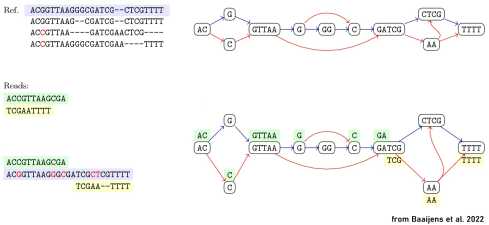
Towards an edit distance between pangenome graphs

Siegfried Dubois[1,2]
Supervised by Thomas Faraut[2], Claire Lemaitre[1] and Matthias Zytnicki[3]

1. Univ Rennes, Inria, CNRS, IRISA, Rennes, F-35000, France
2. GenPhySE, Université de Toulouse, INRAE, ENVT, 31320 Castanet-Tolosan, France
3. MIAT, Université de Toulouse, INRAE, 31320 Castanet-Tolosan, France

DSB - 14th march 2024

Towards a pangenomic era ?



A variation graph :

- contains multiples genomes at once
- stores raw DNA sequences
- paths are genomes and variations

Replacing the reference genome

- allows for higher quality mapping [Eizenga et al. 2020]
- better genotyping of variants [Hickey et al. 2020]

Build a variation graph

From a variant set and a reference

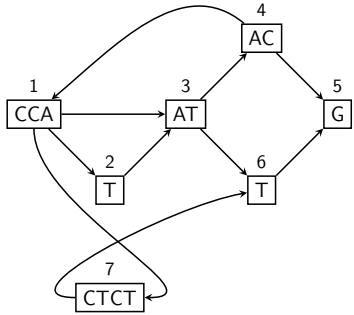
- Variation Graph toolkit (vg) [Hickey et al. 2020]

From a reference and a set of genomes

- minigraph (MG) [Li et al. 2020]
- minigraph-cactus (MGC) [Hickey et al. 2023]

From a set of genomes

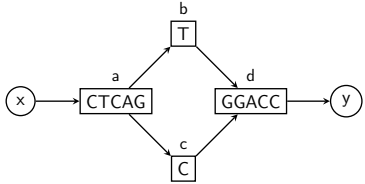
- PanGenome Graph Builder (PGGB) [Garrison et al. 2023]

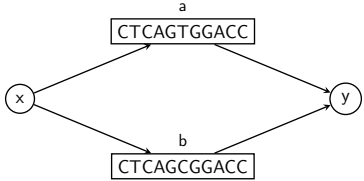


Problem

Graphs obtained from different *state-of-the-art* tools are different :

- number of nodes, edges... are different with the same input data [Leonard et al. 2023, Liao et al. 2023]
- no metric to compare them, nor to locate where the differences are





Definition of a variation graph

A graph $G = (V, E)$ represents a set of genomes $\Gamma = \{\Gamma_0, \Gamma_1, \ldots \Gamma_n\}$:

- each node $u \in V$ is associated to a string (or its *reverse-complement*) which is in at least one genome
- each arc $e \in E$ links two nodes which strings are contiguous in at least one genome and conveys the reading direction

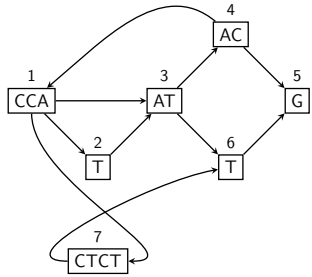


Figure 1 – Variation graph

Complete pangenome graph

We will say that a graph $G(\Gamma) = (V, E, P)$ is a **complete pangenome graph** if :

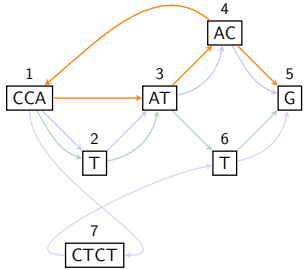
- the graph has a set P of paths (ordered and oriented list of nodes in the graph)
- each path P_i describes the segmentation of a genome Γ_i

$\Gamma_1 = \text{CCATATACG}$

$\Gamma_2 = \text{CCATATTG}$

$\Gamma_3 = \text{CCAATACCCAATACG}$

$\Gamma_4 = \text{CCAAGAGTG}$



$P_1 = 1^+, 2^+, 3^+, 4^+, 5^+$

CCA,T,AT,AC,G

$P_2 = 1^+, 2^+, 3^+, 6^+, 5^+$

CCA,T,AT,T,G

$P_3 = 1^+, 3^+, 4^+, 1^+, 3^+, 4^+, 5^+$

CCA,AT,AC,CCA,AT,AC,G

$P_4 = 1^+, 7^-, 6^+, 5^+$

CCA,AGAG,T,G

Figure 2 – Complete pangenome graph

Idea of our distance

We want to compare two graphs :

- A graph induces a segmentation for each genome
- A difference in segmentation implies different nodes

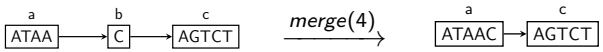
For each genome, we will simultaneously go through both segmentations

- Prevents an all-against-all comparison of the nodes between the two graphs
- They can be computed as independant subproblems

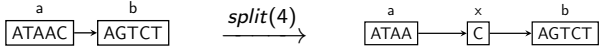
Editions on segmentation

Merges and splits are **editions** on a **segmentation** of a genome.

- **merge(x)** removes the breakpoint at the x^{th} genome position



- **split(x)** adds a breakpoint at the x^{th} genome position



For a genome with two segmentations, there exists a pair of sets M (merges) and S (splits) which allows to transform one segmentation to another. The union of the pair is an edition script.

Definition of our distance

Let $A = (V^A, E^A, P^A)$ and $B = (V^B, E^B, P^B)$ be two complete pangenome graphs with a shared set of genomes Γ.

The segmentation distance of the genome Γ_i will be :

$$d_s(P_i^A, P_i^B) = min(|M| + |S|) \tag{1}$$

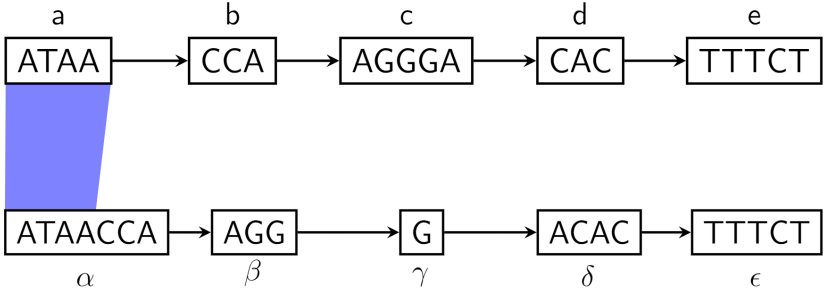
The distance is the sum of the segmentation distances of each genome :

$$d(A, B) = \sum_{i=1}^{|P|} d_s(P_i^A, P_i^B) \tag{2}$$

Algorithm

We compare the two segmentations while we maintain three variables :

- i and j current breakpoint indexes in each segmentation
- p current interval we are looking at

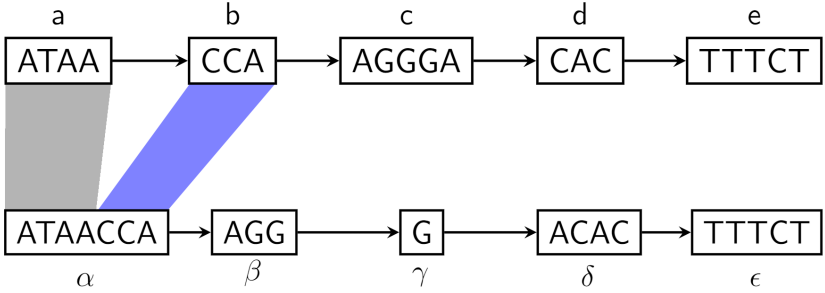


$M = \{\emptyset\}, S = \{\emptyset\}, i = 0, j = 0, p = [1, 4)$

Algorithm

We compare the two segmentations while we maintain three variables :

- i and j current breakpoint indexes in each segmentation
- p current interval we are looking at

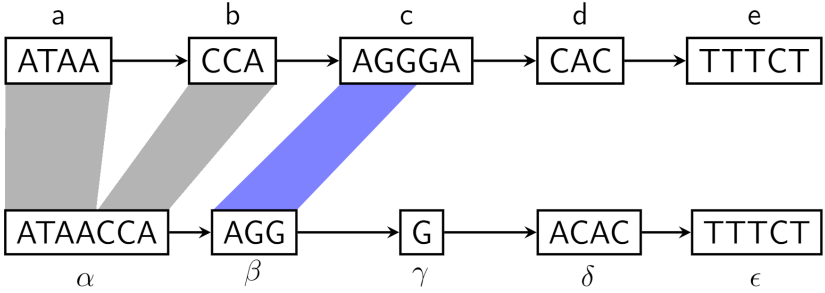


$M = \{4\}, S = \{\emptyset\}, i = 1, j = 0, p = [4, 7)$

Algorithm

We compare the two segmentations while we maintain three variables :

- i and j current breakpoint indexes in each segmentation
- p current interval we are looking at

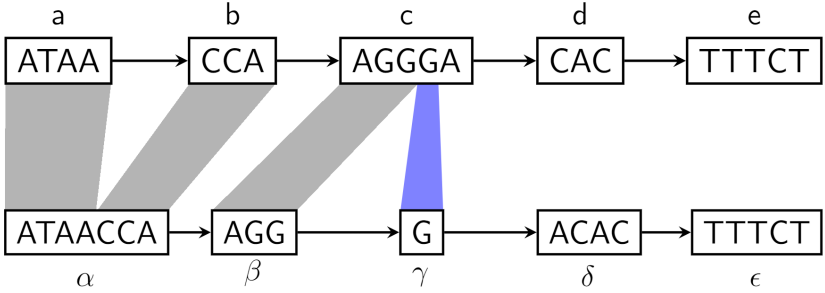


$M = \{4\}, S = \{\emptyset\}, i = 2, j = 1, p = [7, 10)$

Algorithm

We compare the two segmentations while we maintain three variables :

- i and j current breakpoint indexes in each segmentation
- p current interval we are looking at

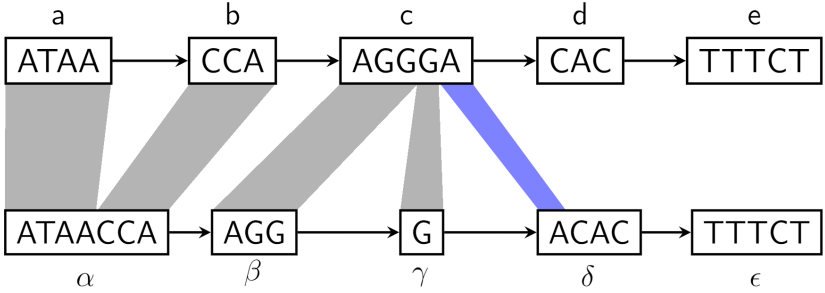


$M = \{4\}, S = \{10\}, i = 2, j = 2, p = [10, 11)$

Algorithm

We compare the two segmentations while we maintain three variables :

- i and j current breakpoint indexes in each segmentation
- p current interval we are looking at

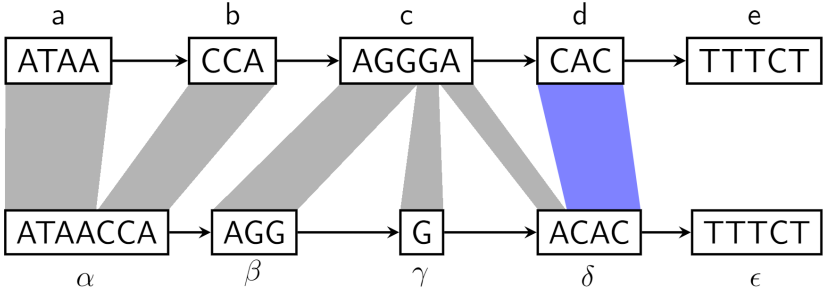


$M = \{4\}, S = \{10, 11\}, i = 2, j = 3, p = [11, 12)$

Algorithm

We compare the two segmentations while we maintain three variables :

- i and j current breakpoint indexes in each segmentation
- p current interval we are looking at

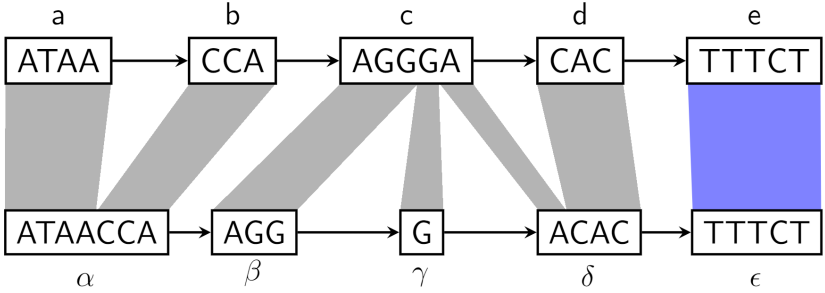


$M = \{4, 12\}, S = \{10, 11\}, i = 3, j = 3, p = [12, 15)$

Algorithm

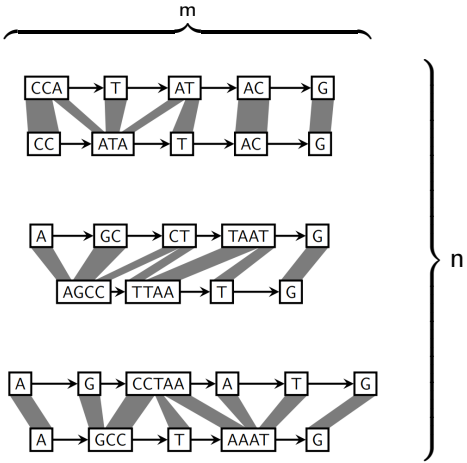
We compare the two segmentations while we maintain three variables :

- i and j current breakpoint indexes in each segmentation
- p current interval we are looking at



$M = \{4, 12\}, S = \{10, 11\}, i = 4, j = 4, p = [15, 20)$

Complexity



$$O(n \times m)$$

With n being the number of genomes and m the length of the genome

PANCAT

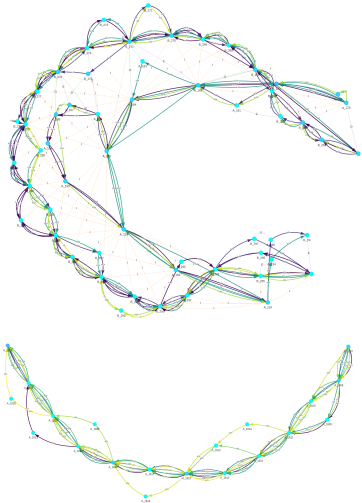
We implemented this algorithm in a Python tool, along with :

- diverse tools for variation graphs
- visualization that displays graphs side-to-side
- library to read, write and edit variation graphs

Regarding distance calculation :

- yeast data, chromosome 1 ($n = 15$ genomes, $n \approx 200$kb, ≈ 50k nodes) : **2.6 sec**.
- HPRC data, chromosome 21 ($n = 90$ genomes, $m = 48$Mb, 2.8M nodes) : **482 sec**.

https://github.com/Tharos-ux/pancat



Data

We build graphs from yeast assemblies [O'Donnell et al. 2023] with *MGC* and *PGGB*

- assemblies for the chromosome 1 of 15 individuals (200kb)
- both tools with default parameters
- multiple graphs for *MGC*, changing genome orders
- one graph for *PGGB*, which is *reference-free*

PGGB MGC

Figure 3 – Visualization with Bandage [Wick et al. 2015]

Distance between MGC graphs

Changing the reference in MGC has a greater impact than changing the order of inclusion of other sequences.

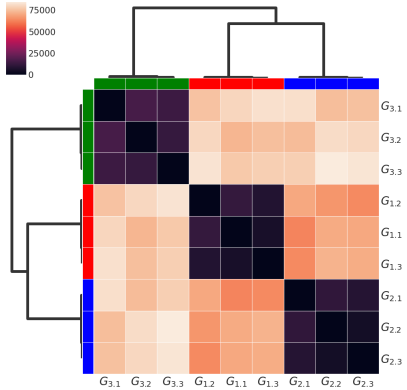


Figure 4 – Green, blue and red markers denotes a shared reference.

Distance between PGGB and MGC graphs

Choice of the reference with MGC has more impact than switching to PGGB.

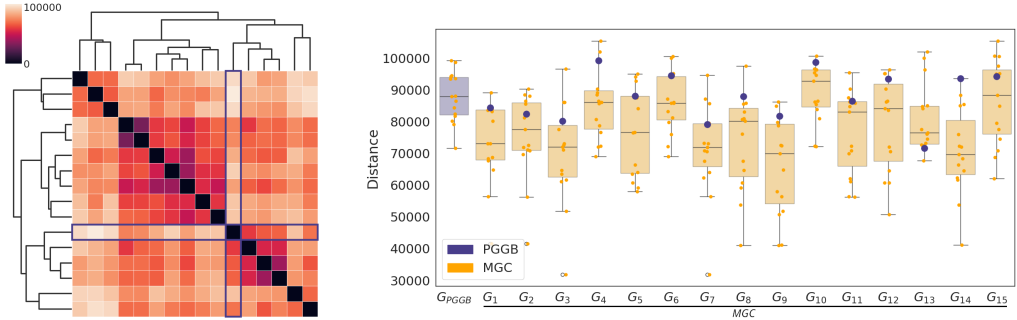


Figure 5 – Blue dots are distances to the PGGB graph, highlighted in blue on the clustermap.

Difference distribution

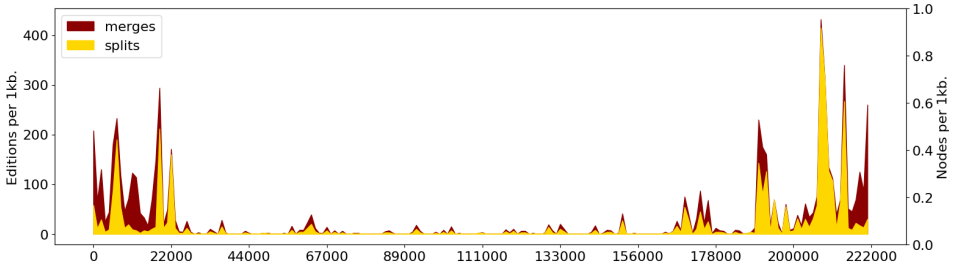


Figure 6 – Editions between the two segmentations of a single genome in a selected MGC graph and the PGGB graph.

Edition is concentred in spots.

Difference distribution

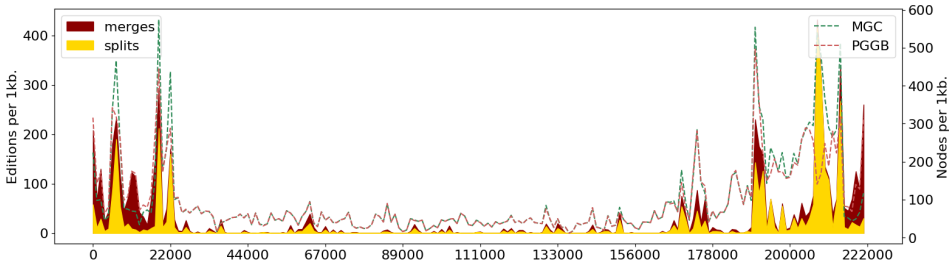
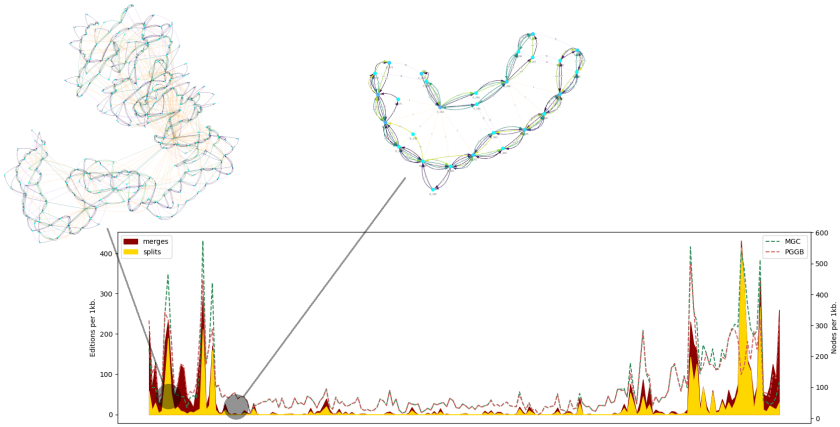


Figure 7 – Editions and nodes distribution between the two segmentations of a single genome in a selected MGC graph and the PGGB graph.

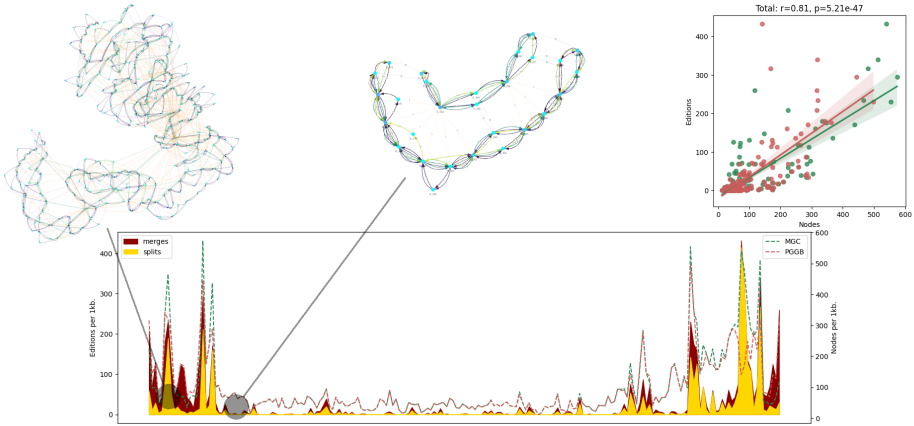
Edition is concentred in spots, and follows partially the distribution of the nodes.

Difference distribution



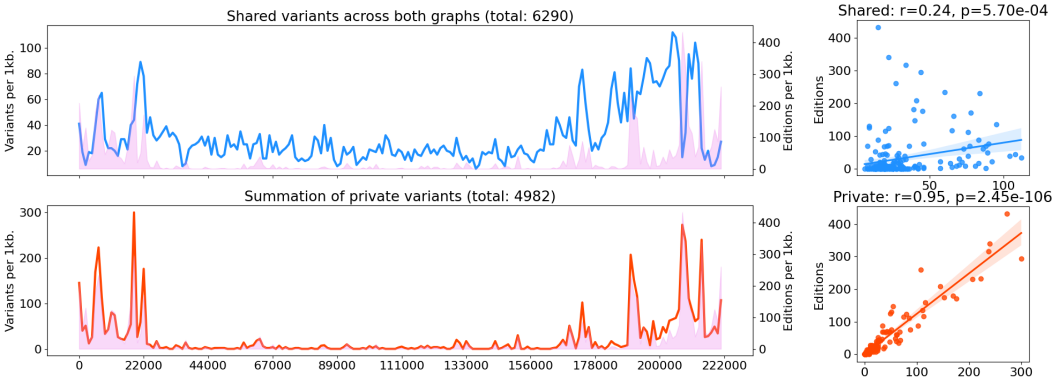
The number of nodes does not fully explain the edition distribution.

Difference distribution



The number of nodes does not fully explain the edition distribution.

Editions and variants



It exists a *quasi*-perfect match between editions and private variants.

Contributions

We defined a distance between pangenome graphs :

- measure that relies on genome segmentation
- possible to pinpoint differences between graphs

We proposed an implementation :

- developped in Python
- a series of utilities, including a visualization tool
- with a library to parse variation graph formats

Perspectives

Distance is computed at *path-level* and not *graph-level*

- Topology differences does not count as editions
- The same operation on multiple paths is accounted multiple times
- Complexity can be lowered to $O(m)$, with m being the number of nodes
- No *out-of-the-box* adaptation for *graph-level*, because of symmetry property

For subsequent works, we want to investigate structural variation in complex sites.

- Explore huge graphs such as the HPRC [Liao et al. 2023]
- Can we find patterns on how splits and merges are related to the topology ?

Is it possible to modify pangenome graphs to reduce differences that are not biologically significant ?

Thanks for your attention !

We acknowledge the GenOuest bioinformatics core facility for providing the computing infrastructure.

PhD founded by the Agroecology and digital technology program.

The tool, **pancat**, is available here : https://github.com/Tharos-ux/pancat
The **gfagraphs** library for GFA files is here : https://pypi.org/project/gfagraphs

Definition of a variation graph

A graph $G = (V, E)$ represents a set of genomes $\Gamma = \{\Gamma_0, \Gamma_1, \ldots \Gamma_n\}$:

- each node $u \in V$ is associated to a string (or its *reverse-complement*) which is in at least one genome
- each arc $e \in E$ links two nodes which strings are contiguous in at least one genome and conveys the reading direction

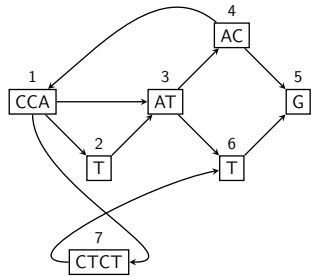


Figure 8 – Variation graph

Definition of a pangenome graph

A graph $G = (V, E, P)$ can be extended by a set $P = \{P_1, P_2 \dots P_n\}$ of paths :

- ordered and oriented list of nodes in the graph
- segmentation of a single embedded genome

We can say G expresses Γ_i if a path P_i is a segmentation of Γ_i

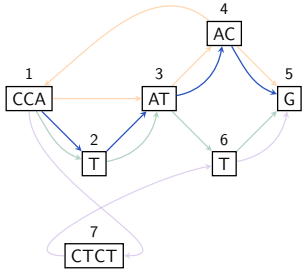


Figure 9 – Paths in a pangenome graph

$P_1 = 1^+, 2^+, 3^+, 4^+, 5^+$

CCA,T,AT,AC,G

$P_2 = 1^+, 2^+, 3^+, 6^+, 5^+$

CCA,T,AT,T,G

$P_3 = 1^+, 3^+, 4^+, 1^+, 3^+, 4^+, 5^+$

CCA,AT,AC,CCA,AT,AC,G

$P_4 = 1^+, 7^-, 6^+, 5^+$

CCA,AGAG,T,G

Complete pangenome graph

We will say that a graph $G = (V, E, P)$ is a **complete pangenome graph** if :

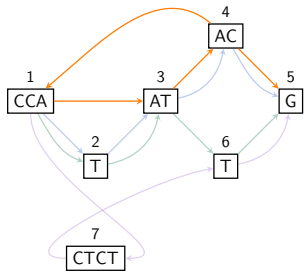
- the graph has a set P of paths
- there's one path per genome ($|P| = |\Gamma|$)
- the graph expresses Γ

Γ_1 = CCATATACG

Γ_2 = CCATATTG

Γ_3 = CCAATACCCAATACG

Γ_4 = CCAAGAGTG



$P_1 = 1^+, 2^+, 3^+, 4^+, 5^+$

CCA,T,AT,AC,G

$P_2 = 1^+, 2^+, 3^+, 6^+, 5^+$

CCA,T,AT,T,G

$P_3 = 1^+, 3^+, 4^+, 1^+, 3^+, 4^+, 5^+$

CCA,AT,AC,CCA,AT,AC,G

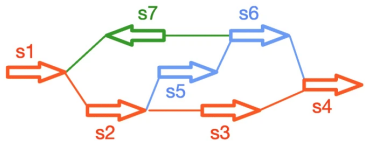
$P_4 = 1^+, 7^-, 6^+, 5^+$

CCA,AGAG,T,G

Figure 10 – Complete pangenome graph

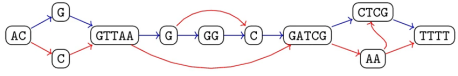
Genome Fragment Assembly

```
S s1 CTGAA   SN:Z:chr1   SO:i:0    SR:i:0
S s2 ACG     SN:Z:chr1   SO:i:5    SR:i:0
S s3 TGGC    SN:Z:chr1   SO:i:8    SR:i:0
S s4 TGTGA   SN:Z:chr1   SO:i:12   SR:i:0
S s5 TTTC    SN:Z:foo    SO:i:8    SR:i:1
S s6 CTGA    SN:Z:foo    SO:i:12   SR:i:1
S s7 GTTAC   SN:Z:bar    SO:i:5    SR:i:2
L s1 + s2 + 0M SR:i:0
L s2 + s3 + 0M SR:i:0
L s3 + s4 + 0M SR:i:0
L s2 + s5 + 0M SR:i:1
L s5 + s6 + 0M SR:i:1
L s6 + s4 + 0M SR:i:1
L s1 + s7 - 0M SR:i:2
L s7 - s6 + 0M SR:i:2
```



from Li et al. 2020

```
Ref.  ACGGTTAAGGGCGATCG--CTCGTTTT
      ACGGTTAAG--CGATCG--CTCGTTTT
      ACCGTTAA----GATCGAACTCG----
      ACCGTTAAGGGCGATCGAA----TTTT
```



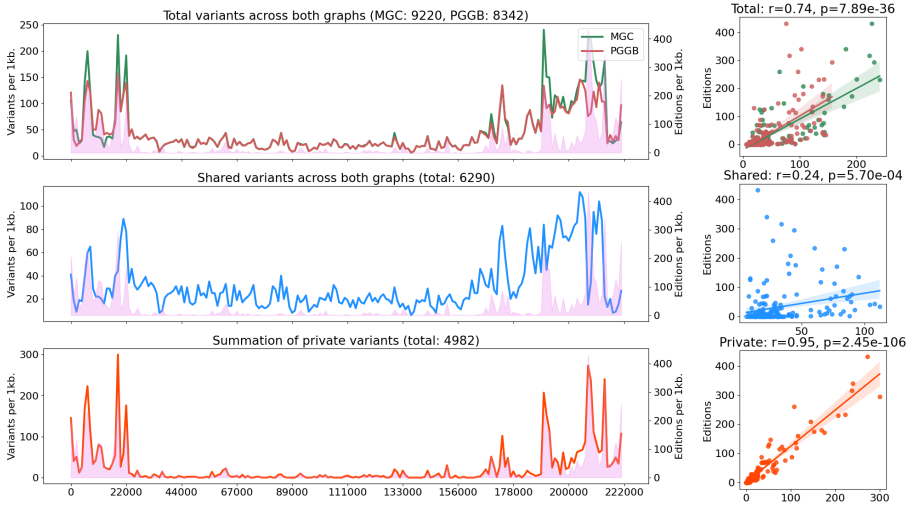
from Baaijens et al. 2022

Algorithm

Algorithm 4 *Computing distance*

1: ▷ p_1 and p_2 are paths that contains nodes $s_0, s_1 \dots s_n$ and $t_0, t_1 \dots t_m$
2: **Input** p_1, p_2 with $p_1 \cong p_2$
3: **Init** $E[] \leftarrow \emptyset$
4: ▷ i and j are the curent positions (in nodes) on p_1 and p_2
5: **Init** $i, j \leftarrow 0$
6: ▷ We iterate while we're not at the end of p_1
7: **while** $|p_1| > i$ **do**
8: **Init** $q \leftarrow \sum_{k=0}^{i} |s_k|$
9: **Init** $E[i] \leftarrow \emptyset$
10: **while** $q \geq \sum_{k=0}^{j} |t_k|$ **do**
11: ▷ We compute the relation r between s_i and t_j
12: $E[i] \leftarrow E[i] + r$
13: $j \leftarrow j + 1$
14: **end while**
15: $i \leftarrow i + 1$
16: **end while**
17: **return** E

Editions and variants



Loss of symmetry

Distance is computed at *path-level* and not *graph-level*

- $d_{path}(G_1, G_2) = 2 \equiv d_{path}(G_2, G_1) = 2$
- $d_{graph}(G_1, G_2) = 1$
- $d_{graph}(G_2, G_1) = 2$

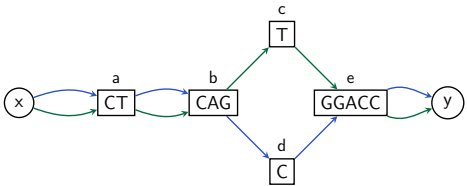


Figure 11 – G_1

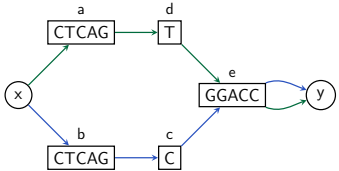


Figure 12 – G_2