Quantifying the Rearrangement Complexity of Pangenomes

Leonard Bohnenkämper[1][0000−0003−4508−0078] and Jens Stoye[1][0000−0002−4656−7155]

Faculty of Technology and Center for Biotechnology (CeBiTec), Bielefeld University, Bielefeld, Germany
`lbohnenkaemper@techfak.uni-bielefeld.de`, `jens.stoye@uni-bielefeld.de`

Abstract. The study of evolution between species (phylogenetics) and the study of evolution within a species (population genetics) are highly related, as the same biological mechanisms are fundamental to both fields. Although both have been studied for a long time, their joint study in a unified setting has been prevented by the different time scales they consider and the different data types they employ. A similar discrepancy holds for their whole-genome specializations, comparative genomics and pangenomics. Two active areas in these fields are genome rearrangement studies and graphical pangenomics, respectively. Since the emergence of graphical pangenomics, these have existed as separate fields, despite observations that central data structures representing genomic variants in both fields are highly similar.

While there exists a wealth of theoretical results for various rearrangement models in comparative genomics, the application to pangenomic data is hampered by the limitations of rearrangement problem formulations. On the practical side, pangenomes typically contain too many individual genomes for classical problems, such as the often NP-hard parsimony problems, to be solved, or for all-vs-all comparisons using rearrangement distances to be performed. On the theoretical side, some assumptions in the formulation of rearrangement problems, such as the assumption of an underlying tree, are inadequate for many pangenomes.

In this work, we propose the *Complete Ancestral Reconstruction for Pangenomes (CARP)* problem, which overcomes these limitations while retaining intuitive relationships to both classical rearrangement problems and pangenome graphs.

Keywords: Pangenomics · Rearrangement Analysis · Single Cut or Join.

1 Introduction

Rearrangements of genetic material have been described even before the molecular structure of DNA was known [44, 4, 15]. These rearrangements and other structural variations have been discovered not only to be ubiquitous, but also to play a significant role in evolution [49, 16, 19], as well as in genetic diseases like cancer [7, 43, 3, 29]. Four years before the first whole eukaryotic genome was sequenced, David Sankoff proposed a framework to computationally quantify rearrangements between two genomes, the rearrangement distance problem [42]. Thirty years later, this basic distance problem has been solved for many different types of rearrangement models, some with simple results, like the Single-Cut-or-Join (SCJ), Double-Cut-and-Join (DCJ) and block interchange models [8, 18, 51], and some with more involved results, like the Reversal, Hannenhalli-Pevzner (HP) and DCJ-indel models [6, 21, 22], to name a few.

In the intervening decades, advances in sequencing technology have greatly enhanced the available material for analysis. This enabled the simultaneous study of several genomes in a clade – the field of pangenomics emerged. With advances in assembly quality, pangenomics itself moved from studying collections of genes [46] to collections of sequences [5] to studying entire genomes [47]. These sets of genomes are typically compressed for analysis, either via lossless text indices such as the Burrows Wheeler Transformation [41, 52], or via lossy graphs such as colored de Bruijn [34, 26, 10] or variation graphs [20, 17]. There exist many different ways to quantitatively characterize pangenomes, such as openness [46, 37] or core percentage [25]. Many of these are already possible using gene-based pangenomes. One truly novel possibility, now that pangenomes are constructed from complete chromosomes, is the quantification of their structural complexity.

Examining the graphical representations of pangenomes, quantifying this complexity is intuitively possible. For example, in Fig. 1, one can hardly deny that Graph B is structurally more complex than Graph A. Of course, an objective, numerical measure is more desirable, especially in less obvious instances.

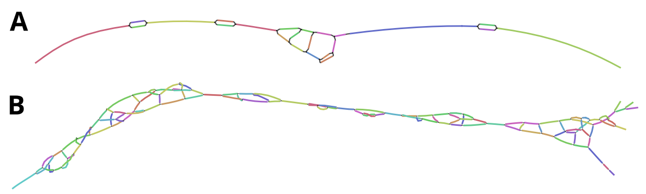

Fig. 1: 1000 BP regions in two compacted de Bruijn graphs ($k = 31$) built with Bifrost [26] and visualized with Bandage [48]. A - built from 56 *Yersinia pestis* genomes. B - built from 48 *Escherichia coli* genomes.

If one were to consider a "pangenome" of exactly 2 genomes, classical rearrangement analyses already give an answer to this quantification, namely the aforementioned rearrangement distance under a certain model. Notably, the rearrangement distance across various models can be quantified using the *breakpoint graph* [21, 22, 51, 9]. This graph bears remarkable similarities to pangenome graphs. Specifically the close relationship between breakpoint and de-Bruijn graphs has been examined in [1] and [32].

Interestingly, the breakpoint graph of two identical genomes, whose distance is therefore 0 under classical rearrangement models, corresponds to a graph without any branching nodes, as seen in the bottom graph in Fig. 3. A pangenome with a graph like this has arguably minimal rearrangement complexity. Therefore, we say such a graph and any pangenome with such a graph is *simple*.

In this work, we propose the *Complete Ancestral Reconstruction for Pangenomes (CARP)* problem (Section 2), a new rearrangement problem designed to quantify the structural complexity of a pangenome based on how many graph altering operations, under a given model, it takes to transform the pangenome into a simple pangenome (see also Fig. 3). We then show how this problem relates to classical rearrangement problems (Section 3). We give a solution to this problem for the SCJ model (Section 4) and demonstrate in a number of experiments (Section 5) how even under this simple model, CARP can be used to derive valuable insights about pangenomes and pangenome graphs. This includes comparative analyses of the rearrangement complexities between different pangenomes, different pangenome graphs constructed from the same sequence set and different regions in the same pangenome graph. We further discuss the potential of the CARP problem formulation (Section 6), particularly once solutions for more advanced rearrangement models become available.

2 A new rearrangement problem

In a pangenome graph, each node represents a sequence segment. We call these segments *markers*. An edge connecting two nodes represents an *adjacency*, meaning that these two markers are immediate neighbors on a chromosome. To that end, an edge can either connect to the beginning or to the end of a marker. Each chromosome is then represented as a path or a cycle in this graph, depending on whether it is linear or circular.

A notable difference between pangenome and breakpoint graphs is that the latter aim to represent only rearrangements, and not local variations such as SNVs or small indels. To capture only structural variations, one can use genes or *collinear blocks* as markers. Collinear blocks can be generated as a by-product of whole genome alignment, such as in Mummer [14] or Mauve [11, 12], but some pangenome graph construction methods, such as Minigraph-Cactus [23] or Pangraph [35], are also able to output collinear blocks. Additionally, there is dedicated software for collinear block detection, such as SibeliaZ [33].

What one considers a rearrangement, is to some degree dependent on a size threshold – note that one could even consider the deletion of a single base a "structural" variation. We will therefore presume that, using one of the aforementioned methods, a suitable level of abstraction that defines markers has been found.

Formally, a marker is a symbol from a large alphabet Σ. A *chromosome* is then associated with a string of oriented markers $S \in (\Sigma \cup \overline{\Sigma})^+$ where $\overline{\Sigma} := \{\overline{m} \mid m \in \Sigma\}$ and $\overline{m}$ denotes marker m in reverse orientation. We call the string S a *chromosome string*. A convenient way of writing a marker m in this context is by referring to its *extremities*, that is, to its beginning m^t (*tail*) and to its end m^h (*head*). Then $m = m^t m^h$ and $\overline{m} = m^h m^t$. Two neighboring markers m and n in a string induce a pair of neighboring extremities, an *adjacency*. Depending on their relative orientation, adjacencies are $m^h n^t$, $m^t n^t$, $m^t n^h$ and $m^h n^h$ for mn, $\overline{m}n$, $\overline{m}\,\overline{n}$ and $m\overline{n}$, respectively. The identity of an adjacency is purely determined by the involved extremities, not their order, i.e. $m^h n^t$ is the same adjacency as $n^t m^h$.

A chromosome may be *linear*, in which case we denote it by $[S]$, or circular, in which case we denote it by (S). A circular chromosome $(s_1 \dots s_l)$ in addition to the adjacencies of $s_1 \dots s_l$ has the additional adjacency induced by $s_l s_1$. For a linear chromosome $[S]$ it is helpful to define the additional adjacencies $\varnothing u$ and $v\varnothing$ for the extremities u and v with which the chromosome starts and ends. We call $\varnothing$ a *telomere*. Note that we regard chromosomes as equal even when denoted in reverse orientation, i.e. $[\bar{s}_n \dots \bar{s}_1] = [s_1 \dots s_n]$ and circular chromosomes as equal even when denoted in different orientation $(s_k \dots s_n s_1 \dots s_{k-1}) = (s_1 \dots s_{k-1} s_k \dots s_n)$. A *genome* is a multiset of chromosomes, and a *pangenome* is a multiset of genomes.

We can now define the pangenome graph that represents the order of these markers as a bi-edged graph [38]. For ease of notation, we write the (undirected) edge between vertices u and v as uv or vu.

Definition 1. *The* Multiple Breakpoint Graph (MBG) $\mathcal{G}(\mathbb{P})$ *of a pangenome* $\mathbb{P}$ *is an undirected graph* $(V, E_{adj} \cup E_{mark})$ *where* V *represents the extremities of* $\mathbb{P}$ *including the telomere node* $\varnothing$ *and* E_{adj}, E_{mark} *are two types of edges:*

- E_{adj} *is a set of undirected edges, which is the union of all adjacencies in* $\mathbb{P}$. *We call* E_{adj} adjacency edges.
- E_{mark} *is a set of undirected edges that contains the edge* $m^t m^h$ *for each marker* m *in addition to a self edge* $\varnothing\varnothing$ *of the telomere* $\varnothing$. *We call* E_{mark} marker edges.

In principle, any graph in `gfa` format can be represented as a MBG if telomeres are added to those nodes at the end of linear paths. However, note that the theory presented here assumes all variants represented in the graph are captured by the rearrangement model used. An example of a MBG is found in Fig. 2.

Recall the earlier motivation of a *simple* pangenome without any branching paths in its pangenome graph as having minimal structural complexity. We can now formulate this more precisely. A pangenome and its MBG are *simple* if for each marker m (excluding $\varnothing$) at most one adjacency edge connects to its beginning m^t and at most one adjacency edge connects to its end m^h. We then wish to quantify the structural complexity of a pangenome by the number of

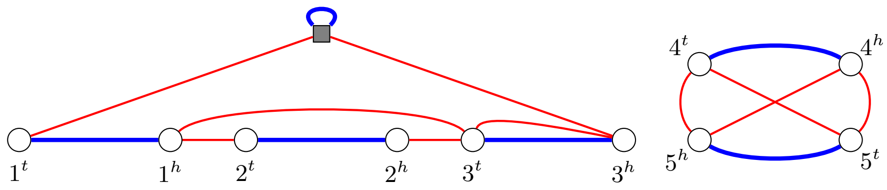

Fig. 2: Multiple Breakpoint Graph (MBG) for the pangenome $\{\{[123],(45)\},\{[133],(45)\},\{[133],(4\bar{5})\}\}$. Blue edges are marker edges, red edges are adjacency edges, and the quadratic grey node is the telomere $\varnothing$.

rearrangement steps necessary to transform it into a simple pangenome. The operations used for this transformation are most meaningful when modeled after biological events, such as reversals or recombinations that occur in chromosomes of the pangenome. They must thus fundamentally operate on chromosomes, i.e. the paths in the graph, not the graph itself. However, not all operations have the same effect on the MBG. There are operations that never affect the structure of the MBG, such as the duplication of a chromosome or the recombination of two chromosomes, as shown in the left box of Fig. 3. Other operations, typically ones that are associated with breaking a chromosome, may have an effect on the graph. Such an operation is shown in the right box of Fig. 3. We call these types of operations Path Altering Operations and Graph Altering Operations, respectively. Formally, an operation o is a *Path Altering Operation (PAO)* if, after applying it to a pangenome $\mathbb{P}$, denoted $\mathbb{P} \xrightarrow{o} \mathbb{P}'$, the resulting pangenome $\mathbb{P}'$ has the same MBG as $\mathbb{P}$, i.e. $\mathcal{G}(\mathbb{P}') = \mathcal{G}(\mathbb{P})$. Otherwise it is a *Graph Altering Operation (GAO)*.

Classical rearrangement operations, such as reversal, translocation, block interchange and DCJ, are all GAOs. In fact, one can reformulate many classical problems as transforming a set of input genomes into the closest set of genomes with a simple MBG by using the model operation and a limited set of supplementary PAOs while minimizing the number of model operations used. We examine this relationship for the distance, median, large parsimony and halving problems in Section 3.2.

However, in these formulations the interaction between lineages is strictly limited – owing to the fact that most classical rearrangement problems (implicitly) assume an underlying tree. In fact, the only PAOs necessary to model these classical problems are coalescence (unifying equal genomes) and some form of chromosome de-duplication (unifying equal chromosomes in the same genome). In a pangenome, however, the interactions between lineages can be much more frequent and complex, such as the transfer of plasmids between bacteria or the sexual reproduction of many eukaryotes. We therefore propose to use a "maximally powerful" set of PAOs to quantify the rearrangement complexity of a pangenome. We call such a set *complete*. Formally, we say a set of PAOs is *complete* if it is able to transform any two pangenomes with the same MBG into each other. Such a set exists; we give it in Theorem 1 (Section 3.3) and we prove that it is complete in Appendix B.

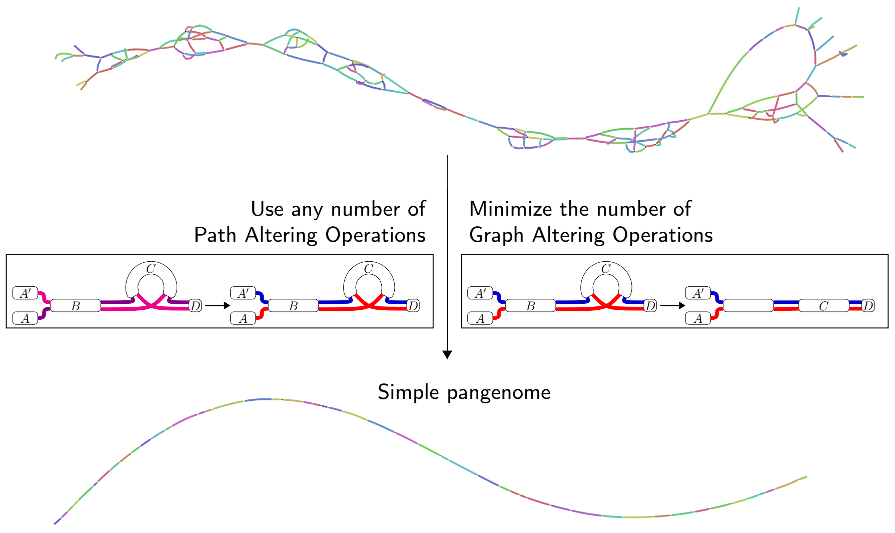

Fig. 3: Schematic representation of the CARP rearrangement problem: A pangenome is transformed into the closest simple pangenome using the minimum number of Graph Altering Operations while utilizing arbitrarily many operations from a complete set of Path Altering Operations. An example of a Path Altering Operation (a homologous recombination between the chromosome $A'B\bar{C}D$ (pink) and the chromosome $ABCD$ (purple) at marker B, transforming them into $A'BCD$ (blue) and $AB\bar{C}D$ (red)) is shown in the left box. An example of a Graph Altering Operation (a reversal of marker C in the chromosome $AB\bar{C}D$ (red)) is shown in the right box. Note that both types of operations change paths directly, but only Graph Altering Operations change edges in the graph.

The definition for the *Complete Ancestral Reconstruction for Pangenomes (CARP)* problem is then as follows: Given a pangenome, a set of GAOs and a complete set of PAOs, find a simple pangenome into which it can be transformed using any number of PAOs from a complete set and a minimal number of GAOs. A visual summary is found in Fig. 3. Formally, we define the problem as follows.

Problem 1 (Complete Ancestral Reconstruction for Pangenomes (CARP)). Given a pangenome $\mathbb{P}$, a complete set of PAOs N and a set of GAOs A, find the shortest sequence of GAOs $o_1 \dots o_n \in A^*$, such that there is a sequence of operations $s = O_0 o_1 O_1 \dots o_n O_n$ transforming $\mathbb{P}$ into a simple pangenome $\mathbb{A}$ where each O_i is a (possibly empty) sequence of PAOs, $O_i \in N^*$. We call the simple pangenome $\mathbb{A}$ a *CARP simplification*, s a *CARP scenario* and n the *CARP measure* of $\mathbb{P}$.

3 Relationship to classical rearrangement problems

CARP is related to classical rearrangement problems, such as the distance, median and large parsimony problem. In fact, all of these can be expressed as specializations of a common superproblem, the *General Ancestral Reconstruction Problem*, which we examine in Section 3.2. In essence, these problems differ only by which path altering operations they permit. We thus first examine a few path altering operations in Section 3.1. Finally, we discuss a complete set of path altering operations in Section 3.3.

3.1 Path Altering Operations relevant to classical problem formulations

In this section, we give examples of PAOs that are useful to understand the relationship between CARP and classical rearrangement problems (see Section 3.2). One of the most fundamental PAOs is the emergence of new lineages, which when regarded backwards in time is known as *coalescence*. In the following, we use the multi-set addition $\oplus$ and multi-set subtraction $\ominus$.

Definition 2 (Coalescence/Divergence). *Given a pangenome $\mathbb{P}$ and genomes $G_1, G_2, \ldots, G_k \in \mathbb{P}$ with $G_1 = G_2 = \ldots = G_k$, a* k-coalescence *on genomes $G_1, G_2, \ldots, G_k$ creates the pangenome $\mathbb{P}' = \mathbb{P} \ominus \{G_2, \ldots, G_k\}$. Given a pangenome $\mathbb{P}$ and a genome $G_1 \in \mathbb{P}$, a* k-divergence *on genome G_1 creates the pangenome $\mathbb{P}' = \mathbb{P} \oplus \{G_2, \ldots, G_k\}$ with $G_2 = \ldots = G_k = G_1$.*

As we examine in Section 3.2, many classical problems examine only tree-like evolution without duplicated markers and can thus be formulated directly using only coalescence. However, some classical problems, such as double distance or halving problems, require the modeling of whole genome duplications. To that end, we first define the duplication and de-duplication of a single chromosome.

Definition 3. *A* k-multiplication *($\rightarrow$) transforms a single chromosome in one of the following ways and a* k-de-multiplication *($\leftarrow$) transforms either k identical or one repetitive circular chromosome in one of the following ways:*
$[S_1] \rightleftarrows [S_1], [S_2], \ldots, [S_k] \quad (S_1) \rightleftarrows (S_1), (S_2), \ldots, (S_k) \quad (S_1) \rightleftarrows (S_1 S_2 \ldots S_k)$
where $S_2 = \ldots = S_k = S_1$.

A whole genome duplication is then characterized by duplicating all of a genome's chromosomes. Again, there is an inverse when viewing time "backwards".

Definition 4. *A* genome duplication operation *transforms a genome by 2-multiplying all of its chromosomes. A* genome halving operation *is the inverse of a genome duplication operation.*

These definitions suffice to characterize many classical rearrangement problems as we examine in the following section.

3.2 The General Ancestral Reconstruction Problem (GARP)

Under the assumption that PAOs do not create new structural complexity while a simple set of genomes possess minimal structural complexity, we can quantify the complexity of a set of genomes by measuring how many GAOs it takes to transform it into a simple pangenome while using some supplementary set of PAOs. This formulation (see Problem 2) is then a common framework for CARP and classical rearrangement problems.

Problem 2 (General Ancestral Reconstruction Problem (GARP)). Given a pangenome $\mathbb{P}$, a set of PAOs N and a set of GAOs A, find the shortest sequence of GAOs $o_1 \dots o_n \in A^*$, such that there is a sequence of operations $s = O_0 o_1 O_1 \dots \dots o_n O_n$ transforming $\mathbb{P}$ into a simple pangenome $\mathbb{A}$ where each O_i is a (possibly empty) sequence of PAOs, $O_i \in N^*$.

Note that CARP (Problem 1) is a specialization of Problem 2, where the set of PAOs is required to be *complete*. In classical rearrangement problems, this is not the case.

For a simple example from classical rearrangement literature, consider the genomic distance problem [42]. In this problem, two genomes X and Y are given that contain the same set of markers, each exactly once. The goal is to find the minimum number of rearrangements to transform X into Y under a given model M. If each rearrangement operation in M has an inverse, which is typically the case, we can model this in the GARP framework.

We leave the set of PAOs N empty and define the set of GAOs A as the operations of M. Then for each scenario transforming X into Y, $X = X_0 \overset{o_1}{\to} X_1 \overset{o_2}{\to} \dots \overset{o_n}{\to} X_n = Y$, there is a GARP solution with the same number of GAOs, namely $o_1 o_2 \dots o_n$. Note that two genomes without duplicate markers are equal if and only if their adjacencies are equal [2], meaning the MBG constructed from both genomes is simple. Since $X_n = Y$, the final MBG is simple. Thus, the number of GAOs required in the GARP formulation is a lower bound for the distance.

Conversely, the distance also bounds the number of GAOs in the GARP scenario. To see why, we can find a classical rearrangement scenario for any GARP scenario. In this case, the scenario can modify both X and Y with $X_0 \overset{x_1}{\to} \dots \overset{x_k}{\to} X_k$ and $Y_0 \overset{y_1}{\to} \dots \overset{y_j}{\to} Y_j$. Because $\mathcal{G}(\{X_k, Y_j\})$ is simple, $X_k = Y_j$. Thus, we can use the inverse operations $y'_1, \dots, y'_j$ of $y_1, \dots, y_j$ respectively to build the classical scenario $X = X_0 \overset{x_1}{\to} \dots \overset{x_k}{\to} X_k = Y_j \overset{y'_j}{\to} \dots \overset{y'_1}{\to} Y_0 = Y$. The problem formulations are thus equivalent.

A slightly more involved example is the large parsimony problem [2] (called "Multiple Genome Rearrangement Problem" there). In this problem, extant genomes $X_1, \dots, X_n$ are given and one is asked to find a binary tree T with $X_1, \dots, X_n$ at the leaves and genomes $Y_1, \dots, Y_{n-1}$ at the internal nodes, such that the total distance along the edges $E(T)$, $\sum_{(U,V)\in E(T)} d(U,V)$, is minimized. Under the assumption that the operations of M are symmetric, the following

GARP formulation is equivalent: In Problem 2, let $\mathbb{P} = \{X_1, \dots, X_n\}$, let N contain only 2-coalescence operations, and let A be the operations of M. We show how to obtain a GARP solution from the large parsimony solution. Assume the children V, W of an internal node U exist in the current pangenome. From the large parsimony solution we can deduce operations $v_1 \dots v_k$ transforming U into V and operations $w_1 \dots w_j$ transforming U into W. Since operations have an inverse, there are operations $v'_k \dots v'_1$ and $w'_j \dots w'_1$ transforming V and W into U, which we can use in the GARP scenario and finally use a 2-coalescence to unify the genomes into U. We can apply this procedure bottom up to arrive at the corresponding GARP scenario. Conversely, one can find the corresponding large parsimony problem solution for a GARP scenario by using the genomes unified by the 2-coalescence as inner nodes and reversing the scenarios to be top-down. In fact, this is how Alekseyev and Pevzner constructed the MGRA heuristic for the large parsimony problem for 2-breaks (DCJ) in [2], although they do not explicitly use coalescence, but instead rely on the concept of "strict transformations", i.e. such transformations that conform to a phylogeny.

In the same way one can find a GARP formulation for many classical rearrangement problems. We give a selection in Table 1. We note that coalescence is the only interaction between genomes in these problems. As mentioned earlier, due to the phylogenetic closeness of the genomes in a pangenome, the interactions in a pangenomic context may well be more frequent and direct. For our pangenome focused specialization of GARP, CARP, we permit much more interaction by using a complete set of PAOs. We give one example of such a complete set in the following section.

3.3 A complete set of Path Altering Operations

We will now show that there is a set of operations, which is a complete set of PAOs and describes meaningful biological interactions. In fact, we do so by extending the PAOs already described in Section 3.1. We only add a direct way by which genomes interact and an operation modeling chromosomal recombination.

While there are many ways how genomes can interact, the most direct one is by transferring chromosomal material.

Definition 5 (Chromosome Transfer). *Given a pangenome $\mathbb{P}$ that includes genomes G_1, G_2 and a multiset of chromosomes $C \subseteq G_1$, a chromosome transfer between G_1 and G_2 creates $\mathbb{P}' = \mathbb{P} \ominus \{G_1, G_2\} \oplus \{G'_1, G'_2\}$ where $G'_1 \coloneqq G_1 \ominus C$ and $G'_2 \coloneqq G_2 \oplus C$.*

If multiple copies of the same chromosome exist in the same genome, new chromosome structures can arise by recombinations, such as shown in Fig. 4. These homologous recombinations are one mechanism of how rearrangements occur [40].

We can define a homologous recombination in our context roughly as taking two (sub-)strings SmT and UmV, $m \in \Sigma$, of one or two chromosomes and transforming them into SmV and UmT. Depending on the connectivity of these substrings, this then has different effects. More formally:

Table 1: Classical genome rearrangement problems on a model M with symmetric operations and an equivalent GARP problem (see Problem 2). A dash (–) represents that no PAOs are needed while brackets indicate those PAOs that one can include into the problem formulation without losing the equivalence. In each case the set of GAOs A equals the set of operations of M and $\mathbb{P}$ is the multiset containing all input genomes.

Problem name	Ref.	Given genomes	Classical Formulation	PAOs (N)
Distance problem	[42]	Genomes X, Y on the same marker set (single copy)	Find the minimum number of operations in M to transform X to Y	– (2-coalescence)
Median problem	[45]	Genomes $X_1, \ldots, X_k$ on the same marker set (single copy)	Find a genome Y that minimizes the total distance to $X_1, \ldots, X_k$	– (k-coalescence)
Large parsimony problem	[2]	Genomes $X_1, \ldots, X_k$ on the same marker set (single copy)	Find a binary tree with leaves $X_1, \ldots, X_k$ and genomes $Y_1, \ldots Y_{k-1}$ as inner nodes with minimal total distance between the genomes at each edge	2-coalescence
Genome halving problem	[45]	Genome Y with each marker occurring twice	Find a genome X in which each marker occurs once, such that the genome obtained by doubling X minimizes the distance to Y	genome halving
Guided genome halving problem	[45]	Genome Y with each marker occurring twice and genome Z with each marker occurring once	Find a genome X in which each marker occurs once, such that the distance of the genome obtained by doubling X to Y plus the distance of X to Z is minimized	genome halving (2-coalescence)

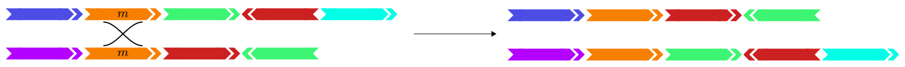

Fig. 4: A homologous recombination may create new chromosome structures not previously present in the pangenome.

Definition 6. *A* homologous recombination *transforms chromosomes of the same genome in one of the following ways*

$$\begin{array}{rcll} [SmT], [UmV] & \rightleftarrows & [SmV], [UmT] & \textit{(Translocation)} \\ [SmT\overline{V}\overline{m}\overline{U}] & \rightleftarrows & [SmV\overline{T}\overline{m}\overline{U}] & \textit{(Inversion)} \\ (SmT\overline{V}\overline{m}\overline{U}) & \rightleftarrows & (SmV\overline{T}\overline{m}\overline{U}) & \textit{(Inversion)} \\ [SmT], (UmV) & \rightleftarrows & [SmVUmT] & \textit{(Circular inclusion/excision)} \\ (SmT), (UmV) & \rightleftarrows & (SmVUmT) & \textit{(Circular fusion/fission)} \end{array}$$

where $m \in \Sigma$ *and* $S, T, U, V \in (\Sigma \cup \overline{\Sigma})^*$.

Note that a homologous recombination is a PAO, because while the chromosome connectivity is transformed, the adjacencies (particularly at marker m) remain the same and therefore the graph remains unchanged.

Transfer and recombination in conjunction with chromosome (de-) multiplication, coalescence and divergence are then a complete set of PAOs, as we show in Appendix B.

Theorem 1. *The set of operations $\tilde{N}$, which consists of (de-) multiply, homologous recombination, coalescence, divergence and chromosome transfer operations is a complete set of PAOs.*

Note that we only discussed a small subset of PAOs here. There are many more possible PAOs. Some rearrangement operations which have been studied recently, such as flanked transpositions [50] or flanked block interchanges [30], are in fact PAOs, although none of them form a complete set by themselves. Nonetheless, this means the complete set of PAOs $\tilde{N}$ discussed here is not the only such set. However, since PAOs are used without a cost in Problem 1, the concrete set of PAOs does not change the CARP measure or CARP simplification as long as the set is complete.

4 Solution under the SCJ model

The *Single Cut or Join (SCJ)* model, established by Feijão and Meidanis [18], is a simple rearrangement model closely related to the breakpoint distance. The model allows to cut a chromosome in one place (single cut) or to join two ends from one or two linear chromosomes together (join). These two operations have analogues in biology, namely double strand breaks and non-homologous end joining. However, care needs to be taken in interpreting genomes reconstructed by SCJ since events with breakpoint reuse are not reflected in SCJ scenarios.

Nonetheless, it is a useful model as many otherwise NP-hard problems are tractable under SCJ. However, reconstruction without a tree (large parsimony problem), remains NP-hard [18]. One might thus expect that CARP is NP-hard as well under SCJ. However, we show here that the problem has a surprisingly simple linear time solution.

Formally, an SCJ is either a *Single Cut* or a *Single Join*. A Single Cut transforms a circular chromosome (S) into a linear chromosome $[S]$ or a linear chromosome $[ST]$ into two linear chromosomes $[S]$ and $[T]$. A *Single Join* transforms a linear chromosome $[S]$ into a circular chromosome (S) or two linear chromosomes $[S]$ and $[T]$ into one linear chromosome $[ST]$. S and T here represent arbitrary strings of oriented markers.

In the MBG, cutting between two markers m, n, i.e. $[SmnT] \to [Sm], [nT]$, creates the telomeric adjacencies $m^h\varnothing, n^t\varnothing$ and removes the adjacency edge $m^h n^t$ if $[SmnT]$ was the only chromosome with that adjacency. Conversely, the join $[Sm], [nT] \to [SmnT]$ creates the adjacency edge $m^h n^t$ and removes $m^h\varnothing, n^t\varnothing$ if $[Sm], [nT]$ are the only linear chromosomes with these (telomeric) adjacencies. The effects for other combinations of orientations and chromosome types are analogous. Note that if uv and $u'v'$ are joined to uu', then $v = v' = \varnothing$.

From this follows that, to arrive at a simple pangenome, we need to expend at least one SCJ operation per non-telomeric adjacency that involves a branching node. We call these adjacencies *contested*. We give an example graph where contested edges are marked in Fig. 5.

Lemma 1. *Transforming a pangenome* $\mathbb{P}$ *with MBG* $\mathcal{G}(\mathbb{P}) = (V, E_{adj} \cup E_{mark})$ *into a simple pangenome* $\mathbb{A}$ *requires at least* $|E_{cnt}|$ *SCJ operations where* $E_{cnt} \coloneqq \{uv \in E_{adj} \mid u = v\} \cup \{uv \in E_{adj} \mid u, v \neq \varnothing, \exists uv' \in E_{adj} \text{ with } v' \neq v\}$. *We call* E_{cnt} *the set of* contested adjacencies.

Proof. We prove this lemma using an injection f between edges in E_{cnt} and SCJ operations in the CARP scenario. Consider an edge $uv \in E_{\text{cnt}}$. We distinguish two cases.

(I) Let uv not be an adjacency in the simple MBG of $\mathbb{A}$. Then there must be a cut c_{uv} removing uv from the graph before $\mathbb{A}$ is reached, i.e. $uv \overset{c_{uv}}{\to} u\varnothing, v\varnothing$. We thus map uv to c_{uv}, i.e. $f(uv) = c_{uv}$.

(II) Let uv be an adjacency in the simple MBG of $\mathbb{A}$. Note that then $u \neq v$, otherwise the final MBG would not be simple. Since uv is contested, there is another adjacency edge at u or at v. Let w.l.o.g. $uv' \in E_{\text{adj}}$ with $v' \neq v$. In the CARP scenario, we may then have a series of alternating cuts and joins exchanging v' for other nodes: $uv' = uv_0 \overset{c_1}{\to} u\varnothing \overset{j_1}{\to} uv_1 \overset{c_2}{\to} u\varnothing \overset{j_2}{\to} uv_2 \ldots \overset{x_n}{\to} uv_n$. The MBG of $\mathbb{A}$, which contains uv, is simple only if $v_n = v \neq \varnothing$. Therefore the last operation in the series must be a join, $x_n = j_n$. We thus map uv to j_n, i.e. $f(uv) = j_n$. $\square$

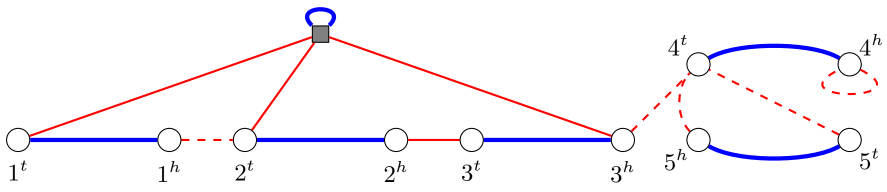

Fig. 5: MBG for the pangenome $\{\{[23], (4\bar{4}5)\}\}, \{[1234\bar{4}\bar{3}\bar{2}]\}\}$ with contested edges shown dashed. Blue edges are marker edges, red edges are adjacency edges, and the quadratic grey node is the telomere $\varnothing$.

We thus need to expend at least one SCJ operation per contested edge. In Appendix C we show that this lower bound is tight. Specifically, one can construct a simplification pangenome by simply cutting each edge in E_{cnt}, which then immediately implies the following:

Theorem 2. *The CARP measure under the SCJ model for a pangenome* $\mathbb{P}$ *with MBG* $\mathcal{G}(\mathbb{P}) = (V, E_{adj} \cup E_{mark})$ *is* $|E_{cnt}|$. *One possible SCJ-CARP simplification* $\mathbb{A}$ *is defined by the adjacencies* $E_{\mathbb{A}} \coloneqq E_{adj} \setminus E_{cnt}$.

CARP under SCJ is thus solvable in time linear with respect to the graph size. Notably, the CARP measure corresponds directly to an easy to count graph metric, namely the edges connecting to branching nodes E_{cnt}. Indeed, this mirrors the results of classical rearrangement theory, where distances often correspond to simple graph structures (e.g. cycles (DCJ distance) or 2-cycles (breakpoint distance)) in specialized graphs, such as the breakpoint graph. The difference here is that the MBG can capture arbitrary (pangenome) graphs.

5 Evaluation

We implemented the SCJ-CARP reconstruction as well as utility functions to filter graph nodes based on size and extracting fixed size regions surrounding nodes from the graph. Relevant for the following experiments are the programs `carp`, which calculates the SCJ-CARP measure optionally filtering small nodes below a threshold and `carp-scan`, which calculates the SCJ-CARP measures for a fixed size region around each node. The implementation as well as snakemake [28] workflows for all following experiments are published on github[1]. The implementation is also available on bioconda under package name `carp`.

We evaluate our method for simulated datasets with regards to its ability to track the number of rearrangement events and to reconstruct ancestral adjacencies, and for its practical implications on biological datasets.

5.1 Simulated experiments

In order to evaluate how well SCJ-CARP quantifies rearragements, we used ZOMBI [13] to simulate phylogenies of genomes from an initial genome with 1000 markers. We used default parameters unless otherwise specified. By default, ZOMBI generates phylogenies with 20 extant genomes.

In order to simulate genomes of varying complexity, we scaled rates in ZOMBI for gene originations, duplications, losses, inversions and transpositions from 0.1 to 1.0 of the default rate. A comparison between this rate scale and the SCJ-CARP measure calculated is shown in Fig. 6 (A). We see that the SCJ-CARP measure tracks the number of simulated rearrangement operations well, particularly for low rates of rearrangements. In fact the Pearson and Spearman correlation coefficients for the entire experiment are 0.89 and 0.91 respectively, indicating a strong correlation between the number of simulated operations and the SCJ-CARP measure.

We then proceeded to evaluate in which way the marker adjacencies in the SCJ-CARP simplification correspond to the ancestral root simulated by ZOMBI. In this analysis, we limited ourselves to non-telomeric adjacencies, since in the SCJ model telomeres are best interpreted as uncertainty about which adjacency should be reconstructed, not as actual ends of chromosomes. We evaluated precision and recall using the adjacencies of the root genome as the ground truth. The results are shown in Fig. 6 (B). We see that the precision is high across all samples with the majority of samples in each step reaching 100% precision. However, recall degenerates quickly with increasing numbers of rearrangements.

[1] https://github.com/gi-bielefeld/scj-carp

This is not suprising as the simplifications reconstructed by `carp` are extremely conservative – like in other problems under SCJ, ancestral genomes tend to fragment unless adjacencies are perfectly conserved.

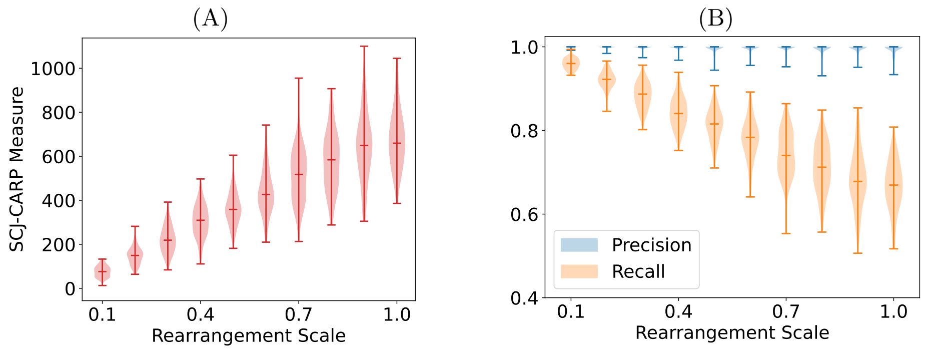

Fig. 6: SCJ-CARP measure (A) and reconstruction precision and recall (B) with increasing numbers of rearrangements.

5.2 Evaluating the SCJ-CARP measure of multiple pangenomes

We downloaded multiple genomes for 8 different taxa using NCBI datasets [36] to compare their structural complexity with SCJ-CARP. We aimed for 10 genomes with NCBI assembly level "complete" when possible and opted for "chromosome+" and a slightly reduced number of genomes when necessary. Details are found in Appendix D. We ran Minigraph-Cactus [23] and SibeliaZ [33] with two different settings to determine collinear blocks. We then ran `maf2synteny` [27] with default settings on the output of both tools.

We evaluated the SCJ-CARP measure on the resulting genome permutation files. Each run of `carp` took less than a second using four threads on a cluster machine. The calculated SCJ-CARP measures are visualized in Fig. 7.

We see that the SCJ-CARP measures rank the relative pangenome complexity consistently even when using different block construction methods. The only disagreements are with respect to the order of *Y. pestis*, *E. coli* and *K. pneumoniae*, which are not well separated using any of the three block types, and the placement of *P. vivax*. Nonetheless, overall correlation between the CARP measures under different block definitions in this experiment is high (see Appendix Fig. 12).

Notably there are vast differences in the absolute SCJ-CARP measures between SibeliaZ and Minigraph-Cactus. Most interesting is the difference for *P. vivax*, for which both SibeliaZ runs evaluate to a much higher SCJ-CARP measure than Minigraph-Cactus, which yields the lowest SCJ-CARP measure overall. We conjecture that this is due to the differing definitions of blocks used by the

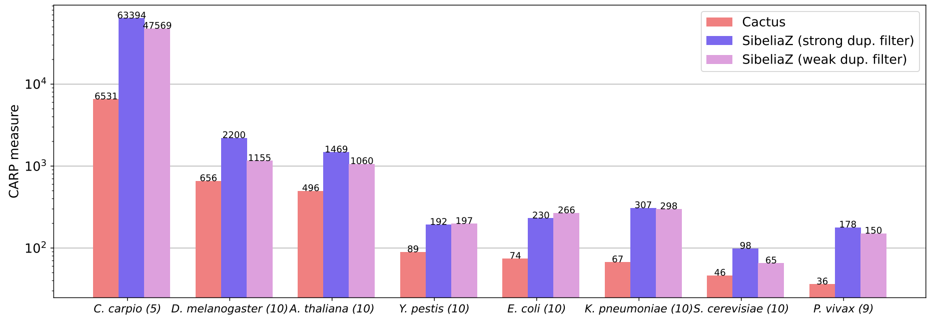

Fig. 7: SCJ-CARP measures on pangenomes with collinear blocks detected by Minigraph-Cactus and SibeliaZ. Two different settings were used for SibeliaZ, a strong duplication filter ($a = 2 \times$ #genomes) and a weak duplication filter ($a = 20 \times$ #genomes). Shown in parentheses behind the taxon name is the number of genomes used as input to the block detection.

two tools. It might also be the case that the SCJ model struggles to capture the rearrangements displayed by SibeliaZ. Another model may score the Minigraph-Cactus and SibeliaZ blocks more similarly. Thus, once solutions to CARP under more complex models are available, the differences between these segmentation approaches may become better explainable.

The discrepancy also emphasizes how crucial the initial block definition is for the comparison of genomes, a known issue in rearrangement analyses. Especially note that the absolute CARP measures on differently determined blocks are usually not comparable. Nonetheless, using blocks determined by the same method, SCJ-CARP gives a relatively coarse but quickly computable measure for the rearrangement complexity of a pangenome.

5.3 Comparing human pangenome graphs using CARP

Given the effect of different marker definitions on the CARP measures observed in the previous section, it stands to reason that the discrepancy between CARP measures of pangenome graphs constructed by different methods can give insights about how these graphs differ structurally. Notably, such structural differences have been observed between Minigraph, Minigraph-Cactus and PGGB during the construction of the first human pangenome reference [31]. There the authors note that Minigraph-Cactus includes smaller variants than Minigraph while still constructing similar nodes for duplicate regions, while PGGB's all-vs-all alignment strategy creates a graph that collapses duplications resulting in an overall more complex graph (see Extended Data Fig. 3 in [31] for an example).

We downloaded the three respective graphs from the human-pangenomics S3 bucket using the CHM13-version of the Minigraph and Minigraph-Cactus graphs. We proceeded to analyze the graphs using `carp`, filtering out nodes below a threshold of progressively larger size. We measured the number of nodes in the graph and the SCJ-CARP measure. We show these results as well as the SCJ-CARP measure relative to the number of nodes in Fig. 8.

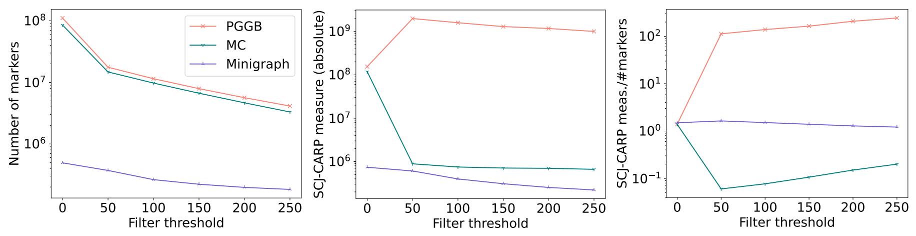

Fig. 8: CARP statistics for various levels of node filtering on human pangenome graphs built by PGGB, Minigraph-Cactus (MC) and Minigraph from [31]. Nodes of a size smaller than the filter threshold were removed and each path bridging two nodes that pass the threshold consisting of only filtered nodes was replaced with a single edge.

Without filtering the nodes (Filter threshold 0), we see that Minigraph is an outlier, while Minigraph-Cactus and PGGB are close together, both in the number of nodes and in the absolute SCJ-CARP measure. Nonetheless, the relative SCJ-CARP measure of all three tools is similar. Progressively filtering nodes, differences between Minigraph, Minigraph-Cactus and PGGB become more apparent. While the number of nodes decreases for all three graphs, again with a similar trajectory for Minigraph-Cactus and PGGB, the SCJ-CARP measure shows that PGGB's complexity increases, while the complexity of Minigraph-Cactus and Minigraph is now much more similar.

From these results one can conclude that while the graphs of PGGB and Minigraph-Cactus have similar numbers of nodes, the way variants are represented in Minigraph and Minigraph-Cactus is much more similar to each other than to PGGB. PGGB in turn tends to produce highly connected small nodes that lead to an increase in connectedness and consequently higher complexity once they are filtered.

These observations are of course consistent with the knowledge that Minigraph-Cactus graphs are constructed by adding smaller variants to Minigraph graphs while PGGB follows a different, alignment-based paradigm. However, note that this knowledge of the graph construction methods is not necessary to interpret the results computed by SCJ-CARP.

On 16 threads, calculations for Minigraph were completed within seconds while calculations for Minigraph-Cactus finished in a matter of minutes. The longest calculation for PGGB completed after about four hours. Overall, the entire analysis was completed within 20 hours. We give all runtimes of SCJ-CARP as Appendix Fig. 13.

We conclude that SCJ-CARP allows an analysis of pangenome graphs that highlights their differing structural qualities without prior knowledge of how these graphs were constructed. Additionally, the time frame for such an analysis is reasonable, taking about one day for human sized pangenomes.

5.4 Scoring region complexities in a *Bacillus subtilis* de Bruijn graph

Typically the fastest way to build a pangenome graph is to construct a de Bruijn graph of all genome sequences in the pangenome of interest. However, de Bruijn graphs tend to be large and hard to interpret, especially in regions with a high complexity ("hair balls"), which typically arise due to nodes that appear very frequently in the pangenome. The SCJ-CARP measure can be used to score the complexity of the subgraph surrounding a node.

To demonstrate this possibility, we downloaded all 1508 available *Bacillus subtilis* genomes in genbank using NCBI datasets [36] and constructed a de Bruijn graph using Bifrost [26] for $k = 31$. The resulting graph had a total number of $3,665,906$ nodes. We then used the `carp-scan` program to extract regions of 300 BP around a node and to score this environment using the SCJ-CARP measure. On 8 threads on a cluster machine, `carp-scan` finished after 8 minutes and 4 seconds. We show the histogram of SCJ-CARP measures of the $3,629,054$ least complex nodes and examples of node environments in Fig. 9.

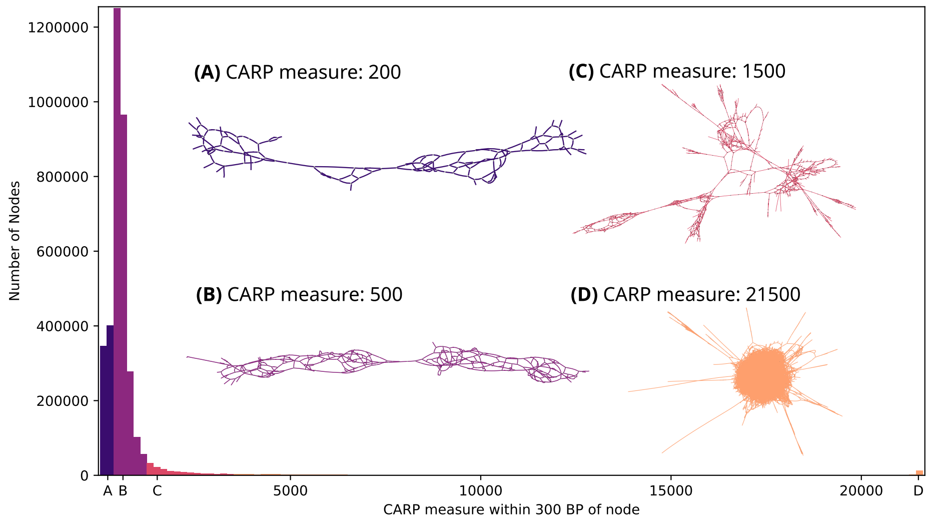

Fig. 9: Histogram of SCJ-CARP measures in the 300 BP environment of nodes in a compacted de Bruijn graph built from 1508 *Bacillus subtilis* genomes. For a better visualization, the histogram is artificially cut after 21500; the highest SCJ-CARP measure of a node environment was 175702. (A) to (D) visualize example environments from distinct regions of the distribution. These visualizations were created with Bandage [48].

We see that the majority of nodes are surrounded by a subgraph with a CARP measure below 1500, which appear to be interpretable (see Examples (A), (B) and (C) in Fig. 9), although ease of interpretation decreases as the complexity identified by the SCJ-CARP measure increases. True "hair ball" subgraphs, such as shown as Example (D) in Fig. 9, seem to be the exception in this graph, but there is a cluster of these nodes around SCJ-CARP measure 21500.

We also performed additional analyses varying the environment size. These histograms can be found in Appendix Fig. 14. Moreover, the `carp-scan` program can color graph nodes based on their surrounding SCJ-CARP measures, which enables to easily visualize where in the graph complexity arises. We give an example in Appendix Fig. 15. Additionally, one can map these complexities back to a reference genome and visualize it with a genome browser. We give an example in Fig. 10 using the UCSC Genome Browser [39].

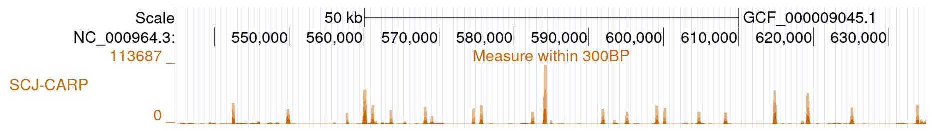

Fig. 10: 300 BP environment complexities in the de Bruijn graph ($k = 31$) mapped to the *Bacillus subtilis* reference `GCF_000009045.1`. Shown is the region `NC_000964.3:534,995-635,004` in the UCSC genome browser [39].

One can thus use SCJ-CARP to identify regions in a pangenome graph that are already interpretable and regions that still need refinement, which could be a useful tool or evaluation metric for pangenome graph construction methods. Moreover, this allows in principle to compare pangenomes and examine at which level structural complexity arises and how it is distributed throughout the pangenome graph.

6 Discussion

We developed CARP, a new rearrangement problem designed to quantify the rearrangement complexity of pangenomes. This problem relates intuitively to both pangenome graphs and classical genome rearrangement, thus forming a first step towards a joint modeling of these two, highly connected areas. We presented a solution to CARP under the simple SCJ model. This solution already enables fast analyses of pangenomes and their graphs, which yield valuable insights about construction methods and structural complexity. We expect this potential to further unfold once solutions under other models become available. For example, one could characterize a pangenome by a vector of CARP measures of different models (i.e. inversions, block interchanges, etc.), indicating which model best explains its complexity. This would also allow a high level comparison between pangenomes that do not share any sequence material. A prerequisite to perform such a comparison is some normalization of the CARP measure. Like other rearrangement-based methods, CARP yields an absolute number of rearrangements, which of course depends on the size and number of the input genomes. Future research may explore normalizing the CARP measure as has been done with other pangenome metrics (see for example [37]).

We have also seen that the CARP measure under the SCJ-model corresponds to a simple to evaluate graph metric in a general pangenome graph, which mirrors the fact that this is the case for many models in classical rearrangement analyses in the specialized breakpoint graph. It thus stands to reason that the CARP measure for other rearrangement models may be related to other known graph theoretic quantities. This would allow the study of abstract graph properties under the more biologically meaningful lens of rearrangement models and, vice versa, the mathematical understanding of a rearrangement model would be enhanced by connecting it to graph theory in general.

Furthermore, it is worth investigating the implications that Path Altering Operations used in a given scenario make about the phylogenetic relationship between genomes of the pangenome. This may well lead to an easily computable rearrangement-based tree reconstruction.

References

1. Alekseyev, M.A., Pevzner, P.A.: Colored de Bruijn graphs and the genome halving problem. IEEE/ACM Transactions on Computational Biology and Bioinformatics **4**(1), 98–107 (2007). https://doi.org/10.1109/TCBB.2007.1002
2. Alekseyev, M.A., Pevzner, P.A.: Breakpoint graphs and ancestral genome reconstructions. Genome Research **19**(5), 943–957 (2009). https://doi.org/10.1101/gr.082784.108
3. Baker, T.M., Waise, S., Tarabichi, M., Van Loo, P.: Aneuploidy and complex genomic rearrangements in cancer evolution. Nature Cancer **5**(2), 228–239 (2024). https://doi.org/10.1038/s43018-023-00711-y
4. Belling, J.: Crossing over and gene rearrangement in flowering plants. Genetics **18**(4), 388 (1933). https://doi.org/10.1093/genetics/18.4.388
5. Bentley, S.: Sequencing the species pan-genome. Nature Reviews Microbiology **7**(4), 258–259 (2009). https://doi.org/10.1038/nrmicro2123
6. Braga, M.D., Willing, E., Stoye, J.: Double cut and join with insertions and deletions. Journal of Computational Biology **18**(9), 1167–1184 (2011). https://doi.org/10.1089/cmb.2011.0118
7. Chen, J.M., Cooper, D.N., Férec, C., Kehrer-Sawatzki, H., Patrinos, G.P.: Genomic rearrangements in inherited disease and cancer. Seminars in Cancer Biology **20**(4), 222–233 (2010). https://doi.org/10.1016/j.semcancer.2010.05.007
8. Christie, D.A.: Sorting permutations by block-interchanges. Information Processing Letters **60**(4), 165–169 (1996). https://doi.org/10.1016/S0020-0190(96)00155-X
9. Compeau, P.E.: DCJ-indel sorting revisited. Algorithms for Molecular Biology **8**(1), 6–6 (2013). https://doi.org/10.1186/1748-7188-8-6
10. Cracco, A., Tomescu, A.I.: Extremely fast construction and querying of compacted and colored de Bruijn graphs with GGCAT. Genome Research **33**(7), 1198–1207 (2023). https://doi.org/10.1101/gr.277615.122

11. Darling, A.C., Mau, B., Blattner, F.R., Perna, N.T.: Mauve: Multiple alignment of conserved genomic sequence with rearrangements. Genome Research **14**(7), 1394–1403 (2004). https://doi.org/10.1101/gr.2289704
12. Darling, A.E., Mau, B., Perna, N.T.: progressiveMauve: Multiple genome alignment with gene gain, loss and rearrangement. PLOS ONE **5**(6), 1–17 (2010). https://doi.org/10.1371/journal.pone.0011147
13. Davín, A.A., Tricou, T., Tannier, E., de Vienne, D.M., Szöllősi, G.J.: Zombi: a phylogenetic simulator of trees, genomes and sequences that accounts for dead linages. Bioinformatics **36**(4), 1286–1288 (2019). https://doi.org/10.1093/bioinformatics/btz710
14. Delcher, A.L., Salzberg, S.L., Phillippy, A.M.: Using MUMmer to identify similar regions in large sequence sets. Current Protocols in Bioinformatics **00**, 10.3.1–10.3.18 (2003). https://doi.org/10.1002/0471250953.bi1003s00
15. Dobzhansky, T., Sturtevant, A.H.: Inversions in the chromosomes of *Drosophila pseudoobscura*. Genetics **23**(1), 28 (1938). https://doi.org/10.1093/genetics/23.1.28
16. Dunham, M.J., Badrane, H., Ferea, T., Adams, J., Brown, P.O., Rosenzweig, F., Botstein, D.: Characteristic genome rearrangements in experimental evolution of *Saccharomyces cerevisiae*. Proceedings of the National Academy of Sciences of the U.S.A. **99**(25), 16144–16149 (2002). https://doi.org/10.1073/pnas.242624799
17. Eizenga, J.M., Novak, A.M., Sibbesen, J.A., Heumos, S., Ghaffaari, A., Hickey, G., Chang, X., Seaman, J.D., Rounthwaite, R., Ebler, J., Rautiainen, M., Garg, S., Paten, B., Marschall, T., Sirén, J., Garrison, E.: Pangenome graphs. Annual Review of Genomics and Human Genetics **21**, 139–162 (2020). https://doi.org/10.1146/annurev-genom-120219-080406
18. Feijao, P., Meidanis, J.: SCJ: A breakpoint-like distance that simplifies several rearrangement problems. IEEE/ACM Transactions on Computational Biology and Bioinformatics **8**(5), 1318–1329 (2011). https://doi.org/10.1109/TCBB.2011.34
19. Feulner, P.G.D., De-Kayne, R.: Genome evolution, structural rearrangements and speciation. Journal of Evolutionary Biology **30**(8), 1488–1490 (2017). https://doi.org/10.1111/jeb.13101
20. Garrison, E., Sirén, J., Novak, A.M., Hickey, G., Eizenga, J.M., Dawson, E.T., Jones, W., Garg, S., Markello, C., Lin, M.F., Paten, B., Durbin, R.: Variation graph toolkit improves read mapping by representing genetic variation in the reference. Nature Biotechnology **36**(9), 875–879 (2018). https://doi.org/10.1038/nbt.4227
21. Hannenhalli, S., Pevzner, P.A.: Transforming men into mice (polynomial algorithm for genomic distance problem). In: Proceedings of IEEE 36th Annual Foundations of Computer Science. pp. 581–592. IEEE Press, Milwaukee, WI, USA (1995). https://doi.org/10.1109/SFCS.1995.492588
22. Hannenhalli, S., Pevzner, P.A.: Transforming cabbage into turnip: polynomial algorithm for sorting signed permutations by reversals. Journal of the ACM **46**(1), 1–27 (1999). https://doi.org/10.1145/300515.300516
23. Hickey, G., Monlong, J., Ebler, J., Novak, A.M., Eizenga, J.M., Gao, Y., Abel, H.J., Antonacci-Fulton, L.L., Asri, M., Baid, G., Baker, C.A., Belyaeva, A., Billis, K., Bourque, G., Buonaiuto, S., Carroll, A., Chaisson, M.J.P., Chang, P.C., Chang, X.H., Cheng, H., Chu, J., Cody, S., Colonna, V., Cook, D.E., Cook-Deegan, R.M., Cornejo, O.E., Diekhans, M., Doerr, D., Ebert, P., Eichler, E.E., Fairley, S., Fedrigo, O., Felsenfeld, A.L., Feng, X., Fischer, C., Flicek, P., Formenti, G., Frankish, A., Fulton, R.S., Garg, S., Garrison, E., Garrison, N.A., Giron, C.G., Green, R.E., Groza, C., Guarracino, A., Haggerty, L., Hall, I.M., Harvey, W.T., Haukness, M., Haussler, D., Heumos, S., Hoekzema, K., Hourlier, T., Howe, K., Jain,

M., Jarvis, E.D., Ji, H.P., Kenny, E.E., Koenig, B.A., Kolesnikov, A., Korbel, J.O., Kordosky, J., Koren, S., Lee, H., Lewis, A.P., Liao, W.W., Lu, S., Lu, T.Y., Lucas, J.K., Magalhães, H., Marco-Sola, S., Marijon, P., Markello, C., Marschall, T., Martin, F.J., McCartney, A., McDaniel, J., Miga, K.H., Mitchell, M.W., Mountcastle, J., Munson, K.M., Mwaniki, M.N., Nattestad, M., Nurk, S., Olsen, H.E., Olson, N.D., Pesout, T., Phillippy, A.M., Popejoy, A.B., Porubsky, D., Prins, P., Puiu, D., Rautiainen, M., Regier, A.A., Rhie, A., Sacco, S., Sanders, A.D., Schneider, V.A., Schultz, B.I., Shafin, K., Sibbesen, J.A., Sirén, J., Smith, M.W., Sofia, H.J., Tayoun, A.N.A., Thibaud-Nissen, F., Tomlinson, C., Tricomi, F.F., Villani, F., Vollger, M.R., Wagner, J., Walenz, B., Wang, T., Wood, J.M.D., Zimin, A.V., Zook, J.M., Li, H., Paten, B., The Human Pangenome Reference Consortium: Pangenome graph construction from genome alignments with minigraph-cactus. Nature Biotechnology **42**(4), 663–673 (2024). https://doi.org/10.1038/s41587-023-01793-w

24. Hierholzer, C., Wiener, C.: Über die Möglichkeit, einen Linienzug ohne Wiederholung und ohne Unterbrechung zu umfahren. Mathematische Annalen **6**(1), 30–32 (1873). https://doi.org/10.1007/BF01442866
25. Hiller, N.L., Janto, B., Hogg, J.S., Boissy, R., Yu, S., Powell, E., Keefe, R., Ehrlich, N.E., Shen, K., Hayes, J., Barbadora, K., Klimke, W., Dernovoy, D., Tatusova, T., Parkhill, J., Bentley, S.D., Post, J.C., Ehrlich, G.D., Hu, F.Z.: Comparative genomic analyses of seventeen *Streptococcus pneumoniae* strains: Insights into the pneumococcal supragenome. Journal of Bacteriology **189**(22), 8186–8195 (2007). https://doi.org/10.1128/jb.00690-07
26. Holley, G., Melsted, P.: Bifrost: highly parallel construction and indexing of colored and compacted de Bruijn graphs. Genome Biology **21**(1), 249 (2020). https://doi.org/10.1186/s13059-020-02135-8
27. Kolmogorov, M., Armstrong, J., Raney, B.J., Streeter, I., Dunn, M., Yang, F., Odom, D., Flicek, P., Keane, T.M., Thybert, D., Paten, B., Pham, S.: Chromosome assembly of large and complex genomes using multiple references. Genome Research **28**(11), 1720–1732 (2018). https://doi.org/10.1101/gr.236273.118
28. Köster, J., Rahmann, S.: Snakemake—a scalable bioinformatics workflow engine. Bioinformatics **28**(19), 2520–2522 (2012). https://doi.org/10.1093/bioinformatics/bts480
29. Krupina, K., Goginashvili, A., Cleveland, D.W.: Scrambling the genome in cancer: causes and consequences of complex chromosome rearrangements. Nature Reviews Genetics **25**(3), 196–210 (2024). https://doi.org/10.1038/s41576-023-00663-0
30. Li, T., Jiang, H., Zhu, B., Wang, L., Zhu, D.: Flanked block-interchange distance on strings. IEEE/ACM Transactions on Computational Biology and Bioinformatics **21**(2), 301–311 (2024). https://doi.org/10.1109/TCBB.2024.3351440
31. Liao, W.W., Asri, M., Ebler, J., Doerr, D., Haukness, M., Hickey, G., Lu, S., Lucas, J.K., Monlong, J., Abel, H.J., Buonaiuto, S., Chang, X.H., Cheng, H., Chu, J., Colonna, V., Eizenga, J.M., Feng, X., Fischer, C., Fulton, R.S., Garg, S., Groza, C., Guarracino, A., Harvey, W.T., Heumos, S., Howe, K., Jain, M., Lu, T.Y., Markello, C., Martin, F.J., Mitchell, M.W., Munson, K.M., Mwaniki, M.N., Novak, A.M., Olsen, H.E., Pesout, T., Porubsky, D., Prins, P., Sibbesen, J.A., Sirén, J., Tomlinson, C., Villani, F., Vollger, M.R., Antonacci-Fulton, L.L., Baid, G., Baker, C.A., Belyaeva, A., Billis, K., Carroll, A., Chang, P.C., Cody, S., Cook, D.E., Cook-Deegan, R.M., Cornejo, O.E., Diekhans, M., Ebert, P., Fairley, S., Fedrigo, O., Felsenfeld, A.L., Formenti, G., Frankish, A., Gao, Y., Garrison, N.A., Giron, C.G., Green, R.E., Haggerty, L., Hoekzema, K., Hourlier, T., Ji, H.P., Kenny,

E.E., Koenig, B.A., Kolesnikov, A., Korbel, J.O., Kordosky, J., Koren, S., Lee, H., Lewis, A.P., Magalhães, H., Marco-Sola, S., Marijon, P., McCartney, A., McDaniel, J., Mountcastle, J., Nattestad, M., Nurk, S., Olson, N.D., Popejoy, A.B., Puiu, D., Rautiainen, M., Regier, A.A., Rhie, A., Sacco, S., Sanders, A.D., Schneider, V.A., Schultz, B.I., Shafin, K., Smith, M.W., Sofia, H.J., Abou Tayoun, A.N., Thibaud-Nissen, F., Tricomi, F.F., Wagner, J., Walenz, B., Wood, J.M.D., Zimin, A.V., Bourque, G., Chaisson, M.J.P., Flicek, P., Phillippy, A.M., Zook, J.M., Eichler, E.E., Haussler, D., Wang, T., Jarvis, E.D., Miga, K.H., Garrison, E., Marschall, T., Hall, I.M., Li, H., Paten, B.: A draft human pangenome reference. Nature **617**(7960), 312–324 (2023). https://doi.org/10.1038/s41586-023-05896-x

32. Lin, Y., Nurk, S., Pevzner, P.A.: What is the difference between the breakpoint graph and the de Bruijn graph? BMC Genomics **15**(6), S6 (2014). https://doi.org/10.1186/1471-2164-15-S6-S6
33. Minkin, I., Medvedev, P.: Scalable multiple whole-genome alignment and locally collinear block construction with SibeliaZ. Nature Communications **11**(1), 6327 (2020). https://doi.org/10.1038/s41467-020-19777-8
34. Muggli, M.D., Alipanahi, B., Boucher, C.: Building large updatable colored de Bruijn graphs via merging. Bioinformatics **35**(14), i51–i60 (2019). https://doi.org/10.1093/bioinformatics/btz350
35. Noll, N., Molari, M., Shaw, L.P., Neher, R.A.: Pangraph: scalable bacterial pan-genome graph construction. Microbial Genomics **9**(6), 001034 (2023). https://doi.org/10.1099/mgen.0.001034
36. O'Leary, N.A., Cox, E., Holmes, J.B., Anderson, W.R., Falk, R., Hem, V., Tsuchiya, M.T.N., Schuler, G.D., Zhang, X., Torcivia, J., Ketter, A., Breen, L., Cothran, J., Bajwa, H., Tinne, J., Meric, P.A., Hlavina, W., Schneider, V.A.: Exploring and retrieving sequence and metadata for species across the tree of life with NCBI datasets. Scientific Data **11**(1), 732 (2024). https://doi.org/10.1038/s41597-024-03571-y
37. Parmigiani, L., Garrison, E., Stoye, J., Marschall, T., Doerr, D.: Panacus: fast and exact pangenome growth and core size estimation. Bioinformatics **40**(12), btae720 (2024). https://doi.org/10.1093/bioinformatics/btae720
38. Paten, B., Eizenga, J.M., Rosen, Y.M., Novak, A.M., Garrison, E., Hickey, G.: Superbubbles, ultrabubbles, and cacti. Journal of Computational Biology **25**(7), 649–663 (2018). https://doi.org/10.1089/cmb.2017.0251
39. Perez, G., Barber, G.P., Benet-Pages, A., Casper, J., Clawson, H., Diekhans, M., Fischer, C., Gonzalez, J.N., Hinrichs, A.S., Lee, C.M., et al.: The UCSC genome browser database: 2025 update. Nucleic Acids Research **53**(D1), D1243–D1249 (2025). https://doi.org/10.1093/nar/gkae974
40. Porubsky, D., Höps, W., Ashraf, H., Hsieh, P., Rodriguez-Martin, B., Yilmaz, F., Ebler, J., Hallast, P., Maria Maggiolini, F.A., Harvey, W.T., Henning, B., Audano, P.A., Gordon, D.S., Ebert, P., Hasenfeld, P., Benito, E., Zhu, Q., Lee, C., Antonacci, F., Steinrücken, M., Beck, C.R., Sanders, A.D., Marschall, T., Eichler, E.E., Korbel, J.O.: Recurrent inversion polymorphisms in humans associate with genetic instability and genomic disorders. Cell **185**(11), 1986–2005.e26 (2022). https://doi.org/10.1016/j.cell.2022.04.017
41. Rossi, M., Oliva, M., Langmead, B., Gagie, T., Boucher, C.: Moni: A pangenomic index for finding maximal exact matches. Journal of Computational Biology **29**(2), 169–187 (2022). https://doi.org/10.1089/cmb.2021.0290
42. Sankoff, D.: Edit distance for genome comparison based on non-local operations. In: Proceedings of the Third Annual Symposium on Combinatorial

Pattern Matching (CPM). pp. 121–135. Springer Berlin Heidelberg (1992). https://doi.org/10.1007/3-540-56024-6_10
43. Schuy, J., Grochowski, C.M., Carvalho, C.M., Lindstrand, A.: Complex genomic rearrangements: an underestimated cause of rare diseases. Trends in Genetics **38**(11), 113–1146 (2022). https://doi.org/10.1016/j.tig.2022.06.003
44. Sturtevant, A.H.: A case of rearrangement of genes in *Drosophila*. Proceedings of the National Academy of Sciences of the U.S.A. **7**(8), 235–237 (1921). https://doi.org/10.1073/pnas.7.8.235
45. Tannier, E., Zheng, C., Sankoff, D.: Multichromosomal median and halving problems under different genomic distances. BMC Bioinformatics **10**(1), 120 (2009). https://doi.org/10.1186/1471-2105-10-120
46. Tettelin, H., Masignani, V., Cieslewicz, M.J., Donati, C., Medini, D., Ward, N.L., Angiuoli, S.V., Crabtree, J., Jones, A.L., Durkin, A.S., DeBoy, R.T., Davidsen, T.M., Mora, M., Scarselli, M., y Ros, I.M., Peterson, J.D., Hauser, C.R., Sundaram, J.P., Nelson, W.C., Madupu, R., Brinkac, L.M., Dodson, R.J., Rosovitz, M.J., Sullivan, S.A., Daugherty, S.C., Haft, D.H., Selengut, J., Gwinn, M.L., Zhou, L., Zafar, N., Khouri, H., Radune, D., Dimitrov, G., Watkins, K., O'Connor, K.J.B., Smith, S., Utterback, T.R., White, O., Rubens, C.E., Grandi, G., Madoff, L.C., Kasper, D.L., Telford, J.L., Wessels, M.R., Rappuoli, R., Fraser, C.M.: Genome analysis of multiple pathogenic isolates of *Streptococcus agalactiae*: Implications for the microbial "pan-genome". Proceedings of the National Academy of Sciences of the U.S.A. **102**(39), 13950–13955 (2005). https://doi.org/10.1073/pnas.0506758102
47. The Computational Pan-Genomics Consortium: Computational pan-genomics: status, promises and challenges. Briefings in Bioinformatics **19**(1), 118–135 (2018). https://doi.org/10.1093/bib/bbw089
48. Wick, R.R., Schultz, M.B., Zobel, J., Holt, K.E.: Bandage: interactive visualization of de novo genome assemblies. Bioinformatics **31**(20), 3350–3352 (2015). https://doi.org/10.1093/bioinformatics/btv383
49. Wilson, A.C., Sarich, V.M., Maxson, L.R.: The importance of gene rearrangement in evolution: Evidence from studies on rates of chromosomal, protein, and anatomical evolution. Proceedings of the National Academy of Sciences of the U.S.A. **71**(8), 3028–3030 (1974). https://doi.org/10.1073/pnas.71.8.3028
50. Xu, H., Tong, X., Jiang, H., Wang, L., Zhu, B., Zhu, D.: On sorting by flanked transpositions. In: Guo, X., Mangul, S., Patterson, M., Zelikovsky, A. (eds.) Bioinformatics Research and Applications. pp. 292–311. Springer Nature Singapore, Singapore (2023). https://doi.org/10.1007/978-981-99-7074-2_23
51. Yancopoulos, S., Attie, O., Friedberg, R.: Efficient sorting of genomic permutations by translocation, inversion and block interchange. Bioinformatics **21**(16), 3340–3346 (2005). https://doi.org/10.1093/bioinformatics/bti535
52. Zakeri, M., Brown, N.K., Ahmed, O.Y., Gagie, T., Langmead, B.: Movi: A fast and cache-efficient full-text pangenome index. iScience **27**(12) (2024). https://doi.org/10.1016/j.isci.2024.111464

Appendix

A Multiplicities in the MBG

To see which pangenome structures a given MBG permits, we need to further characterize how many times a certain edge occurs in a pangenome. To that end it is helpful to classify the number of occurrences of an edge in a path.

Note in some cases $m^t m^h$ might be both a marker and an adjacency edge, leading to ambiguity which edge $m^t m^h$ refers to. In order to keep the notation simple, we assume such edges do not exist in the MBG. In cases where they do exist, one can easily replace the marker $m^t m^h$ by two markers $m_1^t m_1^h$ and $m_2^t m_2^h$, which are always neighboring with adjacency $m_1^h m_2^t$. (For Appendix C, note that $m_1^h m_2^t$ is not contested and is thus not affected by our algorithm cutting contested edges.)

If for a path or cycle $P = v_1 \dots v_n$ and an edge uv there is a j, $1 \leq j < n$, with $\{v_j, v_{j+1}\} = \{u, v\}$ we say that uv *occurs* in P (at position j). If P is a cycle and $\{v_n, v_1\} = \{u, v\}$, we say that uv occurs at position n. We additionally define $\mathrm{occ}(e, P)$ as the total number of times an edge occurs in a path or cycle P.

For example, in path $1^h 2^t 2^h 2^h 2^t 1^h 1^t 3^t$, the edge $1^h 2^t$ occurs twice (at position 1 and at position 5), while the edge $2^h 2^h$ occurs once (at position 3).

For a pangenome $\mathbb{P}$ and its MBG G, we write $\mathfrak{m}_{\mathbb{P}}(uv)$ for the total number of occurrences of the edge uv in $\mathbb{P}$. We call $\mathfrak{m}_{\mathbb{P}}$ the *(edge) multiplicities.*

Given MBG G and multiplicities $\mathfrak{m}$, we write $\deg_{\mathrm{adj}}(\mathfrak{m}, v) := \sum_{vu \in E_{\mathrm{adj}}, u \neq v} \mathfrak{m}(vu) + 2\mathfrak{m}(vv)$ for the *adjacency degree* of a vertex $v \neq \varnothing$, $\deg_{\mathrm{mark}}(\mathfrak{m}, v) := \sum_{vu \in E_{\mathrm{mark}}} \mathfrak{m}(vu)$ for the *marker degree* of a vertex $v \neq \varnothing$, and $\deg_{\mathrm{mark}}(\mathfrak{m}, \varnothing) := 2\mathfrak{m}(\varnothing\varnothing)$, $\deg_{\mathrm{adj}}(\mathfrak{m}, \varnothing) := \sum_{\varnothing u \in E_{\mathrm{adj}}} \mathfrak{m}(\varnothing u)$ for telomere node $\varnothing$. When $\mathfrak{m}$ is unambiguous, we omit it and simply write $\deg_{\mathrm{adj}}(v), \deg_{\mathrm{mark}}(v)$. If edge multiplicities are derived from a pangenome, adjacency degree and marker degree coincide for any vertex v. We formalize this notion with the following definition.

Definition 7. *Given an MBG $G = (V, E_{adj} \cup E_{mark})$ and a multiplicity function $\mathfrak{m}$ we call $(G, \mathfrak{m})$ an* alternating Eulerian graph *if $\deg_{adj}(\mathfrak{m}, v) = \deg_{mark}(\mathfrak{m}, v)$ for all $v \in V$ and $\mathfrak{m}(e) > 0$ for all $e \in E_{adj} \cup E_{mark}$.*

We show in the next subsection that an alternating cycle cover corresponding to the multiplicities exists if and only if G is alternating Eulerian.

A.1 Alternating Eulerian graphs and pangenomes

Lemma 2. *Given an MBG G and multiplicities $\mathfrak{m}$, there is an alternating cycle cover $\mathbb{C}$ with $\mathfrak{m}(e) = \sum_{C \in \mathbb{C}} occ(e, C)$ if and only if $(G, \mathfrak{m})$ is an alternating Eulerian graph.*

In order to prove this lemma, an additional term will be useful. Since we are interested in which cycle covers the graph permits, we distinguish paths that are compatible with the multiplicities from paths that are not compatible.

Definition 8. *An alternating path P or cycle is called* valid *for multiplicities* $\mathfrak{m}$ *if for each edge e that occurs in P holds* $\mathfrak{m}(e) \geq occ(e, P)$.

We will now show the equivalence between alternating Eulerian graphs and graphs that have an alternating cycle cover. In that regard, the following lemma is helpful.

Lemma 3. *Given a valid path P in an alternating Eulerian graph, there is a valid path Q, such that (PQ) is a valid alternating cycle.*

Proof. We give an algorithm similar to the Hierholzer algorithm [24] to find Q in Algorithm 1.

Algorithm 1 Algorithm to complete a valid path to a valid cycle.

```
Initialize an empty path $Q$.
Let $\mathfrak{m}'[e] = \mathfrak{m}(e) - \mathrm{occ}(e, P) \quad \forall e \in E_{\mathrm{adj}} \cup E_{\mathrm{mark}}$.
while $PQ$ is not an alternating cycle do
    Let $uv$ be the last edge in $PQ$.
    if $uv$ is an adjacency edge then
        Find marker edge $vx$ with $\mathfrak{m}'[vx] > 0$.
    else
        Find adjacency edge $vx$ with $\mathfrak{m}'[vx] > 0$.
    end if
    Append $x$ to $Q$.
    $\mathfrak{m}'[vx] \leftarrow \mathfrak{m}'[vx] - 1$.
    $e \leftarrow vx$.
end while
```

Multiplicities $\mathfrak{m}'$ ensure that the path is valid while the case distinction ensures that Q is alternating. Thus the only potential problem is whether edge vx exists. Let uv be an adjacency edge. Say PQ contains a total number of l adjacency edges at v. Then, either the first edge in P is a marker edge starting at v or PQ contains $l - 1$ occurrences of marker edges adjacent to v. In the former case, PQ is already an alternating cycle, so the loop would not have been started. In the latter case, observe that due to the graph being alternating Eulerian, $\deg_{\mathrm{mark}}(v) = \deg_{\mathrm{adj}}(v) \geq l$ and therefore a marker edge vx with $\mathfrak{m}'[vx] > 0$ must exist. The case for a marker edge uv is analogous. □

We can now prove Lemma 2:

Proof. Assume there is an alternating cycle cover. In each cycle, if a vertex v is preceded by an adjacency edge, it is succeeded by a marker edge and vice versa. Thus, $\deg_{\mathrm{adj}}(v) = \deg_{\mathrm{mark}}(v)$.

Using Lemma 3, creating an alternating cycle cover is easy. Take an edge uv with $\mathfrak{m}(uv) > 0$. Then there must be a path Q, such that $C := (uvQ)$ is a valid alternating cycle. Then adjusting the multiplicities $\mathfrak{m}'(e) := \mathfrak{m}(e) - \mathrm{occ}(e, C)$ and removing edges e with $\mathfrak{m}'(e) = 0$, yields another alternating Eulerian graph. Applying this procedure iteratively yields the cycle cover. □

A.2 Diminishing paths

Due to Lemma 2, if we want to find a pangenome that has a particular MBG G, we only need to find a multiplicity function $\mathfrak{m}$, such that $(G, \mathfrak{m})$ is alternating Eulerian. Particularly for applying GAOs most effectively, we want to have a pangenome where certain edges are as rare as possible. Since we start with an alternating Eulerian graph, we only need to find a modified multiplicity function which is smaller for the desired edges.

As an example of how we reduce multiplicities, regard the following path: $1^t1^h2^t2^h$ where $\mathfrak{m}(1^t1^h) = 3$, $\mathfrak{m}(1^h2^t) = 2$ and $\mathfrak{m}(2^t2^h) = 4$. Rewriting the multiplicities to $\mathfrak{m}'(1^t1^h) \coloneqq \mathfrak{m}(1^t1^h) - 1 = 2$, $\mathfrak{m}'(1^h2^t) \coloneqq \mathfrak{m}(1^h2^t) - 1 = 1$ and $\mathfrak{m}'(2^t2^h) \coloneqq \mathfrak{m}(2^t2^h) - 1 = 3$ ensures that the condition of Definition 7 is still met for $\mathfrak{m}'$ at vertices 1^h and 2^t, but not at the ends of the path, 1^t and 2^h. If we can find a cycle with this property, we can thus reduce the multiplicities while the graph stays alternating Eulerian. Generally, such paths can be defined as follows.

Definition 9. *An alternating path or cycle P in MBG G with multiplicities $\mathfrak{m}$ is called* diminishing *if for each edge $e \in E_{adj} \cup E_{mark}$ holds that $occ(e, P) < \mathfrak{m}(e)$. We call $\mathfrak{m}^{\downarrow}_{P}(e') \coloneqq \mathfrak{m}(e') - occ(e', P)$ the* diminution *of P.*

The following lemma generalizes the notion of how to reduce edge multiplicities while keeping the graph alternating Eulerian.

Lemma 4. *Given a diminishing path $P = v_1 \dots v_n$ in an alternating Eulerian graph $(G, \mathfrak{m})$ in $\mathfrak{m}^{\downarrow}_{P}$ holds $\deg_{adj}(\mathfrak{m}^{\downarrow}_{P}, v) = \deg_{mark}(\mathfrak{m}^{\downarrow}_{P}, v)$ for $v \in V$ if $v \notin \{v_1, v_n\}$.*

Proof. Let $v \notin \{v_1, v_n\}$ and $L_v \coloneqq \{2 \leq i \leq n-1 : v_i = v\}$. Then for $i \in L_v$, exactly one of $v_{i-1}v_i$ and v_iv_{i+1} is an adjacency edge and exactly one is a marker edge. Therefore, $\deg_{\mathrm{mark}}(G^{\downarrow}_{P}, v) = \deg_{\mathrm{mark}}(G, v) - |L_v| = \deg_{\mathrm{adj}}(G, v) - |L_v| = \deg_{\mathrm{adj}}(G^{\downarrow}_{P}, v)$. □

We can immediately derive the following corollary.

Corollary 1. *Given a diminishing cycle $C = (v_1 \dots v_n)$ in an alternating Eulerian graph $(G, \mathfrak{m})$, $(G, \mathfrak{m}^{\downarrow}_{C})$ is alternating Eulerian.*

Our goal is thus to reduce multiplicities using diminishing cycles. In this context, it is useful to recognize when a path already contains a diminishing cycle. The following concept helps to detect this situation.

Definition 10. *An alternating path or cycle P is* reducible *if it is of the form $P = s_0m^tm^hs_1m^tm^hs_2$ or $P = s_0m^hm^ts_1m^hm^ts_2$ for some paths s_0, s_1, s_2 and some marker m. Otherwise, P is* irreducible.

For a diminishing reducible path, such as $P = s_0m^tm^hs_1m^tm^hs_2$, we can then instead consider the diminishing cycle $(m^tm^hs_1)$. Note that this also holds for $m^t = m^h = \varnothing$.

B Proof of Theorem 1

We now restate the theorem and prove that a particular set of operations indeed forms an complete set of PAOs.

Theorem 1. The set of operations $\tilde{N}$, which consists of (de-) multiply, homologous recombination, coalescence, divergence and chromosome transfer operations is a complete set of PAOs.

Note that for each operation in $\tilde{N}$, we also have its inverse. It thus suffices to show that any pangenome $\mathbb{P}_a$ can be transformed into $\mathbb{P}^*_a$ where $\mathbb{P}^*_a = \mathbb{P}^*_b$ if and only if $\mathcal{G}(\mathbb{P}_a) = \mathcal{G}(\mathbb{P}_b)$.

We begin by deriving a few helpful lemmas.

Lemma 5. *There is a finite number of irreducible cycles in a multiple breakpoint graph.*

Proof. Each irreducible cycle can contain any marker at most twice, as m and $\overline{m}$. Then for any irreducible cycle we can assign a unique string $s \in (\Sigma \cup \overline{\Sigma} \cup \{\varnothing\})^+$ with $|s| \leq 2|\Sigma| + 2$. Since the number of such strings is finite (finite length and finite alphabet), the number of irreducible cycles is finite as well. □

Notably the only irreducible cycles in a simple MBG are the chromosomes (without duplication) in its input genomes. We further show that a cycle can always be decomposed into a set of irreducible cycles.

Lemma 6. *Any cycle C can be transformed into a set $I(C)$ of irreducible cycles by homologous recombinations.*

Proof. If C is irreducible, the set is $I(C) = \{C\}$. If $C = (S_1 m^t m^h S_2 m^t m^h)$ for some marker m and paths S_1 and S_2, we perform the homologous recombination $(S_1 m^t m^h S_2 m^t m^h) \to C_1 := (S_1 m^t m^h), C_2 := (S_2 m^t m^h)$, recursively apply the procedure to C_1 and C_2, and finally set $I(C) = I(C_1) \cup I(C_2)$. □

Let I^* be the set of all irreducible cycles in the MBG of $\mathbb{P}$. We then define $\mathbb{P}^* := \{I^*\}$ as the pangenome consisting of a single genome (I^*). Note that $\mathbb{P}^*_a = \mathbb{P}^*_b$ if and only if $\mathcal{G}(\mathbb{P}_a) = \mathcal{G}(\mathbb{P}_b)$.

Lemma 7. *Any pangenome $\mathbb{P}$ can be transformed into $\mathbb{P}^+ = \{I\}$, a pangenome consisting of a single genome I with only irreducible cycles by operations in $\tilde{N}$.*

Proof. We first show how to unify all genomes of the pangenome into a single genome. Consider two genomes G_1, G_2. Let $A_1 := G_2 \ominus G_1$ and $A_2 := G_1 \ominus G_2$. We can then create $G_1^+ := G_1 \oplus A_1$ by duplicating the chromosomes from A_1 in G_2 and transferring them to G_1. Analogously, we can create $G_2^+ := G_2 \oplus A_2$. Then, $G_1^+ = G_2^+$. We then unify G_1^+ and G_2^+ via coalescence. One can iterate this procedure until there is a single genome G^+, which can then be transformed into I containing only irreducible cycles by Lemma 6. □

Lemma 8. *Given a genome consisting of irreducible cycles I and an irreducible cycle C in $\mathcal{G}(\{I\})$ with $C \notin I$, there is a series of duplications and homologous recombinations transforming I into a set of irreducible cycles I' with $C \in I'$ and $I \subset I'$.*

Proof. Let $C = (m_1 m_2 \dots m_n)$ where each m_i is some marker in forward or backward orientation. Since all adjacencies of C are from the MBG of $\{I\}$, there are cycles $C_1, \dots, C_n \in I$ with $C_1 = (m_1 m_2 S_1), C_2 = (m_2 m_3 S_2), \dots, C_n = (m_n m_1 S_n)$ for some paths $S_1 \dots S_n$. We duplicate all cycles $C_1, \dots, C_n$. Then, starting with C_1, C_2, we perform homologous recombinations of the following type:

$$(m_1 m_2 | S_1), (m_2 | m_3 S_2) \to (m_1 m_2 m_3 S_2 m_2 S_1)$$

until we merge all cycles into a single cycle with the following structure:

$$(m_1 \dots m_n m_1 S_n m_n \dots S_2 m_2 S_1).$$

We then perform the homologous recombination

$$(m_1 | \dots m_n m_1 | S_n m_n \dots S_2 m_2 S_1) \to (m_1 \dots m_n), (m_1 S_n m_n \dots S_2 m_2 S_1).$$

The remaining cycle $(m_1 S_n m_n \dots S_2 m_2 S_1)$ can then be transformed into irreducible cycles (see Lemma 6). Now I' contains only irreducible cycles, $C = (m_1 \dots m_n) \in I'$, and since all cycles we operated on had a duplicate, we also have $I \subset I'$. □

Using Lemmas 7 and 8, we can derive the following corollary.

Corollary 2. *Any pangenome $\mathbb{P}$ can be transformed into $\mathbb{P}^*$ by operations in $\tilde{N}$.*

Since each operation has an inverse in $\tilde{N}$, this proves Theorem 1. Note that this is by no means the shortest or most realistic scenario. Optimizing for the minimum number of PAOs to transform one pangenome into another one may well be subject to future research.

C SCJ-CARP: Manipulating multiplicities

In principle, the bound in Lemma 1 can be reached by applying a cut $uv \to u\varnothing, v\varnothing$ to each contested edge uv. However note that if uv occurs more than once in the pangenome, we would need multiple cuts to remove it from the pangenome. Therefore we use the complete set of PAOs to ensure there is a contested edge occurring exactly once, arriving at the central theorem, which we restate here.

Theorem 2. The CARP measure under the SCJ model for a pangenome $\mathbb{P}$ with MBG $\mathcal{G}(\mathbb{P}) = (V, E_{\text{adj}} \cup E_{\text{mark}})$ is $|E_{\text{cnt}}|$ where $E_{\text{cnt}} \coloneqq \{uv \in E_{\text{adj}} \mid u = v\} \cup \{uv \in E_{\text{adj}} \mid u, v \neq \varnothing, \exists uv' \in E_{\text{adj}} \text{ with } v' \neq v\}$. One possible SCJ-CARP simplification $\mathbb{A}$ is defined by adjacencies $E_{\mathbb{A}} \coloneqq E_{\text{adj}} \setminus E_{\text{cnt}}$.

Using the concepts introduced in Appendix A.2, we now show that there is a pangenome that contains some $e \in E_{\text{cnt}}$ with $\mathfrak{m}(e) = 1$. The first step is to show that there are diminishing cycles or edges with multiplicity 1. To prove this, we derive a number of useful facts, which we summarize in the following lemma.

Lemma 9. *Let $P = v_1 \dots v_n$ be a diminishing path in an alternating Eulerian graph $(G, \mathfrak{m})$, such that*

- *$(G, \mathfrak{m})$ does not contain any diminishing cycles*
- *P cannot be extended to the right, i.e., there is no path $P' = Pu$ that is diminishing.*

Let U be the set of vertices with $u \in U \iff Pu$ is an alternating path, and let $U_{\geq 2} \subseteq U$ be the set of vertices $u \in U$ for which $\mathfrak{m}(v_n u) \geq 2$. Then all of the following hold:

(i) $\forall u \in U_{\geq 2}$, $v_n u$ occurs in P at some position i and only in reverse order, i.e. $v_i = u$, $v_{i+1} = v_n$.
(ii) $v_n \notin U_{\geq 2}$,
(iii) $|U_{\geq 2}| \leq 1$,
(iv) $\forall u \in U_{\geq 2}$, $v_n u$ occurs exactly once in P,
(v) $\forall u \in U_{\geq 2}$ holds $\mathfrak{m}(v_n u) = 2$,
(vi) $\exists u \in U$ with $\mathfrak{m}(v_n u) = 1$.

Proof. In the following note that the graph not containing diminishing cycles implies that P is irreducible.

(i) If there is any $u \in U_{\geq 2}$, such that $v_n u$ does not occur in P, then $v_1 \dots v_n u$ is a diminishing path, a contradiction. Therefore for all $u \in U_{\geq 2}$, $v_n u$ occurs in P. If there is any $u \in U_{\geq 2}$, such that there is $1 \leq j < n$ with $v_j = v_n$ and $v_{j+1} = u$, then $(v_{j+1} \dots v_n)$ is a diminishing cycle, a contradiction. Thus for each $u \in U$ $v_n u$ occurs in P and only in reverse order. $\square_{\text{(i)}}$

(ii) Assume $v_n \in U_{\geq 2}$. Then, using (i), there is some i with $v_i v_{i+1} = v_n v_n$ and thus $(v_{i+1} \dots v_n)$ is a diminishing cycle, a contradiction. $\square_{\text{(ii)}}$

(iii) Let there be $u, u' \in U_{\geq 2}$ with $u \neq u'$. Then, $v_n u$ and $v_n u'$ must be adjacency edges, because there is exactly one marker edge incident to each vertex. Using (i), both occur in reverse in P. W.l.o.g. let $u v_n$ occur before $u' v_n$ in P. Then we have $P = \dots u v_n \dots u' v_n \dots v_n$. Since $v_n u'$ is an adjacency edge, $v_{n-1} v_n$ must be a marker edge. Since each vertex is adjacent to exactly one marker edge, we have $P = \dots u v_n v_{n-1} \dots u' v_n v_{n-1} \dots v_n$, meaning that P is reducible, a contradiction. $\square_{\text{(iii)}}$

(iv) For $u \in U_{\geq 2}$, let $u v_n$ occur twice in P. As before, we then have $P = \dots u v_n x \dots u v_n y \dots v_n$. Either $u v_n$ is a marker edge or $v_n x$ is a marker edge, in which case $v_n x = v_n y$. In both cases P is reducible, a contradiction. $\square_{\text{(iv)}}$

(v) Because $u \in U_{\geq 2}$ occurs only once in P, if $\mathfrak{m}(v_n u) > 2$, Pu would be a diminishing path. $\square_{\text{(v)}}$

(vi) Using (i) and (ii) we know that for $u \in U_{\geq 2}$, either (a) $P = \dots u v_n v_{j+2} \dots v_{n-1} v_n$ for some j, or (b) $P = \dots u v_n v_n$ and with (iii) that $U_{\geq 2} = \{u\}$.

Regard the adjacency and marker degree of v_n. In (a), $v_n v_{j+2}$ and $v_{n-1}v_n$ have the same type. We then distinguish the cases:

(I) $v_{j+2} \neq v_{n-1}$. Then $v_{n-1}v_n$ and $v_n v_{j+2}$ are adjacency edges. Since P is diminishing $\mathfrak{m}(v_{n-1}v_n) > 1$ and $\mathfrak{m}(v_n v_{j+2}) > 1$. Thus $\deg_{\text{adj}}(v_n) \geq \mathfrak{m}(v_{n-1}v_n) + \mathfrak{m}(v_n v_{j+2}) \geq 4$

(II) $v_{j+2} = v_{n-1}$. Then because P is diminishing $\mathfrak{m}(v_{n-1}v_n) > 2$. Thus $\deg_{\text{adj}}(v_n) = \deg_{\text{mark}}(v_n) \geq \mathfrak{m}(v_{n-1}v_n) \geq 3$.

For (b), we have $\deg_{\text{adj}}(v_n) = \deg_{\text{mark}}(v_n) \geq 2\mathfrak{m}(v_n v_n) \geq 4$.

In all cases (a(I), a(II), b), we have $\deg_{\text{adj}}(v_n) = \deg_{\text{mark}}(v_n) > 2$, and since $U_{\geq 2} = \{u\}$ with $\mathfrak{m}(v_n u) = 2$ (see (v)), there must be at least one other edge $v_n u' \notin U_{\geq 2}$ that has therefore $\mathfrak{m}(v_n u') = 1$. If $|U_{\geq 2}| = 0$, all edges $v_n u$ with $u \in U$ have $\mathfrak{m}(v_n u) = 1$. $\square_{\text{(vi)}}$

The rough idea for the algorithm cutting contested edges (see also Algorithm 2) is thus as follows. As long as there are contested edges of multiplicity 1, we cut them. If we cannot find such an edge any more, we find diminishing cycles, which we then use to reduce the multiplicities in the graph until there is an edge of multiplicity 1, so we can repeat the process. However, it may be the case that none of the edges with multiplicity 1 is contested. If that is the case, a slightly more complex procedure needs to be employed.

We begin by showing that a contested edge uv with some helpful properties always exists.

Lemma 10. *For any alternating Eulerian MBG $(G, \mathfrak{m})$ where $\mathfrak{m}(e) > 1$ $\forall e \in E_{cnt}$, $|E_{cnt}| \neq 0$ and $(G, \mathfrak{m})$ has no diminishing cycles, there is an edge $v\varnothing \in E_{adj}$ with $\mathfrak{m}(v\varnothing) = 1$, such that $uv \in E_{adj}, u \neq v$ with $\mathfrak{m}(uv) = 2$ and $u, \varnothing$ the only vertices adjacent to v.*

Proof. Regard an (irreducible) diminishing path $P = v_1 \ldots v_n$ that cannot be extended in either direction, i.e., there is no path $av_1 \ldots v_n$ or $v_1 \ldots v_n b$ that is diminishing. Then according to Lemma 9 (vi), there are edges $xv_1, v_n y$ with multiplicity 1. (Note that this may also be the same edge.)

Regard $v_n y$. Since $\mathfrak{m}(v_{n-1}v_n) > 1$ and $\mathfrak{m}(v_n y) = 1$, there must be another edge $v_n z$ or $y = v_n$. Analogously, there is wv_1 or $x = v_1$.

Therefore $xv_1, v_n y$ are either adjacency edges or the telomere marker edge $\varnothing\varnothing$. Assume now that $xv_1 = \varnothing\varnothing$. This implies $\deg_{\text{mark}}(\varnothing) = 2$ and, since $\mathfrak{m}(v_1 v_2) \geq 2$, this means that $v_1 v_2 = \varnothing v_2$ is the only adjacency edge incident to $\varnothing$ with $\mathfrak{m}(v_1 v_2) = 2$. However, then $\{v_n, y\} \neq \{v_2, \varnothing\}$, meaning that $v_n y$ is contested, a contradiction. We may thus assume that both xv_1 and $v_n y$ are adjacency edges and therefore $v_1 v_2$ and $v_{n-1}v_n$ are marker edges.

If both $v_1 = v_n = \varnothing$, since $v_1 v_2$ and $v_{n-1}v_n$ are marker edges, $P = \varnothing\varnothing v_3 \ldots \varnothing\varnothing$ is reducible, a contradiction.

Let thus w.l.o.g. $v_n \neq \varnothing$. Then $y = \varnothing$, otherwise $v_n y$ would be a contested edge with $\mathfrak{m}(v_n y) = 1$. This means that $v_n z$ is a contested edge with $\mathfrak{m}(v_n z) > 1$. According to Lemma 9 (iii) and (v) $v_n z$ has $\mathfrak{m}(v_n z) = 2$ and there are no other edges with multiplicity 2 or higher adjacent to v_n. Since any other edge $v_n w$

with $w \neq \varnothing$ and $w \neq z$ would be contested and have $\mathfrak{m}(v_n w) = 1$, such an edge does not exist. Additionally, due to Lemma 9 (ii), $u \neq v$. $\square$

There is then a diminishing path that contains uv and that contains v only once.

Lemma 11. *For any alternating Eulerian MBG* $(G, \mathfrak{m})$ *where* $\mathfrak{m}(e) > 1\ \forall e \in E_{cnt}$*,* $|E_{cnt}| \neq 0$ *and* $(G, \mathfrak{m})$ *has no diminishing cycles, for a pair of vertices* u, v *as described in Lemma 10, there is a diminishing path* $vuu_1 \ldots u_m$ *that cannot be extended to the right, such that* v *does not occur in* $u_1 \ldots u_m$*.*

Proof. Observe that the single edge vu is already a diminishing path. Therefore $vuu_1 \ldots u_m$ exists.

We now show by contradiction that v does not occur in $u_1 \ldots u_m$. Let v' be the other extremity of the marker of v. We distinguish two cases:

(I) v occurs before v' in $u_1 \ldots u_m$ at position k, that is we have $u_1 \ldots u_{k-1} v$, where $u_{k-1} v$ is an adjacency edge. Then consider u_{k-1}. According to Lemma 10, $u_{k-1} = \varnothing$ or $u_{k-1} = u$. In both cases, $vuu_1 \ldots u_{k-1} v$ is not diminishing, a contradiction.

(II) v' occurs before v in $u_1 \ldots u_m$ at position l. Then we have $u_1 \ldots u_{l-1} v' v$. However, then $(v' vuu_1 \ldots u_{l-1})$ is a diminishing cycle, a contradiction. $\square$

We are now ready to construct the multiplicity function $\mathfrak{m}'$, in which uv has multiplicity 1.

Lemma 12. *For any alternating Eulerian MBG* $(G, \mathfrak{m})$ *where* $\mathfrak{m}(e) > 1\ \forall e \in E_{cnt}$*,* $|E_{cnt}| \neq 0$ *and* $(G, \mathfrak{m})$ *has no diminishing cycles, there is* $\mathfrak{m}'$*, such that* $(G, \mathfrak{m}')$ *is alternating Eulerian and* $\exists uv \in E_{cnt}$ *with* $\mathfrak{m}'(uv) = 1$*.*

Proof. Let u, v be vertices as described in Lemmas 10 and 11, i.e., $v\varnothing \in E_{\text{adj}}$ and $\mathfrak{m}(vu) = 2$, and let $P = vuu_1 \ldots u_m$ be the diminishing path that cannot be extended further to the right. Then, according to Lemma 9 (vi), there is an edge $u_m x$ with $\mathfrak{m}(u_m x) = 1$. We distinguish two cases, visualized in Fig. 11.

(I) $\varnothing$ does not occur in P. Then $x = \varnothing$. Then there exists another edge $u_m z$, which is contested. It therefore has multiplicity $\mathfrak{m}(u_m z) = 2$ and occurs once in P_{vu} (see Lemma 9 (iv),(v)). Thus at least one vertex (u_m) occurs twice in P. Let w be the leftmost vertex that occurs more than once in P. Let w' be the other extremity of the marker of w. Then P is of the form $P = P_1 w w' P_2 w' w P_3$ with P_1 containing only vertices that occur once in P and P_3 not containing further occurrences of w. Note that because of this, P_1 and P_3 do not share any vertices. We then set

$$\mathfrak{m}'(xy) = \mathfrak{m}(xy) - \text{occ}(xy, P_1 w) + \text{occ}(xy, w P_3 \varnothing\varnothing v).$$

Note that there are only two vertices at which the condition for an alternating Eulerian path might not hold, namely v and w. Regard the adjacency degree of

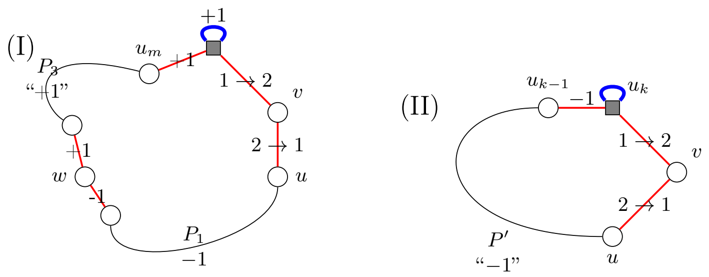

Fig. 11: Case distinction in proof of Lemma 12. Black lines represent arbitrary alternating cycles that do not share edges or vertices. Note that the edges shown explicitly do not occur in the black paths.

v.

$$\begin{aligned}\deg_{\mathrm{adj}}(v,\mathfrak{m}') &= \deg_{\mathrm{adj}}(v,\mathfrak{m}) - \mathrm{occ}(vu, P_1 w) - \mathrm{occ}(v\varnothing, P_1 w) + \mathrm{occ}(vu, wP_3\varnothing\varnothing v) + \mathrm{occ}(v\varnothing, wP_3\varnothing\varnothing v)\\ &= \deg_{\mathrm{adj}}(v,\mathfrak{m}) - 1 - 0 + 0 + \mathrm{occ}(v\varnothing, wP_3) \qquad [v \text{ occurs once as } uv \text{ in } P_1]\\ &= \deg_{\mathrm{adj}}(v,\mathfrak{m}) - 1 - 0 + 0 + 1 = \deg_{\mathrm{adj}}(v,\mathfrak{m}) \qquad [\varnothing \text{ not in } P_3]\end{aligned}$$

Let v' be the other extremity of the marker of v. Note that because v occurs only once in P at the beginning in an adjacency edge, vv' does not occur in P.

$$\begin{aligned}\deg_{\mathrm{mark}}(v,\mathfrak{m}') &= \deg_{\mathrm{mark}}(v,\mathfrak{m}) - \mathrm{occ}(vv', P_1 w) - \mathrm{occ}(vv', wP_3\varnothing\varnothing v)\\ &= \deg_{\mathrm{mark}}(v,\mathfrak{m}) \qquad [vv' \text{ does not occur in } P]\end{aligned}$$

We thus see that at v the degree is unchanged and the condition for an alternating Eulerian graph still holds.

Regard the adjacency and marker degree of w. By definition w does not appear in P_1, P_3. Therefore, its marker degree does not change and for the adjacency degree holds

$$\begin{aligned}\deg_{\mathrm{adj}}(w,\mathfrak{m}') &= \deg_{\mathrm{adj}}(w,\mathfrak{m}) - \sum_{wx\in E_{\mathrm{adj}}} \mathrm{occ}(wx, P_1 w) + \sum_{wx\in E_{\mathrm{adj}}} \mathrm{occ}(wx, wP_3\varnothing\varnothing v)\\ &= \deg_{\mathrm{adj}}(w,\mathfrak{m}) - 1 + 1 = \deg_{\mathrm{adj}}(w,\mathfrak{m}).\end{aligned}$$

Thus, $(G,\mathfrak{m}')$ is alternating Eulerian. Observe that the contested edge uv now has multiplicity $\mathfrak{m}'(uv) = \mathfrak{m}(uv) - \mathrm{occ}(uv, P_1 w) + \mathrm{occ}(uv, P_3\varnothing\varnothing v) = 2-1+0 = 1$.

(II) $\varnothing$ occurs in P. Then let $u_k = \varnothing$ be the first occurrence of $\varnothing$ in P and $P' = vuu_1 \dots u_{k-1}\varnothing$. Note that $u_{k-1} \neq v$ because $\mathfrak{m}(u_{k-1}\varnothing) > 1$ (P is diminishing). We then set

$$\mathfrak{m}'(xy) = \mathfrak{m}(xy) - \mathrm{occ}(xy, vuu_1 \dots u_{k-1}\varnothing) + \mathrm{occ}(xy, v\varnothing).$$

Again note that the adjacency and marker degrees of all nodes except v and $\varnothing$ under $\mathfrak{m}'$ are in agreement. The marker degrees of v and $\varnothing$ do not change as the markers do not occur in $vuu_1 \dots u_{k-1}\varnothing$ or $v\varnothing$.

Regard the adjacency degree of v.

$$\begin{aligned}\deg_{\mathsf{adj}}(v,\mathfrak{m}') &= \deg_{\mathsf{adj}}(v,\mathfrak{m}) - \mathrm{occ}(uv, vuu_1\dots u_{k-1}\varnothing) - \mathrm{occ}(v\varnothing, vuu_1\dots u_{k-1}\varnothing)\\ &\quad + \mathrm{occ}(uv, v\varnothing) + \mathrm{occ}(v\varnothing, v\varnothing)\\ &= \deg_{\mathsf{adj}}(v,\mathfrak{m}) - 1 - 0 + 0 + 1\\ &= \deg_{\mathsf{adj}}(v,\mathfrak{m})\end{aligned}$$

Regard the adjacency degree of $\varnothing$.

$$\begin{aligned}\deg_{\mathsf{adj}}(\varnothing,\mathfrak{m}') &= \deg_{\mathsf{adj}}(\varnothing,\mathfrak{m}) - \sum_{x\varnothing\in E_{\mathrm{adj}}} \mathrm{occ}(x\varnothing, vuu_1\dots u_{k-1}\varnothing) + \sum_{x\varnothing\in E_{\mathrm{adj}}} \mathrm{occ}(x\varnothing, v\varnothing)\\ &= \deg_{\mathsf{adj}}(\varnothing,\mathfrak{m}) - 1 + 1\\ &= \deg_{\mathsf{adj}}(\varnothing,\mathfrak{m})\end{aligned}$$

Therefore $(G,\mathfrak{m}')$ is alternating Eulerian. The multiplicity of uv is

$$\mathfrak{m}'(uv) = \mathfrak{m}(uv) - \mathrm{occ}(uv, vuu_1\dots u_{k-1}\varnothing) + \mathrm{occ}(uv, v\varnothing) = 2 - 1 + 0.$$

□

Using Lemma 2, Corollary 1 and Lemma 12 we can thus ensure that as long as there are contested adjacencies, there is an edge $e \in E_{\mathrm{cnt}}$ with $\mathfrak{m}(e) = 1$. The lower bound in Lemma 1 can thus be reached by cutting all edges in E_{cnt}, proving Theorem 2.

The following algorithm shows how the GAO scenario for SCJ-CARP can be reconstructed. Future research might examine how to reconstruct the shortest PAO scenario. The results from this research could then be used here (see Line 19) to reconstruct both the PAO and GAO sequences. To simply reconstruct a simplification and the number of SCJs in the scenario, Algorithm 2 is not necessary; it suffices to compute E_{cnt} (see Theorem 2).

Algorithm 2 SCJ-CARP scenario reconstruction

```
1: procedure SCJ-CARP-Reconstruction($G \coloneqq (V, E_{\text{adj}} \cup E_{\text{mark}})$,$\mathfrak{m}$)
2:     Initialize empty list $l$
3:     loop
4:         Determine $E_{\text{cnt}}$ for $G$.
5:         while there is $uv \in E_{\text{cnt}}$ with $\mathfrak{m}(E_{\text{cnt}}) = 1$ do
6:             Perform SCJ $uv \to u\varnothing, \varnothing v$ on $G$.
7:             Append $(uv \to u\varnothing, \varnothing v)$ to $l$.
8:         end while
9:         Determine $E_{\text{cnt}}$ for $G$.
10:        if $|E_{\text{cnt}}| = 0$ then
11:            return $l$
12:        end if
13:        while there is a diminishing cycle $C$ do
14:            $\mathfrak{m} \leftarrow \mathfrak{m}^{\downarrow}_{C}$
15:        end while
16:        if $\mathfrak{m}(e) > 1 \quad \forall e \in E_{\text{cnt}}$ then
17:            $\mathfrak{m} \leftarrow \mathfrak{m}'$ by applying Lemma 12.
18:        end if
19:        The PAO scenario could be reconstructed here.
20:    end loop
21: end procedure
```

D Pangenome comparison – datasets

Table 2: Genome datasets and segmentation settings used in Section 5.2. We list the accession numbers of the genomes used (`Accession.Version`), marking the reference used for Minigraph-Cactus with an asterisk ($\star$). We summarize the number of genomes and the NCBI assembly level (Assembly lvl.). The last three columns describe parameters for SibeliaZ: the k-mer size used (k (SbZ)), the associated duplication filter parameters for the strong (a (SbZs)) and for the weak filtration (a (SbZw)) runs.

Taxon	`Accession.Version`	#Genomes	Assembly lvl.	k (SbZ)	a (SbZs)	a (SbZw)
A. thaliana	`GCA_000001735.2`*	10	`chromosome+`	25	20	200
	`GCA_020911765.2`					
	`GCA_023115395.1`					
	`GCA_028009825.2`					
	`GCA_051624255.1`					
	`GCA_051624265.1`					
	`GCA_946405265.1`					
	`GCA_946409395.1`					
	`GCA_946409825.1`					
	`GCA_965117575.1`					
C. carpio	`GCA_018340385.1`*	5	`chromosome+`	25	10	100
	`GCA_000951615.2`					
	`GCA_027406505.1`					
	`GCA_050881965.1`					
	`GCA_051167465.1`					
S. cerevisiae	`GCA_000146045.2`*	10	`complete`	25	20	200
	`GCA_001580425.1`					
	`GCA_003086655.1`					
	`GCA_004014915.1`					
	`GCA_004328465.1`					
	`GCA_021172205.1`					
	`GCA_023508825.1`					
	`GCA_024972935.1`					
	`GCA_024972955.1`					
	`GCA_903819175.2`					
E. coli	`GCA_000005845.2`*	10	`complete`	15	20	200
	`GCA_000008865.2`					
	`GCA_000010385.1`					
	`GCA_000013265.1`					
	`GCA_000019645.1`					
	`GCA_000210475.1`					
	`GCA_002853715.1`					

	`GCA_003018455.1`					
	`GCA_003697165.2`					
	`GCA_048568945.1`					
K. pneumoniae	`GCA_000240185.2`*	10	`complete`	15	20	200
	`GCA_006364295.1`					
	`GCA_011045595.1`					
	`GCA_020911805.1`					
	`GCA_022699305.1`					
	`GCA_022699345.1`					
	`GCA_022749215.1`					
	`GCA_022869665.1`					
	`GCA_043187905.1`					
	`GCA_050156185.1`					
D. melanogaster	`GCA_000001215.4`*	10	`chromosome`	25	20	200
	`GCA_002300595.1`					
	`GCA_002310755.1`					
	`GCA_003401735.1`					
	`GCA_003401745.1`					
	`GCA_003401925.1`					
	`GCA_003402055.1`					
	`GCA_042606445.1`					
	`GCA_047116165.1`					
	`GCA_048772135.1`					
P. vivax	`GCA_000002415.2`*	9	`chromosome`	15	18	180
	`GCA_014843675.1`					
	`GCA_014843685.1`					
	`GCA_014843935.1`					
	`GCA_014843945.1`					
	`GCA_900093535.1`					
	`GCA_900093545.1`					
	`GCA_900093555.2`					
	`GCA_900178095.1`					
Y. pestis	`GCA_000222975.1`*	10	`complete`	15	20	200
	`GCA_000834235.1`					
	`GCA_000834335.1`					
	`GCA_000834495.1`					
	`GCA_000834775.1`					
	`GCA_003798225.1`					
	`GCA_046895405.1`					
	`GCA_046895445.1`					
	`GCA_046895625.1`					
	`GCA_046895675.1`					

E Supplementary Figures

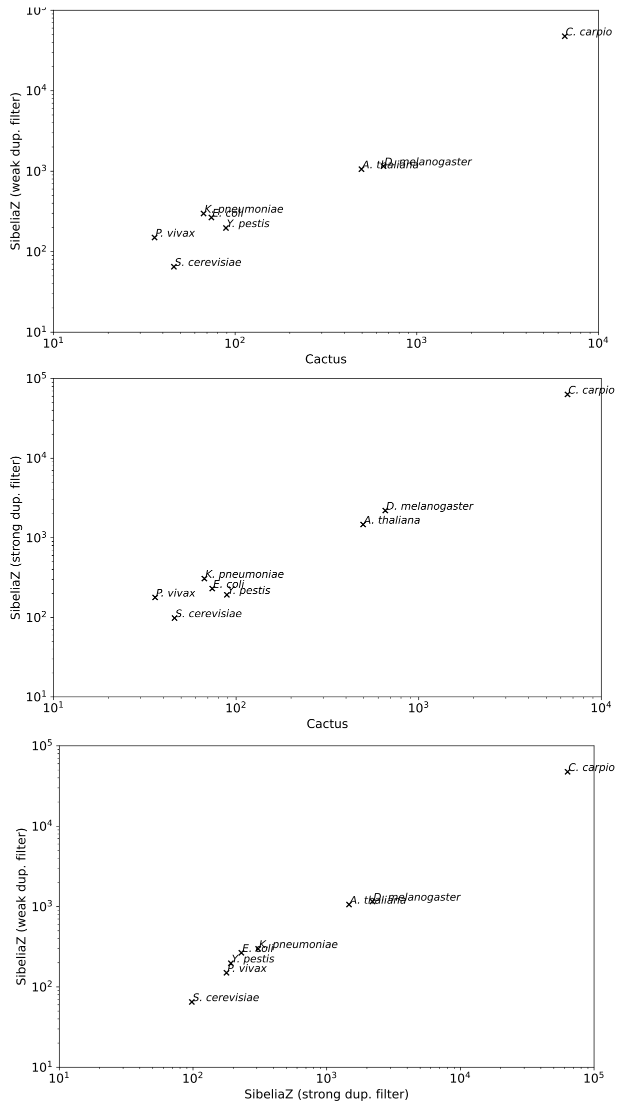

Fig. 12: Pairwise scatterplots of SCJ-CARP measures calculated on different block types as a log-log plot.

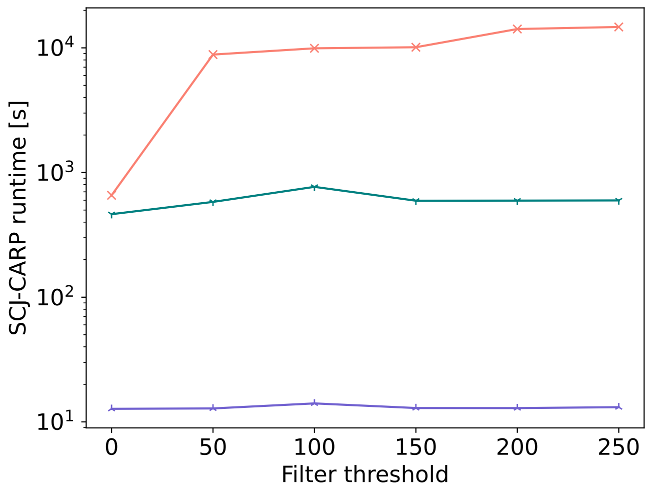

Fig. 13: CARP runtime for various levels of node filtering on human pangenome graphs built by PGGB, Minigraph-Cactus (MC) and Minigraph from [31]. Nodes of a size smaller than the filter threshold were removed and each path bridging two nodes that pass the threshold consisting only of filtered nodes was replaced with a single edge. The SCJ-CARP software was run with 16 threads on a cluster machine.

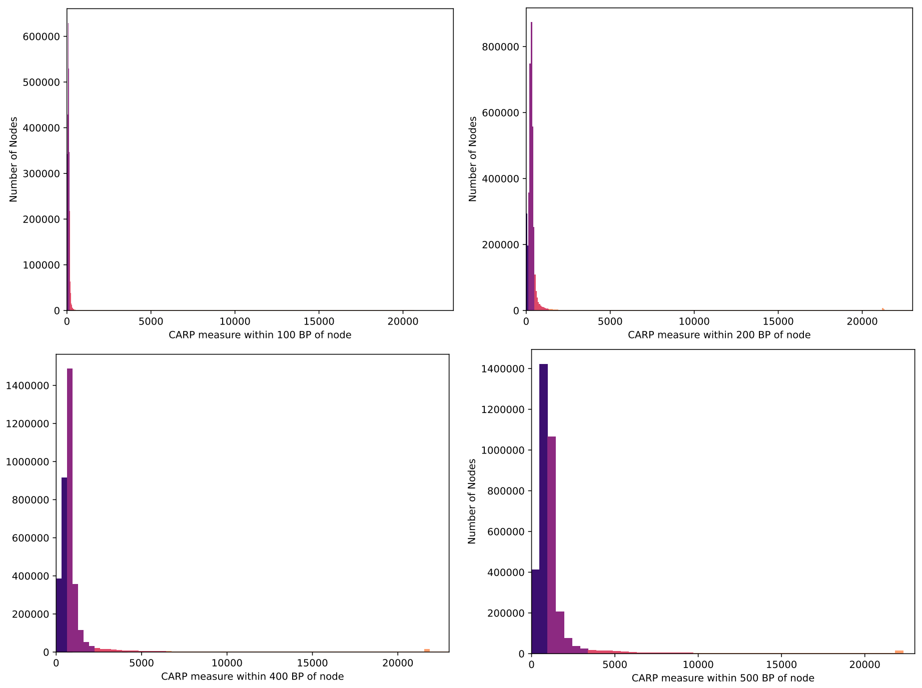

Fig. 14: Alternate histograms for the *Bacillus subtilis* dataset with node environments 100, 200, 400 and 500. Shown is the same excerpt of the histogram as in Fig. 9.

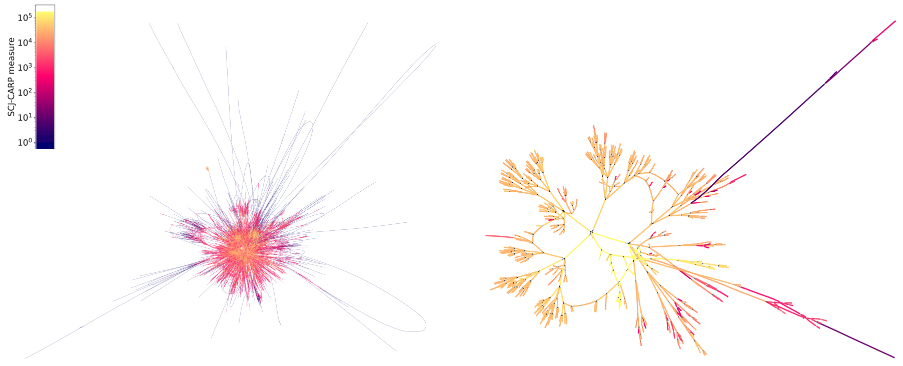

Fig. 15: Coloring of nodes in the de Bruijn graph ($k = 31$) of 1508 *Bacillus subtilis* genomes by SCJ-CARP measure of surrounding 300 BP. Shown is a 20-node (left) and 6-node environment around the most complex node visualized with Bandage [48]. Note that the color scheme is not the same as in the histograms (Figures 9 and 14).