

*Step 4. Finding Reference and Alternate sequences.* The reference sequence associated with variant edge is defined as the sequence derived from the two unique paths from the lowest common ancestor of nodes and (called branch point), to . To identify the branch point of all variants efficiently, I utilize Tarjan's off-line lowest common ancestors algorithm.20

*Step 5. Classification of Variant Edges.* Last, variant edges are classified into three reference-dependent categories based on their structural characteristics relative to the reference tree:

**- Forward edges:** These correspond to **deletions**, where the walk skips over a segment that is present in the reference tree. (**Figure 2.c**)

**- Backward edges:** These represent **duplications**, where the walk includes a segment multiple time.

**- Crossing edges:** These signify **replacements**, where one segment of a walk is replaced by another segment. Single Nucleotide Polymorphisms (SNPs) and Multiple Nucleotide Polymorphisms (MNPs) are crossing edges with singleton reference and alternate sequences with one or multiple variant edges with the same position, respectively. Insertions are crossing edges whose reference allele is empty. (**Figure 2.c**)

**- Inversions:** These edges connect nodes with opposite orientations (positive and negative directions); they cause a segment of the genome to be reversed in orientation relative to the reference sequence. (**Figure 2.c**)

*Computational Efficiency and Scalability.* A key strength of this algorithm is its computational efficiency, operating with a time complexity of , where is the number of nodes in the graph. This efficiency is achieved through the use of DFS traversal for node orientation and reference tree construction. Additionally, we employed Tarjan's algorithm to efficiently compute branch points (lowest common ancestors) for all node pairs simultaneously, as the naive approach fails to handle this problem at scale. Both algorithms are highly optimized for large-scale graphs, ensuring scalability for extensive genomic datasets typical of pangenomic studies.

|  |  |
| --- | --- |
| A diagram of a diagram of a diagram  Description automatically generated | **Figure 2: constructing the reference tree.** **(a)** The initial pangenome graph. Binodes are shown as rectangles. This figure only contains one representative edge for each biedge. **(b)** After directing the binodes, switches with . Directing the binodes does not change the topology of the graph. **(c)** After orienting the nodes, we can partition the nodes into positive and negative directions. The reference tree (blue) is defined as a DFS tree on the positive subgraph. The variant edges are red. Variant edge is a forward edge which represents a deletion (delete the ref. sequence of node ). The variant edge is a crossing edge which represents a replacement (replace ref. sequence of with alt. sequence of ). The variant edge is an inversion (reverse the sequence of node , i.e. the ref sequence of and alt. sequence of reverse . |