Small Parsimony for Natural Genomes in the DCJ-Indel Model

Daniel Doerr
Faculty of Medicine, Heinrich Heine University, Düsseldorf, Germany
daniel.doerr@hhu.de

Cedric Chauve
Department of Mathematic, Simon Fraser University, Canada
cedric.chauve.sfu.ca

The Small Parsimony Problem (SPP) aims at finding the gene orders at internal nodes of a given phylogenetic tree such that the overall genome rearrangement distance along the tree branches is minimized. This problem is intractable in most genome rearrangement models, especially when gene duplication and loss are considered. In this work, we describe an Integer Linear Program algorithm to solve the SPP for natural genomes, *i.e.*, genomes that contain conserved, unique, and duplicated markers. The evolutionary model that we consider is the DCJ-indel model that includes the Double-Cut and Join rearrangement operation and the insertion and deletion of genome segments. We evaluate our algorithm on simulated data and show that it is able to reconstruct very efficiently and accurately ancestral gene orders in a very comprehensive evolutionary model.

Keywords: genome rearrangement; ancestral reconstruction; small parsimony

1. Introduction

Methods for ancestral genome reconstruction, such as those used in evolutionary genomics[34,27,26,28] and genome assembly[37,3], take as input a phylogenetic tree (a species tree) together with gene orders for extant species and aim at computing the gene orders at the internal nodes of the tree, *i.e.*, ancestral species, while optimizing a suitable criterion.

Approaches designed to reconstruct ancestral gene orders can be widely divided into two types: homology-based and parsimony-based. The former rely on the use of *conserved genomic features*, such as gene adjacencies or common intervals, associated to specific internal nodes of the species tree and obtained by the comparison of the extant gene orders[6]. Subsequently, these genomic features can then be assembled into larger linear or circular gene orders for the considered ancestral species, often called *Contiguous Ancestral Regions* (CARs)[23,13]. Conversely, parsimony-based approaches are guided by the principle of minimizing the evolutionary cost, in a given genome rearrangement model, along the branches of the considered species tree. This approach builds upon many tractability results on the pairwise genome rearrangement distance problem[18] and the corresponding computational problem is known as the *Small Parsimony Problem* (SPP). However, even when restricted to the genome median problem, the SPP has been proved to be NP-hard for most genome rearrangement models[29,11,36,20]. The only strong tractability result for the SPP has been obtained in the *Single-Cut-or-Join* (SCJ) model[17,21].

Most methods discussed above consider gene orders with no duplicated genes. However gene duplication and loss play an important role in genome evolution. In most cases, computing the pairwise distance between genomes with duplicates is hard[8,2] and there are very few exact polynomial-time algorithms for reconstructing ancestral gene orders in a framework including gene duplication and loss. The first work toward this goal was due to Sankoff and El-Mabrouk[31] (see also[12]), who introduced the idea of using reconciled gene trees to define the gene content of ancestral genomes and orthology relations between genes. This idea was later used in the homology-based method DUPCAR[22] that requires however a dated species tree to order gene duplication events, and the DeCoSTAR method that does not require dated gene trees[15,3]. Other methods accounting for gene duplication and loss include[16], GapAdj[19] that assumes that gene duplications originate from Whole-Genome Duplications (WGD), PMAG++[39,40], and the homology-based method MULTIRES[30] that requires a preliminary set of CARs as well as an upper bound on the number of duplications per gene. Recently, it was shown that the problem of computing the pairwise distance in a model including SCJ and single-gene duplications is tractable[25] and that the SPP in this moel can be solved by a simple *Integer Linear Program* (ILP); however, results on simulated data showed that considering only single-gene events leads to inaccurate reconstructed ancestral gene orders[24].

Recently, fast ILP-based method were developed for computing the pairwise distance between gene orders with duplicates, especially in genome rearrangement models based on the *Double-Cut and Join* (DCJ) operation[38,7]. ILP-based methods for the pairwise distance with duplicated genes can be traced back to the work of Shao *et al.* [32]. The most recent advance is due to Bohnenkämper *et al.* [9], who designed an extremely efficient ILP for computing the pairwise distance in the *DCJ-indel* model, that allows arbitrary gene orders and considers DCJ for genome rearrangement operations, and the insertion and deletion of segments of consecutive genes for duplications and losses. In our work, we extend the method introduced in[9] to SPP. We also formulate the problem of linearizing degenerate genomes that connects to that of linearizing ancestral circular chromosomes[5,4].

We first describe the general workflow of ancestral gene order reconstruction and background on the DCJ-indel distance (Section 2), followed by our SPP algorithm (Section 3) before providing results on simulated data (Section 4), that show that our algorithm is able to recover efficiently and accurately ancestral gene orders even in the presence of a high level of noise in the input data.

2. Background

Overview. The parsimony-based ancestral reconstruction approach to which our work contributes aims at inferring gene order sequences for internal nodes of a given rooted species phylogeny based on extant gene orders placed at its tips.

From now on, we will refer to gene orders of a species as its *genome*. In this work, we follow the ancestral genome reconstruction framework that was described in[12] and consists in five successive steps, as illustrated in Figure 1[a]: (1) Using the underlying DNA or protein sequences of the genomic markers, gene trees are predicted; (2) These are subsequently reconciled with the species tree. Note that at this point of the workflow, the marker content of the ancestral genomes subject to reconstruction is determined; (3) Reconciled gene trees are then used to infer the evolutionary history of neighborhood relations of adjacent markers, also known as *adjacencies*, using the DeCoSTAR algorithm[15]. This provides weighted candidate ancestral adjacencies, where the weight is considered as a measure of confidence. (4) From the candidate ancestral adjacencies forests, ancestral genomes can be derived. However, candidate ancestral adjacencies may conflict, resulting in some ancestral markers being involved in two or more contradictory adjacencies[35]. We call a genome *degenerate* if its chromosomes are linear, circular, or neither. That is, an ancestral degenerate genome represents a superposition of a set of ancestral genome candidates; (5) As the last step of the reconstruction workflow, ancestral genomes are derived from their degenerate counterparts, based on an optimization criterion that considers jointly parsimony in an evolutionary model and adjacency weights. In the following, we present a method that addresses step (5).

Preliminaries. A *(genomic) marker* $g := \{g^{\mathrm{t}}, g^{\mathrm{h}}\}$ is an element of the universe of markers, denoted by $\mathcal{M}$, defined as a pair of *extremities* g^{t} ("*tail* of g") and g^{h} ("*head* of g"); markers corresponds to genome segments (genes, synteny blocks, ...). In what follows we assume that in a gene order, defined as a total order on marker extremities, the two extremities of a marker are always consecutive, *i.e.*, markers do not overlap. Assuming the doubled stranded nature of DNA, the order of the extremities of a marker encode the strand in which a marker is located: if the tail occurs before the head the marker is assumed to be on the positive strand, and on the negative strand otherwise. *Telomeres* $\mathcal{T} \subset \mathcal{M}$ form a special subset of markers composed of a single extremity, *i.e.*, $t := \{t^{\circ}\}$ for all $t \in \mathcal{T}$; intuitively both the tail and the head are confounded and telomeres are not associated to a specific strand.

Let $\bigcup$ be the operator that takes the union of a collection of sets, *i.e.*, $\bigcup \mathcal{X} := \cup_{X \in \mathcal{X}} X$. We denote the universe of (marker) extremities $\bigcup \mathcal{M}$ by $\mathcal{E}$. Furthermore, we use a function $\mathbf{e} : \mathcal{E} \to \{\mathrm{t}, \mathrm{h}, \circ\}$ to map extremities to their corresponding kind (tail, head or telomere).

An *adjacency* is an unordered pair of extremities $\{\varepsilon, \varepsilon'\} \in \mathcal{E} \times \mathcal{E}$ such that $\varepsilon \neq \varepsilon'$. A *genome* A is a set of unique[b] adjacencies for which holds true that *(i)* $\forall g^{\mathrm{t}} \in \bigcup A$, there exists also extremity $g^{\mathrm{h}} \in \bigcup A$ and vice versa, and *(ii)* each extremity is used only once, *i.e.*, $\forall \{X, X'\} \subseteq A$, $X \cap X' = \emptyset$. As a consequence of *(ii)*, it is implicit

[a]All figures are available in Appendix.
[b]No two genomes contain the same marker and hence the same adjacency.

that the markers of a genome can be ordered in linear and circular segments where the tail and head of any marker are consecutive.

Comparing genomes benefits from the knowledge of evolutionary relationships between non-telomeric markers. Typically, non-telomeric markers are clustered into *families* based on *homology* or *orthology*, indicating a likely common evolutionary origin from an ancestral marker. We model family assignments of marker extremities as a function $\mathbf{f} : \mathcal{E} \setminus \mathcal{T} \to \mathbb{N}$ for which holds true that for any marker $g = \{g^{\text{t}}, g^{\text{h}}\}$, $\mathbf{f}(g^{\text{t}}) = \mathbf{f}(g^{\text{h}})$. Function $\mathbf{m}_A(\varepsilon) := |\{\varepsilon' \in \bigcup A \mid \mathbf{f}(\varepsilon) = \mathbf{f}(\varepsilon') \text{ and } \mathbf{e}(\varepsilon) = \mathbf{e}(\varepsilon')\}|$ indicates the *multiplicity* of an extremity's family in a given genome A, defined as the number of extremities of same kind that belong to the same family in A. A family assignment $\mathbf{f}'$ is $\mathbf{f}$*-derived* if there exists no two extremities $\varepsilon, \varepsilon'$ with $\mathbf{f}'(\varepsilon) = \mathbf{f}'(\varepsilon')$ and $\mathbf{f}(\varepsilon) \neq \mathbf{f}(\varepsilon')$. A family assignment $\mathbf{f}$ is termed $\mathcal{A}$*-resolved* (or simply *resolved* if the context is clear) if for each member A of a set of genomes $\mathcal{A}$, each extremity $\varepsilon \in \mathcal{E} \setminus \mathcal{T}$ has a multiplicity not higher than 1, *i.e.*, $\mathbf{m}_A(\varepsilon) \leq 1$.

Genome rearrangement model. A *Double-Cut-and-Join* (DCJ) operation produces a new genome A' from a given genome A by either acting on a single adjacency or a pair of adjacencies. In the former case, telomeric adjacencies are created by "splitting up" a given adjacency $X = \{a_1, a_2\}$, $X \in A$ and replacing it with a new pair of telomeric adjacencies, *i.e.*, $A' = A \setminus \{X\} \cup \{\{a_1, t\}, \{a_2, t'\}\}$, with $\{t, t'\} \subseteq \mathcal{T}$. In the latter case, the DCJ operation acts on a given a pair of adjacencies $X = \{a_1, a_2\}$, $Y = \{b_1, b_2\}$, $\{X, Y\} \subseteq A$, as follows:

- $A' = A \setminus \{X, Y\} \cup \{\{a_1, b_1\}, \{a_2, b_2\}\}$, or
- $A' = A \setminus \{X, Y\} \cup \{\{a_1, b_2\}, \{a_2, b_1\}\}$, or
- $\{a_1, b_1\} \subseteq \mathcal{T} \Rightarrow A' = A \setminus \{X, Y\} \cup \{\{a_2, b_2\}\}$ (telomeres $\{a_1, b_1\}$ are removed).

An *indel* operation either inserts or deletes a continuous segment of non-telomeric markers of a chromosome, or an entire linear or circular chromosome. The unrestricted use of indels results in solutions to the rearrangement problem, where indels of entire chromosomes dominate the rearrangement scenario. Such biologically irrelevant solutions are prohibited by restricting the removal and insertion of non-telomeric markers to the minimal number that is needed so that the multiplicity of each marker family in one genome equals that of the other. In other words, we do not allow any intermediate genome to have less markers of a family than the minimum of the multiplicity of this marker in both input genomes. This restriction, also called *maximum matching model*, prevails for the remainder of this paper.

Distance calculation. For two genomes A, B and an $\{A, B\}$-resolved family assignment $\mathbf{f}$, the *DCJ indel distance* is defined as $\mathbf{d}_{\text{DCJ}-\text{ID}}(A, B) := d$, were d is the minimum number of DCJ/indel operations Δ_i in any rearrangement scenario $A \xrightarrow{\Delta_1} A_1 \xrightarrow{\Delta_2} A_2 \cdots A_{d-1} \xrightarrow{\Delta_d} B$ that transforms A into B.

Problem 1 (DCJ indel distance[10]). Given two genomes A, B and an $\{A, B\}$-

resolved family assignment $\mathbf{f}$, calculate the DCJ indel distance $\mathbf{d}_{\mathrm{DCJ-ID}}(A, B)$.

In the following, we briefly outline a solution to Problem 1 introduced in[10]. It is convenient to make use of a graph structure to calculate the DCJ-indel distance. The *relational diagram* $R(A, B, \mathbf{f})$ of two genomes A and B and $\{A, B\}$-resolved family assignment $\mathbf{f}$ is a multigraph $G = (V^{AB}, E^{AB})$ with nodes $V^{AB} = V^A \cup V^B$ representing marker extremities of genomes A and B. The graph G has three types of edges: (i) *adjacency edges* $E^{AB}_{\mathrm{adj}} = E^A_{\mathrm{adj}} \cup E^B_{\mathrm{adj}}$, corresponding to the adjacencies of A and B, (ii) *extremity edges* E^{AB}_{ext}, that connect marker extremities between the two genomes according to $\{A, B\}$-resolved family assignment $\mathbf{f}$, and (iii) *indel edges* $E^{AB}_{\mathrm{id}} = E^A_{\mathrm{id}} \cup E^B_{\mathrm{id}}$, each of which connects the extremities of a marker that is removed or inserted in the respective genome (see Figure 2 for an illustration).

Note that under the maximum matching model, the $\{A, B\}$-resolved family assignment $\mathbf{f}$ dictates which marker is removed or inserted, that is, $E^A_{\mathrm{id}} = \{\{g^{\mathrm{t}}, g^{\mathrm{h}}\} \subseteq \bigcup A \mid \mathbf{m}_B(g^{\mathrm{t}}) < 1\}$, and $E^B_{\mathrm{id}} = \{\{g^{\mathrm{t}}, g^{\mathrm{h}}\} \subseteq \bigcup B \mid \mathbf{m}_A(g^{\mathrm{t}}) < 1\}$. Subsequently, we denote by n the number of non-telomeric markers of one genome that are not inserted or deleted, which is identical for both genomes A and B:

$$n := \frac{1}{2}|\{\varepsilon \in \bigcup A \setminus \mathcal{T} \mid \mathbf{m}_B(\varepsilon) = 1\}| \quad = \frac{1}{2}|\{\varepsilon \in \bigcup B \setminus \mathcal{T} \mid \mathbf{m}_A(\varepsilon) = 1\}|$$

Each node in V^{AB} has degree one or two, so the connected components of the graph constitute alternating paths and cycles and each node corresponding to a non-telomeric marker extremity is incident to one adjacency edge and either one extremity or one indel edge. Telomeric extremity nodes have degree one and are incident to an adjacency edge.

The DCJ-indel distance, given by $\mathbf{d}_{\mathrm{DCJ-ID}}(A, B) = n - c - \frac{i}{2} + \delta$, grows inversely to the number c of cycles and twice the number i of paths of odd length in G[7,10], where δ denotes the *indel penalty*, that is, the number of indel operations required to transform A into B. The maximal contribution of a connected component of the relational diagram to the *indel penalty*, the *indel potential*, is quantified based on the concept of runs: an *A-run* is a maximal sequence of indel edges of genome A $(e_1, \ldots, e_l)$, with $e_1, \ldots, e_l \in E^A_{\mathrm{id}}$, that lies on a connected component of graph G, that does not pass over an indel edge of genome B. Analogously, a B-run is a maximal sequence of indel edges of genome B that does not pass over an indel edge of genome A. For a given connected component $C \in G$, let $\mathbf{\Lambda}(C)$ denote the number of runs it contains; the *indel-potential* of C is then given by

$$\lambda(C) = \begin{cases} 0 & \text{if } \mathbf{\Lambda}(C) = 0 \text{ and} \\ \left\lceil \frac{\mathbf{\Lambda}(C)+1}{2} \right\rceil & \text{otherwise.} \end{cases}$$

The sum of indel potentials of all connected components of graph G serves as an upper bound of indel penalty δ: $\delta \leq \sum_{C \in G} \lambda(C)$.

The calculation of the *exact* indel penalty must take into account further DCJ-induced recombinations of connected components that reduce the number of necessary indel operations by merging successive runs. Braga *et al.* show that only recombinations of those connected components that are *paths* lead to optimal rearrangement scenarios. There are 32 groups of path recombinations that can be further classified into five categories with varying negative contributions to the indel potential of its recombinants. Further details can be found in[10]. With the help of tabulation, the indel penalty—and thus the DCJ indel distance—of two genomes under a resolved family assignment can be calculated in linear time[10].

Bohnenkämper *et al.* describe an alternative approach to calculate the indel penalty that makes use of a technique called *capping*[33], where paths are assembled into alternating cycles that correspond to optimal path recombinations. This requires the extension of the relational diagram to a *capped multi-relational diagram* $MR_\circ(A, B, \mathbf{f})$[9] in which node degree is no longer restricted to one and two. In fact, in a capped multi-relational diagram, each node has degree two or higher.

To calculate the indel penalty in the above-described setting, we construct the capped multi-relational diagram H from the existing relational diagram G, i.e, $H \leftarrow G$. Then, if genomes A and B have unequal numbers of telomeres, additional nodes and edges corresponding to *telomeric adjacencies* are added to the graph and are associated with the genome that has fewer telomeres. Without loss of generality, let A be the genome with the lower number of telomeres and $\tau := |\bigcup B \cap \mathcal{T}| - |\bigcup A \cap \mathcal{T}|$. Note that as a result of the genome constraints, τ is even. Then τ new telomeric extremities $\{\varepsilon_1, \ldots, \varepsilon_\tau\} \subseteq \mathcal{T}$ are added to $V_A(H)$. The newly added nodes are pairwise connected by adjacency edges $\{\varepsilon_{2i-1}, \varepsilon_{2i}\}$, $1 \leq i \leq \frac{\tau}{2}$, and associated with edge set $E^A_{\text{adj}}(H)$. Second, additional extremity edges are added to edge set $E^{AB}_{\text{ext}}(H)$, corresponding to the Cartesian product of telomeric extremities $(\bigcup A \cap \mathcal{T}) \times (\bigcup B \cap \mathcal{T}) =: T$. The indel penalty δ is determined by finding a subset of extremity edges $T' \subset T$ in H corresponding to a perfect matching between telomeric nodes $V^A(H) \cap \mathcal{T}$ and $V^B(H) \cap \mathcal{T}$ such that the sum of indel potentials of all connected components of the graph is minimized. Observe that any perfect matching $T' \subset T$ decomposes the graph into alternating cycles, thereby transforming H into a *capped* relational diagram $R_\circ(A, B, \mathbf{f})$.

If such constructed capped relational diagram $G_\circ$ is given, the DCJ-indel distance can be expressed in terms of connected components and their indel potentials:

$$\mathbf{d}_{\text{DCJ}-\text{ID}}(A, B) = n' - \sum_{C \in G_\circ} 1 - \mathbb{1}_{E^{AB}_{\text{ext}}(C) = \emptyset} - \lambda(C)\,, \tag{1}$$

where $\mathbb{1}$ denotes the indicator function and $n' = n + \frac{V^{AB}(H) \cap \mathcal{T}}{4}$.

Cycles that do not contain any extremity edge[c] correspond to circular chromosomes that are entirely inserted or deleted in the rearrangement scenario between the two genomes. These connnected components of the relational diagram are termed *circular singletons* and differ from extremity-edge enclosing cycles in that their presence does not contribute to the count of cycles that reduces the distance but only increases the indel penalty.

[c]Note that the number of extremity edges in a cycle of a relational diagram is even.

Bohnenkämper *et al.* indirectly quantify the indel potential by counting transitions between runs of a connected component: For a given connected component C of the capped relational diagram, a *transition* is a path $\{\varepsilon_1, \varepsilon_2\}, \{\varepsilon_2, \varepsilon_3\}, \dots, \{\varepsilon_{l-1}, \varepsilon_l\} \in E^{AB}_{\text{adj}}(C) \cup E^{AB}_{\text{ext}}(C)$ such that their adjacent edges are indel edges of two different runs, *i.e.*, without loss of generality $\{\varepsilon_0, \varepsilon_1\} \in E^A_{\text{id}}$ and $\{\varepsilon_l, \varepsilon_{l+1}\} \in E^B_{\text{id}}(C)$. To count transitions, one of the edges of the transition is designated as *transition edge*, which we here arbitrarily define as edge $\{\varepsilon_1, \varepsilon_2\}$. Note that by construction, $\{\varepsilon_1, \varepsilon_2\}$ must be part of the edge set E^A_{adj}. With that, the DCJ-indel distance can be separated into four terms

$$\mathbf{d}_{\text{DCJ}-\text{ID}}(A, B) = n' - \sum_{C \in G_\circ} \mathbb{1}_{E^{AB}_{\text{id}}(C) = \emptyset} + \frac{1}{2} \sum_{e \in E^A_{\text{adj}}(G_\circ)} \mathbb{1}_{e \text{ is transition edge}} + \sum_{C \in G_\circ} \mathbb{1}_{E^{AB}_{\text{ext}}(C) = \emptyset} .$$

We refer to Figure 2 for an illustration.

3. Methods

The small parsimony problem for degenerate genomes. A *degenerate genome* is a set of unique adjacencies A that satisfies the following conditions: *(i)* $\forall g^{\text{t}} \in \bigcup A$, there exists also extremity $g^{\text{h}} \in \bigcup A$ and vice versa, and *(ii)* each telomeric extremity is used only once: $\forall \{X, X'\} \subseteq A$, $X \cap X' \cap \mathcal{T} = \emptyset$. Note that a genome is also a degenerate genome, but the reverse does not hold true in general. The *surfeit* of a degenerate genome A is the ratio $\frac{2 \cdot |A|}{|\bigcup A \setminus \mathcal{T}|}$.

Example 1. Consider adjacency sets $A = \{\{1^{\text{t}}_A, 2^{\text{t}}_A\}, \{1^{\text{h}}_A, 2^{\text{h}}_A\}\}$, $B = \{\{1^{\text{h}}_B, 2^{\text{h}}_B\}, \{1^{\text{h}}_B, 3^{\text{h}}_B\}, \{2^{\text{h}}_B, 3^{\text{h}}_B\}, \{1^{\text{t}}_B, 2^{\text{t}}_B\}, \{1^{\text{t}}_B, 3^{\text{t}}_B\}, \{2^{\text{t}}_B, 3^{\text{t}}_B\}\}$, $C = \{\{1^{\text{t}}_C, 2^{\text{t}}_C\}, \{1^{\text{h}}_C, 2^{\text{h}}_C\}, \{1^{\text{t}}_C, 2^{\text{h}}_C\}, \{1^{\text{h}}_C, 2^{\text{t}}_C\}, \{t.1^\circ_C, 2^{\text{t}}_C\}, \{t.2^\circ_C, 1^{\text{h}}_C\}, \{t.3^\circ_C, 2^{\text{h}}_C\}\}\}$, as illustrated in Figure 3. All three are degenerate genomes, but only A is a genome. Their surfeits are 1.0 and 2.0, and 3.5, respectively.

A genome A' is A-*derived* from degenerate genome A (or simply "derived from A") if $A' \subseteq A$ and $\bigcup A' \setminus \mathcal{T} = \bigcup A \setminus \mathcal{T}$. Conversely, a degenerate genome A is *linearizable* if there exists an A-derived genome. In fact, many degenerate genomes are not linearizable.

Example 1 (cont'd). Degenerate genome B is not linearizable. Degenerate genome C is linearizable, and $\{\{1^{\text{t}}_C, 2^{\text{t}}_C\}, \{1^{\text{h}}_C, 2^{\text{h}}_C\}\}$, $\{\{1^{\text{t}}_C, 2^{\text{h}}_C\}, \{1^{\text{t}}_C, 2^{\text{h}}_C\}\}$, $\{\{1^{\text{t}}_C, 2^{\text{t}}_C\}, \{t.2^\circ_C, 1^{\text{h}}_C\}, \{t.3^\circ_C, 2^{\text{h}}_C\}\}$, $\{\{1^{\text{t}}_C, 2^{\text{h}}_C\}, \{t.1^\circ_C, 2^{\text{t}}_C\}, \{t.2^\circ_C, 1^{\text{h}}_C\}\}$ are all C-derived genomes.

Ensuring linearizability of degenerate genomes. The ancestral adjacencies that DeCoSTAR infers in the third step of the ancestral reconstruction workflow do not guarantee that the resulting degenerate genomes are linearizable. Whether a polynomial algorithm exists that can test whether a given degenerate genome A is linearizable is an open question. Therefore, to ensure linearizability, we augment degenerate genomes with additional telomeric adjacencies. That is, for a non-telomeric extremity $\varepsilon \in \bigcup A \cap \mathcal{T}$ of a degenerate genome A such that ε is not already involved in an adjacency with a telomeric extremity, we add a new telomeric adjacency $\{\varepsilon, t_\varepsilon\}$, $t_\varepsilon \in \mathcal{T}$, to A and assign it weight 0. From this procedure parts of the degenerate genome are omitted that are linearizable. To this end, we study connected components of the graph $G_A = (V, E)$ where $V = \bigcup A$ and $E = A$. Nodes of a connected component $C \in G_A$ do not need to be augmented with telomeric adjacencies if $|C|$ is even and C is a cycle, a path, or fully connected.

The small parsimony problem for degenerate genomes. In the following, we study a variant of Problem 1 in which linearizable degenerate genomes and an unresolved family assignment are given with the objective to derive genomes and a resolved family assignment that minimize the DCJ indel distance. For a given pair of linearizable degenerate genomes, often multiple solutions to the DCJ indel distance problem exist. That is, different choices of derived genomes, resolved family assignments, as well as rearrangement scenarios lead to the same optimal DCJ indel distance. The solution space increases with the surfeit of the degenerate genomes as well as the multiplicity of marker extremities under the given unresolved family assignment. At the same time, solutions based on derived genomes that contain excessive numbers of linear chromosomes are not likely to occur in the biological realm. To reduce the solution space as well as to ameliorate biological interpretability, we aim to find resolved genomes with fewer linear chromosomes even if that comes at the expense of an increased DCJ indel distance. At last, we want to facilitate the consideration of prior knowledge about presence or absence of certain adjacencies in degenerate genomes, if such is available:

Problem 2 (Weighted Degenerate DCJ indel distance). Given a weighting scheme $\mathbf{w} : \mathcal{E} \times \mathcal{E} \to \mathbb{R}$, some $\alpha, \beta \in [0, 1]$, two linearizable degenerate genomes A, B and family assignment $\mathbf{f}$, find A-derived genome A', B-derived genome B', and $\mathbf{f}$-derived $\{A', B'\}$-resolved family assignment $\mathbf{f}'$ that minimize the linear combination

$$(1 - \alpha - \beta) \cdot \sum_{X \in A' \cup B'} -\mathbf{w}(X) + \alpha \cdot \mathbf{d}_{\mathrm{DCJ-ID}}(A', B') + \beta \cdot |(\bigcup A' \cup \bigcup B') \cap \mathcal{T}| \,.$$

Because the surfeit of a derived genome only depends on the number of its telomeres, minimizing its number of telomeres is equivalent to minimizing its surfeit.

In this work, we study the following generalization of Problem 2 that represents a variant of the *small parsimony problem* (SPP) under the weighted degenerate DCJ indel distance. To this end, we make use of an additional graph that establishes relationships between a given set of genomes $\mathcal{A} = A_1, \ldots, A_k$ and that we call "phylogeny". A *phylogeny* Γ is a connected graph with nodes $V(\Gamma) = \mathcal{A}$. Nodes $L \subseteq V(\Gamma)$ of degree 1 are termed *leaves.*

Problem 3 (SPP-DCJ). Given a phylogeny Γ and a set of linearizable degenerate genomes $A_1, \ldots, A_k$ corresponding to the node set $V(\Gamma) = \{A_1, \ldots, A_k\}$, find genomes $A'_1 \subseteq A_1, \ldots, A'_k \subseteq A_k$ that minimize the sum of weighted degenerate DCJ indel distances along the edges of Γ.

Finding solutions to SPP-DCJ. Our solution to SPP-DCJ makes use of the capped multi-relational diagram as introduced in[9], subsequently simply denoted by *multi-relational diagram.* The multi-relational diagram represents a superposition of capped relational diagrams of all possible derived genomes of degenerate genomes A and B. For each edge $\{A, B\}$ of a given phylogeny Γ and a given family assignment $\mathbf{f}$, we construct multi-relational diagram $(V^{AB}, E^{AB}) = MR_\circ(A, B, \mathbf{f})$. Our method for solving Problem 3 is an extension of DING[9] and is formulated as an ILP. The key idea of this ILP is to simultaneously calculate solutions to Problem 2 for each pair of genomes $\{A, B\} \in E(\Gamma)$ and family assignment $\mathbf{f}$ while maintaining consistency in the selection of adjacencies of each derived genome among all solutions. But rather than minimizing the sum of objectives specified in Problem 2, the algorithm solves the inverse maximization problem. A solution is encoded as capped relational diagram $R_\circ(A', B', \mathbf{f}')$ and represents A-derived genome A', B-derived genome B' and $\mathbf{f}$-derived $\{A, B\}$-resolved family assignment $\mathbf{f}'$. Details of our algorithm can be found in Appendix Appendix B.

Identifying candidates of circular singletons. For the construction of the ILP described in Algorithm 1 the set of circular singleton candidates $\mathcal{C}^{AB}$, $\{A, B\} \in E(\Gamma)$ must be known. The number of candidates depends heavily on the surfeit of degenerate genomes A and B and is bounded by the number of ordered partitions of edge set E^A_{id} and E^B_{id}, respectively. These numbers are also known as *ordered Bell numbers.* In practice, the size of $\mathcal{C}^{AB}$ is small, because degenerate genomes have typically low surfeit. We construct the set of circular singleton candidates by traversing the multi-relational diagram, thereby identifying alternating cycles of adjacency and indel edges.

Reducing the search space of optimal path recombinations. A substantial factor that impacts the running time are large marker families and the number telomeres in degenerate genomes[9]. Large numbers of telomeres act similarly to large marker families, as in the capping strategy used in the construction of the multi-relational diagram $MR_\circ(A, B, \mathbf{f})$ extremity edges are drawn between all telomeric extremities of the degenerate genomes A and B. This is to include path recombinations in the optimization that may further minimize the DCJ-indel distance. If path recombinations can be determined beforehand, then the search space of optimal path recombinations can be reduced by omitting non-optimal recombinations. However, paths used in the solution of SPP-DCJ are often unknown, as they depend on the choice of indel and extremity edges. Still, for any two telomeres, it is possible to identify the subgraph that spans all possible alternating paths that connect them. We then use these subgraphs to classify the telomere pairs into two groups: those, contained in indel-free subgraphs, and those, contained in indel-enclosing subgraphs. As the former group does not contribute to the indel-penalty, the optimization of their recombinations will maximize the count of indel-free cycles. Conversely, the optimization of indel-enclosing cycles does not affect that count, but will only reduce the indel-penalty. So path recombinations within the two classes of telomeres can be optimized independently.

We identify the class membership of telomeres using a depth-first search (DFS) strategy, which, for a given telomere $t \in V^{AB}$, simultaneously identifies all possible telomeres that can be recombination partners of t. While traversing the graph to find these partners, we also record the occurrence of indel edges. The procedure is repeated for all telomeres. Overall, the classification completes in $O(|V^{AB} \cap T| \cdot |E^{AB}|)$ time. Identified indel-free recombination paths of telomeric extremities from opposite genomes can be directly connected by an extremity edge. However, indel-free paths between telomeres of the same genome and indel-enclosing path combinations of opposite genomes are connected by extremity edges in an all-vs-all manner. If the latter group has an unequal number of telomeres in A and B, additional telomeres are added, as described above in the construction of the multi-relational diagram.

4. Results

We implemented a Python program that constructs the ILP of Algorithm 1 for a given input data set. The source and instructions for its usage is available at `https://github.com/danydoerr/spp_dcj`. The ILP is solved with Gurobi[1].

We evaluated our method on simulated data of moderate scale, using a phylogeny with 10 extant and 9 ancestral genomes. The genomes and their reconciled gene trees were generated with the genome evolution simulator ZOMBI[14]. ZOMBI simulates intra-chromosomal evolution and supports a variety of evolutionary events, including inversions, transpositions, segmental duplications, and segmental deletions. To generate the data sets, we gradually increased the evolutionary scale (number of expected events per branch of the phylogenetic tree), thus creating series of data sets with increasing numbers of evolutionary events. We generated 80 data sets with varying numbers of above-mentioned events. Tables 1 and 2 in the Appendix display these numbers for two representative series of data sets.

Each simulated genome is associated with a node of the phylogeny and corresponds to a single circular chromosome comprised of *ca.* 1,000 markers, the exact number depending on the rate of gene duplication and loss used in the simulation. We conducted three different experiments, each introducing a different type of noise in the data used as input to the ancestral genome reconstruction workflow. In all these experiments, we used the true reconciled gene trees in the input data. The rationale to rely on true reconciled gene trees is to avoid conflating the analysis of the SPP algorithm by factors that are external and therefore unrelated to the ILP when assessing its quality of reconstruction. We discuss this aspect further in conclusion. At last, we set $\alpha = \frac{1}{2}$ and $\beta = \frac{1}{4}$ for calculating the weighted degenerate DCJ-indel distance in all experiments.

Random Noise. In the first experiment, we tested our method for susceptibility of biased and unbiased noise: We generated ancestral degenerate genomes by augmenting the true ancestral genomes of our simulated data sets with random ancestral adjacencies of two kinds: uniform random adjacencies, sampled uniformly from the space of all possible adjacencies of a genome, and *adversarial* random adjacencies formed between extremities of pairs of genes whose families already form adjacencies in the parental degenerate genome. Adversarial adjacencies are particularly detrimental to parsimony-based reconstruction as they are natural candidates to be conserved along a branch of the phylogeny. Altogether, we generated three batches, each comprising 480 data sets, with 0%, 50%, and 100% of the added random adjacencies being adversarial. Within each batch, we modulated the surfeits of the degenerate ancestral genomes within the range of 1.2 to 3 by adding varying numbers of random adjacencies. In that experiment all adjacencies received weight 1.

Except for 26 data sets, SPP-DCJ perfectly recovered the true adjacencies in all three batches. Of the data sets with incorrectly reconstructed adjacencies, 1 occurred in a data set whose degenerate genomes had surfeit 2, while the remaining had surfeit 3. Still, SPP-DCJ erred only on very few adjacencies, scoring after all 99.999% and 99.7% in mean precision and recall, respectively. This baseline experiment shows that a parsimony approach in the evolutionary model implemented in SPP-DCJ is able to recover the true evolutionary signal even in the presence of significant levels of noise, including adversarial noise explicitly designed at simulating parsimonious evolution of false positive adjacencies.

Principled Noise. In our second set of experiments, we reconstructed ancestral genomes based again on including the true ancestral adjacencies in the ancestral degenerate genomes, but augmented by adjacencies recovered by DeCoSTAR[15]. We used DeCoSTAR on the true reconciled gene trees and subsequently derived ancestral adjacencies from the returned adjacency evolution forests. This experiment thus includes a crucial step of the ancestral reconstruction workflow described in Figure 1, that is the inference of ancestral adjacencies based on local parsimony (at the level of pairs of gene families) that is more likely to generate false positive adjacencies (see[35] for similar experiments with this framework), although keeping the true adjacencies in the input too. In the calculation of the weighted degenerate DCJ-indel distance we weighted true adjacencies by one and DeCoSTAR adjacencies by its own assigned weights. In 78 out of 80 experiments, SPP-DCJ obtained a perfect score in recovering the true adjacencies. In the remaining two experiments, 2 (resp. 4) adjacencies were incorrectly selected by SPP-DCJ. This indicates that in rare cases, DeCoSTAR introduces bias that negatively impacts the accurate reconstruction of ancestral gene orders.

DeCoSTAR Reconstruction. In the last experiment we scrutinize the second part of the reconstruction workflow by performing the last three out of five steps (see Figure 1) as if the underlying data were not simulated, but represented actual biological data. In other words, we applied SPP-DCJ only on adjacencies inferred by DecoSTAR. Overall, the precision and recall of adjacencies in the derived genomes of SPP-DCJ do not fall below 97% and 98.7%, respectively. The plot on the top left of Figure 4 visualizes both statistics for all samples as a function of the scale of the phylogeny. As expected, the quality of reconstruction declines with increasing scale, *i.e.*, numbers of evolutionary events.

Runtime analysis. In all experiments described above except those cases of the first experiment where degenerate genomes were excessively large (surfeit 3), Gurobi needed only few seconds to find an optimal solution for SPP-DCJ (cf. Figure 4 top right and bottom left). For the data sets with large degenerate genomes 3, the program completed on average after 16 minutes, as indicated by bottom left plot of Figure 4. This is impressive as SPP-DCJ must calculate the DCJ-indel distance for 18 pairs of degenerate genomes simultaneously. Just as its predecessor, the runtime of SPP-DCJ is only weakly affected by genome size. To prove this point, we generated a series of data sets with genomes comprising 10,000 markers and numbers of evolutionary events comparable to those described in Table 2 in the Appendix. SPP-DCJ found optimal solutions in each run after 47 CPU seconds.

SPP-DCJ extends DING[9] from pairwise distance calculations to the here introduced small parsimony setting. It outperforms its predecessor for genomes with many linear chromosomes. To illustrate this, we ran an experiment where we simulated the evolution of genomes with many linear chromosomes (between 60 and 260), close in nature to haplotype-resolved human genomes, or partially assembled genomes. Our approach in handling optimal recombinations of indel-free paths tremendously reduces the running time. In the experiment, we limited the running time of the pairwise distance solver described in[9] to two hours. While SPP-DCJ completes all but one pairwise calculations within a few seconds, DING reaches the runtime limit in six out of the eleven runs (cf. bottom right plot of Figure 4).

5. Conclusion

In this work we introduced a novel method for solving the SPP in the DCJ-indel model, within an ancestral reconstruction workflow based on the seminal work of Sankoff and El-Mabrouk[31] and taking advantage of recent advances in the field of phylogenomics[15].

Our experiments on simulated data of moderate size provides a proof of concept that SPP-DCJ can reconstruct efficiently accurate ancestral gene orders. Assessing the performance of SPP-DCJ on data of larger scale is a natural avenue for further work, as is a more thorough exploration of the impact of errors in reconciled gene trees[35]. The joint reconstruction of ancestral gene orders and extant scaffolds using DeCoSTAR has recently been applied with good results to a large-scale datasets of mosquito genomes[3]. SPP-DCJ naturally lends itself to such an application as degenerate extant gene orders can be considered by the method; the application of SPP-DCJ to comparative scaffolding is thus a promising research direction. Last, the SPP-DCJ algorithm can work with non-treelike phylogenies. This makes our method applicable to population genomics studies, where introgression events introduce reticulation nodes in the species phylogeny.

References

1. *Gurobi Optimizer*, https://www.gurobi.com/products/gurobi-optimizer.
2. Angibaud S, Fertin G, Rusu I, Thévenin A, Vialette S, On the approximability of comparing genomes with duplicates, *Journal of Graph Algorithms and Applications* **13**(1):19–53, 2009.
3. Anselmetti Y, Tannier E, Bérard S, Chauve C, Phylogenetic signal from rearrangements in 18 *Anopheles* species by joint scaffolding extant and ancestral genomes, *BMC Genomics* **19**:96, 2018. doi:10.1186/s12864-018-4466-7.
4. Avdeyev P, Alekseyev MA, Linearization of ancestral genomes with duplicated genes, *BCB '20: 11th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics, Virtual Event, USA, September 21-24, 2020*, ACM, pp. 53:1–53:10, 2020. doi:10.1145/3388440.3412484.
5. Avdeyev P, Jiang S, Alekseyev MA, Linearization of median genomes under the double-cut-and-join-indel model, *Evolutionary Bioinformatics* **15**:1176934318820534, 2019. doi:10.1177/1176934318820534.
6. Bergeron A, Blanchette M, Chateau A, Chauve C, Reconstructing ancestral gene orders using conserved intervals, Jonassen I, Kim J (eds.), *Algorithms in Bioinformatics, 4th International Workshop, WABI 2004, Bergen, Norway, September 17-21, 2004, Proceedings*, Lecture Notes in Computer Science, Vol. 3240, Springer, pp. 14–25, 2004. doi:10.1007/978-3-540-30219-3_2.
7. Bergeron A, Mixtacki J, Stoye J, A unifying view of genome rearrangements, Bucher P, Moret BME (eds.), *Algorithms in Bioinformatics, 6th International Workshop, WABI 2006, Zurich, Switzerland, September 11-13, 2006, Proceedings*, Lecture Notes in Computer Science, Vol. 4175, Springer, pp. 163–173, 2006. doi:10.1007/11851561_16.
8. Blin G, Chauve C, Fertin G, Rizzi R, Vialette S, Comparing genomes with duplications: A computational complexity point of view, *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **4**(4):523–534, 2007. doi:10.1145/1322075.1322079.
9. Bohnenkämper L, Braga MDV, Doerr D, Stoye J, Computing the rearrangement distance of natural genomes, Schwartz R (ed.), *Research in Computational Molecular Biology - 24th Annual International Conference, RECOMB 2020, Padua, Italy, May 10-13, 2020, Proceedings*, Lecture Notes in Computer Science, Vol. 12074, Springer, pp. 3–18, 2020. doi:10.1007/978-3-030-45257-5_1.
10. Braga MDVM, Willing EE, Stoye JJ, Double cut and join with insertions and deletions., *Journal of computational biology* **18**(9):1167–1184, 2011. doi:10.1089/cmb.2011.0118.
11. Caprara A, The reversal median problem, *INFORMS Journal on Computing* **15**(1):93–113, 2003. doi:10.1287/ijoc.15.1.93.15155.
12. Chauve C, El-Mabrouk N, Guéguen L, Semeria M, Tannier E, Duplication, rearrangement and reconciliation: A follow-up 13 years later, in *Models and Algorithms for Genome Evolution*, , Springer London, pp. 47–62, 2013. doi:10.1007/978-1-4471-5298-9_4.
13. Chauve C, Tannier E, A methodological framework for the reconstruction of contiguous regions of ancestral genomes and its application to mammalian genomes, *PLoS Computational Biology* **4**(11), 2008. doi:10.1371/journal.pcbi.1000234.
14. Davín AA, Tricou T, Tannier E, de Vienne DM, Szöllősi GJ, Zombi: a phylogenetic simulator of trees, genomes and sequences that accounts for dead linages, *Bioinformatics* **36**(4):1286–1288, 2019. doi:10.1093/bioinformatics/btz710.
15. Duchemin W, Anselmetti Y, Patterson M, Ponty Y, Bérard S, Chauve C, Scor-

navacca C, Daubin V, Tannier E, DeCoSTAR: Reconstructing the ancestral organization of genes or genomes using reconciled phylogenies, *Genome Biology and Evolution* **9**(5):1312–1319, 2017. doi:10.1093/gbe/evx069.

16. Earnest-DeYoung JV, Lerat E, Moret BME, Reversing gene erosion - reconstructing ancestral bacterial genomes from gene-content and order data, *Algorithms in Bioinformatics, 4th International Workshop, WABI 2004, Bergen, Norway, September 17-21, 2004, Proceedings*, Lecture Notes in Computer Science, Vol. 3240, pp. 1–13, 2004. doi:10.1007/978-3-540-30219-3_1.
17. Feijão 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. doi:10.1109/TCBB.2011.34.
18. Fertin G, Labarre A, Rusu I, Tannier E, Vialette S, *Combinatorics of Genome Rearrangements*, MIT Press, 2009. ISBN 978-0-262-06282-4.
19. Gagnon Y, Blanchette M, El-Mabrouk N, A flexible ancestral genome reconstruction method based on gapped adjacencies, *BMC Bioinformatics* **13**(S-19):S4, 2012. doi:10.1186/1471-2105-13-S19-S4.
20. Kovác J, On the complexity of rearrangement problems under the breakpoint distance, *Journal of Computational Biology* **21**(1):1–15, 2014. doi:10.1089/cmb.2013.0004.
21. Luhmann N, Lafond M, Thevenin A, Ouangraoua A, Wittler R, Chauve C, The scj small parsimony problem for weighted gene adjacencies, *IEEE/ACM Transactions on Computational Biology and Bioinformatics* **16**(5):1364–1373, 2019. doi:10.1109/TCBB.2017.2661761.
22. Ma J, Ratan A, Raney BJ, Suh BB, Zhang L, Miller W, Haussler D, DUPCAR: reconstructing contiguous ancestral regions with duplications, *Journal of Computational Biology* **15**(8):1007–1027, 2008. doi:10.1089/cmb.2008.0069.
23. Ma J, Zhang L, Suh BB, Raney BJ, Burhans R, Kent WJ, Blanchette M, Haussler D, Miller W, Reconstructing contiguous regions of an ancestral genome., *Genome Research* **16**:1557–1565, 2006.
24. Mane AC, *Genome rearrangement problems accounting for duplicate genes*, Master's Thesis, Simon Fraser University, Department of Mathematics, 2018, https://summit.sfu.ca/item/18563.
25. Mane AC, Lafond M, Feijão PC, Chauve C, The distance and median problems in the single-cut-or-join model with single-gene duplications, *Algorithms for Molecular Biology* **15**(1):8, 2020. doi:10.1186/s13015-020-00169-y.
26. Ming R, Van Buren R, *et al.*, The pineapple genome and the evolution of CAM photosynthesis, *Nature Genetics* **47**(12):1435–1442, 2015. doi:10.1038/ng.3435.
27. Murat F, Zhang R, *et al.*, Karyotype and gene order evolution from reconstructed extinct ancestors highlight contrasts in genome plasticity of modern rosid crops, *Genome Biology and Evolution* **7**(3):735–749, 2015. doi:10.1093/gbe/evv014.
28. Neafsey DE, Waterhouse RM, *et al.*, Highly evolvable malaria vectors: The genome of 16 *Anopheles* mosquitoes, *Science* **347**(6217):1258522, 2015. doi:10.1126/science.1258522.
29. Pe'er I, Shamir R, The median problems for breakpoints are NP-complete, *Electronic Colloquium on Computational Complexity (ECCC)* **5**(71), 1998.
30. Rajaraman A, Ma J, Reconstructing ancestral gene orders with duplications guided by synteny level genome reconstruction, *BMC Bioinformatics* **17**(S-14):201–212, 2016. doi:10.1186/s12859-016-1262-8.
31. Sankoff D, El-Mabrouk N, Duplication, rearrangement, and reconciliation, in *Comparative Genomics*, , Springer Netherlands, pp. 537–550, 2000. doi:10.1007/978-94-011-4309-7_46.

32. Shao M, Lin Y, Moret BME, An exact algorithm to compute the double-cut-and-join distance for genomes with duplicate genes, *Journal of Computational Biology* **22**(5):425–435, 2015. doi:10.1089/cmb.2014.0096.
33. Shao M, Moret BME, On computing breakpoint distances for genomes with duplicate genes, *Journal of Computational Biology* **24**(6):571–580, 2017. doi:10.1089/cmb.2016.0149.
34. Smith J, Timoshevskaya N, *et al.*, The sea lamprey germline genome provides insights into programmed genome rearrangement and vertebrate evolution, *Nature* **50**:270–277, 2018. doi:10.1038/s41588-017-0036-1.
35. Tannier E, Bazin A, Davín A, Guéguen L, Bérard S, Chauve C, Ancestral Genome Organization as a Diagnosis Tool for Phylogenomics, in *Phylogenetics in the Genomic Era*, eds., Scornavacca C, Delsuc F, Galtier N, No commercial publisher — Authors open access book, pp. 2.5:1–2.5:19, 2020.
36. Tannier E, Zheng C, Sankoff D, Multichromosomal median and halving problems under different genomic distances, *BMC Bioinformatics* **10**, 2009. doi:10.1186/1471-2105-10-120.
37. Waterhouse RM, Agazenov S, *et al.*, Evolutionary superscaffolding and chromosome anchoring to improve *Anopheles* genome assemblies, *BMC Biology* **18**(1):1, 2020. doi:10.1186/s12915-019-0728-3.
38. Yancopoulos S, Attie O, Friedberg R, Efficient sorting of genomic permutations by translocation, inversion and block interchange, *Bioinformatics* **21**(16):3340–3346, 2005. doi:10.1093/bioinformatics/bti535.
39. Yang N, Hu F, Zhou L, Tang J, Reconstruction of ancestral gene orders using probabilistic and gene encoding approaches, *PLOS ONE* **9**(10):1–11, 2014. doi:10.1371/journal.pone.0108796.
40. Zhou L, Tang J, Ancestral reconstruction with duplications using binary encoding and probabilistic model, *7th International Conference on Bioinformatics and Computational Biology (BICoB 2015)*, pp. 97–104, 2015.

Appendix A. Figures

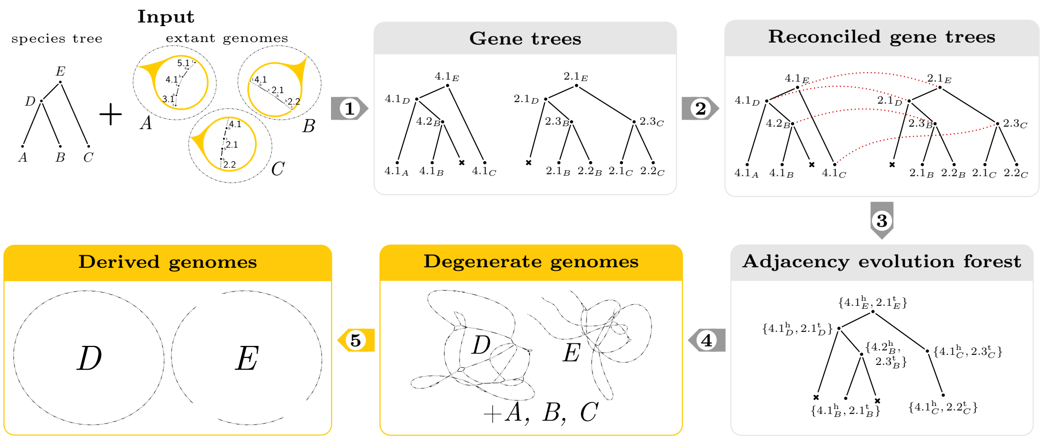

Fig. 1. Ancestral genome reconstruction workflow followed in this paper, from a given phylogeny and genomes at its extant nodes. The first digit in the label of an illustrated genomic marker, *e.g.* **4**.1, indicate its gene family membership.

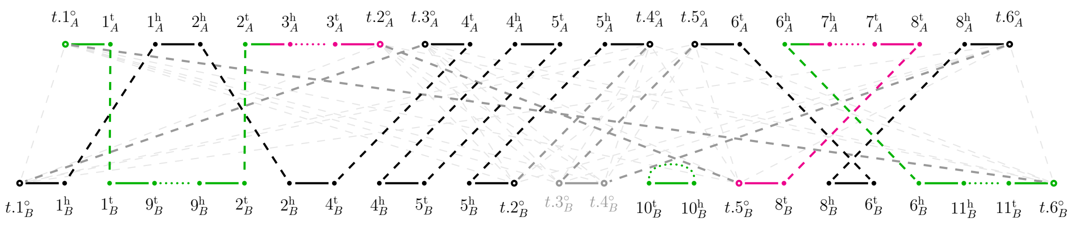

Fig. 2. **Non-gray part**: Relational diagram between two exemplary genomes A and B. Solid lines show edges of E_{adj}^{AB}, dotted lines show edges of E_{id}^{AB}, and dashed lines show edges of E_{ext}^{AB}. Paths of A- and B-runs (dotted lines) are highlighted in green and magenta, respectively. **Dark-gray part:** Additional telomeric extremities and telomeric extremity edges that, together with black and colored edges make up a capped relational diagram of A and B. **Light-gray edges** correspond to candidates of telomeric adjacencies.

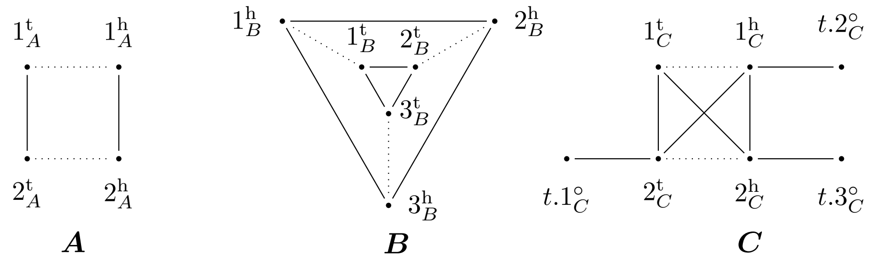

Fig. 3. Visualization of genome A, degenerate genome B, and linearizable degenerate genome C. Solid edges correspond to adjacencies, dotted lines connect extremities of the same marker.

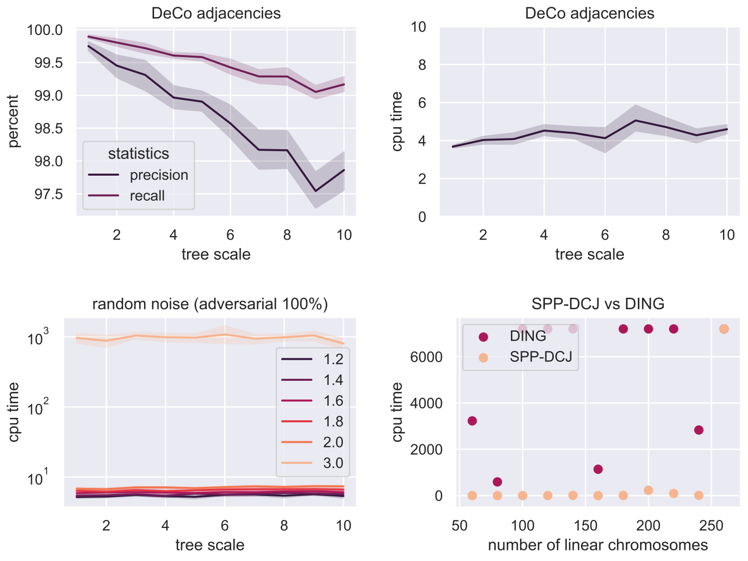

Fig. 4. **(Top left)** Precision and recall of reconstructed ancestral adjacencies in experiments with ancestral degenerate genomes defined by DeCoSTAR's inferred adjacencies. **(Top right)** Runtime of experiments using DeCoSTAR's inferred adjacencies. **(Bottom left)** Runtime of SPP-DCJ on noisy data with varying surfeit and 100% adversarial noise as a function of tree scale. **(Bottom right)** Runtime comparisons of SPP-DCJ and DING for pairwise distance computations.

Appendix B. ILP

Our solution to SPP-DCJ makes use of the capped multi-relational diagram as introduced in[9], subsequently simply denoted by *multi-relational diagram.* The multi-relational diagram represents a superposition of capped relational diagrams of all possible derived genomes of degenerate genomes A and B. For each edge $\{A, B\}$ of a given phylogeny Γ and a given family assignment $\mathbf{f}$, we construct multi-relational diagram $(V^{AB}, E^{AB}) = MR_\circ(A, B, \mathbf{f})$. Similarly to the relational diagram introduced in Section 2, the multi-relational diagram is a composition of node sets $V^{AB} = V^A \cup V^B$ and edge sets $E^{AB} = E^{AB}_{\text{adj}} \cup E^{AB}_{\text{ext}} \cup E^{AB}_{\text{id}}$.

- A node of the multi-relational diagram corresponds either to an extremity of degenerate genomes A or B, or to an additional telomeric extremity required to match the number telomeres in both degenerate genomes. Here, we take advantage of the following observation: Although the number of telomeres of a degenerate genome can be odd, only an even subset thereof can be simultaneously part of a derived genome. In other words, if a degenerate genome has x telomeres, at most $2\lfloor \frac{x}{2} \rfloor$ telomeres need counterparts in the opposing degenerate genome. Let $l = |\bigcup B \cap \mathcal{T}| - |\bigcup A \cap \mathcal{T}|$ be the difference of telomere counts of both degenerate genomes. Then the set of additional telomeres added to the relational diagram of A and B is defined by sets $T^A = \{t_i \in \mathcal{T} \mid 1 \leq i \leq 2\lfloor \frac{l}{2} \rfloor\}$ and $T^B = \{t_i \in \mathcal{T} \mid 1 \leq i \leq 2\lfloor \frac{-l}{2} \rfloor\}$. Observe that by construction, the size of both sets is even and at most one of the two sets is non-empty. Then the node sets associated with degenerate genomes A and B are defined by $V^A = \bigcup A \cup T^A$ and $V^B = \bigcup B \cup T^B$.
- Likewise, the set of adjacency edges $E^{AB}_{\text{adj}} = E^A_{\text{adj}} \cup E^B_{\text{adj}}$ comprises adjacencies of A and B as well as additional telomeric adjacencies between nodes of T^A or T^B, *i.e.*, $E^A_{\text{adj}} = A \cup \{\{t_{2i-1}, t_{2i}\} \mid 1 \leq i \leq 2\lfloor \frac{l}{2} \rfloor\}$, $E^B_{\text{adj}} = B \cup \{\{t_{2i-1}, t_{2i}\} \mid 1 \leq i \leq 2\lfloor \frac{-l}{2} \rfloor\}$.
- Extremity edges connect marker extremities of degenerate genome A with those of degenerate genome B according to family assignment $\mathbf{f}$, but they also connect all telomeric extremities of A with all telomeric extremities of B, *i.e.*, $E^{AB}_{\text{ext}} = \{\{\varepsilon, \varepsilon'\} \in V^A \times V^B \mid \varepsilon, \varepsilon' \in \mathcal{T}$ or $\mathbf{f}(\varepsilon) = \mathbf{f}(\varepsilon')$ and $\mathbf{e}(\varepsilon) = \mathbf{e}(\varepsilon')\}$.
- Indel edges $E^{AB}_{\text{id}} = E^A_{\text{id}} \cup E^B_{\text{id}}$ connect extremities of markers whose families are overrepresented in the respective genome, *i.e.*, $E^A_{\text{id}} = \{\{g^{\text{t}}, g^{\text{h}}\} \subseteq \bigcup A \mid \mathbf{m}_A(g^{\text{t}}) > \mathbf{m}_B(g^{\text{t}})\}$, and $E^B_{\text{id}} = \{\{g^{\text{t}}, g^{\text{h}}\} \subseteq \bigcup B \mid \mathbf{m}_B(g^{\text{t}}) > \mathbf{m}_A(g^{\text{t}})\}$.

Our method for solving Problem 3 is an extension of DING[9] and is formulated as an ILP (Algorithm 1). The key idea of the ILP is to simultaneously calculate solutions to Problem 2 for each pair of genomes $\{A, B\} \in E(\Gamma)$ and family assignment $\mathbf{f}$ while maintaining consistency in the selection of adjacencies of each derived genome among all solutions. A solution is encoded as capped relational diagram $R_\circ(A', B', \mathbf{f}')$ and represents A-derived genome A', B-derived genome B' and

Algorithm 1 ILP for solving Problem SPP-DCJ.

Objective:

`Maximize`

$$(1-\alpha-\beta)\sum_{X\in A\cup B}\mathbf{w}(X)+\alpha\left(\sum_{\substack{1\le i\le|V^{AB}\setminus\mathcal{T}|,\\ AB\in E(\Gamma)}} z_i^{AB}-\frac{1}{2}\cdot\sum_{\substack{e\in E^{AB},\\ AB\in E(\Gamma)}} t_e^{AB}-\sum_{\substack{C\in\mathcal{C}^{AB}\\ AB\in E(\Gamma)}} s_C\right)-\beta\sum_{\substack{v\in V^{AB}\cap\mathcal{T}\\ AB\in E(\Gamma)}} o_v$$

For each multi-relational diagram $MR_\circ(A,B,\mathrm{f})$, $\{A,B\}\in E(\Gamma)$:

Constraints:

$$\begin{array}{lrl}
(\mathtt{C.01}) & o_v = 1 & \forall\ v\in V^{AB}\setminus\mathcal{T}\\
(\mathtt{C.02}) & \displaystyle\sum_{\{u,v\}\in E_{\text{adj}}^{AB}} x_{uv} = o_v & \forall\ v\in V^{AB}\\
 & \displaystyle\sum_{\{u,v\}\in E_{\text{id}}^{AB}\cup E_{\text{ext}}^{AB}} x_{uv}^{AB} = o_v & \forall\ v\in V^{AB}\\
(\mathtt{C.03}) & x_e^{AB} = x_f^{AB} & \forall\ e,f\in E_{\text{ext}}^{AB} \text{ such that } e \text{ and } f \text{ are siblings}\\
(\mathtt{C.04}) & y_j^{AB}+i(1-x_{v_iv_j}) \ge y_i^{AB} & \forall\ \{v_i,v_j\}\in E_{\text{adj}}^{AB}\ ,\\
 & y_j^{AB}+i(1-x_{v_iv_j}^{AB}) \ge y_i^{AB} & \forall\ \{v_i,v_j\}\in E_{\text{id}}^{AB}\cup E_{\text{ext}}^{AB}\ ,\\
(\mathtt{C.05}) & i(1-x_{v_iv_j}^{AB}) \ge y_i^{AB} & \forall\ \{v_i,v_j\}\in E_{\text{id}}^{AB}\\
(\mathtt{C.06}) & i\cdot z_i^{AB} \le y_i^{AB} & \forall\ 1\le i\le|V^{AB}\setminus\mathcal{T}|\\
(\mathtt{C.07}) & 1-x_{uv}^{AB} \ge r_v^{AB} & \forall\ \{u,v\}\in E_{\text{id}}^{A}\ ,\\
 & x_{u'v'}^{AB} \le r_{v'}^{AB} & \forall\ \{u',v'\}\in E_{\text{id}}^{B}\\
(\mathtt{C.08}) & r_v^{AB}-r_u^{AB}-(1-x_{uv}^{AB}) \ge t_{uv}^{AB} & \forall\ \{u,v\}\in E^{AB}\\
(\mathtt{C.09}) & \displaystyle\sum_{e\in E_{\text{adj}}^{AB}(C)} x_e+\sum_{e\in E_{\text{id}}^{AB}(C)} x_e^{AB}+1 \le s_C & \forall\ C\in\mathcal{C}^{AB}\\
(\mathtt{C.10}) & \displaystyle\sum_{\substack{d\in E_{\text{id}}^{A}\\ d\cap e\neq\emptyset}} x_d^{AB}-t_e^{AB} \ge 0 & \forall\ e\in E_{\text{adj}}^{A}\\
 & t_e^{AB} = 0 & \forall\ e\in E_{\text{id}}^{AB}\cup E_{\text{ext}}^{AB}\\
(\mathtt{C.11}) & \displaystyle\sum_{v\in V^{A}\cap\mathcal{T}} o_v-2a_A = 0 & \\
 & \displaystyle\sum_{v\in V^{B}\cap\mathcal{T}} o_v-2a_B = 0 &
\end{array}$$

Domains:

$$\begin{array}{lrl}
(\mathtt{D.01}) & x_e\in\{0,1\} & \forall\ e\in E_{\text{adj}}^{AB}\\
 & x_e^{AB}\in\{0,1\} & \forall\ e\in E_{\text{id}}^{AB}\cup E_{\text{ext}}^{AB}\\
(\mathtt{D.02}) & 0\ \le y_i^{AB}\le i & \forall\ 1\le i\le|V^{AB}|\\
(\mathtt{D.03}) & z_i^{AB}\in\{0,1\} & \forall\ 1\le i\le|V^{AB}\setminus\mathcal{T}|\\
(\mathtt{D.04}) & r_v^{AB}\in\{0,1\} & \forall\ v\in V^{AB}\\
(\mathtt{D.05}) & t_e^{AB}\in\{0,1\}\in\{0,1\} & \forall\ e\in E^{AB}\\
(\mathtt{D.06}) & o_v\in\{0,1\} & \forall\ v\in V^{AB}\\
(\mathtt{D.07}) & s_C\in\{0,1\} & \forall\ C\in\mathcal{C}^{AB}\\
(\mathtt{D.07}) & a_A,a_B\in\mathbb{N} &
\end{array}$$

$\mathbf{f}$-derived $\{A, B\}$-resolved family assignment $\mathbf{f}'$.

Rather than minimizing the sum of objectives specified in Problem 2, the algorithm solves the inverse maximization problem. Note that n, *i.e.*, the number of non-telomeric markers that are shared between each pair of derived genomes A', B', is fixed and amounts to

$$\frac{1}{2}\sum_{\varepsilon \in A} \min\left(\mathbf{m}_A(\varepsilon), \mathbf{m}_B(\varepsilon)\right) \quad = \frac{1}{2}\sum_{\varepsilon \in B} \min\left(\mathbf{m}_A(\varepsilon), \mathbf{m}_B(\varepsilon)\right)$$

and thus requires no optimization. Similarly, each quadruple of telomeres[d] may increase the number of cycles by at most 1 while simultaneously increasing n' by 1 (cf. Eq. 1): the number of telomeres used in a solution does not impact the DCJ-indel distance. The objective of Algorithm 1 translates the linear combination of Problem 3 into weighted sums over binary variables z_i^{AB}, t_e^{AB}, s_C, and o_v that count indel-free cycles, transition edges, circular singletons, and telomeric extremities, respectively. Here, $\mathcal{C}^{AB}$ represents the set of all candidates of circular singletons.

The choice of adjacencies across all relational diagrams that make use of the same genome is synchronized by using the same variable for each of its adjacencies (cf. D.01). Conversely, indel and extremity edges depend on the particular pairwise comparison. Also, variables involved in identifying/counting indel-free cycles (cf. D.02, D.03), indels (cf. D.04), and transition edges (cf. D.05) are not synchronized, and therefore are optimized independently. Note the capped relational diagram does not permit components with only telomeric extremities. Thus allows us to omit telomeric extremities from the process of counting indel-free cycles.

As alluded to before, the set of extremities in a genome derived from a degenerate genome may vary, leading to different surfeits. More precisely, the set of telomeric extremities is mutable while, per definition, each derived genome shares the same set of non-telomeric extremities. To this end, our ILP makes use of additional variables that indicate the *presence* of an extremity in a derived genome that is part of the solution (cf. D.06). Components corresponding to circular singletons are counted by binary variables specified in D.07. Last, we also count the number of linear chromosomes in derived genomes of the solution (cf. D.08).

Valid solutions to Problem 2 for each pair of degenerate genomes $\{A, B\} \in E(\Gamma)$ are guaranteed by constraints C.01-09. The presence of each node associated with a non-telomeric extremity in capped relational diagram $R_\circ(A', B', \mathbf{f}')$ is ensured by setting the corresponding variable o_v for each v in V^{AB} to one. Constraints C.02 enforce that each connected component in $R_\circ(A', B', \mathbf{f}')$ represents an alternating cycle, where each adjacency edge is followed by either an extremity or indel edge. The next constraint, C.03, implements the definition of a family assignment which specifies that head and tail of marker in $\{A, B\}$-resolved assignment $\mathbf{f}'$ must belong to the same family. Here, we denote by *sibling* a pair of edges $\{g^\mathrm{t}, h^\mathrm{t}\}, \{g^\mathrm{h}, h^\mathrm{h}\} \subseteq E_\mathrm{adj}^{AB}$, such that $\{g^\mathrm{t}, g^\mathrm{h}\}, \{h^\mathrm{t}, h^\mathrm{h}\} \in \mathcal{M}$.

[d]Telomeres in a capped relational diagram appear in multiples of four

Variables y^{AB} label each cycle in relational diagram $R_\circ(A', B', \mathbf{f}')$ by a number (cf. C.04). This number will be zero, if the cycle contains one or more indel edges (cf. C.05). Otherwise, the number will correspond to the smallest index of any node in the cycle (cf. C.06). Constraints C.07 label runs of genome A as zero and runs of genome B as one. Constraint C.08 enforces that the indicator of a transition edge t_{uv}^{AB} corresponding to edge $\{u, v\} \in E^{AB}$ is set to one if (i) the edge is part of relational diagram $R_\circ(A', B', \mathbf{f}')$ and (ii) its incident nodes have unequal run labels.

Constraint C.09 sets the indicator s_C for a connected component C to one if C is a circular singleton. This entails the prerequisite that the set of all candidates of circular singletons $\mathcal{C}^{AB}$ must be known. The construction of this set is explained further below.

Constraints C.10 and C.11 are *optional* in the sense that they are not required to construct a valid solution to SPP-DCJ. However, these constraints reduce the search and/or solution space, which helps the ILP solver in reducing the computation time. Constraint C.10 limits the choice of transition edge between to neighboring indel runs to an adjacency edge of genome A. Last, constraints C.11 let the solver know that valid genomes contain an even number of telomeric extremities.

Appendix C. Simulation Details

Table 1. *Data set 1*: Total numbers of evolutionary events produced by ZOMBI run on tree of 10 extant species with parameter settings: *genome size*: 1000, *duplication*: 2, *dupl. extension*: 0.5, *loss*: 2, *loss extension*: 0.5, *inversion*: 2, *inversion extension*: 0.05, *transposition*: 2, *transposition extension*: 0.05, *origination*: 0.

Tree scale	**Dup. events**	**Dup. genes**	**Loss events**	**Lost genes**	**Inver-sions**	**Transpo-sitions**
1	4	10	8	14	4	4
2	9	16	13	25	16	8
3	5	15	17	35	13	12
4	26	47	28	60	18	15
5	29	51	35	72	18	28
6	33	80	35	70	27	26
7	32	69	27	57	32	30
8	50	89	34	63	47	34
9	40	90	48	102	48	45
10	55	107	52	117	55	44

Table 2. *Data set 2*: Total numbers of evolutionary events produced by ZOMBI run on tree of 10 extant species with parameter settings: *genome size*: 1000, *duplication*: 2, *dupl. extension*: 0.8, *loss*: 2, *loss extension*: 0.8, *inversion*: 2, *inversion extension*: 0.5, *transposition*: 2, *transposition extension*: 0.5, *origination*: 0.

Tree scale	**Dup. events**	**Dup. genes**	**Loss events**	**Lost genes**	**Inver-sions**	**Transpo-sitions**
1	3	5	4	4	6	3
2	4	5	9	12	7	10
3	15	18	17	22	11	22
4	21	29	17	24	17	21
5	24	33	30	37	22	18
6	35	48	39	44	23	37
7	52	67	26	30	40	42
8	39	46	40	51	37	39
9	45	60	45	59	53	38
10	43	57	47	63	53	54