BMC Bioinformatics

RESEARCH ARTICLE Open Access

Chromosome structures: reduction of certain problems with unequal gene content and gene paralogs to integer linear programming

Vassily Lyubetsky[1,2], Roman Gershgorin[1] and Konstantin Gorbunov[1*]

Abstract

Background: Chromosome structure is a very limited model of the genome including the information about its chromosomes such as their linear or circular organization, the order of genes on them, and the DNA strand encoding a gene. Gene lengths, nucleotide composition, and intergenic regions are ignored. Although highly incomplete, such structure can be used in many cases, e.g., to reconstruct phylogeny and evolutionary events, to identify gene synteny, regulatory elements and promoters (considering highly conserved elements), etc. Three problems are considered; all assume unequal gene content and the presence of gene paralogs. The distance problem is to determine the minimum number of operations required to transform one chromosome structure into another and the corresponding transformation itself including the identification of paralogs in two structures. We use the DCJ model which is one of the most studied combinatorial rearrangement models. Double-, sesqui-, and single-operations as well as deletion and insertion of a chromosome region are considered in the model; the single ones comprise cut and join. In the reconstruction problem, a phylogenetic tree with chromosome structures in the leaves is given. It is necessary to assign the structures to inner nodes of the tree to minimize the sum of distances between terminal structures of each edge and to identify the mutual paralogs in a fairly large set of structures. A linear algorithm is known for the distance problem without paralogs, while the presence of paralogs makes it NP-hard. If paralogs are allowed but the insertion and deletion operations are missing (and special constraints are imposed), the reduction of the distance problem to integer linear programming is known. Apparently, the reconstruction problem is NP-hard even in the absence of paralogs. The problem of contigs is to find the optimal arrangements for each given set of contigs, which also includes the mutual identification of paralogs.

Results: We proved that these problems can be reduced to integer linear programming formulations, which allows an algorithm to redefine the problems to implement a very special case of the integer linear programming tool. The results were tested on synthetic and biological samples.

Conclusions: Three well-known problems were reduced to a very special case of integer linear programming, which is a new method of their solutions. Integer linear programming is clearly among the main computational methods and, as generally accepted, is fast on average; in particular, computation systems specifically targeted at it are available. The challenges are to reduce the size of the corresponding integer linear programming formulations and to incorporate a more detailed biological concept in our model of the reconstruction.

(Continued on next page)

* Correspondence: gorbunov@iitp.ru
[1]Institute for Information Transmission Problems of the Russian Academy of Sciences (Kharkevich Institute), Bolshoy Karetny per. 19, build.1, Moscow 127051, Russia
Full list of author information is available at the end of the article

(Continued from previous page)

Keywords: Chromosome structure, Chromosomal rearrangement, Ancestral genome, Evolution along the tree, Reconstruction of ancestral genomes, Transformation of chromosome structures, Parsimony principle, Integer linear programming, Efficient algorithms

Background

Introduction

Chromosome structure is a large-scale view on the genome; it can be considered as a very limited model of the genome taking into account only the mutual arrangement of genes (ignoring their length and nucleotide composition) on both DNA strands as well as the chromosome type (linear or circular), including gene names (identifiers) [1, 2]. Instead of the term "chromosome structure", the terms "genome" or even "genotype" are used sometimes [3–5].We prefer the term "chromosome structure", [6], to outline the distinction between the genome as a biological notion and the considered model. Below we consider the DCJ model widely used in studies of this kind, e.g., [3, 7]. The model includes *standard* DCJ operations: *double-*, *sesqui-*, and *single*-operations; the last ones comprise cut and join operations. They were proposed in [7] and later studied in dozens of publications, for example, in [8–10] where a detailed review of the results and further references are given. The biological mechanisms of the operations are described, e.g., in ([10], chapter 5). Two structures have *equal gene content* if they have no paralogs and contain the same set of names. In the case of unequal gene content, structures can have paralogs, and *supplementary* operations are considered: deletion and insertion of a chromosome connected region [4, 11]; these operations were actively studied, e.g., in [4, 8, 12] where further references are given. The popularity of this model stems from the simplicity and elegance of the underlying mathematical constructs as well as from the ability to model many types of genomic rearrangements. Although highly incomplete, such model can be used in many cases, e.g., to reconstruct phylogeny and evolutionary events, to identify gene synteny, regulatory elements and promoters (considering highly conserved elements), etc.; e.g., ref. to [10, 13]. Remind that paralogs are duplicated genes in the same genome, and the problem of their identification in different genomes is hard and important. The role of the structures with paralogs were described in detail, e.g., in [5, 14, 15].

In the context of chromosome structures, three well-known problems are considered. They are formally described in sections 1.3 and 4.1; here their concepts are introduced together with the corresponding references. The *distance* problem determines the distance between two chromosome structures, i.e., the minimum number of operations required to transform one chromosome structure into another, and the corresponding minimum transformation. Paralogs should be identified so that the resulting structures considered as structures without paralogs have the minimum distance. It is easy to prove that the allowance for paralogs makes the distance problem NP-hard.

A linear-time algorithm was proposed for the distance problem in the absence of paralogs for both equal [3] and unequal [4, 16] gene content. This problem is reduced to integer linear programming formulation (ILP) in [5, 14, 15], where its definition was considerably simplified; specifically, balanced gene content in [5], structure reduction to equal gene content by elimination of unwanted regions with paralogs in [14], and ignoring paralogous genes in [15]. More precisely, in [15] such structures can have paralogs, but after the identification of paralogs, the genes present in one out of both structures (which is a real-life situation) are eliminated and not considered later, which does not seem to be justified in any way. Balanced gene content means the same set of names but with possible paralogs.

In the *reconstruction* problem a phylogenetic tree with chromosome structures in the leaves is given. It is required to assign structures to inner nodes of the tree to minimize the *total distance* between terminal structures of each edge. Thus it can be called a small phylogeny problem; the term "reconstruction" is widely used, e.g., in [13]. As previously, unequal gene content and paralogs in all nodes are allowed. Paralogs should be identified such that the total distance for all resulting structures without paralogs is minimum. It is easy to prove that this problem is NP-hard even in the absence of paralogs. Only heuristic algorithms are known for the problem, among which the algorithms in [6, 13, 17] should be noted. These as well as other publications mentioned above present numerous relevant references; it allows us to avoid detailed historical review here due to publication size limitations.

Thus, *exact* algorithms presented here solve two above problems by reducing them to ILPs. Let us recall that an algorithm is called exact if it is mathematically proved that it always results in a global minimum (hereafter, *minimum point*) of the minimized function involved in the problem statement. The significance of this reduction stems from the appearance of fast methods solving ILP tasks in recent 20 years (e.g., [18, 19]). Note, many combinatorial problems (possibly including ILP) have low complexity on average but can be pretty hard in some special cases. For example, hard inputs are rare for the simplex algorithm for linear programming [20, 21].

Another example, a simple algorithm for solving almost all instances of the famous set partition problem, that is NP-hard, is also proposed in [22].

Finally, the computation of the distance between two chromosome structures with paralogs was reduced to ILP for circular chromosomes in [17]. Here, we define such reduction for arbitrary structures with unequal gene content and paralogs as well as for the reconstruction of such structures along the phylogenetic tree. The computation of a sequence of operations (for the minimum transformation) was also considered previously, e.g., in [16, 17, 23, 24]. An algorithm with a linear complexity solving the distance problem without paralogs and with preset weights of operations (which minimizes the total weight of sequence of operations) that is not based on reduction to ILP was obtained in [23, 24] as well as in our study prepared for publication.

The statement of the *contig* problem is given separately in section 4.1 after the first two problems are clarified.

Definitions of notions

The definitions relevant to the distance problem can be found in publications in different modifications or the problem can have no strict definition at all. Accordingly, we will briefly review the relevant definitions.

Chromosome structure is defined as a directed graph composed of non-intersecting paths (of nonzero length) and cycles (including loops). Loops correspond to circular chromosomes comprising a single gene. Each graph edge represents a gene with no account of its length, and the edge is given the *name* of this gene. The edge direction shows the gene transcription direction. Two extremities of neighboring genes are combined (or *merged*) into a graph node.

In this context, an edge with an assigned name is referred to as a *gene*, while a path or cycle is referred to as a *chromosome*. Repeated names can occur in a structure, they correspond to paralogous genes distinguished by the index *j*: paralogous genes with name *k* get *full names* of the form *k.j*. Full names are unique; a structure with full names only has no paralogs.

Let *adjacency* denote a pair of merged gene extremities, a node of degree 2 in a structure. Here, the extremity is a 5′- or 3′-end of a gene considering that the term "end" is linked to ends of graph edges.

Hereafter, a and b denote two chromosome structures; a is meant to be transformed into b. A gene present in both a and b is referred to as a *common* gene; a gene present in only one structure a or b, a *special* gene; accordingly, there are a- and b-special genes. In the case of unequal gene content, two *supplementary* operations can be applied to a structure in addition to the *standard* ones mentioned above: *deletion* and *insertion*. The former is the removal of a connected region of a-special genes together with its extremities. Such region can be removed from a circular or linear chromosome (cycle or path); the whole chromosome can be removed as well. If the removed region has neighboring genes on both sides, their extremities are merged. The latter operation, inversely, inserts a connected region of b-special genes; in this case, a chromosome is cut in a node and pairs of the new free ends are merged. More precisely, the region can be inserted into or to a boundary of a chromosome or form a new circular or linear chromosome (cycle or path).

Let us recall the notion of common graph $a + b$ for two structures a and b given in [17] for *unequal gene content without paralogs*. For equal gene content, such graph was first defined in [25] as the breakpoint graph. For unequal gene content without paralogs, a similar graph was first defined in [12] under the same name. Following [12, 25], $a + b$ will be referred as the breakpoint graph here. Thus, it is an undirected graph without loops whose nodes are *conventional*, i.e., the extremities of common genes with their names (e.g., 3_h or 3_t), and *special*, i.e., any maximal by inclusion connected regions of a-special or b-special genes. The latter are referred to as *blocks*. A block belongs to one of the structures a or b, and the special node corresponding to it is called an a- or a b-node, and a set (more precisely, a sequence) of gene names corresponding the block is assigned to it; the latter serves as the special node name. The breakpoint graph edges are as follows. A *conventional* edge connects two conventional nodes if the extremities corresponding to them are merged in a or b; a *special* edge connects a conventional node to a special one if the extremity corresponding to a conventional node is merged in a or b with the boundary of the block corresponding to the special node. Double conventional edges are also possible here. A *loop* in $a + b$ corresponds to a cycle that is a block; stated differently, a special node of this block is connected to itself. A special edge incident to a special node of degree 1 is referred to as a *hanging* edge.

In any case, the breakpoint graph is undirected and includes non-intersecting connected components: paths including isolated nodes and cycles including loops. Non-hanging special edges occur in it in pairs as edges incident to the same special node; it is convenient to consider such pairs as a double edge; subject to this provision, the alternation of a- and b-edges is preserved. Accordingly, the component *size* is the quantity of conventional edges in it plus half the quantity of special non-hanging edges. The size of isolated conventional nodes and loops equals 0, while that for isolated special nodes equals −1.

A breakpoint graph is considered *final* (or of the *final form*) if all its components are conventional nodes, or cycles without special edges of size (or length) 2, one edge from a and the other from b. If the a, b, c marks are neglected, the *final* graph $a + b$ has the form $c + c$ for a certain structure c.

Four *standard* operations are allowed on a breakpoint graph, they correspond to the standard operations on a structure. Let us describe them in brief (for details see [16, 17, 23]). *Double-cut-and-paste* is the removal of two edges with the same label (e.g., a) and joining four resulting free ends in a new way by two edges with the same label. If this gives rise to an edge with two special nodes (both of which pertaining to either a or b), it is replaced with one special node to which the concatenated sequence of the sequences of two initial special nodes is assigned (Fig. 1a). Hereafter, for the breakpoint graph, an edge removal indicates the removal of only its internal part. *Sesqui-cut-and-paste* is the removal of an edge and joining in a new way with an edge with the same label of one of its free ends with a conventional free node non-incident to an edge with this label or with a special node of degree not exceeding 1 with the same label (which can be followed by a similar replacement of two special nodes). *Join* is inserting an edge (say with the label a) between free nodes, where each node is either conventional non-incident to an edge labeled a or special of degree not exceeding 1 with the same label (which can also be followed by the subsequent replacement of the special nodes if any). *Cut* is the removal of any edge.

In addition, only one *supplementary* operation on breakpoint graphs is allowed (it corresponds to the deletion operation on a structure): the *removal* of a special node (i.e., a block). Specifically, if this node s has the degree 2, it is removed and the edges incident to it are combined into one edge labeled as the neighbors of node s (Fig. 1b); if the node has degree 1, it is removed together with the edge incident to it (the conventional node is preserved); and if the node has degree 0 or has a loop, the isolated node and the loop are removed.

In [16, 23], we have reduced the problem of structure a transformation into structure b using the above six operations with allowed unequal gene content (without paralogs) to the problem of their breakpoint graph $a + b$ transformation into the final form using these five operations. For equal gene contents, such transformation was proposed in [25]; for unequal gene contents without paralogs, this idea was implemented in [12].

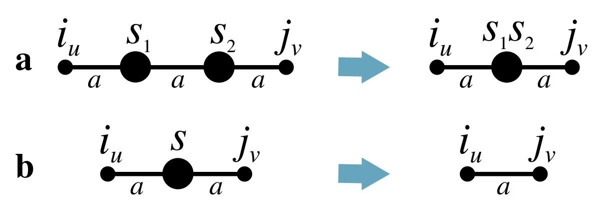

Fig. 1 **a** Concatenation of any two neighboring special nodes s_1 and s_2(both from **a**). The nodes s_1 and s_2 are replaced with one special node s_1s_2 (the concatenated sequence of the sequences of two initial special nodes). Similarly for (**b**). **b** Removal of a special node. Large point is an a-special node s and the resulting combined edge is marked (**a**). Similarly for (**b**)

Statements of two problems

Hereafter, the structures can always have *unequal gene content* and *include paralogs*. The identification of paralogs (e.g., paralogs of a gene with the name k) means that they are given *unique* new names k.1, k.2, …. This form of paralog identification will be referred to as *numbering* of paralogs, and new names of the form $k.j$ will be referred to as *full names* (of paralogs of gene k). The numbering makes it possible to establish a partial bijection between two sets of paralogs of gene k that belong to structures a and b, respectively. It is only partial since paralogs can disappear and emerge in the course of transformation (a to b) or evolution. If a gene has no paralogs, we can take that it has no index j or, better, assign it the same fixed index, e.g., 1.

It is important that the definitions of the common and special genes depend on the numbering of all paralogs of all genes, i.e., on the index j. Different paralog numberings in structures a and b can substantially change the breakpoint graph and its transformation to the final form.

At first, we define two problems to solve; the former is the *distance* problem. We are given two structures a and b with different gene content and paralogs. It is required to number paralogs of all genes in the structures to minimize the distance between the resulting structures without paralogs as well as to calculate this distance and to find the minimum sequence of operations.

The latter is the *reconstruction* problem. We are given a root and, generally speaking, non-binary tree T. Structures a_1, …, a_n with different gene content and paralogs are defined in the tree leaves (their quantity is n). It is required to number all paralogs in the leaves and to identify mutually coherent numbered structures (in the inner nodes) with the minimum *total distance* calculated as the sum of distances for all edges of the tree, as well as to calculate the total distance. Only the names k present in the leaves are allowed in the inner nodes, and the upper limit $s(k)$ of the index j is fixed for each k in these a priori unknown structures. Clearly, the appearance of new names in the inner nodes will not decrease the total distance. The distance on each edge is calculated as in the former problem. *Arrangement* is the assignment of a numbered structure to each node of the tree so that the leaves are assigned the initial predefined structures. Given the arrangement, the node and its structure are not distinguished. The minimum point of the specified function of the total distance in the latter problem is called the *minimum arrangement*, which is wanted; if there are several minimum points, we consider any one of them. Let $F^*(A)$ be the total distance at any arrangement A.

Section 2 presents an exact algorithm to solve the distance problem through its reduction to ILP. Section 3 presents an exact algorithm to solve the reconstruction problem by the same reduction *if* there is a minimum point (a minimum arrangement) for objective function F', such that at the point, for any tree edge and for any circular chromosome at one of the edge ends, there is a gene from this chromosome present at the other end of the edge. This *condition* is applicable only to the problem of reconstruction and is marked by (*). Without this condition, our algorithm gives only an approximation F' to the minimum value F^*; the difference between F' and F^* is majorized.

The more general statement of the distance problem, which was considered, in particular, in [17, 23, 24], assigned each operation a weight, a strictly positive rational number, and the sequence transforming a into b with the minimum *total weight* of operations is sought. This generalization of the reconstruction problem is considered in [23, 24] on the basis of a direct algorithm and also can be reduced to ILP in a similar way as here. The latter more general consideration is omitted here for brevity. We have demonstrated that the problem of finding such total weight and the corresponding sequence of operations in this setup of the problem is reduced to the problem of breakpoint graph transformation to the final form if the weights of all standard operations are equal or obeyed some other constraints [16, 23].

The problem of *contigs* is to find the optimal concatenations of each given set of contigs providing their unequal gene content and identification of paralogs (see Section 4.1).

Method and results

Solution of the distance problem

Linear minimized function and its linear constraints

Below a reduction algorithm for the distance problem to integer linear programming (ILP) is described. We formulate the objective function *F*, variables and constraints of the ILP task, and also prove the key equality (1) in the Theorem 1.

Let a and b are given chromosome structures with unequal gene content and paralogs. Let us do arbitrarily numberings for gene paralogs as well as for genes without paralogs; the resulting numbered structures will be denoted as a' and b'. The numberings are called *initial*. We will deal only with numbered structures below. Let *adjacency* denote a pair of merged gene extremities that is a node of degree 2 in a' or b'.

Let us introduce Boolean variables z_{kij} to indicate whether genes $k.i$ in a' and $k.j$ in b' correspond to each other in terms of a partial bijection of paralogs in a' and b'; thus $z_{kij} = 1$ if i corresponds to j, otherwise $z_{kij} = 0$. Specifically, $\sum_i z_{kij} \le 1$ for any fixed indexes k and j; and analogously for the sum over index j. Based on biological considerations, lower bounds can be set on this sum, e.g., $1 \le \sum_{i,j} z_{kij}$ for certain values of k.

A gene is called *common* if it becomes common after paralogs in b' are renumbered according to the z_{kij} values. Specifically, if $z_{kij} = 1$, the gene $k.j$ in b' is renamed to $k.i$ and becomes synonymous to $k.i$ in a', after which the genes out of the z-bijection are arbitrarily numbered to keep the structures numbered. Similarly, a gene is called *special* if it becomes special after renumbering. The structures resulting from such renumbering in b' will be referred to as $a'(z)$ and $b'(z)$. A circular chromosome composed of only special genes will be called *special*. Circular chromosome will be referred to as *1-circular* if it composed of a single gene; otherwise it is *m-circular*. For each circular chromosome d in a', let us define $o(d,a) = \left(\sum_{k.i\in d,k.j\in b'} z_{kij}\right)/n_d$ where n_d is the quantity of genes in d. For a linear chromosome d, we set $o(d) = 1$; $0 \le o(d) \le 1$. It holds that d is special if and only if $o(d,a) = 0$. The value of $o(d,a)$ indicates the proportion of genes in d that are in z-bijection with genes in b'. The proportion $o(d,b)$ for a chromosome d in b' is defined similarly. References to a or b are usually omitted.

Let us equalize the gene contents in $a'(z)$ and $b'(z)$ just by adding to $a'(z)$ special $b'(z)$-genes except the genes from special $b'(z)$-chromosomes; a similar addition is made to $b'(z)$. All added genes are combined into circular chromosomes, some from $a'(z)$ and some from $b'(z)$. The resulting chromosomes as well as their genes and gene adjacencies will be referred to as *new*. New adjacencies are defined by a new variable t, which is formally described below. Thus obtained structures referred to as $a^{-}(z,t)$ and $b^{-}(z,t)$ released from special chromosomes (if any) are denoted as $a''(z,t)$ and $b''(z,t)$. Let us introduce the breakpoint graph

$$G^{'}(z,t) = a^{''}(z,t) + b^{''}(z,t)$$

It is proved as in [12] that the distance between $a^{-}(z,t)$ and $b^{-}(z,t)$ equals $\Phi(z,t)$ for any z and t. It follows that, for a fixed z, the minimum by t distance between $a^{-}(z,t)$ and $b^{-}(z,t)$ equals $\min_t \Phi(z,t)$; for any z, $t_0 = t_0(z)$ defines the value of t corresponding to this minimum. Here

$$\Phi(z,t) = (C_0 + n + s_a + s_b) - \mathrm{C}_1 - 0.5C_2,$$

where C_0 is the total number of special chromosomes in $a'(z)$ and $b'(z)$, C_1 is the number of cycles in G', C_2 is the number of even paths in G', n is the number of common genes in $a'(z)$ and $b'(z)$ counted once, and s_a,

s_b are the quantities of new genes in $a^{-}(z,t)$ and $b^{-}(z,t)$. *Even* (odd) path is a path of even (odd) length. Notice that natural constraints are imposed on z and t in the definition of Φ. Following [12], it is easy to verify that the distance between $a^{-}(z,t_0)$ and $b^{-}(z,t_0)$ equals the distance between $a'(z)$ and $b'(z)$ for any z. There is no z variable in [12] since paralogs are not considered there; the t variable is not used either. Thus, solving the distance problem requires finding $\min_z \min_t \Phi(z,t)$. By definition, a new adjacency corresponds to the *new edge* in $G'(z)$; the remaining edges in G' are called *old*.

Now let us define the variable t which describes new adjacencies. For each pair $s = (g,g')$ of different gene extremities in a', we define a Boolean variable t_{bs} to indicate whether g and g' form a new adjacency in $b''(z,t)$. Specifically, $t_{bs} \le 1 - \sum_j z_{kij}$, $t_{bs} \le n_g \cdot o(d_g)$, $\sum_{g'} t_{bgg'} \le 1$, and $\sum_{g'} t_{bgg'} \ge o\left(d_g\right) - \sum_j z_{kij}$, where $k.i$ is a gene with the extremity g, d_g is the chromosome containing $k.i$, n_g is quantity of genes in d_g. Similar variable t_{as} and constraints are defined for extremities in b'. Often we will omit the indexes a and b near t.

Items 1–3 below describe the summands of the function Φ by means of equivalent ILP formulation (of minimization). To this end, let us sequentially describe the summands C_1, C_2, and $C_0 + n + s_a + s_b$ in Φ. Thus, the objective function will be equal to

$$F = \left(\sum_d n_d + \sum_d (1 - n_d) o_d - \sum_{k,i,j} z_{kij}\right) - \sum_s p_s - 0.5\left(\sum_g r_g - \sum_g l_g\right)$$

where d runs over all chromosomes in a' or b' and n_d is the quantity of genes in chromosome d. The summand $\sum_d n_d$ is a constant and has no effect on the minimum value. The variables o_d, p_s, r_p, l_p and their linear constraints will be defined in items 1–3 below. The critical point is the equality

$$\min_{z,t} \Phi(z,t) = \min F(o,z,p,r,l). \qquad (1)$$

1) Here we use the counting cycles idea from [5]. Let us describe the quantity C_1 of cycles in the breakpoint graph G'. Let us do numbering of all adjacencies (g,g') in a' and b' starting from one; and m_s is the number of an adjacency s. Let us for each s introduce an integer (non-Boolean) variable u_s with the constraint $0 \le u_s \le m_s$. We require that $u_s = 0$ for all adjacencies s in a' from special chromosomes d in $a'(z)$; with regard to other constraints, it is expressed as the inequality $u_s \le m_s \sum_{k,i \in d} \sum_j z_{kij}$ for any circular chromosome d. And symmetrically for adjacencies in b'.

Two extremities of two genes are defined to be of the *same type* if both of them are either 5′-ends or 3′-ends and belong to paralogs in different structures. We require that $u_s = 0$ for any adjacency s in a' such that one of its extremities belongs to a common gene and is a boundary of a path in G'. Specifically, let g be an extremity of gene $k.\ i \in a'$ adjacent to any extremity in s. For each gene $k.j$ in b' with an extremity of the same type as g that is a boundary of a path in b', the constraint $u_s \le m_s(1 - z_{kij})$ is imposed. The constraints are symmetrical for b'.

Further, we require that $u_s = 0$ for any adjacency s in a' such that one of its extremities belongs to a special a-gene and is not a boundary of a path through the end of a terminal new edge of a path in G'. Specifically, for each extremity g_1 in a' that is a boundary of a path in a', we impose that $u_s \le m_s(1 - t_{g1g})$ where s includes g. The constraints are symmetrical for b'.

We require that u_s is constant at all edges in a cycle or path in G'. Specifically, for each pair of adjacencies $s1 = (g,g_1)$ and $s2 = (g',g_2)$ in a' and b', respectively, with g and g' being of the same type, we impose

$$u_{s1} \le u_{s2} + m_{s1}(1 - z_{kjj'}),\ u_{s2} \le u_{s1} + m_{s2}(1 - z_{kjj'})$$

where $k.j$ and $k.j'$ are genes with the extremities g and g'. These two constraints ensure that $u_{s1} = u_{s2}$ for two neighboring edges $s1$ and $s2$ in G' that are both old edges. For each pair of different adjacencies $s1 = (g_1,g_2)$ and $s2 = (g_3,g_4)$ of extremities both in a' or b', we impose that $u_{s1} \le u_{s2} + m_{s1}(1 - t_{g2g3})$, $u_{s2} \le u_{s1} + m_{s2}(1 - t_{g2g3})$. These constraints ensure that $u_{s1} = u_{s2}$ for two edges in G' that are both old edges and spaced by exactly one new edge.

For each adjacency s, we define the Boolean variable p_s to indicate whether u_s is equal to its upper bound m_s at the minimum point of the function F. Specifically, $p_s \cdot m_s \le u_s$. Indeed, if $u_s < m_s$, then $p_s = 0$. Otherwise, p_s can take any of two values, but since variables p_s are summands of F with negative coefficients, we have $p_s = 1$.

Since u_s has a constant value on all edges in a cycle and all upper bounds are unequal, there is exactly one edge at the minimum point whose u_s equals its upper bound. Indeed, exactly one of p_s equals 1 in a cycle at the minimum point. In a path, the constraints imply that $u_s = 0$ so that neither of them can reach the maximum; hence, $p_s = 0$ in a path. Considering that any cycle contains at least one old edge, the quantity of variables u_s that reaches its maximum is equal to the quantity of cycles, thus $C_1 = \sum_s p_s$ at the minimum point of F.

2) Let us describe the quantity C_2 of *even* paths in the graph G'. Let us introduce three-valued (0, 1 or −1) integer variables r_{ag1} and r_{bg2} for any gene extremity g_1 and g_2 in a' and b' such that, at the minimum

point of *F*, the *sum* of the variables (if g_1 and g_2 are in z-bijection and have the same type, r_{bg2} is omitted) by the nodes of a path or a cycle in G' equals 1 if it is an even path; otherwise it equals 0. At the minimum point of *F*, it follows from the constraint that the values of r at adjacent nodes in G' are not equal to 1 and 1 or 0 and 1. Specifically, for each adjacency (g_1,g_2) in a' or b', we impose that $r_{ag1} + r_{ag2} \le 0$ or $r_{bg1} + r_{bg2} \le 0$, respectively. For each pair of different extremities g_1 and g_2 from a' which do not form an adjacency, we impose that $r_{ag1} + r_{ag2} \le 2(1 - t_{ag1g2})$. Similar constraints are imposed for b'. For each pair (g,g') of extremities of the same type from a' and b', respectively, we impose that $-2\left(1-z_{kjj'}\right)\le r_g-r_{g'}\le 2\left(1-z_{kjj'}\right)$, where $k.j$ and $k.j'$ are genes with extremities g and g'. These constraints ensure that $r_g + r_{g'} \le 0$ if (g,g') is an edge in G'; also if g and g' are in z-bijection, then $r_{ag} = r_{bg'}$. Considering that the variables r_g are summands of *F* with some negative coefficients, they equal 1 at the minimum point at isolated nodes in G'. The lengths of cycles in G' are even, and the values of r_g in their nodes either alternate between 1 and −1 or constantly equal 0. Therefore, the above sum along a cycle equals 0. The r_g values alternate on non-zero even paths being equal to 1 at the path boundaries; accordingly, the sum along an even path equals 1. On an odd path, such alternation can be interrupted by zero values, but again the sum along its nodes equals 0. Hence, it follows that the sum indicates each even path. For a special chromosome d, $\sum_{g\in d} r_g = 0$ at the point of minimum of *F* since this sum is clearly not greater than 0.

Let us define the sum described in the beginning of item 2. For each extremity g of a gene in a', we define an integer variable l_g, which equals r_{ag} if g is an extremity of a common gene, or equals 0 otherwise. This is provided by the constraints $-\sum_j z_{kij}\le l_g\le \sum_j z_{kij}$, $l_g\le r_{ag}+2\left(1-\sum_j z_{kij}\right)$, $r_{ag}\le l_g+2\left(1-\sum_j z_{kij}\right)$, where $k.i$ is a gene with extremity g. Thus, the node g in G', an extremity of a common gene, corresponds to three variables r_{ag}, r_{bg}, and l_g, which take equal values. This allows us to cancel the summands r_{ag} and $-l_g$ when summing up all r_{ag}, r_{bg}, and $-l_g$. The node g, an extremity of a special gene in $a'(z)$, corresponds to two variables r_{ag} and l_{ag}, the latter equals 0. The node g, an extremity of a special gene in $b'(z)$, corresponds to one variable r_{bg}. Therefore, $C_2=\sum_g r_g-\sum_g l_g$ in a minimum point of *F*.

3) Let us describe the summand $C_0 + n + s_a + s_b$. For each chromosome d in a' or b', we define a Boolean variable o_d to indicate whether this chromosome is special m-circular at the minimum point of *F*. Specifically, if d is m-circular then $o_d \le 1 - o(d)$; if d is a 1-circular or a linear chromosome, then $o_d = 0$. Indeed, $o_d = 0$ follows from the above constraint if d is not special or is special and 1-circular. For a special m-circular chromosome $o_d = 1$ at the minimum point of *F* considering that variables o_d are summands of *F* with negative coefficients.

Let us show that in a minimum point of *F* we have

$$C_0+n+s_a+s_b=\sum_d n_d+\sum_d(1-n_d)o_d-\sum_{k,i,j} z_{kij},$$

where d runs over all chromosomes in the first sum and over all m-circular chromosomes in the second sum, and n_d is the quantity of genes in d. The number n is equal to the sum of all z_{kij} values, while the numbers s_a and s_b are equal by the definition as follows: $s_a = n_b - n$ and $s_b = n_a - n$, where n_a and n_b are quantities of genes in structures $a'(z)$ and $b'(z)$, respectively, not in special chromosomes. Thus, $n + s_a + s_b = n_a + n_b - n$. Considering that $C_0=\sum_d o_d+U$, $n=\sum_{kij} z_{kij}$, and $n_a+n_b=\sum_d n_d(1-o_d)-U$, where U is the quantity of 1-circular chromosomes, the desired equality is readily derived from the previous equality.

Theorem 1 For given a and b, the minimum paralog numbering and minimum value of the distance are defined by the minimum point of *F*.

Proof Let the function *F* reaches the minimum at the point x_0. It follows from items 1–3 that the function $\Phi(z, t)$ calculated at the point $y_0 = (z_0,t_0)$, which is a part of x_0 coordinates, equals $F(x_0)$. Such y_0 is the minimum for $\Phi(z, t)$. Indeed, if there is (z,t), for which the value of $\Phi(z, t)$ is strictly lower, then (z,t) can be extended to the point where *F* is equal to Φ, which is impossible. The extension is as follow. The point (z,t) together with given a' and b' uniquely define G'; p, r, l are defined by G'; and o_d is defined by $a'(z)$ and $b'(z)$. □.

Clearly, the number of variables and constraints in it quadratically depends on the data size of the initial problem.

Note 1 After solving the ILP task, one can use (as in [16]) the obtained z and the structures $a'(z)$ and $b'(z)$ to find the minimum sequence of operations transforming $a'(z)$ into $b'(z)$.

Examples for the distance problem based on synthetic data

Example 1

Let the structure a include three circular chromosomes with unidirectional genes: (1, 3); (1, 2, 2); (3, 5, 2, 4) and the structure b also include three circular chromosomes: (4, 2); (1, 2, 1); (4, 5, 5, 3) with unidirectional genes. Let us introduce the initial numbering; for a', it is (1.1, 3.1); (1.2, 2.1, 2.2); (3.2, 5.1, 2.3, 4.1); for b', it is (4.1, 2.1); (1.1, 2.2, 1.2); (4.2, 5.1, 5.2, 3.1). The ILP program of the Pulp python package returned the following solution: the number of operations transforming a' into b' equals 4. At the minimum point, the paralogs in b' are renumbered as follows: 1.1 to 1.2, 1.2 to 1.1, 2.1 to 2.3, 2.2 to 2.1, 3.1 to 3.2, 5.1 to 5.2, 5.2 to 5.1. The program execution time was about 1.5 h.

Example 2

We are given two structures with the following arrangement of genes on the chromosomes; a: (1, 2, −3, 4, 5, 6), (3), [10], [−7, 8, 9] and b: (1), (2), (9), (4, 6, −3, 5), [8], [−7, 10, 3]. Here minus sign indicates the complementary strand, while round and square brackets indicate circular and linear chromosomes, respectively. The initial numberings are as follows; a', the gene 3 is 3.1 and 3.2 in the large and small cycles, respectively; b', the gene 3 is 3.1 and 3.2 in the path and cycle, respectively. The ILP program of the Pulp python package returned the following solution: the number of operations transforming a' into b' equals 7. At the minimum point the paralogs in b' are renumbered as follows: 3.1 to 3.2, 3.2 to 3.1. The program execution time was about 3 h.

Solution of the reconstruction problem

Below a reduction of the algorithm for the reconstruction problem to integer linear programming (ILP) is described. We formulate the objective function F', variables and constraints of the ILP task, while the Theorem 2 proves that ILP can solve the problem. Let T be a fixed rooted possibly non-binary tree. Recall that *leaf* edge link to a tree leaf and *inner* edge means a non-leaf tree edge. T-Edge and G''-edge emphasize that this edge belongs to T and G'', respectively, but not to any structure. The structure in a node x is usually denoted by x; in this sense we do not distinguish a node and its structure.

Linear minimized function and its linear constraints

The argumentation is largely the same as in the distance problem fully described in Section 2 above, and it will not be reproduced in detail here. The specialties distinguishing the solution of the reconstruction problem from that of the distance one will be emphasized. Hereafter, a and b are nodes and, at the same time, structures in the beginning and end of a T-edge, respectively; an edge is often designated as $e = (a,b)$. Let us fix the initial paralog numberings in all given structures assigned to the leaves; they are called *initial*. For a leaf b, the given initially numbered structure is designated as b', while any numbered structure is designated as u', a', and likewise. Let M denote a set of all full names $k.i$, where $1 \leq i \leq s(k)$. Recall that circular chromosomes composed solely of special genes are called *special*.

We define the variable z_{ukij} for each leaf u and each gene $k.i$ from u' and $k.j$ from M; it equals 1 if $k.i$ is renamed to $k.j$; otherwise $z_{ukij} = 0$. The existence and uniqueness of $k.j$ is ensured by the following constraints:

$$\text{for fixed } k \text{ and } i, \sum_j z_{ukij} = 1; \text{for fixed } k \text{ and } j, \sum_i z_{ukij} \leq 1.$$

The index u is usually omitted.

We define the variable $y_{vk.i}$ for each inner node v and each gene $k.i$ from M; it equals 1 if $k.i$ is missing from v; otherwise it equals 0. For each inner node v and each pair (g,g') of different extremities from M, we define the variable $x_{vgg'}$; it equals 1 if g and g' are present and merged in the node v; otherwise it equals 0. The variables $x_{vgg'}$ are not specified in leaves since their values are fixed there. Specifically, $\sum_{g' \neq g} x_{vgg'} \leq 1 - y_{vk.i}$ implies that any extremity g of any gene $k.\ i \in M$ missing in v is not merged, where g' runs over all extremities from M; and the constraint implies that $\sum_{g'} x_{vgg'} \leq 1$ for any fixed v and g. The index v is usually omitted.

In order to avoid degenerate scenarios with empty ancestral structures, we lay the condition that if a gene is absent from an inner node v, it is absent from at least a half of its direct descendants. Specifically, the following constraint is imposed on each name $k.j$ from M:

$$y_{vk.j} \leq 1.5 - \frac{1}{n_v}\left[\sum_{v'}\left(1 - y_{v'k.j}\right) + \sum_{v'}\sum_i z_{v'kij}\right],$$

where n_v is the total number of direct descendants v' of v; in the first and second sums, v' runs over the inner nodes and leaves, respectively. This constraint can be simplified for a binary tree:

$$y_{vk.j} \leq w\left(v'\right) + w\left(v''\right),$$

where v' and v'' are direct descendants of the node v, and $w(v^\alpha) = y_{v^\alpha k.j}$ if v^α is not a leaf or $w(v^\alpha) = 1 - \sum_i z_{v^\alpha kij}$ otherwise.

As in Section 2 we equalize the gene contents in $a'(z)$ and $b'(z)$ where the variable z defines identical bijections for inner edges. But now we add to $a'(z)$ all special $b'(z)$ genes; respectively, to $b'(z)$ all special $a'(z)$ genes; we denote obtained structures $a^+(z,t)$ and $b^+(z,t)$. Thus, special chromosomes are not removed. Therefore the breakpoint graph G'' of $a^+(z,t)$ and $b^+(z,t)$ may be different from the graph G' defined in Section 2.

For each edge $e = (a,b)$ and each pair $s = (g,g')$ of different gene extremities from M we define the Boolean variable t_{ebs} to make sure that if $t_{ebs} = 1$, then g and g' form a new adjacency in $b^+(z,t)$. Similar variable t_{eas} is introduced for a, but if b is a leaf, t_{eas} is defined only for the extremities present in b'. The index e can be omitted. Let $k.j$ be a gene with extremity g. For a leaf edge e, the constraints are as follows:

$$t_{ebs}{\le}1{-}y_{ak.j}\,,\,t_{ebs}{\le}1{-}\sum_{i}z_{bkij},\sum_{g1\in M}t_{ebgg1}{\le}1,\sum_{g1\in b^{'}}t_{eagg1}{\le}1,$$
$$\sum_{g1\in M}t_{ebgg1}{\ge}1{-}y_{ak.j}{-}\sum_{i}z_{bkij};\ t_{eas}{\le}1$$
$$+y_{ak.\alpha}{-}z_{bkj\alpha},\sum_{g1\in b^{'}}t_{eagg1}{\ge}y_{ak.\alpha}+z_{bkj\alpha}{-}1.$$

Actually, the last two constraints assume the systems of inequalities for each value of α, such that $1 \le \alpha \le s(k)$.

For an inner edge e, we impose that:

$$t_{ebs}{\le}1{-}y_{ak.j},t_{ebs}{\le}y_{bk.j},\sum_{g1\in M}t_{ebgg1}{\le}1,\sum_{g1\in M}t_{bgg1}{\ge}y_{bk.j}{-}y_{ak.j}.$$

Similar constraints are imposed for t_{eas}.

For any leaf edge $e \in T$, let $|M|$ be the quantity of elements in M, and c_e be $|M|$ plus the quantity of genes in b. The objective *function* F' (for the task of minimization) equals the sum of two expressions. The first one is the sum

$$\left(c_e{-}\sum_{k.i\in M}y_{ak.i}{-}\sum_{k.j\in M}f_{k.j}\right){-}\sum_{s}p_s{-}0.5\left(\sum_{g}r_g{-}\sum_{g}l_g\right)$$

calculated over all leaf T-edges e. The second one is the sum

$$\left(2\cdot|M|{-}\sum_{k.i}y_{ak.i}{-}\sum_{k.i}y_{bk.i}{-}\sum_{k.j}f_{k.j}\right){-}\sum_{s}p_s{-}0.5\left(\sum_{g}r_g{-}\sum_{g}l_g\right)$$

calculated over all inner T-edges e. The variables except y and corresponding constrains are defined in the following items 1–3. They correspond to items 1–3 in Section 2, which described the algorithm of reduction for the distance problem.

1) Let $e = (a,b)$ be a T-edge and $G''(e) = a^+(z,t) + b^+(z,t)$. Let us define the variables u_{es} and p_{es} as well as the constraints ensuring that the number $C_1{}'$ of cycles in the graph $G''(e)$ at the minimum point of F' equals $\sum_s p_{es}$. Specifically, for each pair $s = (g,g')$ of different extremities from M for an inner edge $e = (a,b)$, we define the integer non-negative variables u_{eas} and u_{ebs} and Boolean variables p_{eas} and p_{ebs}. For a leaf edge e and its b', we define the integer non-negative variable u_{ebs} and Boolean variable p_{ebs}, where s is any adjacency in b'. Both variables u_{eas} and u_{ebs} obey $u_s \le m_s$. Here, m_s is the *number* of the mentioned pair s, where s runs over all pairs where the variables u_{eas} and u_{ebs} are defined for any fixed $e \in T$. For Boolean variable p_{es}, we impose that $p_{es}{\cdot}m_s \le u_s$.

Let $e = (a,b)$ be a leaf edge. We impose that $u_{as} \le m_{as} \cdot x_{as}$ ensuring that $u_{as} = 0$ for any pair s of non-merged extremities from M. For a, let s include g which is an extremity of a gene $k.j$ from M. Each variable u_{as} and each extremity of a gene $k.\,j^{'} \in b^{'}$ of the same type as g and a boundary of a path in b' are imposed that $u_{as}{\le}m_{as}\left(1{-}z_{kj^{'}j}\right)$. These constraints ensure that $u_s = 0$ if the extremity g belongs to a common gene of $a'(z)$ and $b'(z)$, and in $G''(e)$ we have: g is a boundary of a path and, at the same time, is an extremity of an G''-edge marked a. For b, let an adjacency $s \in b^{'}$ and includes $g \in k.\,j$. Each variable u_{bs} and each i ($1 \le i \le s(k)$) are imposed that $u_{bs}{\le}m_{bs}(1{-}z_{kji}+\sum_{g1\in M}x_{ag^{'}g1})$, where g' is the extremity of a gene $k.\,i \in M$ of the same type as g. These constraints ensure that $u_s = 0$ if g belongs to a common gene of $a'(z)$ and $b'(z)$, and in $G''(e)$ we have: g is a boundary of a path and, at the same time, an extremity of a G''-edge marked b. Each extremity g_1 from M is imposed that $u_{as}{\le}m_{as}\left(1{-}t_{bg1g}+\sum_{g2}x_{ag1g2}\right)$, which ensures that $u_s = 0$ if $g \in s$, g belongs to a special gene in $a'(z)$ and g in $G''(e)$ is not a boundary of a path but the end of a terminal new G''-edge of the path. Each extremity g_1 in b' that is a boundary of a path in b' is imposed the constraint $u_{bs} \le m_{as}(1 - t_{ag1g})$, ensuring that $u_{bs} = 0$ if the extremity $g \in b^{'}$, g belongs to a special gene in $b'(z)$ and g in $G''(e)$ is not a boundary of a path but the end of a terminal new G''-edge of the path.

Recall that now we consider a leaf edge $e = (a,b)$. Each pair $(s1,s2)$, where $s1 = (g,g_1)$ is a pair of extremities from M and $s2 = (g',g_2)$ is an adjacency from b' where g and g' are of the same type and belongs to paralogs $k.j$ and $k.j'$, is imposed the constraints:

$$u_{as1}{\le}u_{bs2}+m_{as1}\left(1{-}z_{kj^{'}j}\right),u_{bs2}{\le}u_{as1}+m_{bs2}\left(2{-}z_{kj^{'}j}{-}x_{agg1}\right).$$

It follows that $u_{s1} = u_{s2}$ for neighboring old G''-edges $s1$ and $s2$ of $G''(e)$. Each pair $(s1,s2)$, where $s1 = (g_1,g_2)$ and $s2 = (g_3,g_4)$ are pairs of extremities from M, is imposed that

$$u_{as1} \leq u_{as2} + m_{as1}\left(2 - t_{bg2g3} - x_{ag3g4}\right), u_{as2} \leq u_{as1} + m_{as2}\left(2 - t_{bg2g3} - x_{ag1g2}\right),$$

all g_1, g_2, g_3, g_4 are pairwise different. These constraints ensure that $u_{s1} = u_{s2}$ for two old G''-edges (marked a) of $G''(e)$ spaced by exactly one new G''-edge. Each pair ($s1$,$s2$) where $s1 = (g_1,g_2)$ and $s2 = (g_3,g_4)$ are different adjacencies from b' are imposed that

$$u_{s1} \leq u_{s2} + m_{s1}\left(1 - t_{ag2g3}\right), u_{s2} \leq u_{s1} + m_{s2}\left(1 - t_{ag2g3}\right).$$

These constraints ensure that $u_{s1} = u_{s2}$ for two old G''-edges (marked b) of the graph $G''(e)$ spaced by exactly one new G''-edge.

For an inner edge $e = (a,b)$, let us impose that $u_{as} \leq m_{as}x_{as}$, $u_{bs} \leq m_{bs}x_{bs}$, ensuring that $u_{as} = 0$ or $u_{bs} = 0$ for non-merged $s = (g,g')$. Each variable u_{as} is imposed that $u_{as} \leq m_{as}\left(y_{bk.j} + \sum_{g1} x_{bgg1}\right)$ where s includes $g \in k.\ j$. Similar constraints are imposed for u_{bs}. It ensures that $u_s = 0$ if g belongs to a common gene and is a boundary of a path in $G''(e)$. The equality $u_s = 0$ (for u_{as} and u_{bs}) in the case when the extremity g belongs to a special gene (in $a'(z)$ or $b'(z)$) and a boundary edge of a path (in $G''(e)$) is provided in the same manner as for u_{as} on a leaf edge. Each pair ($s1$,$s2$), where $s1 = (g,g_1)$ and $s2 = (g,g_2)$ are different pairs of extremities from M, is imposed that

$$u_{as1} \leq u_{bs2} + m_{s1}\left(1 - x_{bgg2}\right), u_{bs2} \leq u_{as1} + m_{s2}\left(1 - x_{agg1}\right).$$

These constraints ensure that $u_{s1} = u_{s2}$ for old neighboring G''-edges $s1$ and $s2$ in $G''(e)$. Each pair ($s1$,$s2$), where $s1 = (g_1,g_2)$ and $s2 = (g_3,g_4)$ are pairs of extremities from M, is imposed that

$$\begin{aligned} u_{as1} &\leq u_{as2} + m_{as1}\left(2 - t_{bg2g3} - x_{ag3g4}\right), u_{as2} \leq u_{as1} \\ &+ m_{as2}\left(2 - t_{bg2g3} - x_{ag1g2}\right), u_{bs1} \leq u_{bs2} \\ &+ m_{bs1}\left(2 - t_{ag2g3} - x_{bg3g4}\right), u_{bs2} \leq u_{bs1} \\ &+ m_{bs2}\left(2 - t_{ag2g3} - x_{bg1g2}\right) \end{aligned}$$

(all g_1, g_2, g_3, g_4 are pairwise different). These constraints ensure that $u_{s1} = u_{s2}$ for two old G''-edges of the graph $G''(e)$ spaced by exactly one new G''-edge.

The statement that $C_1' = \sum_s p_s$ at the minimum point, for any $e \in T$, is proved in the same way as in Section 2.

2) Let us define the variables and constraints ensuring that the quantity C_2' of even paths in $G''(e)$ on an edge $e = (a,b)$ at the minimum point of F' equals $\sum_g r_g - \sum_g l_g$. Let us define for each extremity g from M an integer variable r_{eag} that runs over the values 0, +1, −1. And similarly for b if b is inner; otherwise only for each extremity g in b'.

The constraint $-2(1 - y_{ak.\ \ i}) \leq r_{eag} \leq 2(1 - y_{ak.\ \ i})$ implies that $r_{eag} = 0$ for any extremity g of any gene $k.\ i \in M$ missing in v. And similarly for b if b is inner.

Each pair of different extremities g_1 and g_2 from M is imposed that $r_{eag1} + r_{eag2} \leq 2(1 - x_{ag1g2})$, ensuring that $r_{eag1} + r_{eag2} \leq 0$ if these extremities are merged. For an inner edge e, similar constraints are imposed with the index a replaced by b; otherwise, they are imposed only for each adjacency (g_1,g_2) from b' with zero in the right part. It is also imposed that $r_{eag1} + r_{eag2} \leq 2(1 - t_{ebg1g2})$, ensuring that $r_{eag1} + r_{eag2} \leq 0$ if g_1 and g_2 form a new adjacency. For an inner edge e, similar constraints are imposed with the index a replaced by b and vice versa; otherwise this constraint is imposed only for pairs (g_1,g_2) of extremities from b' that do not form an adjacency. For a leaf edge e, each pair (g,g'), where extremities g (of $k.j$) and g' (of $k.j'$) are of the same type, g is from M, and g' is from b', the constraint is imposed that

$$-2\left(1 - z_{bkj'j} + y_{ak.j}\right) \leq r_{eag} - r_{ebg'} \leq 2\left(1 - z_{bkj'j} + y_{ak.j}\right),$$

ensuring that $r_{ebg} = r_{eag'}$ for z-bijection extremities g and g' of the same type if g is present in a. For an inner edge e and each extremity $g \in M$ of $k.i$, we impose that $r_{eag} \leq r_{ebg} + 2(y_{ak.\ \ i} + y_{bk.\ \ i})$, $r_{ebg} \leq r_{eag} + 2(y_{ak.\ \ i} + y_{bk.\ \ i})$. These constraints ensure that $r_{eag} = r_{ebg}$ for a gene g common for $a'(z)$ and $b'(z)$.

For each edge e and gene $k.j$ from M, we define the Boolean variable $f_{ek.j}$ to indicate whether the gene $k.j$ is common for $a'(z)$ and $b'(z)$. Specifically, for an inner edge e we impose that

$$f_{ek.j} \geq 1 - y_{ak.j} - y_{bk.j}, f_{ek.j} \leq 1 - y_{ak.j}, f_{ek.j} \leq 1 - y_{bk.j};$$

while for a leaf edge, the variable $y_{bk.j}$ is replaced with $1 - \sum_i z_{bkij}$ yielding:

$$f_{ek.j} \geq \sum_i z_{bkij} - y_{ak.j}, f_{ek.j} \leq \sum_i z_{bkij}.$$

For each extremity g of gene $k.i$ from M, we define the integer variable l_{eg}, which equals r_{eag} if g is an extremity of a common gene in $a'(z)$ and $b'(z)$, or equals 0 otherwise. The corresponding constraints are as follows:

$$-f_{ek.i} \leq l_{eg} \leq f_{ek.i}, l_{eg} \leq r_{eag} + 2\left(1 - f_{ek.i}\right), r_{eag} \leq l_{eg} + 2\left(1 - f_{ek.i}\right).$$

Now the statement that $C_2' = \sum_g r_g - \sum_g l_g$ for any $e \in T$ is proved in the same manner as in the distance problem.

3) On each edge $e \in T$, where $e = (a,b)$, each of the first two parentheses in the definition F' equals the number of common genes in $a'(z)$ and $b'(z)$ counted once plus the total number of special genes

in the same structures; this sum will be referred to as *X*. Indeed, the values of $c_e - \sum_{k,i} y_{ak.i}$ and $2 \cdot |M| - \sum_{k,i} y_{ak.i} - \sum_{k,i} y_{bk.i}$ equal to the total number of all genes in *a* and *b*. These values minus $\sum_{k,j} f_{k.j}$, the number of common genes counted once, gives *X*.

Let $\Psi(e, x, y, z)$ be equal to $C_0 + n + s_a + s_b$ in Φ from Section 2. Here, Ψ is actually considered on the edge $e = (a,b)$, and the summands are defined as in Section 2. We obtain $X - \Psi = C_3 - C_0$, where C_3 is the total number of genes in special chromosomes in $a'(z)$ and $b'(z)$, and C_0 is the total number of special chromosomes in $a'(z)$ and $b'(z)$. We define that $E = \sum_{e \in T} (C_3 - C_0)(e)$. For any arrangement *A* and the initial numberings, *E*(*A*) is defined analogously.

Theorem 2 states that our reduction algorithm upon the condition (*) is *exact.* To this end, let us introduce the definitions. Assumed that the arguments (*x,y,z,t,-f,u,p,r,l*) of the function *F*′ *extend* the arguments (*x,y,z*) of the function *F**, if the variable *t* for each *e* defines new adjacencies in $a^+(z,t)$ and $b^+(z,t)$ such that the distance between the structures is minimum, and other variables are defined through $a'(z)$, $b'(z)$, and G'' such that the above constraints as well as the equalities $C_1{}' = \sum_s p_s$ and $C_2{}' = \sum_g r_g - \sum_g l_g$ are satisfied for each edge *e*. Clearly, there is an extension for each arrangement *A* = (*x,y,z*); any of them is denoted as A_+. Recall that an arrangement *A* defines structures *a* and *b* at the ends of the edge *e* = (*a,b*).

Theorem 2

Upon (*), the minimum values of functions *F**(*A*) and *F*′(*x,y,z,t,f,u,p,r,l*) are equal. Otherwise, the difference between the minimum values is not greater than the total quantity of special chromosomes in the minimum point of *F*′.

Lemma

For any structures $a'(z)$ and $b'(z)$ we have $Q_2 = Q_1 + C_3$ where Q_1 and Q_2 are the maximal values of $C_1 + 0.5 \cdot C_2$ and $C_1' + 0.5 \cdot C_2'$, respectively.

Proof of lemma

Let the maximums of Q_1 and Q_2 be reached at the points t_0 and t'_0, respectively. We can add to the structures $a''(z,t_0)$ and $b''(z,t_0)$ the removed special chromosomes and new chromosomes that are identical to these special chromosomes. Respectively, C_3 cycles of length 2 are added to the breakpoint graph $a''(z,t_0) + b''(z,t_0)$. Thus, $Q_2 \geq Q_1 + C_3$.

To prove the inverse relation let us consider the distance *d* between $a'(z)$ and $b'(z)$. As we know $d = C_0 + n + s_a + s_b - Q_1$. On the other hand, following [12] it is easy to verify that $d \leq C_0' + n + s_a + s_b + C_3 - Q_2$ where C_0' is the quantity of new chromosomes that remain unchanged under a transformation of $a^+(z, t_0')$ into $b^+(z, t_0')$. Evidently $C_0' \leq C_0$. Thus, $Q_2 \leq Q_1 + C_3$. □.

Proof of theorem 2

For any arrangement *A* and edge $e = (a, b) \in T$, it is valid that $F^*(A) = \sum_{e \in T} \Phi(e, t_0)$, where $\Phi(a, b, t) = C_0 + n + s_a + s_b - C_1 - 0.5 \cdot C_2$ and C_0, *n*, s_a, s_b, C_1, C_2 are defined as in Section 2; t_0 is also defined there. By the Lemma we have on each *T*-edge *e* that $C_1' + 0.5 \cdot C_2' = C_1 + 0.5 \cdot C_2 + C_3$. This and item 3 imply that

$$F'(A_+) = F^*(A) + E(A) - C_3(A) = F^*(A) - C_0(A).$$

It follows from items 1–2 that any minimum point of the function *F*′ is an extension of its coordinates *x,y,z*. Let A_+ be a minimum point of the *F*′. If the condition (*) is satisfied for it, then *E*(*A*) = 0; *A* is the minimum arrangement since $C_0(A) \geq 0$ for any *A*. If (*) is not satisfied, let A_+ be the point of minimum of *F*′, *A** be a minimum arrangement. Then

$$F'(A_+) = F^*(A) - C_0(A) \geq F^*(A^*) \\ -C_0(A), F'(A_+) \leq F'(A*_+) \leq F^*(A^*).\square$$

The constants 2·|*M*| and c_e can be omitted in the minimization.

Notice that the condition (*) limits special (broadly speaking, circular) chromosomes in the structures, i.e., limits the relationship between the parental structure and its direct descendants. Our computer experiments (data not shown) have demonstrated that the solution of the second problem with *F** using a heuristic algorithm (described in [17]) differed little from that with *F*′ using ILP. Indeed, the evolutionary scenario for mitochondrial chromosome structures generated by the heuristic algorithm in [17] included no special chromosomes.

Clearly, the number of variables and constraints in it cubically depend on the size of the initial data.

Examples for the reconstruction problem on synthetic data

Example 1

Let us consider a tree ((*c*, *d*),(*e*, *f*)) with four leaf structures and three genes in each structure distributed among circular chromosomes: structure *c*, (1, 2, −1); *d*, (1, 1, −2); *e*, (2, 1, −1); *f*, (1, 1, 2). Other designations in all examples are as in Section 2.2.

The initial numberings are as follows: *c*, (1.1, 2, −1.2); *d*, (1.1, 1.2, −2); *e*, (2, 1.1, −1.2); *f*, (1.1, 1.2, 2).

The ILP program of the Pulp python package returned the solution with the total number of operations being 3, one in each edge. The result swaps 1.1 and 1.2 in the leaves *e* and *f*; the chromosome (1.1, 1.2, 2) appears in the root; (1.1, 2, −1.2) is ancestral for the nodes *c* and *d* and (1.1, 2, 1.2) is ancestral for the nodes *e* and *f*. The program execution time was about 13 h.

Example 2

Let us consider the same tree with five genes in each leaf structure distributed among linear chromosomes: structure *c*, [2, 3, −4, 1], [1]; *d*, [3, −4, 1], [1, 2]; *e*, [1, −2, 3], [1, 4]; and *f*, [1, −2, 3, 4], [1].

The initial numberings is as follows: the paralogs of gene 1 in each structure have the name 1.1 in the first chromosome and 1.2 in the second chromosome.

The ILP program of the Pulp python package returned the solution with the total number of operations being 6, one in each edge. The result swaps 1.1 and 1.2 in the leaves *c* and *d*; the chromosome [1.1, 2, 3, 4, 1.2] appears in the root; [1.1, 2, 3, −4, 1.2] is ancestral for the nodes *c* and *d* and [1.1, −2, 3, 4, 1.2] is ancestral for the nodes *e* and *f*. The program execution time was about 20 h.

Examples for the reconstruction problem on biological data

The orthologs of plastid and mitochondrial proteins were obtained using our algorithm and databases available at http://lab6.iitp.ru/ppc/ and http://lab6.iitp.ru/mpc/. The mitochondrial, plastid, and bacterial chromosome structures were extracted from genome annotations in GenBank by our script.

Example 1

Let us consider the example from [17], specifically, the tree given in ([17], Figure 4) and the mitochondrial chromosome structures in its leaves listed in ([17], Table 3); which are also given in Table 1 where they are marked by (*l*) after the species name. The mitochondrial chromosomes belong to the sporozoan class Aconoidasida. The ILP program of the package of Joint Supercomputer Center of the Russian Academy of Sciences (http://www.jscc.ru/eng/index.shtml) returned the solution specified in other lines of Table 1. The program execution time was about 2 days. The resulting reconstruction of the mitochondrial chromosome structures is slightly different from that obtained in ([17], Table 3) using the heuristic algorithm in [17]. The result is close to those obtained in [17]. Specifically, the gene *ls*2 encoding a fragment of the large subunit ribosomal RNA becomes in the inner nodes the separate linear chromosome which likely reflects frequent relocations of the fragment. Although ribosomal RNA genes are rarely fragmented, it is arguable that the small fragments can be highly mobile in this case. The tree generated using protein alignments in apicomplexan parasites [26] is in good agreement with the chromosome structure tree.

Example 2

Let us exemplify the reconstruction for plastid chromosome structures with paralogs in brown algae. They are also given in Table 2 marked by (*l*) after the species name. The following chromosome structure tree was built: (Ectocarpus_siliculosus, (Fucus_vesiculosus, Saccharina_japonica)). The tree that was generated using highly conserved elements identified in the complete plastid genomes of all considered species [27] is in good agreement with the chromosome structure one. The reconstruction result is presented in Table 2. The program execution time was about 5 days.

Example 3

Let us exemplify the reconstruction for chromosome structures with paralogs from *Rhizobium* spp. The corresponding tree generated here using chromosome structures is given in Table 3 in the lines marked by (*l*) is shown in Fig. 2. The reconstruction result is presented in other lines of Table 3. The program execution time was about 11 days.

Solution of the problem of optimal arrangement of contigs

Let us apply the developed approach to the *contig* problem, optimal genome assembly from contigs. The biological significance of the problem is discussed in [28].

Contig problem statement

Sequencing results in a set *a* of contigs (or scaffolds, or sequences of a higher level, etc.), each of which includes several genes with their own direction of transcription. Here, a contig is considered as a *path* of genes each with a name not necessarily unique (paralogs) and a direction (Fig. 3a). Therefore, *a* is a structure comprised of paths. Two contigs can be concatenated in four ways considering that a contig is a double-stranded DNA region with undefined beginning and end. A set of contigs can be concatenated into a long path or cycle; these variants are essentially equivalent, and we will consider the second one as in [28]. It is convenient to consider that each contig ends with an extremity of one of its genes.

The *contig* problem is as follows. We are given two sets *a* and *b* of contigs, and it is required to concatenate contigs from *a* into one cycle and contigs from *b* into another cycle, and simultaneously find paralog numberings (see Section 1.3) with the minimum distance between the cycles without paralogs (Fig. 3b). Naturally, these cycles are considered as structures comprised of sole cycle each. Similarly to the solution below, a more

Table 1 Reconstruction obtained by reduction to ILP for mitochondrial chromosome structures in sporozoan class Aconoidasida. The data in the tree leaves are in the lines marked by (*l*) after the species name. It was obtained from genomes represented in GenBank

Plasmodium fragile – Babesia bovis	*ls5 ls6 ls2 (L) ss4 ss6 ls7 ss3 ls3 ls9 ss2 ls4 *cox3 ls8 ss5 ss1 cox1 cytb ls1 (C)
Theileria annulata – Babesia bovis	cox1 *cox3 ls1 *ls3 *cytb *ls5 ls4 (L)
Theileria annulata – Theileria parva	cox1 *cox3 ls1 *ls3 *cytb *ls5 ls4 (L)
Theileria annulata (l)	cox1 *cox3 ls1 *ls3 *cytb *ls5 ls4 (L)
Theileria parva (l)	cox1 *cox3 ls1 *ls3 *ls2 *cytb *ls5 ls4 (L)
Babesia bovis (l)	cox1 *cox3 ls1 *ls2 *ls3 *cytb *ls4 ls5 (L)
Plasmodium fragile – Plasmodium berghei	ss4 ss6 ls7 ss3 ls3 ls9 ss2 ls4 *cox3 ls8 ss5 ss1 cox1 cytb ls1 (C) ls2 (L)
Plasmodium juxtanucleare – Plasmodium berghei	ls1 ss4 ss6 ls7 ls6 ss3 ls3 ls9 ss2 ls4 ls5 *cox3 ls8 ss5 ss1 cox1 cytb ls2 (L)
Plasmodium juxtanucleare – Leucocytozoon sabrazesi	ls1 ss4 ss6 ls7 ls6 ss3 ls3 ls9 ss2 ls4 ls5 *cox3 ls8 ss5 ss1 cox1 cytb ls2 (L)
Plasmodium juxtanucleare – Plasmodium gallinaceum	ls1 ss4 ss6 ls7 ls6 ss3 ls3 ls9 ss2 ls4 ls5 *cox3 ls8 ss5 ss1 cox1 cytb ls2 (L)
Plasmodium juxtanucleare (l)	ls1 ss4 ss6 ls7 ls6 ss3 ls3 ls9 ss2 ls4 ls5 *cox3 ls8 ss5 ss1 cox1 cytb ls2 (L)
Plasmodium gallinaceum (l)	ls1 ss4 ss6 ls7 ls6 ss3 ls3 ls9 ss2 ls4 ls5 *cox3 ls8 ss5 ss1 cox1 cytb ls2 (L)
Leucocytozoon sabrazesi (l)	ls1 ss4 ss6 ls7 ls6 ss3 ls3 ls9 ss2 ls4 ls5 *cox3 ls8 ss5 ss1 cox1 cytb ls2 (L)
Plasmodium berghei (l)	ls1 ss4 ss6 ls7 ls6 ss3 ls3 ls9 ss2 ls4 ls5 *cox3 ls8 ss5 ss1 cox1 cytb ls2 (L)
Plasmodium fragile – Plasmodium relictum	ss4 ss6 ls7 ss3 ls3 ls9 ss2 ls4 *cox3 ls8 ss5 ss1 cox1 cytb ls1 (C) ls2 (L)
Plasmodium reichenowi – Plasmodium relictum	ss4 ss6 ls7 ss3 ls3 ls9 ss2 ls4 *cox3 ls8 ss5 ss1 cox1 cytb ls1 (C) ls2 (L)
Plasmodium floridense – Plasmodium relictum	ls7 ss3 ls3 ls9 ss2 ls4 *cox3 ls8 ss5 ss1 cox1 cytb ls1 ss6 (C)
Plasmodium floridense (l)	ss3 ls3 ls9 ss2 ls4 *cox3 ls8 ss5 ss1 cox1 cytb ls1 ss4 ss6 ls7 (L)
Plasmodium relictum (l)	ss3 ls3 ls9 ss2 ls4 *cox3 ls8 ss5 ss1 cox1 cytb ls1 ss4 ss6 ls7 (C)
Plasmodium reichenowi – Plasmodium mexicanum	ss3 ls3 ls9 ss2 *cox3 ls8 ss5 ss1 cox1 cytb ls1 ss4 ss6 ls7 (L)
Plasmodium reichenowi – Plasmodium falciparum	ss3 ls3 ls9 ss2 *cox3 ls8 ss5 ss1 cox1 cytb ls1 ss4 ss6 ls7 (L)
Plasmodium reichenowi (l)	ss3 ls3 ls9 ss2 *cox3 ls8 ss5 ss1 cox1 cytb ls1 ss4 ss6 ls7 (L)
Plasmodium falciparum (l)	ss3 ls3 ls9 ss2 *cox3 ls8 ss5 ss1 cox1 cytb ls1 ss4 ss6 ls7 (L)
Plasmodium mexicanum (l)	ss3 ls3 ls9 ss2 *cox3 ls8 ss5 ss1 cox1 cytb ls1 ss4 ss6 ls7 (L)
Plasmodium fragile – Plasmodium simium	ss4 ss6 ls7 ss3 ls3 ls9 ss2 ls4 *cox3 ls8 ss5 ss1 cox1 cytb ls1 (C)
Plasmodium fragile – Leucocytozoon fringillinarum	ss3 ls3 ls9 ss2 ls4 *cox3 ls8 ss5 ss1 cox1 cytb ls1 ss4 ss6 ls7 (L)
Plasmodium fragile – Plasmodium vivax	ls1 ss6 ls7 ss3 ls3 ls9 ss2 ls4 *cox3 ls8 ss5 ss1 cox1 cytb (C)
Plasmodium fragile – Plasmodium knowlesi	ls1 ss6 ls7 ss3 ls3 ls9 ss2 ls4 *cox3 ls8 ss5 ss1 cox1 cytb (C)
Plasmodium fragile – Leucocytozoon majoris	ls1 ss6 ls7 ss3 ls3 ls9 ss2 ls4 *cox3 ls8 ss5 ss1 cox1 cytb (C)
Plasmodium fragile (l)	ls1 ss6 ls7 ss3 ls3 ls9 ss2 ls4 *cox3 ls8 ss5 ss1 cox1 cytb (C)
Leucocytozoon majoris (l)	ss3 ls3 ls9 ss2 ls4 *cox3 ls8 ss5 ss1 cox1 cytb ls1 ss6 ls7 (C)
Plasmodium knowlesi (l)	ls1 ss6 ls7 ss3 ls3 ls9 ss2 ls4 *cox3 ls8 ss5 ss1 cox1 cytb (C)
Plasmodium vivax (l)	ls1 ss6 ls7 ss3 ls3 ls9 ss2 ls4 *cox3 ls8 ss5 ss1 cox1 cytb (C)
Leucocytozoon fringillinarum (l)	ss3 ls3 ls9 ss2 ls4 *cox3 ls8 ss5 ss1 cox1 cytb ls1 ss6 ls7 ss4 (C)
Plasmodium simium (l)	ls1 ss6 ls7 ss3 ls3 ls9 ss2 ls4 *cox3 cox1 cytb ls8 ss5 ss1 (C)

If a structure has two chromosomes, they are given on separate lines. Circular and linear chromosomes are marked by (C) and (L), respectively. The symbol * means the complementary chain

general case is considered when contigs from *a* and, similarly, from *b* are concatenated into structures of another fixed shape.

An almost linear (to be precise, $n \cdot f(n)$, where $f(n)$ is the inverse Ackermann´s function) algorithm was proposed in [28]; it exactly solves the contig problem on the condition of equal gene content of two sets of contigs (n genes in each) and without paralogs. Below is the solution of the problem with this *condition released* based on its reduction to ILP. The presence of paralogs makes the problem NP-hard. In addition, the solution in [28] relies on the algebraic theory of permutation groups, which absolutely differs from our approach and relies on a different distance. Specifically, in the case of equal gene content, our distance (in the terms specified in Sections 1–2) equals $n - C_1 - 0.5C_2$, where C_1

Table 2 Reconstruction obtained by reduction to ILP for plastid chromosome structures with paralogs in brown algae

Ectocarpus siliculosus (*l*)	rpl32_1 rpl21_1 *rps4 *rps16 *rps1 rpl9 rpl11 rpl1 rpl12 *rps10 *tufa *rps7 *rps12 *rpl31 *rps9 *rpl13 *rpoa *rps11 *rps13 *rpl36 *rps5 *rpl18 *rpl6 *rps8 *rpl5 *rpl24 *rpl14 *rps17 *rpl29 *rpl16 *rps3 *rpl22 *rps19 *rpl2 *rpl23 *rpl4 *rpl3 *rpl21_2 *rpl32_2 *rpl35 rpl20 *rpl19 rpl27 rpl34 rps20 rpob rpoc1 rpoc2 rps2 rps14 *rps18 *rpl33 clpc rbcl (C)
Fucus vesiculosus (*l*)	*rpl19 rpl27 rpl34 rps20 rpob rpoc1 rpoc2 rps2 rpl35 rpl20 rbcl rps14 *clpc rpl33 rps18 *rpl32_2 rps16 rps4 rps1 rpl9 rpl11 rpl1 rpl12 *rps10 *tufa *rps7 *rps12 *rpl31 *rps9 *rpl13 *rpoa *rps11 *rps13 *rpl36 *rps5 *rpl18 *rpl6 *rps8 *rpl5 *rpl24 *rpl14 *rps17 *rpl29 *rpl16 *rps3 *rpl22 *rps19 *rpl2 *rpl23 *rpl4 *rpl3 *rpl21_2 (C)
Saccharina japonica (*l*)	*rps2 *rpoc2 *rpoc1 *rpob *rps20 *rpl34 *rpl27 rpl19 rpl35 rpl20 rbcl rps14 *rps18 *rpl33 clpc rpl32_1 rpl21_1 rpl3 rpl4 rpl23 rpl2 rps19 rpl22 rps3 rpl16 rpl29 rps17 rpl14 rpl24 rpl5 rps8 rpl6 rpl18 rps5 rpl36 rps13 rps11 rpoa rpl13 rps9 rpl31 rps12 rps7 tufa rps10 *rpl12 *rpl1 *rpl11 *rpl9 rps1 *rps4 *rps16 (C)
Inner non-root node	*rpl19 rpl27 rpl34 rps20 rpob rpoc1 rpoc2 rps2 rpl35 rpl20 rbcl rps14 rpl32_2 *rps18 *rpl33 clpc rpl32_1 rpl21_1 *rps4 *rps16 *rps1 rpl9 rpl11 rpl1 rpl12 *rps10 *tufa *rps7 *rps12 *rpl31 *rps9 *rpl13 *rpoa *rps11 *rps13 *rpl36 *rps5 *rpl18 *rpl6 *rps8 *rpl5 *rpl24 *rpl14 *rps17 *rpl29 *rpl16 *rps3 *rpl22 *rps19 *rpl2 *rpl23 *rpl4 *rpl3 *rpl21_2 (C)
Tree root	rpl32_1 rpl21_1 *rps4 *rps16 *rps1 rpl9 rpl11 rpl1 rpl12 *rps10 *tufa *rps7 *rps12 *rpl31 *rps9 *rpl13 *rpoa *rps11 *rps13 *rpl36 *rps5 *rpl18 *rpl6 *rps8 *rpl5 *rpl24 *rpl14 *rps17 *rpl29 *rpl16 *rps3 *rpl22 *rps19 *rpl2 *rpl23 *rpl4 *rpl3 *rpl21_2 *rpl19 rpl27 rpl34 rps20 rpob rpoc1 rpoc2 rps2 rps14 rpl32_2 *rps18 *rpl33 clpc rpl35 rpl20 rbcl (C)

Paralog numbers are given after the underscore. For other designations, see Table 1

is the quantity of cycles and C_2 is the quantity of even paths in the breakpoint graph; while the distance used in [28] can be calculated using the same expression but with C_2 being the quantity of all paths. We fix arbitrary initial numberings of paralogs, and the structures a and b with fixed numberings are denoted as a' and b'.

In the next section the reduction of the contig problem to ILP is presented, which simultaneously determines the numberings and the above mentioned two cycles with the minimum distance between them. The resulting cycles will be referred to as *minimum*. Our solution for two sets of contigs can be similarly extended to an arbitrary number of sets. In this problem, it is important to discriminate between *outer* and *inner adjacencies*. The former merge the extremities of contigs, while the latter merge the extremities of genes within contigs. The contig problem concerns the selection of outer adjacencies that transform two given sets of contigs into two cycles with the minimum distance between them while the inner adjacencies remain unaltered. However, calculation of the distance between cycles includes the variation of inner adjacencies. The distance calculation allows all six operations mentioned in Section 1. Thus, both adjacency types are used altogether.

Solution of the contig problem

A reduction algorithm for the contig problem to ILP is described below.

For each pair $s = (g_1, g_2)$ of extremities of different contigs in a', we define the Boolean variable t_{as}. It equals 1 if g_1 and g_2 form the outer adjacency; otherwise $t_{as} = 0$. Similarly for b'. The usual constraints ensure that each contig extremity is merged with exactly one contig extremity.

For each ordered pair $d = (c_1, c_2)$ of different contigs from either given set a' or b', we define the Boolean variable v_d to indicate whether the contig c_2 is concatenated with c_1 and is placed after it; the set of all values $v_d = 1$ consistently determines the clockwise order on a required cycle. First, the usual constraints ensure that each contig is concatenated with exactly one contig on either side. The constraint $v_d \le t_{s1} + t_{s2} + t_{s3} + t_{s4}$ (for pairs s_1, s_2, s_3, s_4 of extremities of the contigs c_1 and c_2) provides the relation between the order and the outer adjacencies. Let us define the integer (non-Boolean) variables w_{ac} and w_{bc}, where c runs over all contigs in a' or b' and $1 \le w_c \le N$ (N is the quantity of contigs in the corresponding set). The variable w_{ac} numbers all contigs in strictly increasing order according to their position in the cycle, this order is violated only in the last contig. Similarly for w_{bc}. For each ordered pair $d = (c_1, c_2)$ of different contigs from either set a' or b', we define the Boolean variable r_d to indicate the contig where this order is violated. It equals 1 if $v_d = 1$ and $w_{c2} \le w_{c1}$, or 0 otherwise. The corresponding constraints are as follows: $r_d \le v_d$, $Nr_d \le N - (w_{c2} - w_{c1})$, $Nr_d \ge w_{c1} - w_{c2} + 1$. Finally, $\sum_d r_d = 1$ ensures that all contigs of the set are concatenated into a single circular chromosome, where they are numbered by the variable w in strictly increasing order.

The further reduction of the contig problem to ILP corresponds to the layout in [17] (or the general case of such reduction was considered in Section 2 above). Namely, let us introduce the Boolean variable z_{kij}, where $z_{kij} = 1$ if the gene $k.i$ in a' corresponds to the gene $k.j$ in b'; otherwise $z_{kij} = 0$. The standard constraints ensure that z_{kij} defines a partial bijection of k-paralogs. If $z_{kij} = 1$, the gene $k.j$ in b' is renamed to $k.i$ and becomes synonymous to $k.i$ in a', after which the genes in the z-bijection are arbitrarily numbered to keep the structures numbered. Structures resulting from such renumbering in b', *are denoted as* $a'(z)$ and $b'(z)$. Adjacencies of the contigs in the cycle are defined by the variable t as in Section 2. The resulting two cycles will be referred to as $a'(z,t)$ and $b'(z,t)$. Notice that these structures

Table 3 Reconstruction obtained by reduction to ILP for chromosome structures in *Rhizobium* spp.

Rhizobium_N324_CP013630 (*I*)	*rpsA *rpsO rplT rpsT rpoN rpoE_1 rpsU_1 rpoZ *rplI *rpsR *rpsF *rpsI *rplM rplK rplA rplJ rplL rpoB rpoC rpsL rpsG rpsJ rplC rplD rplW rplB rpsS rplV rpsC rplP rpsQ rplN rplX rplE rpsN rpsH rplF rplR rpsE rplO rpsM rpsK rpoA rplQ rpsB *rpsD rpoE_2 rplY rpoH_1 rpsU_2 rpoE_3 rpsP rplS *rpoH_2 rplU (C)
Rhizobium_phaseoli_straiNN261_CP013580 (*I*)	*rpsA *rpsO rplT rpsT rpoN rpoE_1 rpsU_1 rpoZ *rplI *rpsR *rpsF *rpsI *rplM rplK rplA rplJ rplL rpoB rpoC rpsL rpsG rpsJ rplC rplD rplW rplB rpsS rplV rpsC rplP rpsQ rplN rplX rplE rpsN rpsH rplF rplR rpsE rplO rpsM rpsK rpoA rplQ rpsB *rpsD rpoE2 rplY rpoH_1 rpsU_2 rpoE_3 rpsP rplS rpoH_2 rplU (C)
Rhizobium_N324_CP013630 – *Rhizobium_phaseoli_N261_CP013580*	*rpsA *rpsO rplT rpsT rpoN rpoE_1 rpsU_1 rpoZ *rplI *rpsR *rpsF *rpsI *rplM rplK rplA rplJ rplL rpoB rpoC rpsL rpsG rpsJ rplC rplD rplW rplB rpsS rplV rpsC rplP rpsQ rplN rplX rplE rpsN rpsH rplF rplR rpsE rplO rpsM rpsK rpoA rplQ rpsB *rpsD rpoE2 rplY rpoH_1 rpsU_2 rpoE_3 rpsP rplS rpoH_2 rplU (C)
Rhizobium_etli_CP001074 (*I*)	*rpsA *rpsO rplT rpsT rpoN rpoE_1 rpsU_1 rpoZ *rplI *rpsR *rpsF *rpsI *rplM rplK rplA rplJ rplL rpoB rpoC rpsL rpsG rpsJ rplC rplD rplW rplB rpsS rplV rpsC rplP rpsQ rplN rplX rplE rpsN rpsH rplF rplR rpsE rplO rpsM rpsK rpoA rplQ rpsB *rpsD rpoE2 rplY rpoE_3 *rpoE_4 rpoH_1 rpsU_2 rpsP rplS rpoH_2 rplU (C)
Rhizobium_etli_CP001074 – *Rhizobium_phaseoli_N261_CP013580*	*rpsA *rpsO rplT rpsT rpoN rpoE_1 rpsU_1 rpoZ *rplI *rpsR *rpsF *rpsI *rplM rplK rplA rplJ rplL rpoB rpoC rpsL rpsG rpsJ rplC rplD rplW rplB rpsS rplV rpsC rplP rpsQ rplN rplX rplE rpsN rpsH rplF rplR rpsE rplO rpsM rpsK rpoA rplQ rpsB *rpsD rpoE_2 rplY rpoH_1 rpsU_2 rpoE_3 rpsP rplS rpoH_2 rplU (C)
Rhizobium_gallicum_CP006877 (*I*)	*rpsA rpsO rplT rpsT rpoN rpoE_1 rpoZ *rplI *rpsR *rpsF *rpsI *rplM rplK rplA rplJ rplL rpoB rpoC rpsL rpsG rpsJ rplC rplD rplW rplB rpsS rplV rpsC rplP rpsQ rplN rplX rplE rpsN rpsH rplF rplR rpsE rplO rpsM rpsK rpoA rplQ rpsB *rpsD rpoE_2 rplY rpoH_1 *rpsU_1 rpsU_2 rpsP rplS *rpoH_2 rplU (C)
Rhizobium_etli_CP001074 – *Rhizobium_gallicum_CP006877*	*rpsA *rpsO rplT rpsT rpoN rpoE_1 rpsU_1 rpoZ *rplI *rpsR *rpsF *rpsI *rplM rplK rplA rplJ rplL rpoB rpoC rpsL rpsG rpsJ rplC rplD rplW rplB rpsS rplV rpsC rplP rpsQ rplN rplX rplE rpsN rpsH rplF rplR rpsE rplO rpsM rpsK rpoA rplQ rpsB *rpsD rpoE2 rplY rpoH_1 rpsU_2 rpsP rplS rpoH_2 rplU (C)
Rhizobium_NXC14_CP021030 (*I*)	*rpsA *rpsO rplT rpsT rpoN rpoE_1 rpoE_3 rpsU_1 rpoZ *rplI *rpsR *rpsF *rpsI *rplM rplK rplA rplJ rplL rpoB rpoC rpsL rpsG rpsJ rplC rplD rplW rplB rpsS rplV rpsC rplP rpsQ rplN rplX rplE rpsN rpsH rplF rplR rpsE rplO rpsM rpsK rpoA rplQ rpsB *rpsD rpoE_2 rplY rpoH_1 rpsU_2 rpsP rplS rpoH_2 rplU (C)
Rhizobium_etli_CP001074 – *Rhizobium_NXC14_CP021030*	*rpsA *rpsO rplT rpsT rpoN rpoE_1 rpsU_1 rpoZ *rplI *rpsR *rpsF *rpsI *rplM rplK rplA rplJ rplL rpoB rpoC rpsL rpsG rpsJ rplC rplD rplW rplB rpsS rplV rpsC rplP rpsQ rplN rplX rplE rpsN rpsH rplF rplR rpsE rplO rpsM rpsK rpoA rplQ rpsB *rpsD rpoE_2 rplY rpoH_1 rpsU_2 rpsP rplS rpoH_2 rplU (C)
Rhizobium_leguminosarum_AM236080 (*I*)	*rpsA *rpsO rplT rpsT rpoN rpsU_1 *rplI *rpsR *rpsF *rpsI *rplM rplK rplA rplJ rplL rpoB rpoC rpsL rpsG rpsJ rplC rplD rplW rplB rpsS rplV rpsC rplP rpsQ rplN rplX rplE rpsN rpsH rplF rplR rpsE rplO rpsM rpsK rpoA rplQ rpsB *rpsD *rpoD *rpoZ rpoH_1 rpsU_2 rpoE_3 rplU (C)
Rhizobium_etli_CP001074 – *Rhizobium_leguminosarum_AM236080*	*rpsA *rpsO rplT rpsT rpoN rpoE_1 rpsU_1 rpoZ *rplI *rpsR *rpsF *rpsI *rplM rplK rplA rplJ rplL rpoB rpoC rpsL rpsG rpsJ rplC rplD rplW rplB rpsS rplV rpsC rplP rpsQ rplN rplX rplE rpsN rpsH rplF rplR rpsE rplO rpsM rpsK rpoA rplQ rpsB *rpsD rpoH_1 rpsU_2 rpoH_2 rplU (C)
Rhizobium_tropici_CIAT_899_CP004015 (*I*)	*rpsA *rpsO rplT rpsT *rpoN *rplI *rpsR *rpsI *rplM rplK rplA rplL rpoB rpoC rpsL rpsG rpsJ rplC rplD rplW rplB rplV rpsC rplP rpsQ rplN rplX rpsH rplF rplR rpsE rplO rpsM rpsK rpoA rplQ rpsB *rpsD rpoH_1 *rpoH_2 rplU (C)
Rhizobium_etli_CP001074 – *Rhizobium_tropici_CIAT_899_CP004015*	*rpsA *rpsO rplT rpsT rpoN rpsU1 rpoZ *rplI *rpsR *rpsF *rpsI *rplM rplK rplA rplJ rplL rpoB rpoC rpsL rpsG rpsJ rplC rplD rplW rplB rpsS rplV rpsC rplP rpsQ rplN rplX rplE rpsN rpsH rplF rplR rpsE rplO rpsM rpsK rpoA rplQ rpsB *rpsD rpoH_1 rpsU_2 rpoH_2 rplU (C)
Rhizobium_IRBG74_HG518322 (*I*)	*rpsO rplT rpsT *rpoN rpoZ *rpsR *rpsF *rpsI *rplM rpsB *rpsD *rplQ *rpoA *rpsK *rpsM *rplO *rpsE *rplR *rplF *rpsH *rpsN *rplE *rplX *rplN *rplP *rpsC *rplV *rpsS *rplB *rplW *rplD *rplC *rpsJ *rpsG *rpsL *rpoC *rpoB *rplL *rplJ *rplA *rplK *rpoD rplY rpoH_1 rpsP rplS *rplU (C) rpsU_1 *rpsU_2 rpsA (L)
Rhizobium_LPU83_HG916852 (*I*)	rpsO rpsA rplT rpsT rpoN rpoZ *rplI *rpsR *rpsF rplK rplA rplJ rplL rpoB rpoC rpsL rpsG rpsJ rplC rplD rplW rplB rpsS rplV rpsC rplP rpsQ rplN rplX rplE rpsN rpsH rplF rplR rpsE rplO rpsM rpsK rpoA rplQ *rpsI *rplM rpsD *rpsB *rpoD rplY rpoH_1 rpsU_1 *rpsU_2 rpsU_3 rpsP rplS rpoH2 *rplU (C)
Rhizobium_IRBG74_HG518322 – *Rhizobium_LPU83_HG916852*	rpsO rpsA rplT rpsT rpoN rpoZ *rplI *rpsR *rpsF *rpsI *rplM rplK rplA rplJ rplL rpoB rpoC rpsL rpsG rpsJ rplC rplD rplW rplB rpsS rplV rpsC rplP rpsQ rplN rplX rplE rpsN rpsH rplF rplR rpsE rplO rpsM rpsK rpoA rplQ rpsD *rpsB *rpoD rplY rpoH_1 rpsU_1 *rpsU_2 rpsU_3 rpsP rplS rpoH_2 *rplU (C)
Rhizobium_NT26_FO082820 (*I*)	rpsO rpsA rplT *rpoN rpoZ *rplI *rpsR *rpsF *rpsI *rplM rplK rplA rplJ rplL rpoB rpoC rpsL rpsG rpsJ rplC rplD rplW rplB rpsS rplV rpsC rplP rplN rplX rplE rpsN rpsH rplF rplR rpsE rplO rpsM rpsK rpoA rplQ rpsB rpsU_2 *rpsD *rpoD rplY rpoH_1 rpsU_1 rplU rpsP rplS *rpoH_2 (C)
Rhizobium_IRBG74_HG518322 – *Rhizobium_NT26_FO082820*	rpsO rpsA rplT rpsT rpoN rpoZ *rplI *rpsR *rpsF *rpsI *rplM rplK rplA rplJ rplL rpoB rpoC rpsL rpsG rpsJ rplC rplD rplW rplB rpsS rplV rpsC rplP rpsQ rplN rplX rplE rpsN rpsH rplF rplR rpsE rplO rpsM rpsK rpoA rplQ rpsB *rpsD *rpoD rplY rpoH_1 rpsU_1 rpsP rplS rpoH_2 *rplU (C)
Tree root	rpsO rpsA rplT rpsT rpoN rpsU_2 rpoZ *rplI *rpsR *rpsF *rpsI *rplM rplK rplA rplJ rplL rpoB rpoC rpsL rpsG rpsJ rplC rplD rplW rplB rpsS rplV rpsC rplP rpsQ rplN rplX rplE rpsN rpsH rplF rplR rpsE rplO rpsM rpsK rpoA rplQ rpsB *rpsD *rpoD rplY rpoH_1 rpsU_1 rpsP rplS rpoH_2 rplU (C)

For other designations, see Tables 1 and 2

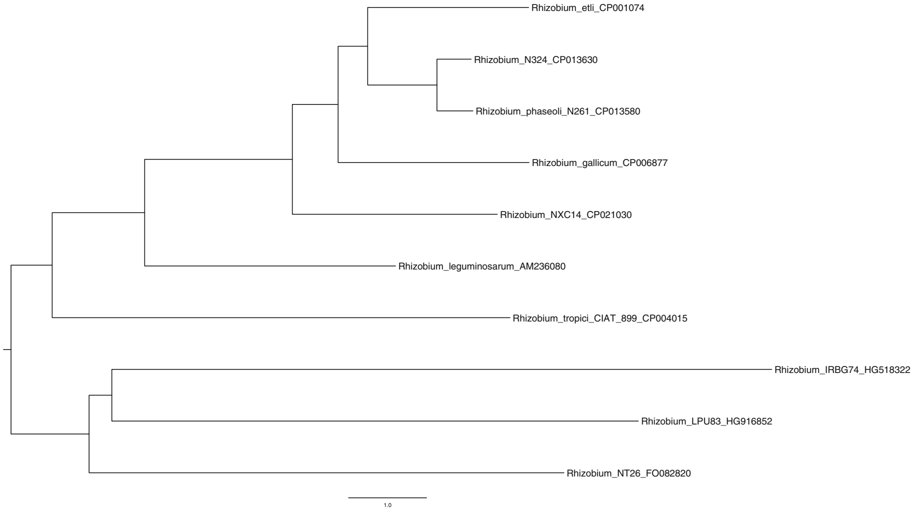
Rhizobium_etli_CP001074
Rhizobium_N324_CP013630
Rhizobium_phaseoli_N261_CP013580
Rhizobium_gallicum_CP006877
Rhizobium_NXC14_CP021030
Rhizobium_leguminosarum_AM236080
Rhizobium_tropici_CIAT_899_CP004015
Rhizobium_IRBG74_HG518322
Rhizobium_LPU83_HG916852
Rhizobium_NT26_FO082820
1.0

Fig. 2 Tree of chromosomal structures of *Rhizobium* spp. generated using the chromosome structures given in Table 3 in the lines marked by (*I*) after the species name. The reconstruction result is presented in the other lines of the same Table 3

have unequal gene content. Let us define $G = a'(z,t) + b'(z,t)$; this breakpoint graph is composed of cycles. It is close to G' in Section 2, although equal gene contents were considered there. Let us focus on the differences of the current procedure from that in [17] remembering the presence of outer adjacencies.

1) The quantity B of blocks in G is expressed by the variable x_{as} for each s, where s is an inner adjacency or a pair of contigs extremities in a'. It equals 1 if s is a boundary of a block in $a'(z,t)$, and 0 otherwise. Similarly for x_{bs} and $b'(z,t)$. Specifically, each s in a', is imposed the constraint $x_{as} \geq \sum_j z_{ki1j} - \sum_j z_{li2j} + (t_s - 1)$, where $k.i_1$ and $l.i_2$ are genes in a' with these extremities. Similarly for s in b'. For an inner adjacency s, the summand $t_s - 1$ is omitted. Let the objective function be

$$H = 0.5 \cdot \sum_s x_s + \sum_s y_s - \sum_s p_s.$$

Thus, $B = 0.5 \cdot \sum_s x_s$ at the minimum point of H.

2) The sum S_1 of integer parts of half-lengths of the maximal connected regions of conventional edges in G is expressed through the Boolean variables y_{as} and y_{bs} for all s as in subsection (1). It equals 0 if s is a boundary or within a block in $a'(z,t)$ or $b'(z,t)$; while for the adjacencies of common genes, y_{as} and y_{bs} on the edges of G alternate within each such region and equal to zero at the ends of odd regions. Specifically, for each pair s_1 in a' and s_2 in b', where gene $k.i$ is adjacent to gene $k_1.i_1$ in s_1 and gene $k.j$ is adjacent to gene $k_2.i_2$ in s_2 we impose that

$$y_{as1} + y_{bs2} \geq z_{kij} + \sum_j z_{k1i1j} + \sum_j z_{k2ji2} - 2 + (t_{as1} - 1) + (t_{bs2} - 1),$$

where the summands $t_{as1} - 1$ and $t_{bs2} - 1$ are omitted for inner adjacencies $s1$ and $s2$, respectively. It implies that y_s cannot equal 0 at both neighboring conventional edges. Consequently, it implies that the minimum quantity of unities on the region is reached for the arrangement where zeros alternate with unities starting with zero. Thus, $S_1 = \sum_s y_s$.

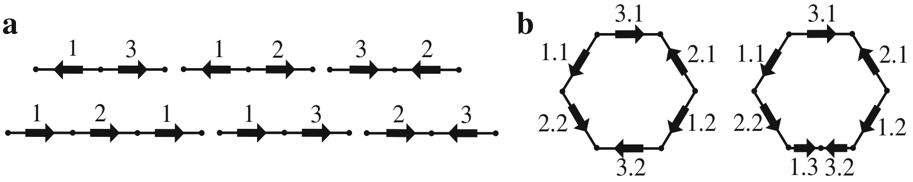

Fig. 3 **a** Given sets *a* and *b* composed of three contigs each. **b** Problem solution: the minimum cycles for (**a**) (left) and (**b**) (right)

3) The quantity S_2 of cycles in G composed of conventional edges is expressed in the variables u_s and p_s for s as in subsection (1) (see also section 2 and [17]). For each s, we impose that $u_s \le m_s \sum_j z_{kij}$ (for s from a') or $u_s \le m_s \sum_j z_{kji}$ (for s from b'), where $k.i$ is a gene with an extremity from s. For each pair s of contig extremities, we impose that $u_s \le m_s t_s$. In addition, for each pair $s1$ and $s2$ that include extremities of genes $k.i$ from a' and $k.j$ from b', we impose that

$$u_{s1} \le u_{s2} + m_{s1}(1-z_{kij}) + m_{s1}(1-t_{s2}), u_{s2} \le u_{s1} + m_{s2}(1-z_{kij}) + m_{s2}(1-t_{s1}),$$

where, the summand m_{s2} $(1 - t_{s1})$ and m_{s1} $(1 - t_{s2})$ are omitted for inner adjacencies $s1$ and $s2$, respectively. Then $S_2 = \sum_s p_s$ at the minimum point. The proof is similar to the proof that the quantity C_1 of cycles in G' equals $\sum_s p_s$ in Section 2.

Therefore, the minimum value of function H equals $B + S_1 - S_2$, which equals the distance between the desired cycles [16]. Indeed, lemma 5 and theorem 6 in [16] suggest that the distance equals $B + S + D–P$ where B is the quantity of special nodes (that is, blocks) in G; S equals S_1 plus the quantity S_3 of such odd regions at a boundary of any path minus S_2; D is the sum of defects of components in the graph G; P is the quantity of operations, optimized through the interaction of chains in the graph G, [16, item 3.4]. Circular G has no paths, hence, D, S_3, and P equal zero.

Clearly, the number of variables and constraints in it quadratically depend on the size of the data.

Examples for the contig problem on synthetic data

Example 1

We are given two sets, a (upper) and b (lower), each composed of three contigs (Fig. 3a). The initial numberings are as follows (left to right): a', [1.1, 3.1], [1.2, 2.1], and [3.2, 2.2]; b', [1.1, 2.1, 1.2], [1.3, 3.1], and [2.2, 3.2]. Other designations in all examples are as in Section 2.2. The ILP program of the Pulp python package returned the desired minimum cycles for a' and b' (on the left and on the right in Fig. 3b, respectively). The program execution time was about 6 h.

Example 2

We are given two sets, a: [−2,1,3], [5,2,−3], [−2,−4,3], [−5,−4,1], [−1,4] and b: [3,−2,−4], [3,−1,4,5], [−1,1], [2,−3,−5], [3,−1,−5], [−4,2]. The ILP program of the Pulp python package returned the following minimum cycles for a and b (outer adjacencies are indicated by the symbol "|"): a, (1.2, 3.2 | −2.3, −4.2, 3.1 | −1.3, 4.3 | 5.1, 2.1, −3.3 | −5.2, −4.1, 1.1 | −2.2) and b, (1.2 | 3.2, −2.3, −4.2 | 3.1, −1.3, −5.3 | 3.4, −1.4, 4.3, 5.1 | 2.1, −3.3, −5.2 | −4.1, 2.2 | −1.1). The program execution time was about 11 h.

Conclusions

Three problems are considered; all assume unequal gene content and the presence of gene paralogs. These problems are: (1) to determine the minimum number of operations required to transform one chromosome structure into another and the corresponding transformation itself including the paralog identification; (2) to reconstruct along a tree the chromosome structures given in its leaves; (3) to find the optimal arrangements for each given set of contigs, which also includes the paralog identification.

We proved that these problems can be reduced to integer linear programming, which allows an efficient algorithm to redefine the problems to implement integer linear programming tools. The results were tested on synthetic and biological samples.

Abbreviations

DNA: Deoxyribonucleic acid; ILP: Integer linear programming; RNA: Ribonucleic acid

Acknowledgements

A part of the calculations were performed at the Joint Supercomputer Centers of the Russian Academy of Sciences and Moscow Lomonosov University.

Funding

The study was supported by the Russian Science Foundation (project no. 14–50-00150).

Availability of data and materials

The mitochondrial genomes from Aconoidasida were obtained from GenBank, https://www.ncbi.nlm.nih.gov/genomes/GenomesGroup.cgi?opt=mitochondrionid&taxid=2759. The plasid genomes from brown algae were obtained from GenBank, https://www.ncbi.nlm.nih.gov/genomes/GenomesGroup.cgi?opt=plastid&taxid=2759#. The genomes from *Rhizobium* spp. were obtained from GenBank, ftp://ftp.ncbi.nlm.nih.gov/genomes/genbank/bacteria/. The developed reduction algorithms are implemented in the software programs freely downloadable at [29]. The following free and commercial external packages were applied: Pulp python package available at https://pythonhosted.org/PuLP/, ILP-JCP package available at http://www.jscc.ru/eng/index.shtml (Joint Supercomputer Center of the Russian Academy of Sciences); IBM CPLEX Optimizer available at https://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/index.html; IBM Decision Optimization on Cloud available at https://www.ibm.com/us-en/marketplace/decision-optimization-cloud#product-header-top.

Authors' contributions

The proofs were found by KYG and VAL. The programming and testing of the algorithms and programs were performed by RAG. The reduction of the initial problems to ILP was done jointly by the authors. VAL and KYG wrote the manuscript. All the authors have read and approved the final manuscript.

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Author details
[1]Institute for Information Transmission Problems of the Russian Academy of Sciences (Kharkevich Institute), Bolshoy Karetny per. 19, build.1, Moscow 127051, Russia. [2]Faculty of Mechanics and Mathematics, Lomonosov Moscow State University, Leninskiye Gory 1, Main Building, Moscow 119991, Russia.

Received: 19 March 2017 Accepted: 15 November 2017
Published online: 06 December 2017

References

1. Hannenhalli S, Pevzner P. Transforming men into mice (polynomial algorithm for genomic distance problem). In 36th Annual IEEE Symposium on Foundations of Computer Science. Proc FOCS. 1995:581–92.
2. Blanchette M, Kunisawa T, Sankoff D. Gene order breakpoint evidence in animal mitochondrial phylogeny. J Mol Evol. 1999;49(2):193–203.
3. Bergeron A, Mixtacki J, Stoye J. A unifying view of genome rearrangements. Algorithms Bioinform, LNCS. 2006;4175:163–73.
4. Braga MDV, Willing E, Stoye J. Double cut and join with insertions and deletions. J Comput Biol. 2011;18(9):1167–84.
5. Shao M, Lin Y, Moret B. An exact algorithm to compute the DCJ distance for genomes with duplicate genes. In: Proc. of RECOMB 2014, LNBI, vol. 8394. Heidelberg: Springer Verlag; 2014. p. 280–92.
6. Gorbunov KY, Gershgorin RA, Lyubetsky VA. Rearrangement and inference of chromosome structures. Mol Biol (Moscow). 2015;49(3):327–38. https://doi.org/10.7868/S0026898415030076.
7. Yancopoulos S, Attie O, Friedberg R. Efficient sorting of genomic permutations by translocation, inversion and block interchange. Bioinformatics. 2005;21:3340–6.
8. Braga MDV, Stoye J. Sorting linear genomes with rearrangements and Indels. IEEE/ACM Trans Comput Biol Bioinform. 2015;12(3):1–13.
9. Yin Z, Tang J, Schaeffer SW, Bader DA. Exemplar or matching: modeling DCJ problems with unequal content genome data. J Comb Optim. 2016;32(4):1165–81.
10. Chauve C, El-Mabrouk N, Tannier E. (eds.) Models and Algorithms for Genome Evolution. Computational Biology, Springer; 2013;19. doi:10.1007/978-1-4471-5298-9.
11. Yancopoulos S, Friedberg R. DCJ path formulation for genome transformations which include insertions, deletions, and duplications. J Comput Biol. 2009;16:1311–38.
12. Compeau PEC. DCJ-indel sorting revisited. Algorithms Mol Biol. 2013;8:6.
13. Avdeyev P, Jiang S, Aganezov S, Hu F, Alekseyev MA. Reconstruction of ancestral genomes in presence of gene gain and loss. J Comput Biol. 2016;23(3):150–64.
14. Shao M, Moret B. Comparing genomes with rearrangements and segmental duplications. Bioinformatics. 2015;31:i329–38.
15. Martinez FV, Feijão P, Braga MDV, Stoye J. On the family-free DCJ distance and similarity. Algorithms Mol Biol. 2015;10:13. https://doi.org/10.1186/s13015-015-0041-9.
16. Gorbunov KY, Lyubetsky VA. Linear algorithm of the minimal reconstruction of structures. Probl Inf Transm. 2017;53(1):55–72.
17. Lyubetsky VA, Gershgorin RA, Seliverstov AV, Gorbunov KY. Algorithms for reconstruction of chromosomal structures. BMC Bioinform. 2016;17:40. https://doi.org/10.1186/s12859-016-0878-z.
18. Klotz Ed, Newman Alexandra M. Practical guidelines for solving difficult linear programs. Surv Oper Res Manag Sci 2013; 18(1–2):1–17.
19. Ed K, Newman Alexandra M. Practical guidelines for solving difficult mixed integer linear programs. Surv Oper Res Manag Sci. 2013;18(1–2):18–32.
20. Vershik AM, Sporyshev PV. An estimate of the average number of steps in the simplex method, and problems in asymptotic integral geometry. Sov Math Dokl. 1983;28:195–9.
21. Smale S. On the average number of steps of the simplex method of linear programming. Math Program. 1983;27(3):241–62. https://doi.org/10.1007/BF02591902.
22. Seliverstov A. On probabilistic algorithm for solving almost all instances of the set partition problem. In: Weil P, editor. Computer science – theory and applications, CSR 2017. Lecture notes in computer science, vol. 10304. Cham: Springer; 2017. p. 285–93. https://doi.org/10.1007/978-3-319-58747-9_25.
23. Gorbunov KY, Lyubetsky VA. A linear algorithm for the shortest transformation of graphs with different operation costs. J Commun Technol Electron. 2017;62(6):653–62.
24. Gorbunov KY, Lyubetsky VA. A modified algorithm for transformation of chromosomal structures: a condition of absolute exactness. In: CEUR workshop proceedings (CEUR-WS.Org), selected papers of the first international scientific conference "convergent cognitive information technologies (convergent 2016)", Moscow, Russia, vol. 1763; 2016. p. 162–72. in Russian.
25. Alekseyev MA, Pevzner PA. Multi-break rearrangements and chromosomal evolution. Theor Comput Sci. 2008;395(2–3):193–202.
26. Bachvaroff TR, Gornik SG, Concepcion GT, Waller RF, Mendez GS, Lippmeier JC, Delwiche CF. Dinoflagellate phylogeny revisited: using ribosomal proteins to resolve deep branching dinoflagellate clades. Mol Phylogenet Evol. 2014;70:314–22. https://doi.org/10.1016/j.ympev.2013.10.007.
27. Lyubetsky VA, Gershgorin RA, Rubanov LI, Seliverstov AV, Zverkov OA. Evolution and systematics of plastids of rhodophytic branch. In: Proceedings of the Moscow conference on computational molecular biology (MCCMB'17), Moscow, Russia; 2017, July 27–30, 4 pp.
28. Chin Lung L. An efficient algorithm for the Contig ordering problem under algebraic rearrangement distance. J Comput Biol. 2015;22(11):975–87. https://doi.org/10.1089/cmb.2015.0073.
29. Utility for generation of ILP problems represented in IBM lp format. http://lab6.iitp.ru/en/ilp_generatorggl/3. Accessed 24 July 2017.