Advances on Genome Duplication Distances

Yves Gagnon[1], Olivier Tremblay Savard[2], Denis Bertrand[3], and Nadia El-Mabrouk[4]

[1] DIRO, Université de Montréal, H3C 3J7, Canada, y.gagnon@umontreal.ca
[2] DIRO, olivier.tremblay-savard@umontreal.ca
[3] DIRO, bertrden@iro.umontreal.ca
[4] DIRO, mabrouk@iro.umontreal.ca

Abstract. Given a phylogenetic tree involving Whole Genome Duplication events, we contribute to the problem of computing the rearrangement distance on a branch of a tree linking a duplication node d to a speciation node or a leaf s. In the case of a genome G at s containing exactly two copies of each gene, the *genome halving problem* is to find a perfectly duplicated genome D at d minimizing the rearrangement distance with G. We generalize the existing exact linear-time algorithm for genome halving to the case of a genome G with missing gene copies. In the case of a known ancestral duplicated genome D, we develop a greedy approach for computing the distance between G and D that is shown time-efficient and very accurate for both the rearrangement and DCJ distances.

1 Introduction

The increasing number of completely sequenced and annotated genomes now allow to study evolution from a genomic point of view, in contrast to the classical method of comparing single gene sequences. In particular, inferring the structure of ancestral genomes is a major step towards answering to numerous biological questions such as the mechanisms of evolution, the variation in rearrangement and loss rates among the different branches of a phylogenetic tree, and the consequence of such variations on the genetic and physiological specificities of species. Even though manual approaches can not be avoided when analyzing specific biological datasets, the availability of automated methods can largely facilitate and orient the study [19]. In this context, since 1995, the computational biology community working on genome rearrangements has contributed to provide many accurate and rapid algorithms dedicated to the evolutionary study of a set of genomes represented as ordered sequences of genes [8, 16, 21]. However, most of these methods can not be applied to genomes with multiple gene copies, in particular genomes arising from whole genome duplication events.

Whole genome duplication (WGD) is a spectacular evolutionary event that has the effect of simultaneously doubling all the chromosomes of a genome. Right after the WGD event, a genome D_{predup} is transformed into a *perfectly duplicated genome* $D = (D_{predup} \oplus D_{predup})$ containing a complete set of duplicated

chromosomes. However, this initial perfect duplicate status is obscured by subsequent rearrangement events and gene losses, eventually leading to an extant *rearranged duplicated genome* (RD genome) containing exactly two copies of each gene, or a *rearranged duplicated genome with losses* (RDL genome), containing at most two copies of each gene. Evidence of WGD events has shown up across the whole eukaryote spectrum, from the protist *Giardia* to the yeast species [11], including most plant lineages [18], several fishes [22], amphibians [23], and even to mammalian species [17].

Consider a set of genomes that have been subject to WGD events during their evolution, and a phylogenetic tree reflecting the speciation events leading to these genomes. Then, under the assumptions that WGD is the only mechanism leading to gene duplicates and that, in each genome, at least one gene reflects the doubling status of the genome, WGD events can be placed on the phylogenetic tree as new internal nodes, called *WGD nodes* [25]. The *rearrangement phylogeny problem* seeks for ancestral gene orders leading to a most "plausible" evolutionary scenario. The parsimony approach is based on inferring gene orders at the internal nodes of the tree so that the sum of distances among all branches is minimized. When studying genome rearrangements, the most natural distance between two gene orders (distance on a branch) is the minimum number of rearrangements required to transform one gene order into the other. The rearrangements that have been most studied by the genome rearrangement community are inversions and reciprocal translocations (including fusion and fission). In the case of two genomes G and H with no gene duplicates and the same gene content, a key result in the field of genome rearrangement is the Hannenhalli and Pevzner (HP) formula [14, 21] for computing the *rearrangement distance* (minimum number of inversion and translocations required to transform G into H), leading to a polynomial-time algorithm. Another distance that has been extensively studied in the last years is the Double Cut-and-Join (DCJ) distance which represents all known rearrangement events and gives rise to simplest formal results [5, 6, 24].

In the case of genomes with no gene duplicates, one of the main approaches to the rearrangement phylogeny problem is based on iterating an algorithm for the median problem to all overlapping triplets of the phylogenetic tree [7, 8, 15]. A prerequisite for applying such methodology to a phylogeny with WGD nodes is to be able to compute the distance on a branch of the phylogeny. However, this is far from being straightforward, as the orthology relationship between duplicated genes is not set. In particular, computing the distance between an RD genome G and a perfectly duplicated genome D (called the *double distance* in [10, 20]) has been shown to be NP-hard for the DCJ distance [20]. When the ancestral genome D is unknown, the *genome halving problem* seeks for a perfectly duplicated genome D minimizing the rearrangement distance between G and D. In 2003, we have presented the first formal result related to genome duplication, which is an exact linear-time algorithm for solving the genome halving problem [9].

In this paper, we contribute to solving a number of problems related to the computation of the rearrangement and DCJ distances on a branch of a phyloge-

netic tree connecting a first WGD node to a speciation node or a leaf, in both cases of a known and an unknown preduplicated genome (label of the WGD node). In the case of an unknown ancestral genome, our result is a generalization of the genome halving algorithm to a genome G with missing gene copies (i.e. G is an RDL genome instead of an RD genome). In the case of a known ancestral genome D, we present a very efficient and accurate greedy heuristic for computing both the rearrangement and DCJ distance between G and D.

2 Preliminaries

Let Σ be a set of n genes. A *string* is a sequence of genes from Σ, where each gene is signed (+ or −) depending on its orientation. The *reverse* of a string $X = x_1x_2\dots x_r$ is the string $-X = -x_r\ -x_{r-1}\dots -x_1$. A *chromosome* is a string, and a *genome* is a collection of chromosomes. A *unichromosomal* genome has a single chromosome, and a *multichromosomal* genome has at least two nonnull chromosomes $C_1, C_2, \dots C_N$. A *circular chromosome* is a string $x_1 \dots x_r$, where x_1 is considered to follow x_r. A chromosome that is not circular is *linear*. To represent its endpoints, we add an "artificial gene", denoted O, at each extremity. In other words, a linear chromosome is a string of the form $Ox_1 \dots x_rO$.

In this paper, we consider both uni- and multichromosomal genomes. As most unichromosomal genomes are formed by a circular chromosome, and most multichromosomal genomes are formed by linear chromosomes, only circular unichromosomal genomes, and linear multichromosomal genomes are considered here.

2.1 Evolutionary events and genomic distances

All the following evolutionary events apply to both uni- and multichromosomal genomes, except translocations that are only relevant for multichromosomal genomes.

- A *reversal* (or *inversion*) is an operation that replaces some proper substring of a chromosome into its reverse.
- A *translocation* between two chromosomes $X = X_1X_2$ and $Y = Y_1Y_2$ is an event transforming the two chromosomes into X_1Y_2, Y_1X_2 (prefix-prefix), or into $X_1(-Y_1), (-Y_2)X_2$ (prefix-suffix). Two special cases of reciprocal translocations are *fusions* (if one of the two chromosomes generated by the translocation is an empty string) and *fissions* (if one of the two input chromosomes is the empty string).
- A *Whole Genome Duplication* (WGD) is an event transforming a multichromosomal genome $G = \{C_1, C_2, \dots C_N\}$ into a multichromosomal genome $D = \{C_1, C'_1, C_2, C'_2, \dots C_N, C'_N\}$ containing $2N$ chromosomes where, for each $1 \leq i \leq N$, $C_i = C'_i$. In the case of a circular genome G represented by the string $x_1x_2\dots x_r$, a WGD transforms G into a circular genome D represented by either of the two strings : $x_1x_2\dots x_r\ x_1x_2\dots x_r$, or $x_1x_2\dots x_r - x_r \dots - x_2 - x_1$.

– Finally, a *loss* is an operation removing a proper substring from a chromosome.

A *rearrangement event* will refer to an inversion or a translocation event. The *rearrangement distance* between two genomes G and H (with the same gene content or not), denoted $d_R(G, H)$, is the minimum number of rearrangement events required in a scenario transforming G into H. In the case of genomes with single gene copies, computing the inversion and/or translocation distance has been shown to be a polynomial-time problem, and the best developed method runs in linear time [3, 4].

Another distance that has been extensively studied in the last years is the DCJ distance [5, 6, 24]. Given a genome G, a Double-Cut-and-Join (DCJ) is an operation that "cuts" two adjacencies pq and rs in a genome, and replaces them by either pr and qs, or ps and qr. The DCJ distance is an "artificial" distance in the sense that some DCJ operations are not relevant from a biological point of view. However, it is interesting from a theoretical point of view as it leads to a unifying formulae including all previously studied rearrangement events, as well as transpositions, for which no polynomial-time exact method is known. Computing the DCJ distance between two signed permutations is a linear-time problem.

2.2 Genome definitions

In this section, we consider G to be a genome defined on a set Σ of genes, i.e. g is in G iff $g \in \Sigma$.

• G is a *singleton genome* iff each gene is present exactly once in G.

• G is a *rearranged duplicated (RD) genome* iff each gene is present exactly twice in G.

• G is a *perfectly duplicated genome* (or *duplicated genome* for short) iff:

– *The multichromosomal case:* G is an RD genome containing an even number $2N$ of chromosomes, with two identical copies of each chromosome. If D is the set of the N different chromosomes, then we write $G = (D \oplus D)$.
– *The circular case:* G is an RD genome and there is a string D such that G is exactly D followed by D (we write $G = D \oplus D$), or D followed by $-D$ (we write $G = D \oplus -D$).

• G is a *rearranged duplicated genome with losses (RDL genome)* iff each gene in Σ is present at least once and at most twice in G.

• G is a *duplicated genome with losses (DL genome)* if each gene of Σ is present in one or two copies in G, and if a duplicated genome D can be obtained from G by an appropriate insertion of an additional copy of each *singleton* (gene present in one copy in G).

A DL genome A is said to be *induced* by an RDL genome G if each gene in Σ has the same copy number in A and G.

Let G be an RD genome and H be an RDL genome. We can define the evolutionary cost $\mathcal{E}(G, H)$ as the minimum number of inversions, translocations and losses required to transform G into H.

2.3 The breakpoint graph

In a series of papers published in 1995 [12–14], Hannenhalli and Pevzner (hereafter HP) developed polynomial-time algorithms for computing the rearrangement distance (inversion only, translocation only, or inversion+translocation) between two singleton genomes G and H on Σ. The algorithms all depend on a bicolored graph $\mathcal{B}(G, H)$, called the *breakpoint graph*, constructed from G and H as follows (Tesler's formalism [21]).

Graph $\mathcal{B}(G, H)$: If gene x of Σ has a positive sign, replace it by the pair $x^t x^h$, and if it is negative, replace it by $x^h x^t$. Then the set V of vertices of $\mathcal{B}(G, H)$ is the set of x^t and x^h for all x in Σ. Any two vertices of V that are adjacent in some chromosome in G, other than x^t and x^h deriving from the same x, are connected by a black edge (thick lines in Figure 2(b)), and any two adjacent vertices in H are connected by a gray edge (thin lines in Figure 2(b)). Notice that adjacencies to O are not represented.

In the case of circular chromosomes, each vertex in V is incident to exactly one black and one gray edge, and thus the graph uniquely decomposes into $c(G, H)$ disjoint cycles of alternating edge colors.

In the case of G and H being multichromosomal genomes, let an *endpoint vertex of* G (resp. of H) be a vertex of V adjacent to O in G (resp. in H). Then any vertex has degree zero if it is an endpoint in both G and H, one if it is an endpoint in exactly one of the two genomes or two otherwise. Thus, the graph decomposes into $c(G, H)$ cycles and $p(G, H)$ paths of alternating edge colors. Note that a path may contain only one vertex and no edges. We denote by p_{GG} (resp. p_{HH}) the number of paths linking two endpoints of G (resp. of H). If G and H have the same number of chromosomes, then $p_{GG} = p_{HH}$. Otherwise, suppose w.l.o.g. that G has more chromosomes than H, then $p_{HH} \leq p_{GG}$.

The rearrangement distance: Although somehow different algorithms are required for sorting by translocation only, inversion only or inversion+translocation, all results in [12–14] (revisited by Tesler [21] for multichromosomal genomes) can be summarized by a unique formulae given bellow:

$$\mathsf{HP}: \quad d_R(G, H) = n + N - C(G, H) + h(G, H)$$

where n is the number of genes, N is the number of chromosomes of G in the case of a multichromosomal genome and $N = O$ in the case of a circular genome, $C(G, H) = c(G, H) + p(G, H) - p_{GG}$ is the dominating parameter, with $p(G, H) = p_{GG} = 0$ in the case of circular genomes, and $h(G, H)$ is a correction parameter that has a different value depending on the considered model. In all cases, it is related to the decomposition of $\mathcal{B}(G, H)$ into components, where a *component* is a maximal set of crossing cycles. A component is termed good if it can be transformed into a set of cycles of size 1 by increasing the number of cycles at each step, and bad otherwise. The parameter $h(G, H)$ reflects the number of bad components of the graph. As the probability for a component to be bad is low, the value of $h(G, H)$ is usually low compared to the dominating parameter $C(G, H)$.

The DCJ distance: Based on the breakpoint graph, the DCJ distance between G and H can be expressed as follows [5, 20]:

$$\mathsf{DCJ}: \quad d_{DCJ}(G,H) = n - \left(c(G,H) + \frac{p_{even}}{2} \right)$$

where p_{even} is the number of paths with an even number (≥ 0) of edges.

3 Genome Halving with Losses

Given an RD genome G, the *Genome Halving* problem is to find a duplicated genome D minimizing the rearrangement distance with G. In other words, we define $d_R(G)$ as the minimum rearrangement distance between G and any duplicated genome D. Then the problem is to find a duplicated genome D such that $d_R(G) = d_R(G, D)$.

In [9] we have developed an exact linear-time algorithm, called Algorithm Dedouble, for the reversals-only version of the problem (in the case of unichromosomal genomes), the translocations-only version, and the version with both reversals and translocations. The approach was to start from a *partial breakpoint graph* $\mathcal{B}(G)$, i.e. the breakpoint graph with the set of edges restricted to the black edges representing G, and to complete this graph with a set of "valid" gray edges, i.e. gray edges representing a duplicated genome D, maximizing the number of cycles and paths (parameter $c(G, D)$ and $p(G, D)$ in the HP formula). The second step was then to perform modifications on the obtained graph in order to remove bad components that can be avoided, and obtain a duplicated genome D minimizing the rearrangement distance with G (i.e. minimizing the HP formula).

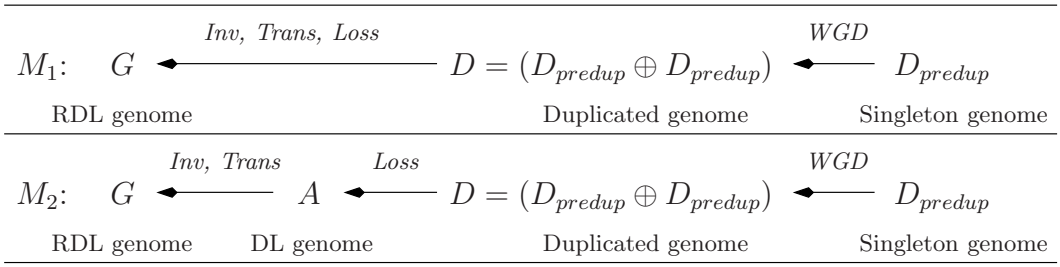

Fig. 1. Evolutionary models M_1 and M_2 considered for a present-day rearranged duplicated genome with losses G. Direction of evolution is represented by arrows orientation.

Here, we seek to generalize Algorithm Dedouble to a present-day genome G containing both duplicated genes and singletons, i.e. to an RDL genome. Let G be a present-day RDL genome. We assume that G has evolved from an ancestral

singleton genome through a WGD, and a sequence of inversions, translocations and loss events. We are then interested in finding such a pre-duplicated singleton genome D_{predup} minimizing the number of rearrangements needed to obtain G (see model M_1 in Figure 1). Note that we do not attempt to minimize the number of losses.

The following theorem allows to reduce the evolutionary model to a simpler one (model M_2 in Figure 1), where all losses occur first, followed by all rearrangement events.

Theorem 1. *Let G be an RDL genome and D be a duplicated genome. Then there exists a DL genome A induced by G such that $d_R(G, A) = d_R(G, D)$.*

Proof: By induction on $e = \mathcal{E}(G, D)$
1. The property is trivially verified for $e = 0$ and $e = 1$.
2. Suppose the induction hypothesis is verified for a given $e \geq 1$. Now suppose that $\mathcal{E}(G, D) = n + 1$, and let $\mathcal{E} = E_1, E_2, \ldots E_n, E_{n+1}$ be a sequence of $e + 1$ events transforming a duplicated genome D into an RDL genome G. Let G' be the genome obtained after performing the sequence of e events $\mathcal{E}' = E_1, E_2, \ldots E_n$ on D. Then $e = \mathcal{E}(G', D)$ as otherwise (if e is not the minimum number of events transforming D into G') $e+1$ would not be the minimum number of events transforming D into G. Moreover, by the induction hypothesis, there exists a DL genome A' for which $d_R(G', A') = d_R(G', D)$.

If E_{n+1} is a rearrangement event, the DL genome A induced by G is equal to A'. Then, we have $d_R(G, A) = d_R(G, A') = d_R(G', A') + 1$ and $d_R(G, D) = d_R(G', D) + 1$. Therefore, $d_R(G, A) = d_R(G, D)$.

Otherwise, E_{n+1} is a loss event. Let A be the DL genome obtained from A' by removing the genes that are removed by the loss operation E_{n+1}. Then, it is easy to see that a minimum sequence of k rearrangement events transforming A' into G' can be converted into a sequence of k rearrangement events transforming A into G (just by removing the lost genes from the inverted or translocated segments). Therefore $d_R(G, A) \leq d_R(G', A')$. Similarly, a minimum sequence of k rearrangement events transforming A into G can be converted into a sequence of k rearrangement events transforming A' into G'. Therefore, $d_R(G, A) = d_R(G', A') = d_R(G, D)$ □

Corollary 1. *Let G be an RDL genome, and A be a DL genome induced by G minimizing the cost $d_R(G, A)$. If D is the duplicated genome obtained from A, then $d_R(G) = d_R(G, D)$.*

Proof: Let A be a DL genome induced by G minimizing the cost $d_R(G, A)$, and D be the duplicated genome obtained from A. Then we have $d_R(G, D) = d_R(G, A)$. Suppose $d_R(G) \neq d_R(G, A)$, i.e. $d_R(G, A) > d_R(G)$. Let D' be a duplicated genome such that $d_R(G, D') = d_R(G)$. Then, from Theorem 1, there is a DL genome A' such that $d_R(G, A') = d_R(G, D') = d_R(G)$. And thus $d_R(G, A') < d_R(G, A)$, which is a contradiction with the fact that A minimizes the rearrangement cost □

Therefore, finding a duplicated genome D such that $d_R(G) = d_R(G, D)$ can be reduced to the problem of finding a DL genome A induced by G such that $d_R(G, A)$ is minimal over all DL genomes induced by G. In other words, loss events can be ignored.

To find such DL genome A, we use a generalization of Algorithm Dedouble, called Algorithm Dedouble-RDL(G), that proceeds as follows:

1. Consider the RD genome G' obtained from G by "gluing" singletons to an adjacent gene. More precisely, consider a given orientation for chromosomes. Then, for each maximum sequence S of singletons in G: (1) if S is a chromosome, then just remove this chromosome; (2) otherwise, if S is connected to a left extremity of a chromosome, then replace its successor x (the gene representing the right adjacency of S in G) by the artificial gene $x' = Sx$; (3) otherwise, if S is not connected to a left extremity of a chromosome, then replace its predecessor x (possibly already updated in step (2)) by a new artificial gene x' representing the sequence xS.
2. Use Algorithm Dedouble to infer a duplicated genome A' from G'.
3. Recover a DL genome A from A' by replacing each of its artificial genes by its corresponding sequence of singletons, and by adding all removed chromosomes of G (formed exclusively by singletons).

The following theorem immediately follows from the fact that Algorithm Dedouble outputs a doubled genome A' minimizing the distance to G', and that singletons are preserved in the same order in G and A.

Theorem 2. *Let G be an RDL genome and A be the DL genome resulting from* Algorithm Dedouble-RDL(G). *Then $d_R(G, A) = d_R(G)$.*

4 An algorithm for the Double Distance

Given an RD genome G and a duplicated genome $D = (D_{predup} \oplus D_{predup})$, how to compute the distance between G and D? The problem of computing the DCJ distance between G and D has already been shown to be an NP-hard problem [20]. The difference in complexity (polynomial versus NP-hard) between computing the distance of two singleton genomes versus computing the distance between a RD genome and a duplicated genome (or between two RD genomes) is due to the missing one-to-one orthology relationship between genes. In other words, given a labeling of the genes in G, the problem is to find a labeling of the genes in D leading to a minimum distance between G and D.

Consider a given beginning gene, in the case of a circular genome, or a given order and left-to-right orientation of chromosomes in the case of a multichromosomal genome G. Then, for each gene x (present in two copies in G and also in D), label the first occurrence of x in G as x_1 and the second as x_2. Let $\mathcal{B}(G)$ be the partial breakpoint graph for G. To complete this partial graph, each double adjacency (x^r, y^s) in D (where $r, s \in \{t, h\}$) should be represented in the completed graph $\mathcal{B}(G, D)$ by either of the following pairs of gray edges:

$\{(x_1^r, y_1^s), (x_2^r, y_2^s)\}$, or $\{(x_1^r, y_2^s), (x_2^r, y_1^s)\}$. Each of these two cases leads to a different labeling of the gene copies in D. The problem is then to choose the pairs of gray edges allowing to minimize the HP formulae in the case of the rearrangement distance, or the DCJ formulae in the case of minimizing the DCJ distance.

Here, we focus on maximizing the dominating value $C(G, D)$ in the HP formulae. In the case of Genome Halving, this simplification has been called the Weak Genome Halving Problem [1]. We similarly define our simplified problem as follows:

Weak Double Distance Problem. *For a given labeled RD genome G and a duplicated genome D, find a labeling of gene copies in D that maximizes the parameter $C(G, D)$ in the breakpoint graph $\mathcal{B}(G, D)$ of the labeled genomes G and D.*

Notice that, in the case of a circular genome, a labeling of D maximizing the parameter $C(G, D)$ also maximizes the DCJ formulae, as $C(G, D) = c(G, D)$ in this case. In the multichromosomal case, a labeling of D maximizing $C(G, D)$ is likely to also maximize the DCJ formulae, though there is no guarantee for that.

Clearly, the "best" exhaustive approach trying all possible labelings for D has a worst running-time complexity in $O(n.2^n)$ for $n = |\Sigma|$. Indeed, D has 2^n possible labelings, and for each labeling, the most efficient approach for computing the rearrangement distance between G and D is linear.

4.1 Circular genomes

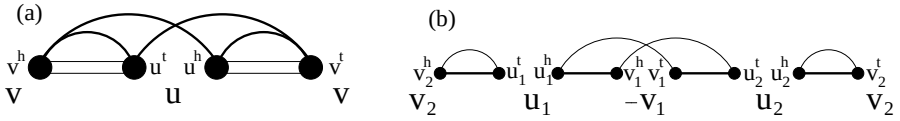

Fig. 2. (a) The contracted breakpoint graph $\mathcal{CB}(D, G)$ constructed for the circular RD genome $G = (u, -v, u, v)$ and the circular duplicated genome $D = (u, v) \oplus (u, v)$. Gray edges (thin lines) represent genome D and black edges (thick lines) represent genome G. (b) The breakpoint graph $\mathcal{B}(G, D)$ corresponding to the labeling $G = (u_1, -v_1, u_2, v_2)$ and $D = (u_1, v_1) \oplus (u_2, v_2)$. Given the above labeling of G, the labeling of D, leading to 3 cycles, is optimal. The resulting rearrangement distance is 1.

Let G be a circular RD genome and D be a circular duplicated genome. We consider the contracted breakpoint graph representation $\mathcal{CB}(D, G)$ defined as follows: the set of vertices of $\mathcal{CB}(D, G)$ is $V = \{x^r, \text{ for all } x \in \Sigma \text{ and } r \in \{t, h\}\}$. Any two vertices which are adjacent in D (except the extremities of a same gene) are connected by two parallel gray edges, and any two adjacent in G (except the extremities of a same gene) are connected by a black edge (see Figure 2.(a)).

Such representation has previously been used in the context of genome halving for circular [2] and multichromosomal genomes [10], with the difference that each gray edge was represented exactly once. It follows that each vertex of $\mathcal{CB}(D, G)$ is adjacent to exactly two gray edges and two black edges.

Then, for each cycle of alternating edge color (just called cycle in the rest of this paper) in $\mathcal{CB}(D, G)$, there is a labeling of D giving rise to a corresponding cycle in $\mathcal{B}(G, D)$. This observation leads to a greedy approach for labeling the genome D, or equivalently completing the partial graph $\mathcal{B}(G)$. Formally, a *completed graph* $\mathcal{B}(G, D)$ is a graph obtained from $\mathcal{B}(G)$ by adding gray edges such that each vertex of $\mathcal{B}(G, D)$ is adjacent to exactly 2 edges (one black and one gray), and such that the set of gray edges represent a given labeling of genome D (Figure 2.(b)).

The general idea of Algorithm Complete-Graph(G,D) given in Figure 3 is: at each step, pick a cycle of minimum size from $\mathcal{CB}(D, G)$, construct the corresponding cycle in $\mathcal{B}(G)$, and then remove from $\mathcal{CB}(D, G)$ all used edges. The algorithm stops when the partial graph is completed.

```
Algorithm Complete-Graph(G,D)
1.   For CSize = 1 to n Do ;
2.       For CVertex = b_1^l to b_n^l Do
3.           If CB(D,G) is empty (i.e. no edges left)
4.               Return ;
5.           If there is a cycle C_CB of size CSize beginning at CVertex Then
6.               Construct the corresponding cycle C_B in B(G);
7.               Remove from CB(D,G) all edges of C_CB;
8.           End If
9.       End For
10.  End For
```

Fig. 3. A greedy approach for completing the partial graph $\mathcal{B}(G)$ with gray edges representing the genome D. Here, $n = |\Sigma|$ is the number of different genes, and $b_1, b_2, \ldots b_n$ is a left-to-right ordering of the black edges of $\mathcal{CB}(D, G)$. For each i, b_i^l is the vertex representing the left adjacency of b_i. The size of a cycle is the number of black (or equivalently gray) edges of the cycle.

The following proposition immediately follows from the fact that all gray edges of $\mathcal{CB}(D, G)$ are placed in $\mathcal{B}(G)$.

Proposition 1. *Given a circular RD genome G and a circular duplicated genome D, the output of* Algorithm Complete-Graph (G,D) *is a completed graph $\mathcal{B}(G, D)$.*

As each vertex is adjacent to two black edges, finding a cycle of size k beginning at a given vertex of $\mathcal{CB}(D, G)$ (line 5) can be done in $O(2^k)$ time. Therefore,

the algorithm has a worst running-time complexity bounded by $\Sigma_{k=1}^{n} n.2^k$, which is not better than the exhaustive approach in $O(n.2^n)$. However, as demonstrated in the experimental part of this paper, it is actually a much faster approach in practice. This is due to the edge removal step (line 7), which allows to reduce the graph quickly, and to stop the process after a small number of iterations.

4.2 Multichromosomal genomes

In the case of G and D being multichromosomal genomes, we define the contracted breakpoint graph $\mathcal{CB}(D, G)$ as before, except that it contains an additional vertex O such that any endpoint vertex in D is connected to O by two gray edges, and any endpoint vertex in G is connected to O by a black edge. It follows that, except O that is adjacent to $2N_G$ black edges and $2N_D$ gray edges, N_G being the number of chromosomes of G, and N_D the number of chromosomes of D, each other vertex is adjacent to exactly two gray edges and two black edges.

Algorithm Complete-Graph(G,D) can be used in the case of multichromosomal genomes if we replace line 3 with " **If** $\mathcal{CB}(D, G)$ is acyclic". The output of the algorithm is then an acyclic partially completed graph, where the only remaining paths connect two vertices that are both endpoints of G, or both endpoints of D. Indeed, if there is a remaining path that connects two endpoints of different genomes, $\mathcal{CB}(D, G)$ is not acyclic. This path is in fact a cycle since it starts from O with an edge type and ends at O with a different edge type. Then, to complete the graph $\mathcal{B}(G)$, it suffices to add the remaining paths of $\mathcal{CB}(D, G)$.

Due to the $2(N_G + N_D)$ edges incident to O, the worst-time complexity is the one for circular genomes multiplied by $N_G.N_D$, i.e. $O(n.N_G.N_D.2^n)$. Hopefully in practice, n is not a tight upper bound as exploration eventually stops for much smaller cycle sizes.

5 Results

Since the generalization of the genome halving problem to a present-day RDL genome has been proven to be an exact algorithm executing in linear time, we only test the performance of the proposed method to compute the double distance. We generated datasets through simulated evolutions between a duplicated genome D and an RD genome G for both circular and multichromosomal genomes, as follows.

Simulated datasets: We first determine n, the number of genes, and N, the number of chromosomes in D. Then, we generate D, and a series of rearrangement events are performed on D to obtain G. Those are simply the rearrangements allowed by our model, namely inversions only in the case of circular genomes or inversions and translocations (including fusions and fissions) in the case of multichromosomal genomes. The number of events, μ, is a parameter chosen prior to the data generation, and the size of each rearrangement is chosen randomly. As for the rates of rearrangement operations, we chose (Inv : Trans : Fus+Fiss)

$= (5 : 4 : 1)$ to follow the rates reported for a lineage where a WGD occurred [11].

In order to validate the distances obtained with our greedy approach, we use an exact algorithm described below.

Exact algorithm: Let L (*resp.* L^*) be a complete (*resp.* partial) labeling of the gene copies of D, and $\mathcal{B}(G, D_{L^*})$ the breakpoint graph where the only defined gray edges are those adjacent to the genes of L^*. The idea is to compute a lower bound for $d_R(G, D)$ as we progressively construct L^*. More precisely, if at one step we have c cycles and p paths in $\mathcal{B}(G, D_{L^*})$, we know that the number of cycles in $\mathcal{B}(G, D_L)$ will be at most equal to $c + p$. Thus it is possible to use the following lower bound in a branch and bound strategy: $d_R(G, D) \geq n - c - p$.

Due to the high running-time complexity of the exact method, validation with the exact distance can only be done for "simple" datasets obtained with a low number of genes, and a low number of rearrangements. For datasets that were too complex for the exact algorithm, we estimated the accuracy of our greedy algorithm for the double distance by comparing the inferred distance with the number of rearrangements performed between D and G in the simulated evolution.

5.1 Time efficiency

Since the running-time complexity is function of n for the exact approach, we generated genomes containing different number of genes to evaluate the time efficiency of our greedy heuristic. For the exact method, n varies from 10 to 100, with an increment of 10. The parameters μ and N are arbitrarily fixed to 15 and 4 respectively. For the greedy heuristic, n varies from 100 to 1000 with an increment of 100. With μ fixed to 15, the running-time of the heuristic does not vary (below 0.001 seconds for all values of n). Thus, the number of rearrangements has been changed to $\mu = n$ in order to see a variation in the running-time. For each of those n values, multiple datasets were generated and the running time was averaged.

We can clearly observe the exponential running-time of the exact approach when the number of genes increases (see Figure 4 left). In contrast, our greedy algorithm is less limited by the genome size and more by the number of rearrangements. In Figure 4 right, we can see that even for datasets with a high number of rearrangements ($\mu = n$), the running-time is less than the anticipated worst-time complexity and remains under, or close to 1 second.

5.2 Heuristic accuracy

Comparison with the exact approach. We now test whether our greedy heuristic infers an accurate rearrangement distance by comparing its results against those of the exact approach. Recall that because of the high running-time complexity of the exact approach, we can only perform this algorithm on simple datasets

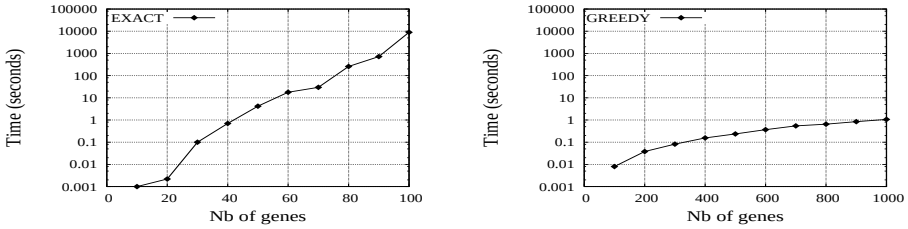

Fig. 4. Left: Running-time of the exact algorithm computing the double distance between D and G with various number of genes and a fixed number of rearrangements (μ=15). Right: Running-time of the heuristic approach to compute the double distance with various number of genes and rearrangements ($\mu = n$).

exhibiting low numbers of genes and rearrangements. The genomes are generated with n fixed to 25, N to 4 and μ varying from 0 to 50 by increments of 5. For each value of μ, 500 datasets were simulated. The error rate is the proportion of datasets for which the exact method found a more accurate distance than our greedy algorithm. Results are averaged over all datasets showing a comparable number of rearrangement events.

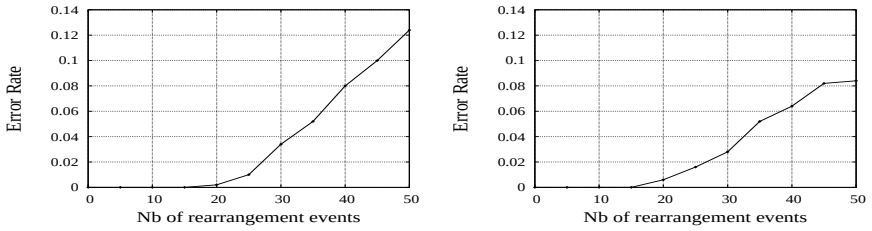

Fig. 5. Comparison of the greedy heuristic with the exact approach, showing the error rate of the inferred rearrangement distance for circular (Left) and multichromosomal genomes (Right).

As observed in Figure 5, the error rate of the greedy approach is close to 0 when the number of rearrangements is less than 25. Notice that the distance inferred by the greedy algorithm is in average really close to the optimal distance for both types of genomes (circular and multichromosomal). Moreover, when the distance is not the same, it differs only by 1 rearrangement. Naturally, the error rate of the greedy approach is more apparent when the number of rearrangements increases. This behavior is due to the fact that when a high number of rearrangements is performed, different cycles of equal size can be selected and a choice must be made affecting the remaining set of cycles. As stated before, in

this experiment we seek to optimize the rearrangement distance, but we obtain similar results if we seek to optimize the DCJ distance (results not shown).

Complex datasets. As a final experiment, simulations were performed with $n = 1000$, $N = 8$ and μ varying from 0 to 1000. The distances obtained with our greedy approach are compared with μ. Results shown in Figure 6 demonstrate that our method infers distances close to the number of rearrangement events performed on the original genome (for circular and multichromosomal genomes). However, when the number of rearrangement events increases, our approach underestimates that value. As in the comparison with the exact approach, the results are similar with the DCJ distance (not shown).

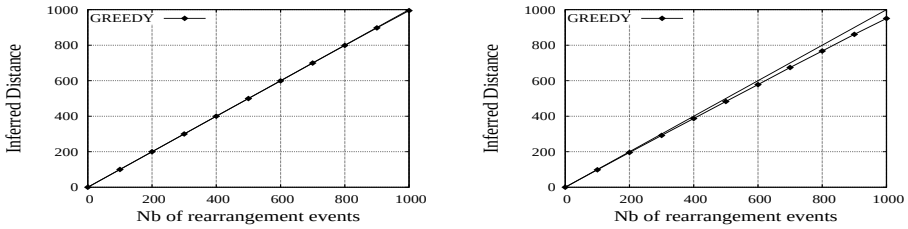

Fig. 6. Inferred rearrangement distances with complex datasets for circular genomes (Left) and multichromosomal genomes (Right).

6 Conclusion

We presented a linear time algorithm to solve the genome halving problem for genomes with missing gene copies. We also presented a greedy heuristic to compute the distance between an RD genome and a duplicated genome for the rearrangement and DCJ distances. Our experiments on simulated datasets showed that our greedy approach is time-efficient and accurate.

The proposed heuristic for the double distance could be easily generalized to compute the distance between two RD genomes. Moreover, it could be adapted to genomes that have undergone more than one WGD, thus increasing the running-time complexity as the number of possible labelings for a gene would increase. Our algorithm could then be used for the rearrangement phylogeny problem with genomes that have evolved through one or more whole genome duplications (without gene losses). Indeed, this method would allow to compute distances efficiently on all branches of such a phylogeny and consequently, an algorithm for the median problem could be used on the tree.

However, there is evidence of massive gene losses in lineages that have undergone a whole genome duplication event [11]. Thus, another interesting future work will concern the generalization of Algorithm Complete-Graph(G,D) to an

RDL genome G. Note that the approach of Algorithm dedouble-RDL can not be used directly (i.e. removing the singleton genes from G and the corresponding copies in D, and reinserting them after having completed the breakpoint graph) because of the constraints imposed by the duplicated genome D.

References

1. M.A. Alekseyev and P.A. Pevzner. Colored de bruijn graphs and the genome halving problem. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 4(1):98 – 107, 2007.
2. M.A. Alekseyev and P.A. Pevzner. Whole genome duplications, multi-break rearrangements, and genome halving problem. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 665 – 679, 2007.
3. D.A. Bader, B.M.E Moret, and M. Yan. A linear-time algorithm for computing inversion distance between signed permutations with an experimental study. *Journal of Computational Biology*, 8:483 – 491, 2001.
4. A. Bergeron, J. Mixtacki, and J. Stoye. Reversal distance without hurdles and fortresses. In *Combinatorial Pattern Matching, LNCS*, volume 3109, pages 388 – 399, 2004.
5. A. Bergeron, J. Mixtacki, and J. Stoye. A unifying view of genome rearrangements. In *Algorithms in Bioinformatics, LNCS*, volume 4175 of *WABI*, pages 163 – 173, 2006.
6. A. Bergeron, J. Mixtacki, and J. Stoye. A new linear time algorithm to compute the genomic distance via the double cut and join distance. *Theoretical Computer Science*, 410(51):5300 – 5316, 2009.
7. M. Blanchette, T. Kunisawa, and D. Sankoff. Gene order breakpoint evidence in animal mitochondrial phylogeny. *J.Mol.Evol.*, 49:193 – 203, 1999.
8. G. Bourque and P.A. Pevzner. Genome-scale evolution: Reconstructing gene orders in the ancestral species. *Genome Research*, 12:26 – 36, 2002.
9. N. El-Mabrouk and D. Sankoff. The reconstruction of doubled genomes. *SIAM Journal on Computing*, 32(1):754 – 792, 2003.
10. H. Gavranović and E. Tannier. Guided genome halving: probably optimal solutions provide good insights into the preduplication ancestral genome of Saccharomyces cerevisiae. In *Pacific Symposium on Biocomputing*, volume 15, pages 21 – 30, 2010.
11. J.L. Gordon, K.P. Byrne, and K.H. Wolfe. Additions, losses, and rearrangements on the evolutionary route from a reconstructed ancestor to the modern saccharomyces cerevisiae genome. *PloS Genetics*, 5(5):e1000485, 2009.
12. S. Hannenhalli. Polynomial-time algorithm for computing translocation distance between genomes. In *LNCS*, volume 937, pages 162 – 176, 1995.
13. S. Hannenhalli and P. A. Pevzner. Transforming cabbage into turnip (polynomial algorithm for sorting signed permutations by reversals). *J. ACM*, 48:1 – 27, 1999.
14. S. Hannenhalli and P.A. Pevzner. Transforming men into mice. In *Proceedings of the IEEE 36th Annual Symposium on Foundations of Computer Science*, pages 581 – 592, 1995.
15. B. Moret, L. Wang, T. Warnow, and S. Wyman. New approaches for reconstructing phylogenies from gene order data. In *ISMB-2001*, pages 165 – 173, 2001.
16. B.M.E. Moret, J. Tang, and T. Warnow. *Reconstructing phylogenies from gene-content and gene-order data*, pages 321 – 352. Mathematics of Evolution and Phylogeny. Oxford Univ. Press, 2005.

17. J.L. Boore P. Dehal. Two rounds of whole genome duplication in the ancestral vertebrate. *Plos Biology*, 3(10):e314, 2005.
18. J. Salse, S. Bolot, M. Throude, V. Jouffe, B. Piegu, U.M. Quraishi, T. Calcagno, R. Cooke, M. Delseny, and C. Feuillet. Identification and characterization of shared duplications between rice and wheat provide new insight into grass genome evolution. *The Plant Cell*, 20:11 – 24, 2008.
19. D. Sankoff. Reconstructing the history of yeast genomes. *PLOS Genetics*, 5(5), 2009.
20. E. Tannier, C. Zheng, and D. Sankoff. Multichromosomal median and halving problems under different genomic distances. *BMC Bioinformatics*, 10(120), 2009.
21. G. Tesler. Efficient algorithms for multichromosomal genome rearrangements. *Journal of Computer and System Sciences*, 65:587 – 609, 2002.
22. J.H. Postlethwait *et al.* Vertebrate genome evolution and the zebrafish gene map. *Nature Genetics*, 18:345 – 349, 1998.
23. R.H. Xu *et al.* Differential regulation of neurogenesis by the two xenopus gata-1 genes. *Molecular and Cellular Biology*, 17:436 – 443, 1997.
24. S. Yancopoulos, O. Attie, and R. Friedberg. Efficient sorting of genomic permutations by translocation, inversion and block interchange. *Bioinformatics*, 21:3340 – 3346, 2005.
25. C. Zheng, Q. Zhu, and D. Sankoff. Descendants of whole genome duplication within gene order phylogeny. *Journal of Computational Biology*, 15(8):947 – 964, 2008.