Comparing Genomes with Duplications: A Computational Complexity Point of View

Guillaume Blin, Cedric Chauve, Guillaume Fertin, Romeo Rizzi, and Stéphane Vialette

Abstract—In this paper, we are interested in the computational complexity of computing (dis)similarity measures between two genomes when they contain duplicated genes or genomic markers, a problem that happens frequently when comparing whole nuclear genomes. Recently, several methods [1], [2] have been proposed that are based on two steps to compute a given (dis)similarity measure M between two genomes G_1 and G_2: First, one establishes a one-to-one correspondence between the genes of G_1 and the genes of G_2; second, once this correspondence is established, it explicitly defines a permutation and it is then possible to quantify their similarity using classical measures defined for permutations like the number of breakpoints. Hence, these methods rely on two elements: a way to establish a one-to-one correspondence between genes of a pair of genomes and a (dis)similarity measure for permutations. The problem is then, given a (dis)similarity measure for permutations, compute a correspondence that defines an optimal permutation for this measure. We are interested here in two models to compute a one-to-one correspondence: the *exemplar* model, where all but one copy is deleted in both genomes for each gene family, and the *matching* model, which computes a maximal correspondence for each gene family. We show that, for these two models and for three (dis)similarity measures on permutations, namely, the number of common intervals, the maximum adjacency disruption (MAD) number, and the summed adjacency disruption (SAD) number, the problem of computing an optimal correspondence is NP-complete and even APX-hard for the MAD number and the SAD number.

Index Terms—Comparative genomics, computational complexity, common intervals, maximum adjacency disruption number, summed adjacency disruption number.

1 Introduction

The comparison of whole genomes from the gene order point of view has been a very active research domain since the early 1990s. In this context, genomes are modeled by sequences of integers, each integer representing a single gene or a genomic marker.[1] In phylogeny reconstruction, the main problem is thus to compute a (dis)similarity measure between the corresponding integer sequences which approximates the true evolutionary distance between these genomes (see, for instance, [3] for one of the first papers using this approach and [4] for a recent application to vertebrate genomes). Most of the mathematical models developed to compute such (dis)similarity measures are based on the assumption that a given integer appears exactly once in each considered genome. The rationale of this approach is that genomes are thus simply represented by permutations. However, aside some particular cases such as mitochondrial genomes [3], due to several evolutionary mechanisms (duplication/loss or whole genomes duplications [5]), duplicated genes are very common in genomes. As a result, real data cannot be naturally modeled by permutations.

The first way to overcome such a limitation is to consider genomes at a higher scale than genes, for example, *synteny blocks* [4]. However, if one wants to stay at the level of genes or, more generally, short genomic markers, one has to deal with the fact that genomes are modeled by sequences of integers where some integers may appear more than once in a given genome. Such genes that appear at several occurrences are said to belong to *nontrivial gene families*. Two genes represented by the same integer are said to have the same *label*. Recently, a new two-step permutation-based approach has been proposed for computing (dis)similarity measures between genomes. The first step consists of transforming the two sequences into a single permutation P by establishing a one-to-one correspondence between pairs of genes having the same label (and then, by resorting to renaming procedure, we can always assume that one of the two permutations is the identity permutation, see Section 2). In the second step, a permutation-based (dis)similarity measure is computed from the permutation P. The main line of research following this approach seeks the permutation P that *optimizes* the (dis)similarity measure. The classical criterion retained to define the optimal (dis)similarity measure is the parsimony criterion: One tries to compute the permutation P that induces the maximal (respectively,

1. From now on, we use only the word gene, without loss of generality.

- *G. Blin is with the IGM-LabInfo, UMR CNRS 8049, Université de Marne-la-Vallée, 77454 Marne-la-Vallée Cedex 2, France. E-mail: gblin@univ-mlv.fr.*
- *C. Chauve is with the Department of Mathematics, Simon Fraser University, 8888 University Drive, V5A 1S6, Burnaby (BX), Canada. E-mail: cedric.chauve@sfu.ca.*
- *G. Fertin is with the Laboratoire d'Informatique de Nantes-Atlantique (LINA), FRE CNRS 2729, Université de Nantes, 2 rue de la Houssinière, 44322 Nantes Cedex 3, France. E-mail: fertin@lina.univ-nantes.fr.*
- *R. Rizzi is with the Dipartimento di Matematica e Informatica, Università di Udine, Italy. E-mail: Romeo.Rizzi@simi.uniud.it.*
- *S. Vialette is with the Laboratoire de Recherche en Informatique (LRI), UMR CNRS 8623, Faculté de Sciences d'Orsay, Université Paris-Sud, 91405 Orsay, France. E-mail: vialette@lri.fr.*

Manuscript received 12 July 2006; revised 28 Nov. 2006; accepted 17 Jan. 2007; published online 5 Mar. 2007.
For information on obtaining reprints of this article, please send e-mail to: tcbb@computer.org, and reference IEEECS Log Number TCBBSI-0137-0706. Digital Object Identifier no. 10.1109/TCBB.2007.1069.

minimal) similarity (respectively, dissimilarity) measure. Note, however, that there exist other methods that are based on the principle of transforming a pair of integer sequences into a permutation but do not aim at optimizing a (dis)similarity measure for the resulting permutation (see [6], [7], [8] for example).

There are two main approaches for computing a one-to-one correspondence between two integer sequences. In the *exemplar* model, introduced by Sankoff [1], for every nontrivial gene family, all but one copy in each genome is deleted. The pair of genes that is conserved for each family is called a pair of *ancestral homologs* as the goal of the exemplar method is to find the pair of genes which best reflects the original position of the ancestral gene in the common ancestor genome. The *matching* model is more general as it allows us to conserve more than one copy of a gene family and seeks a maximal one-to-one correspondence between these copies [2]. Several distances have been considered under the exemplar and matching models which are either based on minimizing the number of evolutionary events that allow us to transform a genome into the other, for events like reversals[2] [1], [9], [10], [11], [12], [13], reversals and insertions and deletions [14], [15], reversals and translocations [16], or on maximizing a similarity measure based on conserved structure in permutations like the number of adjacencies (which is equivalent to minimizing the number of breakpoints) [1], [9], [12], [13], [17] or the number of conserved intervals [18], [19], [20], [21]. As far as we know, none of the above problems has been shown to be solvable in polynomial time and, in fact, most of them have been shown to be **NP**-complete as soon as duplicates are present in the genomes (see Tables 1 and 2, in Section 6).

In this paper, we present new results on the algorithmic complexity of computing different (dis)similarity measures between pairs of genomes that contain duplicates. We describe results for three (dis)similarity measures, namely, number of common intervals, Maximum Adjacency Disruption (MAD) number, and Summed Adjacency Disruption number (SAD), which will be defined in Section 2. In Section 3, we focus on the problem of computing the number of common intervals in genomes containing duplicates and show that the problem is **NP**-complete in both the matching and exemplar models. In Sections 4 and 5, we prove that, under both models, both the MAD and SAD problems are **APX**-hard when genomes contain duplicates.

2 Preliminaries

In this section, we precisely define the three similarity measures we are interested in, together with the exemplar and matching models. As mentioned in the introduction, the three considered measures (number of common intervals, MAD, and SAD) are defined for duplication-free genomes only and, hence, one has to first disambiguate the data by inferring homologs, that is, a nonambiguous mapping between the genes of the two genomes.

We need some notations. A *genome* is a sequence of unsigned integers. Let G be a genome of size n. As mentioned above, a *gene family* is any integer that occurs in G, regardless of its number of occurrences. A *gene* is an occurrence of a gene family in G and we denote by $G[i]$ the gene that occurs at position i in G. Let $occ(G, g)$ denote the maximum number of occurrences of a gene g in genome G and let $occ(G)$ be the maximum of $occ(G, g)$ over all genes g in G. The genome G is said to be *duplication-free* if $occ(G) = 1$. Let G_1 and G_2 be two genomes. A *matching* $\mathcal{M}$ between G_1 and G_2 is a set of pairwise disjoint pairs $\mathcal{M} = \{(i_1, j_1), (i_2, j_2), \ldots, (i_k, j_k)\}$ such that $G_1[i_\ell] = G_2[j_\ell]$, for all $1 \leq \ell \leq k$. A *maximum matching* between G_1 and G_2 is a matching of maximum cardinality. Suppose that G is duplication-free, let i and j be such that $1 \leq i < j \leq n$ and write $a = G[i]$ and $b = G[j]$. The *distance* between a and b in G, written $\mathrm{Dist}(G, a, b)$, is defined by $\mathrm{Dist}(G, a, b) = |j - i|$.

Given two genomes containing duplications, the first step is thus to establish a nonambiguous mapping between the genes of the two genomes. In the *exemplar* model, for all gene families, all but one occurrence in each genome is deleted. In other words, we are looking for a matching $\mathcal{M} = \{(i_1, j_1), (i_2, j_2), \ldots, (i_k, j_k)\}$ between G_1 and G_2 such that 1) $G_1[i_\ell] \neq G_1[i_{\ell'}]$, for all $1 \leq \ell < \ell' \leq k$, and 2) each gene family occurs in one pair of $\mathcal{M}$. In the *matching* model, the goal is to map as many genes as possible, that is, to find a maximum matching between G_1 and G_2. The rationale of this preliminary step is that we may now assume that the two genomes are duplication-free. Indeed, suppose the first step results in the matching $\mathcal{M}$, we thus modify the two genomes G_1 and G_2 as follows:

1. we delete all genes in G_1 and G_2 that are not part of the matching $\mathcal{M}$ and
2. we rename the genes of G_1 and G_2 according to the index of the associated pair in $\mathcal{M}$.

Observe that the resulting genomes are both of size $|\mathcal{M}|$. According to the above (for both the exemplar and the matching models) discussion, if a gene family occurs in one genome but not in the other, then all occurrences of this gene family will be deleted in the end. Therefore, we may thus assume in the sequel that any gene family of G_1 is a gene family of G_2 and vice versa.

We now turn to precisely defining the three similarity measures in which we are interested. As mentioned before, we assume now that the two genomes are duplication-free, that is, both G_1 and G_2 are permutations of size n. Moreover, for convenience, by first resorting to an easy renaming procedure, we can always assume that one of the two genomes, say, G_1, is the identity permutation, that is, $G_1 = 1\ 2\ \ldots\ n$.

Number of common intervals. A *common interval* between G_1 and G_2 is a substring of G_1, that is, a consecutive sequence of genes of G_1, for which exactly the same content can be found in a substring of G_2. For example, if $G_1 = 1\,2\,3\,4\,5$ and $G_2 = 1\,4\,3\,5\,2$, then 1, 2, 3, 4, 5, 3 4, 3 4 5, 2 3 4 5, and 1 2 3 4 5 are common intervals. Notice that there exist at least $n + 1$ common intervals between G_1 and G_2 since each individual gene is always a common interval and G_1 itself is

2. The reversal model considers *signed permutations*, where each element has a sign, positive or negative, that indicates which strand on the genome is the corresponding gene located. However, the three (dis)similarity measures we consider in this paper do not take signs into account and, thus, we do not discuss signed permutations here.

also a common interval. This lower bound is tight, as shown by $G_1 = 1\,2\,3\,4$ and $G_2 = 2\,4\,1\,3$. Furthermore, if $G_1 = G_2$, the number of common intervals between G_1 and G_2 is $\frac{n(n+1)}{2}$, that is, each possible substring of G_1 is a common interval.

Maximum Adjacency Disruption (MAD) Number. This notion was introduced by Sankoff and Haque [22]. The MAD number between G_1 and G_2, denoted $\mathrm{MAD}(G_1, G_2)$, is defined by

$$\mathrm{MAD}(G_1, G_2) = \max\{\mathcal{M}_1, \mathcal{M}_2\},$$

where $\mathcal{M}_1 = \max\{\mathrm{Dist}(G_2, G_1[i], G_1[i+1]) : 1 \leq i \leq n-1\}$ and $\mathcal{M}_2 = \max\{\mathrm{Dist}(G_1, G_2[i], G_2[i+1]) : 1 \leq i \leq n-1\}$.

The rationale of this double maximization measure lies in the fact that, in general, $\mathcal{M}_1 \neq \mathcal{M}_2$. For instance, if $G_1 = 1\,2\,3\,4\,5$ and $G_2 = 1\,4\,3\,5\,2$, then $\mathcal{M}_1 = 4$ and $\mathcal{M}_2 = 3$ and, hence, $\mathrm{MAD}(G_1, G_2) = \max\{4, 3\} = 4$.

Summed Adjacency Disruption (SAD) Number. This notion was also introduced by Sankoff and Haque [22] and can be seen as a global variant of the MAD number. The SAD number between G_1 and G_2, denoted $\mathrm{SAD}(G_1, G_2)$, is defined by

$$\mathrm{SAD}(G_1, G_2) = \sum_{1 \leq i \leq n-1} \mathrm{Dist}(G_2, G_1[i], G_1[i+1]) + \sum_{1 \leq i \leq n-1} \mathrm{Dist}(G_1, G_2[i], G_2[i+1]).$$

Going back to our example, $G_1 = 1\,2\,3\,4\,5$ and $G_2 = 1\,4\,3\,5\,2$, one obtains

$$\mathrm{SAD}(G_1, G_2) = (4 + 2 + 1 + 2) + (3 + 1 + 2 + 3) = 18.$$

Of particular importance from a computational complexity point of view, we observe that the MAD and SAD numbers are *dissimilarity* measures, that is, the associated optimization problem is a minimization one; on the contrary, the number of common intervals is a *similarity* measure, that is, the associated optimization problem is a maximization one.

3 Number of Common Intervals

In this section, we investigate the algorithmic complexity of computing the number of common intervals between two genomes in both the exemplar and matching models. Let ECOMI (respectively, MCOMI) denote the problem of computing the maximum number of common intervals in the exemplar (respectively, matching) model. We show that both ECOMI and MCOMI are **NP**-complete, even for restricted instances. The proof we give below is valid for both models since it shows **NP**-completeness in the case where $occ(G_1) = 1$. However, in order to simplify notations, we will mention in the proof only the exemplar model (that is, the ECOMI problem). The proof is made by reduction from VERTEXCOVER. Starting from any instance of VERTEXCOVER (that is, a graph $G = (V, E)$ with $V = \{v_1, v_2 \ldots v_n\}$ and $E = \{e_1, e_2 \ldots e_m\}$), we will first describe a polynomial-time construction of two genomes, G_1 and G_2, such that $occ(G_1) = 1$ and $occ(G_2) = 2$. We first describe G_1:

$$G_1 = b_1, b_2 \ldots b_m, x, a_1, C_1, a_2, C_2 \ldots a_n, C_n, y, b_{m+n}, b_{m+n-1} \ldots b_{m+1}.$$

The a_is, the b_is, x, and y are genes, whereas C_is are sequences of genes. They are defined as follows:

- for any $1 \leq i \leq n$, $a_i = 2(i-1)m + i$;
- for any $1 \leq i \leq n$, $C_i = (a_i + 1), (a_i + 2) \ldots (a_i + 2m)$;
- for any $1 \leq i \leq n + m$, $b_i = a_n + 2m + i$;
- $x = b_{n+m} + 1$; and
- $y = b_{n+m} + 2$.

It can be seen that no gene appears more than once in G_1, thus $occ(G_1) = 1$. Now, we describe the construction of G_2:

$$G_2 = y, a_1, D'_1, b_{m+1}, a_2, D'_2, b_{m+2} \ldots a_{n-1}, D'_{n-1}, b_{m+n-1}, a_n, D'_n, b_{m+n}, x.$$

The duplicated genes in G_2 are $b_1, b_2 \ldots b_n$ and are spread within the D'_is. Moreover, each b_i, $1 \leq i \leq n$, will appear only twice in G_2. We now describe the contents of D'_i, $1 \leq i \leq n$. Basically, D'_i is constructed in two steps:

1. We first construct, for each i, a sequence of genes D_i, which is a specific shuffle of the contents of $C_i = (a_i + 1), (a_i + 2) \ldots (a_i + 2m)$. More precisely, let $\min = a_i + 1$ and $\max = a_i + 2m$, then
$$D_i = (a_i + 3), (a_i + 5) \ldots (a_i + 2m - 3), (a_i + 2m - 1), \min, \max, (a_i + 2), (a_i + 4) \ldots (a_i + 2m - 4), (a_i + 2m - 2).$$
2. For any $1 \leq i \leq n$, we obtain D'_i by adding some b_js $(1 \leq j \leq m)$ into D_i, according to the initial graph G we are given. More precisely, for any edge e_j that is incident to a vertex v_i in G, we add the gene b_j between the jth and the $(j+1)$th gene of D_i. This process gives us the D'_is.

Note that no two b_js $(1 \leq j \leq m)$ can appear contiguously in a D'_i and that no D'_i starts or ends with a b_j (all D'_is start and end with a gene that only appears in C_i in G_1). In the following, any interval of size one (that is, any singleton), as well as the whole genome, will be called a *trivial interval*.

Lemma 1. *For any exemplar genome G_2^E of G_2, the only nontrivial common intervals that occur between G_2^E and G_1 are necessarily taken in G_1 within the sequence a_iC_i, for any $1 \leq i \leq n$.*

Proof. We will first prove that, for any exemplar genome G_2^E obtained from G_2, any interval of size greater than or equal to 2 that contains x (respectively, y) also contains y (respectively, x) and thus corresponds to the whole genome. Suppose, indeed, that there is a common interval, different from a singleton, containing x and not y. Let us call this interval I_x. Now, let us look at what other genes I_x could contain in G_1:

- If I_x contains b_m in G_1, since b_m belongs to a D'_i in G_2^E, this means that I_x contains b_{m+n} in G_2^E and thus contains y in G_1, a contradiction.

- If I_x contains a_1 in G_1, I_x contains, in particular, b_{m+n} in G_2^E and thus contains y in G_1, a contradiction.

Hence, any common interval I_x that contains x also contains y. Now, suppose that a common interval I_y, different from a singleton, contains y and not x and let us look at what other genes I_y could contain in G_1:

- If I_y contains b_{m+n} in G_1, then it contains all of the D'_is in G_2^E and, in particular, it contains all of the b_js, $1 \leq j \leq m$. Thus, it contains x in G_1, a contradiction.
- If I_y contains $a_n + 2m$ in G_1, then it contains, in particular, b_{m+n-1} in G_2^E and thus contains b_{m+n} in G_1. We are now back to the previous case.

Hence, the only common interval containing x (respectively, y) is the whole genome G_1. Thus, if there are common intervals that are nontrivial, they must be, in G_1, strictly on the left of x, strictly between x and y, or strictly on the right of y. We will separately investigate these three cases:

1. Intervals strictly on the left of x in G_1. Since no two b_js, $1 \leq j \leq m$, are contiguous in G_2^E, any such interval would contain at least one gene in a given D'_i, which occurs only in C_i in G_1, a contradiction.
2. Intervals strictly on the right of y in G_1. Similarly, any such interval would contain an a_i in G_2^E, a contradiction.
3. Intervals strictly between x and y in G_1. Independent of the way G_2^E is exemplarized, we see that no common interval in G_1 can contain, at the same time, a_i and a_{i+1}, $1 \leq i \leq n-1$. Thus, the only possible common intervals between G_1 and G_2^E must be taken within a given substring of the form a_iC_i $(1 \leq i \leq n)$ in G_1 and the lemma is proven. □

Lemma 2. *For any given $1 \leq i \leq n$, let Δ_i be a subsequence of D'_i that does not contain any b_j. If $2 \leq |\Delta_i| \leq 2m-1$, then it is not a common interval.*

Proof. Let Δ_i be a subsequence of D'_i that does not contain any b_j and let $2 \leq |\Delta_i| \leq 2m-1$. By Lemma 1, Δ_i can only be a common interval with a substring of C_i, which, by construction, contains consecutive integers. Thus, since $|\Delta_i| \geq 2$, it must contain at least two consecutive integers. However, by construction, any two consecutive integers in D'_i are extremities of an interval that contains both the minimum value m and the maximum value M of D'_i. However, since, in C_i, m and M are the left and right extremities, Δ_i is at least as big as C_i. Since, by construction, $|C_i| = 2m$ and since we supposed that $|\Delta_i| \leq 2m-1$, this cannot happen. Hence, Δ_i is not a common interval. □

Lemma 3. *For any exemplar genome G_2^E of G_2 and for any $1 \leq i \leq n$, only two cases can occur:*

1. *In G_2^E, all of the b_js have been deleted from D'_i and, in that case, there are exactly two nontrivial common intervals involving D'_i.*
2. *In G_2^E, at least one b_j has been left within D'_i and, in that case, there is no nontrivial common interval involving D'_i.*

Proof. By Lemma 1, we know that any nontrivial interval is composed in G_1 of elements of the sequence a_iC_i, for any $1 \leq i \leq n$. Hence, it is composed, in any exemplar genome G_2^E obtained from G_2, of elements of the sequence $a_iD'_i$, for any $1 \leq i \leq n$.

Suppose first that all of the b_js in D'_i have been deleted in our exemplar genome G_2^E, thus transforming it into the exemplar subsequence D_i. By Lemma 1, we know that any nontrivial interval is composed in G_1 of elements of the sequence a_iC_i, for any $1 \leq i \leq n$. Hence, it is composed, in any exemplar genome G_2^E obtained from G_2, of elements of the sequence a_iD_i, for any $1 \leq i \leq n$. In that case, it can be easily seen that, for any $1 \leq i \leq n$:

1. interval a_iC_i in G_1 is a common interval to a_iD_i in G_2^E and
2. interval C_i in G_1 is a common interval to D_i in G_2^E.

Moreover, by Lemma 2, no strict subsequence Δ_i of D_i such that $2 \leq |\Delta_i| \leq |D_i| - 1$ is a common interval (we recall that $|D_i| = |C_i| = 2m$ by construction). Hence, if all of the b_js in D'_i have been deleted to obtain D_i, then only two common nontrivial intervals exist in G_2^E: a_iD_i (which is common with a_iC_i in G_1) and D_i (which is common with C_i in G_1).

Suppose now that at least one b_j in D'_i has not been deleted in G_2^E. First, we note that no nontrivial common interval can include b_j, since b_j does not appear in C_i. Hence, any possible nontrivial interval involving D'_i is a substring Δ_i of D'_i that does not contain any b_j. However, since no b_j is an extremity of D'_i, it implies that, necessarily, $|\Delta_i| \leq 2m-1$. However, by Lemma 2, we know that, in that case, Δ_i is not a common interval. □

Lemma 4. *Let G be a graph and G_1 and G_2 be the two genomes obtained by the construction described above. G admits a Vertex Cover VC such that $|VC| \leq k$ iff there exists an exemplar genome G_2^E obtained from G_2 having at least $\mathcal{I} = 2(n-k) + I_T$ common intervals, where I_T is the number of trivial common intervals.*

Proof. ($\Rightarrow$) Suppose there exists in G a Vertex Cover VC such that $|VC| = k' \leq k$. Let $VC = \{v_{i_1}, v_{i_2} \ldots v_{i_{k'}}\}$. In G_2, delete the b_js in the substrings D'_i for any $i \notin \{i_1, i_2 \ldots i_{k'}\}$. If, after doing this, there remain some b_js which appear twice, remove one copy of each arbitrarily. Since, in G_2, 1) only the b_js are duplicated, 2) each b_j occurs exactly twice in G_2, and 3) VC is a Vertex Cover of G, we conclude that, with those deletions, we end up with an exemplar genome G_2^E. In G_2^E, we have at least $(n-k)$ substrings of the form D'_i for which all of the b_js have been deleted. Thus, by Lemma 3, we know that they each imply two nontrivial common intervals, which sums up to at least $2(n-k)$. To those intervals, we add the trivial ones. Hence, on the whole,

we get at least $\mathcal{I} = 2(n-k) + I_T$ common intervals between G_1 and G_2^E.

($\Leftarrow$) Suppose there exists an exemplar genome G_2^E obtained from G_2 and having at least $\mathcal{I} = 2(n-k) + I_T$ common intervals. Then, there are at least $2(n-k)$ nontrivial common intervals. However, by Lemma 1, we know that they can only occur within the substrings a_iC_i, $1 \leq i \leq n$, in G_1, that is within the substrings a_iD_i', $1 \leq i \leq n$, in G_2^E. By Lemma 3, we know that, in at least $(n-k)$ such substrings, all of the b_js, $1 \leq j \leq m$, have been deleted. Since G_2^E is exemplar, this means that the b_js have remained in at most k substrings of the form a_iD_i'. By construction, each b_j has been included in a D_i' because the edge e_j is incident to the vertex v_i in the graph G. Since one copy of each b_j has remained in G_2^E and since they are included in at most k substrings of the form a_iD_i', we conclude that those substrings imply a Vertex Cover which is of at most size k in G. □

As a direct consequence of Lemma 4, we can say that the ECOMI problem is **NP**-complete. Moreover, as mentioned before, the proof and the result are also valid for the MCOMI problem since our construction implies that $occ(G_1) = 1$. We thus have the following theorem.

Theorem 1. *The ECOMI and MCOMI problems are both **NP**-complete, even when $occ(G_1) = 1$ and $occ(G_2) = 2$.*

We also consider, for the matching model, instances for which the constraints do not rely on the maximum number of duplicates per family but on the number of families that contain duplicates. With this restriction, we obtain the following result.

Theorem 2. *The MCOMI problem is **NP**-complete even when $f(G_1) = f(G_2) = 1$, where $f(G)$ denotes the number of different families of genes that contain duplicates in G.*

Proof. The proof is directly derived from the proof by Blin and Rizzi [18] in which the authors studied *conserved intervals*, a measure that is closely related to common intervals. More precisely, a conserved interval is a common interval for which the extremities are conserved [23]. Hence, any conserved interval is by definition a common interval, although the converse is not true in general. However, the construction given in [18] has the property that any common interval is in fact also a conserved interval. Hence, the reduction they provide is also valid for the MCOMI problem, and the result follows. □

4 Maximum Adjacency Disruption (MAD)

Let EMAD (respectively, MMAD) denote the problem of computing the minimum MAD number in the exemplar (respectively, matching) model. In this section, we prove inapproximability results for both the EMAD and MMAD problems. More precisely, we show that, for no $\varepsilon > 0$, EMAD (respectively, MMAD) admits a $(2-\varepsilon)$-approximation algorithm unless $\mathbf{P} = \mathbf{NP}$. This inapproximability result does not rely on the **PCP** theorem. We will also remark, however, how reconsidering the reduction proposed in view of the **APX**-hardness results based on the **PCP** theorem can one replace the constant 2 above with a strictly bigger constant. The proof is split into two: We first study the complexity of a restricted form of SAT, which we call UNIFORM-SAT, and, in particular, we show that it is **NP**-complete. Next, we show that a $(2-\varepsilon)$-approximation algorithm for EMAD (respectively, MMAD), for some $\varepsilon > 0$, would imply the existence of a polynomial time algorithm for UNIFORM-SAT. Finally, we obtain the inapproximability result for EMAD (respectively, MMAD).

In the following, 3SAT will denote the restriction of SAT for which each clause contains at most three literals. We introduce a restricted form of 3SAT called UNIFORM-SAT, as follows: An instance $\langle X, \mathcal{C} \rangle$ of 3SAT is an instance of UNIFORM-SAT when the following two conditions are met:

1. for each clause $C \in \mathcal{C}$, either all literals occurring in C are positive occurrences of variables from X or all literals occurring in C are negated occurrences of variables from X and
2. for each variable $x \in X$, x has at most three positive and at most two negated occurrences within $\mathcal{C}$.

A 3SAT formula $F = \bigwedge_{C \in \mathcal{C}} C$ is called 3-*bounded* if no variable has more than three occurrences within C and is called (2, 2)-*bounded* if no variable has more than two positive occurrences and no more than two negated occurrences within C. The following two facts are known:

1. the decision problem 3SAT is **NP**-complete even when restricted to 3-bounded formulas [24] and
2. the optimization problem MAX-3SAT is **APX**-hard even when restricted to 3-bounded formulas [25].

Since both problems admit a trivial self-reduction in case a variable has only positive (or only negated) occurrences, then the following two facts also hold:

1. 3SAT is **NP**-complete even when restricted to (2, 2)-bounded formulas and
2. MAX-3SAT is **APX**-hard even when restricted to (2, 2)-bounded formulas.

Notice that, of the above two results, only the second is related to the **PCP** theorem, whereas the first was known much before its appearance.

The following reduction links the complexity of UNIFORM-SAT to the complexity of (2, 2)-bounded 3SAT. Given a generic instance $\langle X, \mathcal{C} \rangle$ of (2, 2)-bounded 3SAT, where $X = \{x_1, x_2, \ldots, x_n\}$ and $\mathcal{C} = \{C_1, C_2, \ldots, C_m\}$, consider the instance $\langle Y, \mathcal{P} \rangle$ of UNIFORM-SAT, where $Y = \{y_i^j : i = 1, 2, \ldots, n, j = 0, 1, 2, 3\}$ and $\mathcal{P} = \mathcal{P}_{var} \cup \mathcal{P}_{cla}$, where

$$\mathcal{P}_{var} = \{(y_i^j \vee y_i^{j+1 \bmod 4}), (\neg y_i^j \vee \neg y_i^{j+1 \bmod 4}) : i = 1, 2, \ldots, n, j = 0, 1, 2, 3\}$$

and $\mathcal{P}_{cla} = \{P_1, P_2, \ldots, P_m\}$, where, for $j = 1, 2, \ldots, m$, the clause P_j is obtained from the clause C_j as follows: For each literal ℓ occurring in C_j and assuming that ℓ is the tth positive (or the tth negated) literal of variable x_i (with $i = 1, 2, \ldots, n$ and $t = 1, 2$) occurring within the clauses $C_1, C_2, \ldots, C_m$ as taken in this order, then the literal y_i^{2t-1} (respectively, the literal y_i^{2t-2}) is placed in the clause P_j. In practice, the clause P_j is a clause made only of positive literals which is meant to represent the original clause C_j.

At the same time, the all-positive or all-negated clauses in $\mathcal{P}_{var}$ are there to enforce the consistency of the truth values of the variables y_i^0, y_i^1, y_i^2, and y_i^3, which are meant to represent either the positive (y_i^1 and y_i^3) or the negated (y_i^0 and y_i^2) occurrences of variable x_i within $\mathcal{C}$.

The above is clearly a polynomial time reduction; besides, we have the following lemmas.

Lemma 5. *Let $t_X : X \mapsto \{0,1\}$ be a truth assignment over X which satisfies at least c of the clauses in $\mathcal{C}$. Then, there exists a truth assignment $t_Y : Y \mapsto \{0,1\}$ over Y which satisfies at least $c + 8n$ of the clauses in $\mathcal{P}$.*

Proof. Consider the assignment t_Y defined by $t_Y(y_i^1) := t_Y(y_i^3) := t_X(x_i)$ and by $t_Y(y_i^0) := t_Y(y_i^2) := \neg t_X(x_i)$. Note that each of the $8n$ clauses in $\mathcal{P}_{var}$ is satisfied under t_Y. Moreover, for each $j = 1, 2, \ldots, m$, the clause P_j is satisfied under t_Y if and only if the clause C_j is satisfied under t_X. □

Lemma 6. *Let $t_Y : Y \mapsto \{0,1\}$ be a truth assignment over Y which satisfies at least $c + 8n$ of the clauses in $\mathcal{P}$. Then, in polynomial time, we can derive from t_Y a truth assignment $t_X : X \mapsto \{0,1\}$ over X which satisfies at least c of the clauses in $\mathcal{C}$.*

Proof. Truth assignment t_Y is called *canonical* if for each $i = 1, 2, \ldots, n$, the truth values of the variables y_i^0, y_i^1, y_i^2, and y_i^3 are *consistent*, that is, when $t_Y(y_i^0) = t_Y(y_i^2) \neq t_Y(y_i^1) = t_Y(y_i^3)$. Notice that by possibly redefining at most two truth values among $t_Y(y_i^0)$, $t_Y(y_i^2)$, $t_Y(y_i^1)$, and $t_Y(y_i^3)$, we can always assume that t_Y is canonical. Indeed, at least two extra clauses from $\mathcal{P}_{var}$ are satisfied in restoring the consistency among the variables y_i^0, y_i^1, y_i^2, and y_i^3, while, at the same time, since at most two truth values have been affected, at most two clauses from $\mathcal{P}_{cla}$ may lose in satisfaction. In other words, we can make t_Y canonical by a majority vote on y_i^0, y_i^1, y_i^2, and y_i^3, for each $i = 1, 2, \ldots, n$, while preserving the fact that at least $c + 8n$ of the clauses in $\mathcal{P}$ are satisfied under t_Y. Once t_Y is canonical, the arguments spent within the proof of the previous lemma are fully reversible. □

The above two lemmas imply that UNIFORM-SAT is **NP**-complete.

Theorem 3. *Deciding whether a given UNIFORM-SAT formula is satisfiable is **NP**-complete.*

Theorem 3 here does not need the **PCP** theorem and is all that is required in the following for proving that, for no $\varepsilon > 0$, EMAD (respectively, MMAD) admits a $(2-\varepsilon)$-approximation algorithm unless $\mathbf{P} = \mathrm{NP}$. With dependence on **PCP**, Lemmas 5 and 6 also imply the following result, which, besides being of independent interest, can be used to show that the right constant for the approximability of EMAD (respectively, MMAD) is not 2.

Theorem 4. *Given a UNIFORM-SAT formula, the problem of finding a truth assignment maximizing the number of satisfied clauses is **APX**-hard.*

Proof. We will proceed as follows: Assume we are given a $(1-\varepsilon)$-approximation algorithm A for UNIFORM-SAT and design a $(1 - 25\,\varepsilon)$-approximation algorithm for a (2, 2)-bounded 3SAT, which rests on algorithm A as a subroutine. The **APX**-hardness of UNIFORM-SAT then follows from the **APX**-hardness of a (2, 2)-bounded 3SAT.

After receiving an instance $\langle X, \mathcal{C} \rangle$ of (2, 2)-bounded 3SAT in the input, we construct the instance $\langle Y, \mathcal{P} \rangle$ of UNIFORM-SAT, as described above. Assume that the optimal truth assignment $t_{X,opt}$ for $\langle X, \mathcal{C} \rangle$ satisfies at least *opt* clauses in $\mathcal{C}$. Clearly, $opt \geq \frac{n}{3}$ since there clearly exists a truth assignment under which, for each variable $x \in X$, at least one of the occurrences of x in $\mathcal{C}$ belongs to a satisfied clause since each clause contains at most three literals. Moreover, by Lemma 5, there exists a truth assignment $t_{Y,opt}$ over Y satisfying at least $8n + opt$ clauses in $\mathcal{P}$. By running algorithm A, we are hence guaranteed to find a truth assignment $t_{Y,apx}$ over Y satisfying at least $(8\,n + opt)(1-\varepsilon)$ clauses in $\mathcal{P}$. Moreover, Lemma 6 (whose proof can be easily converted into a polynomial time algorithm) shows how, starting from this truth assignment $t_{Y,apx}$, can one obtain a truth assignment $t_{X,apx}$ over X such that the clauses in $\mathcal{C}$ that are satisfied under $t_{X,apx}$ are at least

$$(8n + opt)(1-\varepsilon) - 8n \geq opt - \varepsilon\, opt - 8\,\varepsilon\, n$$
$$\geq opt - \varepsilon\, opt - 24\,\varepsilon\, opt \geq (1 - 25\varepsilon)\, opt.$$

□

We now prove that both the EMAD and MMAD problems are **APX**-hard. The result holds for both problems since we prove it in the case where $occ(G_1) = 1$, where they coincide. The result rests on a reduction from UNIFORM-SAT. Assume we are given an instance $\langle X, \mathcal{C} \rangle$ of UNIFORM-SAT, where $X = \{x_1, x_2, \ldots, x_n\}$. Here, $\mathcal{C}$ can be partitioned into the family $\mathcal{P} = \{P_1, P_2, \ldots, P_{m_p}\}$ of clauses comprised of only positive literals and the family $\mathcal{N} = \{N_1, N_2, \ldots, N_{m_n}\}$ of clauses comprised of only negated literals. Let M_ε be a sufficiently big positive integer that we will later fix in order to force our conclusions. Let us now detail the construction of the two genomes G_1 and G_2 from any instance of the UNIFORM-SAT problem. Here, G_1 is the simple (that is, duplication-free) genome G_1 of length $L_1 = 2\,M_\varepsilon + m_p + m_n + n - 1$, defined as follows:

$$G_1 = 1\ 2\ 3 \ldots L_1.$$

A gene at position i in G_1 with $i \leq m_p$ or $i \geq L_1 - m_n + 1$ is called an *$*$-gene*. Genome G_2 has length $L_2 = 2\,M_\varepsilon + 6\,n - 1$ and conforms to the following pattern, where we have found it convenient and pertinent to spot out the displacement of the $*$-genes within genome G_2:

$$G_2 = m_p + 1, m_p + 2, \ldots, m_p + M_\varepsilon, *, *, *, *, *, m_p + M_\varepsilon + 1,$$
$$*, *, *, *, *, m_p + M_\varepsilon + 2, \ldots\ldots, *, *, *, *, *,$$
$$m_p + M_\varepsilon + n, m_p + M_\varepsilon + n + 1, m_p + M_\varepsilon + n + 2,$$
$$\ldots, m_p + 2\,M_\varepsilon + n - 1.$$

We will specify later the precise identity of the $*$-genes within genome G_2. For now, notice that, in G_2, we have precisely n runs of five consecutive $*$-genes. We put these runs into a 1, 1-correspondence with the n variables in X so that the ith run corresponds to variable x_i, for $i = 1, 2, \ldots, n$. For each $i = 1, 2, \ldots, n$, let $\mathcal{P}_i$ and $\mathcal{N}_i$ be the lists of index

sets of the clauses from $\mathcal{P}$ and $\mathcal{N}$ which contain variable x_i. For example, if x_i appears in P_3, in P_7, and in N_2, then $\mathcal{P}_i = (3, 7)$, whereas $\mathcal{N}_i = (2)$. Notice that the lengths of the lists $\mathcal{P}_i$ and $\mathcal{N}_i$ are at most 3 and 2, respectively. From the list $\mathcal{P}_i$, we obtain a list $\mathcal{P}'_i$ of length precisely 3 by possibly iterating the last element in $\mathcal{P}_i$ the required number of times (that is, $3 - |\mathcal{P}_i|$ times). A list $\mathcal{N}'_i$ of length precisely 2 is similarly obtained from list $\mathcal{N}_i$. Now, for each $i = 1, 2, \ldots, n$, the ith run of five consecutive $*$-genes consists, more precisely, of the following five characters:

$$(*, *, *, *, *) \rightarrow (\mathcal{P}'_i[1], \mathcal{P}'_i[2], \mathcal{P}'_i[3], L_1 - m_n + \mathcal{N}'_i[1], L_1 - m_n + \mathcal{N}'_i[2]).$$

The above is clearly a polynomial time reduction. It can also be easily seen that there are no duplications in G_1, whereas each gene appears at most nine times in G_2 (that is, $occ(G_1) = 1$ and $occ(G_2) \leq 9$). Besides, we have the following lemmas.

Lemma 7. *Let $t_X : X \mapsto \{0, 1\}$ be a satisfying truth assignment for $\langle X, \mathcal{C}\rangle$. Then, there exists an exemplar subgenome G_2^E of G_2 whose MAD number satisfies $\mathrm{MAD}(G_1, G_2^E) \leq M_\varepsilon + m_p + m_n + n$.*

Proof. For each clause $P_j \in \mathcal{P}$, choose a variable x_i occurring in P_j and such that $t_X(x_i) = 1$ (remember that t_X is a satisfying truth assignment) and color with red one copy of gene j occurring within the ith run of five consecutive $*$-genes in G_2. Similarly, for each clause $N_j \in \mathcal{N}$, choose a variable x_i occurring in N_j and such that $t_X(x_i) = 0$ (again, at least one such variable must exist since t_X is a satisfying truth assignment) and color with red one copy of gene $(L_1 - m_n) + j = 2\,M_\varepsilon + m_p + n - 1 + j$ occurring within the ith run of five consecutive $*$-genes in G_2. Now, obtain G_2^E from G_2 by deleting all of the $*$-genes except those marked red. Notice that G_2^E is indeed an exemplar genome on the genes $1, 2, \ldots, L_1$.

We now verify that

$$\mathrm{MAD}(G_1, G_2^E) \leq M_\varepsilon + m_p + m_n + n,$$

which is better done in two separate steps. First, we check out that any two genes j and $j + 1$ that are adjacent in G_1 are at most $M_\varepsilon + m_p + m_n + n$ positions apart in G_2^E. This follows from the fact that $L_1 = 2\,M_\varepsilon + m_p + m_n + n - 1$ and considering that the first M_ε positions in G_2^E are taken by genes $j \in [m_p + 1, m_p + M_\varepsilon]$, whereas the last M_ε positions in G_2^E are taken by genes $j \in [L_1 - m_n - M_\varepsilon + 1, L_1 - m_n]$. Moreover, for

$$j \in [m_p + 1, m_p + M_\varepsilon - 1] \cup [L_1 - m_n - M_\varepsilon + 1, L_1 - m_n - 1],$$

genes j and $j + 1$ are also adjacent in G_2^E (more generally, for $j \in [m_p + 1, L_1 - m_n - 1]$, genes j and $j + 1$ both also have a unique occurrence in G_2, where they are at most six positions apart, and they are at most four positions apart in G_2^E). Second and last, we check out that any two genes i and j which are adjacent in G_2^E are at most $M_\varepsilon + m_p + m_n + n$ positions apart in G_1. Here, if neither i nor j are $*$-genes, then i and j are also adjacent in G_1, that is, $j = i \pm 1$. Furthermore, if precisely one among i and j, say, j, is an $*$-gene, then $m_p + M_\varepsilon \leq i \leq m_p + M_\varepsilon + n$ since, otherwise, i could not be adjacent to an $*$-gene in G_2^E; hence, if $j < i$, then $i - j \leq m_p + M_\varepsilon + n$, whereas if $i < j$, then $j - i \leq m_n + M_\varepsilon + n$. Thus, the only interesting case is when both i and j are $*$-genes, that is, both i and j belong either to the interval $[1, m_p]$ or to the interval $[L_1 - m_n + 1, L_1]$. It suffices here to notice that, in this case, i and j come from the same interval. Indeed, this follows from the fact that i and j are adjacent in G_2^E and, hence, correspond to occurrences of the same variable. However, then these two occurrences must either be both positive or both negative since they both have been colored with red in the marking phase. □

Lemma 8. *For any exemplar genome G_2^E of G_2 such that $\mathrm{MAD}(G_1, G_2^E) < 2\,M_\varepsilon + n$, we can derive in polynomial time from G_2^E a satisfying truth assignment for $\langle X, \mathcal{C}\rangle$.*

Proof. Since $\mathrm{MAD}(G_1, G_2^E) < 2\,M_\varepsilon + n$, then, in obtaining G_2^E from G_2 and for each $i = 1, 2, \ldots, n$, it must be the case that, in the ith run of five consecutive $*$-genes in G_2, either the genes $\mathcal{P}'_i[1]$, $\mathcal{P}'_i[2]$, and $\mathcal{P}'_i[3]$ have all been deleted or the genes $\mathcal{N}'_i[1] + L_1 - m_n$ and $\mathcal{N}'_i[2] + L_1 - m_n$ have both been deleted. Consider the truth assignment $t_X : X \mapsto \{0, 1\}$ such that, for each $i = 1, 2, \ldots, n$, $t_X(x_i) = 1$ iff both $\mathcal{N}'_i[1] + L_1 - m_n$ and $\mathcal{N}'_i[2] + L_1 - m_n$ have been deleted. We claim that $t_X(x_i)$ is a satisfying truth assignment. Indeed, for each clause $P_j \in \mathcal{P}$, we know that at least a copy of gene j has been retained in G_2^E. This copy must come from one of the runs of five consecutive $*$-genes in G_2, say, from the ith run. It follows that x_i occurs in P_j and that $t_X(x_i) = 1$. Similarly, for each clause $N_j \in \mathcal{N}$, we know that at least a copy of the gene $(L_1 - m_n) + j$ (that is, of gene $2\,M_\varepsilon + m_p + n - 1 + j$) has been retained in G_2^E. This copy must come from one of the runs of five consecutive $*$-genes in G_2, say, from the ith run. It follows that x_i occurs in N_j and that $t_X(x_i) = 0$. □

Theorem 5. *For no $\varepsilon > 0$, EMAD (respectively, MMAD) admits a $(2 - \varepsilon)$-approximation algorithm unless $\mathbf{P} = \mathbf{NP}$.*

Proof. We proceed as follows: We assume that we are given a $(2 - \varepsilon)$-approximation algorithm A for EMAD (respectively, MMAD) and design a polynomial time algorithm for UNIFORM-SAT which rests on algorithm A as a subroutine. The theorem then follows from the **NP**-completeness of UNIFORM-SAT, as stated in Theorem 3. After receiving an instance $\langle X, \mathcal{C}\rangle$ of UNIFORM-SAT in the input, we construct the instance $\langle G_1, G_2\rangle$ of EMAD (respectively, MMAD), as described above. If $\langle X, \mathcal{C}\rangle$ is satisfiable, then, by Lemma 7, there exists an exemplar subgenome G_2^E of G_2 such that

$$\mathrm{MAD}(G_1, G_2^E) \leq M_\varepsilon + m_p + m_n + n.$$

By running algorithm A, we are hence guaranteed to find an exemplar subgenome G_{apx}^E of G_2 such that

$$\mathrm{MAD}(G_1, G_{apx}^E) \leq (M_\varepsilon + m_p + m_n + n)(2 - \varepsilon) \leq 2\,M_\varepsilon + 2\,m_p + 2\,m_n + 2\,n - \varepsilon\,M_\varepsilon.$$

Now, after choosing $M_\varepsilon \geq \frac{2m_p+2m_n+2n}{\varepsilon}$, we conclude that the solution G^E_{apx} produced by algorithm A satisfies $\mathrm{MAD}(G_1, G^E_{apx}) \leq 2M_\varepsilon$. Moreover, Lemma 8 (whose proof can be easily converted into a polynomial time algorithm) shows how, starting from G^E_{apx}, one can obtain a satisfying truth assignment for $\langle G_1, G_2\rangle$. Conversely, if $\langle X, \mathcal{C}\rangle$ is not satisfiable, then, by Lemma 8, $\mathrm{MAD}(G_1, G^E_{apx}) \geq 2M_\varepsilon + n$ must hold for the solution returned by algorithm A as it holds for any solution and we can realize that $\langle X, \mathcal{C}\rangle$ was not satisfiable comparing this fact against Lemma 7. □

Remark 1. There actually exists a constant $c > 2$ such that EMAD (respectively, MMAD) admits no c-approximation algorithm unless $\mathbf{P} = \mathbf{NP}$. We can get to this stronger conclusion if, in the proof of Theorem 5 above, we apply Theorem 4 instead of Theorem 3. Moreover, explicit values of c for which this stronger statement holds can also be worked out.

5 Summed Adjacency Disruption (SAD)

Let ESAD (respectively, MSAD) denote the problem of computing the minimum SAD number in the exemplar (respectively, matching) model. In this section, we prove that both problems ESAD and MSAD, expressed on two genomes G_1 and G_2 such that $|G_1| \leq |G_2|$, cannot be better than $\log(|G_1|)$ approximated (here and in the rest of the paper, logarithms are assumed to be base e). This result holds for both the exemplar and the matching models since we prove it in the case where $occ(G_1) = 1$, for which the two problems coincide. The inapproximability of ESAD (respectively, MSAD) is obtained starting from the inapproximability of SETCOVER. This result will hence depend on the **PCP** theorem but will deliver stronger SETCOVER-like inapproximability thresholds than for the EMAD and MMAD problems discussed in the previous section.

Let $\langle V, \mathcal{S}\rangle$ be an instance of SETCOVER, where $V = \{1, 2, \ldots, n\}$ and $\mathcal{S} = \{S_1, S_2, \ldots, S_m\}$ is a family of subsets of V. We can assume n is even, say, $n = 2k$, and each set S_i contains precisely $k = \frac{n}{2}$ elements, say, $s^i_1, s^i_2, \ldots, s^i_k$. The well-known inapproximability results for SETCOVER also hold under these assumptions since we can think of enlarging a groundset V, originally on k elements, by adding a set V' of k new elements, adding V' to $\mathcal{S}$, and enlarging the other sets in $\mathcal{S}$ with elements from V' until their size rises up to k. Let $M = m^2n^2$ play the role of a sufficiently big positive integer. Let us now detail the construction of the two genomes, G_1 and G_2, from any instance of the SETCOVER problem. Here, G_1 is a simple (that is, duplication-free) genome of length $L_1 = M + n + m$ as given by $G_1 = 1, 2, 3 \ldots L_1$. Genome G_2 has length $L_2 = M + m(k+1)$ and is constructed as follows:

$$
\begin{aligned}
G_2 = {} & n+1, n+2, \ldots, n+M, s^1_1, s^1_2, \\
& \ldots, s^1_k, n+M+1, s^2_1, s^2_2, \ldots, s^2_k, n+M+2, \ldots \\
& \ldots, s^{m-1}_1, s^{m-1}_2, \ldots, s^{m-1}_k, n+M+m-1, s^m_1, s^m_2, \ldots, \\
& s^m_k, n+M+m.
\end{aligned}
$$

The above is clearly a polynomial time reduction; in addition, we have the following lemmas.

Lemma 9. *Let $\mathcal{S}' \subset \mathcal{S}$ be a set cover of V with $|\mathcal{S}'| \leq s$. Then, there exists an exemplar subgenome G^E_2 of G_2 whose SAD number satisfies $\mathrm{SAD}(G_1, G^E_2) \leq 2sM + 5M$.*

Proof. For each element $v \in V$, choose a set S_i in $\mathcal{S}'$ such that $v \in S_i$, that is, $s^i_j = v$ for some $j = 1, 2, \ldots, k$. Color with red this copy of gene v, that is, the copy of gene v occurring in the position $M + (k+1)(i-1) + j$ of G_2. Now, obtain G^E_2 from G_2 by deleting all of the copies of the first n genes, except those marked with red. Notice that G^E_2 is indeed an exemplar genome on the genes $1, 2, \ldots, L_1$.

We now verify that $\mathrm{SAD}(G_1, G^E_2) \leq 2sM + 5M$. Indeed,

$$
\begin{aligned}
\mathrm{SAD}(G_1, G^E_2) = {} & \\
\sum_{i=1}^{M+m+n-1} & \mathrm{Dist}(G^E_2, G_1[i], G_1[i+1]) + \\
\sum_{i=1}^{M+m+n-1} & \mathrm{Dist}(G_1, G^E_2[i], G^E_2[i+1]),
\end{aligned}
$$

where, assuming that m and n are sufficiently big $(m, n \geq 4)$,

$$
\begin{aligned}
\sum_{i=1}^{M+m+n-1} & \mathrm{Dist}(G^E_2, G_1[i], G_1[i+1]) \leq \\
& \sum_{i=1}^{n-1} \mathrm{Dist}(G^E_2, G_1[i], G_1[i+1]) + \\
& (M+m+n) + \\
& \sum_{i=n+1}^{M+n-1} \mathrm{Dist}(G^E_2, G_1[i], G_1[i+1]) + \\
& \sum_{i=M+n}^{M+m+n-1} \mathrm{Dist}(G^E_2, G_1[i], G_1[i+1]) \\
\leq {} & n(n+m) + (M+m+n) + M + mn \\
& \leq 2M + 3mn^2 \\
& \leq 2M + m^2n^2 \\
& \leq 3M,
\end{aligned}
$$

and where, again assuming that m and n are sufficiently big $(m, n \geq 4)$,

$$\sum_{i=1}^{M+m+n-1} \mathrm{Dist}(G_1, G_2^E[i], G_2^E[i+1]) =$$
$$\sum_{i=1}^{M-1} \mathrm{Dist}(G_1, G_2^E[i], G_2^E[i+1]) +$$
$$\sum_{i=M}^{M+m+n-1} \mathrm{Dist}(G_1, G_2^E[i], G_2^E[i+1]) =$$
$$(M-1) +$$
$$\sum_{i=M}^{M+m+n-1} \mathrm{Dist}(G_1, G_2^E[i], G_2^E[i+1])$$
$$\leq M + \sum_{S_i \notin \mathcal{S}'} 1 + \sum_{S_i \in \mathcal{S}'} (2(M+m+n+n^2))$$
$$\leq M + m + 2\,s\,(M+m+n+n^2)$$
$$\leq M + 2\,s\,M + m^2 n^2$$
$$\leq 2\,s\,M + 2\,M.$$

To better explain the upper bound on the term $\sum_{i=M}^{M+m+n-1} \mathrm{Dist}(G_1, G_2^E[i], G_2^E[i+1])$ used in the above chain of inequalities, denote with p_i, $i = 0, 1, \ldots, m$, the absolute position of the gene $M+n+i$ inside the genome G_2^E. (Thus, $p_0 = M$ and $p_n = M+n+m$). Clearly,

$$\sum_{i=M}^{M+m+n-1} \mathrm{Dist}(G_1, G_2^E[i], G_2^E[i+1]) =$$
$$\sum_{i=1}^{m} \sum_{j=p_{i-1}}^{p_i-1} \mathrm{Dist}(G_1, G_2^E[j], G_2^E[j+1]).$$

Now, when $S_i \notin \mathcal{S}'$, then the two genes $n+M+(i-1)$ and $n+M+i$ are adjacent both in G_2^E and in G_1, where $\sum_{j=p_{i-1}}^{p_i-1} \mathrm{Dist}(G_1, G_2^E[j], G_2^E[j+1]) = 1$. Also, for each $i = 1, 2, \ldots, m$,

$$\mathrm{Dist}(G_1, G_2^E[p_{i-1}], G_2^E[p_{i-1}+1]) \leq M+m+n$$

and $\mathrm{Dist}(G_1, G_2^E[p_i - 1], G_2^E[p_i]) \leq M+m+n$ since $M+m+n$ is the length of G_1. Furthermore, $p_i - p_{i-1} \leq 1+k \leq n$, and, for each

$$j = 1, 2, \ldots, p_i - p_{i-1} - 2,$$
$$\mathrm{Dist}(G_1, G_2^E[p_{i-1}+j], G_2^E[p_{i-1}+j+1]) \leq n.$$

□

Lemma 10. *For any exemplar subgenome G_2^E of G_2 such that $\mathrm{SAD}(G_1, G_2^E) < 2\,s\,M$, we can derive, from G_2^E and in polynomial time, a set cover $\mathcal{S}' \subset \mathcal{S}$ of V such that $|\mathcal{S}'| \leq s$.*

Proof. Let $\mathcal{S}'$ be the family of those $S_i \in \mathcal{S}$ for which there exists a $v \in S_i$, say, $v = s_i^j$, such that, in obtaining G_2^E from G_2, the copy s_i^j of gene v has not been deleted. Notice that $\mathcal{S}'$ is a cover of V since all genes $1, 2, \ldots, L_1$ occur in G_2^E. Moreover, $|\mathcal{S}'| \leq s$ follows from $\mathrm{SAD}(G_1, G_2^E) < 2\,s\,M$. Indeed,

$$\sum_{i=M}^{M+m+n-1} \mathrm{Dist}(G_1, G_2^E[i], G_2^E[i+1]) \leq SAD(G_1, G_2^E) \leq 2\,s\,M.$$

However, for every i such that $S_i \in \mathcal{S}'$, the genes $M+n+(i-1)$ and $M+n+i$ are not consecutive in G_2^E. Let us denote with p_{i-1} and p_i the absolute positions of genes $M+n+(i-1)$ and $M+n+i$ within the genome G_2^E. Thus, whenever $S_i \in \mathcal{S}'$, then $p_i > p_{i-1}+1$ and we have $\mathrm{Dist}(G_1, G_2^E[p_{i-1}], G_2^E[p_{i-1}+1]) \geq M$ and

$$\mathrm{Dist}(G_1, G_2^E[p_i - 1], G_2^E[p_i]) \geq M$$

since $G_2^E[p_{i-1}+1] \leq n$ and $G_2^E[p_i - 1] \leq n$. It follows that $|\mathcal{S}'| \leq \frac{2\,s\,M}{2\,M} = s$. □

Theorem 6. *There exists a constant $c > 0$ such that ESAD (respectively, MSAD) admits no $(c \log |G_1|)$-approximation algorithm unless $\mathbf{P} = \mathbf{NP}$, where $|G_1|$ is the length of the smallest genome.*

Proof. It is well known that the SETCOVER cannot be approximated within $(1-\varepsilon)\log n$ (where n is the number of elements) for any $\varepsilon > 0$ (see [26]) nor within $c \log m$ (where m is the number of sets (see [27]) for some $c > 0$). To be more precise, it has been proved in [27] that the instance of the Set Cover produced through the reduction in [26] is characterized by having $m \leq n^5$. Thus, for no $\varepsilon > 0$, SETCOVER can be $(1-\varepsilon)$-approximated, even when restricting attention to instances in which $\log m \leq 5 \log n$. This means that there exists a constant c' such that no polynomial algorithm approximates SETCOVER within $c'(\log m + \log n)$, with c' chosen small enough (consider any $c' < \frac{1}{6}$). We claim that ESAD (respectively, MSAD) admits no $(\frac{c'}{4} \log |G_1|)$-approximation algorithm. We proceed as follows: We assume that we are given a $(\frac{c'}{4} \log |G_1|)$-approximation algorithm A for ESAD (respectively, MSAD) and design a $c'(\log m + \log n)$-approximation algorithm for SETCOVER which rests on algorithm A as a subroutine. The theorem then follows from the above collected inapproximability facts about SETCOVER. After receiving in input an instance $\langle V, \mathcal{S} \rangle$ of SETCOVER, we construct the instance $\langle G_1, G_2 \rangle$ of ESAD (respectively, MSAD) as described above. Notice that $|G_1| \leq 2\,M$ and, hence,

$$\log |G_1| \leq \log 2m^2 n^2 \leq 3\,(\log m + \log n).$$

Let *opt* be the minimum size of a set cover for $\langle V, \mathcal{S} \rangle$. Then, by Lemma 9, there exists an exemplar subgenome G_2^E of G_2 such that $\mathrm{SAD}(G_1, G_2^E) \leq 2\,opt\,M + 5\,M$. By running algorithm A, we are hence guaranteed to find an exemplar subgenome G_{apx}^E of G_2 such that

$$\mathrm{SAD}(G_1, G_{apx}^E \leq (2\,opt\,M + 5\,M)\frac{c'}{4}\log |G_1|$$
$$\leq \left(\frac{8}{3}\,opt\,M\right)\frac{c'}{4}\,3\,(\log m + \log n)$$
$$\leq (2\,opt\,M)\,c'\,(\log m + \log n).$$

Indeed, in the derivation of the above chain of inequalities, we can conveniently assume that the value of *opt* is sufficiently big since, if *opt* was bounded by any constant, then an optimal solution to the original SETCOVER instance could be found in polynomial time. Now, Lemma 10 (whose proof can be easily converted

TABLE 1
Results Concerning the Exemplar Model

Exemplar Model		
Measure	**Complexity**	**Approximability**
Breakpoints	**NP**-complete [9] even when $occ(G_1) = 1$ and $occ(G_2) = 2$	**APX**-hard [21] even when $occ(G_1) = 1$ and $occ(G_2) = 2$
Reversals	**NP**-complete [9] even when $occ(G_1) = 2$ and $occ(G_2) = 2$	**APX**-hard [13] even when $occ(G_1) = 2$ and $occ(G_2) = 2$
Conserved Intervals	**NP**-complete [18] even when $occ(G_1) = 1$ (*)	**APX**-hard [21] even when $occ(G_1) = 1$ and $occ(G_2) = 2$
Common Intervals	**NP**-complete (Theorem 1) even when $occ(G_1) = 1$ and $occ(G_2) = 2$	**APX**-hard [21] even when $occ(G_1) = 1$ and $occ(G_2) = 2$
MAD	**NP**-complete (Theorem 5) even when $occ(G_1) = 1$ and $occ(G_2) \leq 9$	**APX**-hard (Theorem 5) even when $occ(G_1) = 1$ and $occ(G_2) \leq 9$
SAD	**NP**-complete (Theorem 6) even when $occ(G_1) = 1$	**APX**-hard (Theorem 6) even when $occ(G_1) = 1$

(*) *We note that this result can actually be extended to the case where* $occ(G_1) = 1$ ***and*** $occ(G_2) = 2$ *by reducing the problem from* VERTEX-COVER *instead of* SETCOVER.

TABLE 2
Results Concerning the Matching Model

Matching Model		
Measure	**Complexity**	**Approximability**
Breakpoints	**NP**-complete even when $occ(G_1) = 1$ and $occ(G_2) = 2$ [9] even when $f(G_1) = f(G_2) = 1$ [2]	**APX**-hard [21] even when $occ(G_1) = 1$ and $occ(G_2) = 2$
Reversals	**NP**-complete [10] even when $occ(G_1) = 2$ and $occ(G_2) = 2$	
Conserved Intervals	**NP**-complete even when $occ(G_1) = 1$ [18] (*) even when $f(G_1) = f(G_2) = 1$ [18]	**APX**-hard [21] even when $occ(G_1) = 1$ and $occ(G_2) = 2$
Common Intervals	**NP**-complete even when $occ(G_1) = 1$ and $occ(G_2) = 2$ (Theorem 1) even when $f(G_1) = f(G_2) = 1$ (Th. 2)	**APX**-hard [21] even when $occ(G_1) = 1$ and $occ(G_2) = 2$
MAD	**NP**-complete (Theorem 5) even when $occ(G_1) = 1$ and $occ(G_2) \leq 9$	**APX**-hard (Theorem 5) even when $occ(G_1) = 1$ and $occ(G_2) \leq 9$
SAD	**NP**-complete (Theorem 6) even when $occ(G_1) = 1$	**APX**-hard (Theorem 6) even when $occ(G_1) = 1$

(*) *We note that this result can actually be extended to the case where* $occ(G_1) = 1$ ***and*** $occ(G_2) = 2$ *by reducing the problem from* VERTEX-COVER *instead of* SETCOVER.

into a polynomial time algorithm) shows how, starting from $G^E_{apx'}$, one can obtain a set cover S' with $|\mathcal{S}'| \leq \frac{1}{2M}(2\,opt\,M)\,c'\,(\log m + \log n) = opt\,c'\,(\log m + \log n)$. □

6 Summary of the Results and Discussion

In this section, we give a summary of the results from this paper, as well as some other results concerning the complexity of computing several classical (dis)similarity measures, under both the exemplar and the matching models. We found it interesting to end this paper by giving an overview of the existing results in this area since several recent papers, by different groups of authors, have investigated the problem. Hence, in addition to the number of common intervals, MAD number and SAD number, we include results concerning the number of conserved intervals (initially defined in [23]), number of breakpoints, and number of reversals. However, we should note that the three above-mentioned measures take signs into account, which is not the case for common intervals, MAD, and SAD.

We recall that $occ(G)$ denotes the maximum of $occ(G, g)$ over all genes g in G, where $occ(G, g)$ denotes the maximum number of occurrences of a gene g in genome G (regardless of the signs). We also recall that $f(G)$ denotes the number of different families of genes that contain several occurrences in genome G.

The results concerning the exemplar model are summarized in Table 1, whereas the ones concerning the matching model are summarized in Table 2.

The main conclusion that we can draw from these two tables is that, as soon as $occ(G_1) = 1$ and $occ(G_2) = 2$, the computation of five out of the six above-mentioned measures becomes **NP**-complete under both the exemplar and matching models. In that sense, we are able to draw the exact border between polynomial problems ($occ(G_1) = occ(G_2 = 1)$ and **NP**-complete problems ($occ(G_1) = 1$ and $occ(G_2) = 2$), except for the number of reversals where a gap exists (we do not know the complexity of the problem when $occ(G_1) = 1$ and $occ(G_2) = 2$).

Another interesting parameter to consider for the complexity of those problems is $f(G)$, the number of families of genes that are duplicated in genome G. Concerning this parameter, only a few results are known (breakpoints and conserved and common intervals, in the matching model only).

Concerning the approximability of the problems, it turns out that, even when $occ(G_1) = 1$, we are able to say that five out of the six measures lead to **APX**-hard problems. For the number of reversals, it is **APX**-hard in the exemplar model when $occ(G_1) = occ(G_2) = 2$ [13]. However, for three of those five cases (breakpoints and conserved and common intervals) similar to the complexity results, we know that the problem is **APX**-hard even when $occ(G_1) = 1$ and $occ(G_2) = 2$, whereas, in the others, the value of $occ(G_2)$ is either unbounded (SAD) or bounded by constant 9 (MAD).

7 Conclusion

In this paper, we have investigated the algorithmic complexity of the problem of computing similarity measures between genomes in the case where they contain duplicates. This has been done for three measures: common intervals, MAD, and SAD. We have shown that the three problems are **NP**-complete, for both the exemplar and matching variants. Moreover, we have provided **APX**-hardness results concerning MAD and SAD. Those results, together with the ones concerning conserved intervals, breakpoints, and reversals, basically show that, as soon as duplicates are present, the problem becomes hard and even hard to approximate, even in very restricted instances.

Several lines of research would be interesting to follow, some of which we mention below:

- Make Tables 1 and 2 even more precise. In particular, 1) complete the cases for which no result

is known or a gap exists (that is, the number of reversals), 2) study more deeply the complexity and approximability results with respect to the parameter f, and 3) tighten, if possible, the results concerning the (in)approximability of the problems, notably for the number of reversals in the exemplar model.

- Find Fixed-Parameter Tractable algorithms for those problems in order to circumvent **NP**-completeness and **APX**-hardness of the problems.
- Find good heuristics for those problems, as done, for instance, in [17] and [28] (among many others), in which the authors are able to compare their proposed heuristic to the exact results.

REFERENCES

[1] D. Sankoff, "Genome Rearrangement with Gene Families," *Bioinformatics,* vol. 15, no. 11, pp. 909-917, 1999.

[2] G. Blin, C. Chauve, and G. Fertin, "The Breakpoint Distance for Signed Sequences," *Proc. Algorithms and Computational Methods for Biochemical and Evolutionary Networks (CompBioNets '04),* pp. 3-16, 2004.

[3] D. Sankoff, G. Leduc, N. Antoine, B. Paquin, B. Lang, and R. Cedergren, "Gene Order Comparisons for Phylogenetic Inference: Evolution of the Mitochondrial Genome," *Proc. Nat'l Academy of Sciences USA,* vol. 89, no. 14, pp. 6575-6579, 1992.

[4] G. Bourque, E. Zdobnov, P. Bork, P. Pevzner, and G. Tesler, "Comparative Architectures of Mammalian and Chicken Genomes Reveal Highly Variable Rates of Genomic Rearrangements across Different Lineages," *Genome Research,* vol. 15, no. 1, pp. 98-110, 2005.

[5] D. Sankoff, "Gene and Genome Duplication," *Current Opinion in Genetics Development,* vol. 11, no. 6, pp. 681-684, 2001.

[6] J. Korbel, D. Snel, M. Huynen, and P. Bork, "Shot: A Web Server for the Construction of Genome Phylogenies," *Trends in Genetics,* vol. 18, no. 3, pp. 158-162, 2002.

[7] E. Belda, A. Moya, and F. Silva, "Genome Rearrangement Distances and Gene Order Phylogeny in γ-Proteobacteria," *Molecular Biology Evolution,* vol. 22, no. 6, pp. 1456-1467, 2005.

[8] G. Blin, A. Chateau, C. Chauve, and Y. Gingras, "Inferring Positional Homologs with Common Intervals of Sequences," *Proc. RECOMB Int'l Workshop Comparative Genomics (RCG '06),* pp. 24-38, 2006.

[9] D. Bryant, "The Complexity of Calculating Exemplar Distances," *Comparative Genomics: Empirical and Analytical Approaches to Gene Order Dynamics, Map Alignment, and the Evolution of Gene Families,* pp. 207-212, Kluwer Academic, 2000.

[10] X. Chen, J. Zheng, Z. Fu, P. Nan, Y. Zhong, S. Lonardi, and T. Jiang, "Assignment of Orthologous Genes via Genome Rearrangement," *IEEE/ACM Trans. Computational Biology and Bioinformatics,* vol. 2, no. 4, pp. 302-315, Oct.-Dec. 2005.

[11] K. Swenson, M. Marron, J. Earnest-DeYoung, and B. Moret, "Approximating the True Evolutionary Distance between Two Genomes," *Proc. Seventh Workshop Algorithm Eng. and Experiments and Second Workshop Analytic Algorithmics and Combinatorics (ALENEX/ANALCO '05),* pp. 121-129, 2005.

[12] N.C. Thach, "Algorithms for Calculating Exemplar Distances," honours Year Project Report, Nat'l Univ. of Singapore, 2005.

[13] Z. Chen, B. Fu, and B. Zhu, "The Approximability of the Exemplar Breakpoint Distance Problem," *Proc. Second Int'l Conf. Algorithmic Aspects in Information and Management (AAIM '06),* pp. 291-302, June 2006.

[14] M. Marron, K. Swenson, and B. Moret, "Genomic Distances under Deletions and Insertions," *Theoretical Computer Science,* vol. 325, no. 3, pp. 347-360, 2004.

[15] J. Tang and B. Moret, "Phylogenetic Reconstruction from Gene-Rearrangement Data with Unequal Gene Content," *Proc. Eighth Int'l Workshop Algorithms and Data Structures (WADS '03),* pp. 37-46, 2003.

[16] Z. Fu, X. Chen, V. Vacic, P. Nan, Y. Zhong, and J. Tang, "A Parsimony Approach to Genome-Wide Ortholog Assignment," *Proc. 10th Ann. Int'l Conf. Research in Computational Molecular Biology (RECOMB '06),* pp. 578-594, Apr. 2006.

[17] C. Nguyen, Y. Tay, and L. Zhang, "Divide-and-Conquer Approach for the Exemplar Breakpoint Distance," *Bioinformatics,* vol. 21, no. 10, pp. 2171-2176, 2005.

[18] G. Blin and R. Rizzi, "Conserved Interval Distance Computation between Non-Trivial Genomes," *Proc. 11th Ann. Int'l Conf. Computing and Combinatorics (COCOON '05),* pp. 22-31, Aug. 2005.

[19] G. Bourque, Y. Yacef, and N. El-Mabrouk, "Maximizing Synteny Blocks to Identify Ancestral Homologs," *Proc. RECOMB Int'l Workshop Comparative Genomics (RCG '05),* pp. 21-34, Sept. 2005.

[20] Z. Chen, R. Fowler, B. Fu, and B. Zhu, "Lower Bounds on the Approximation of the Exemplar Conserved Interval Distance Problem of Genomes," *Proc. 12th Ann. Int'l Conf. Computing and Combinatorics (COCOON '06),* pp. 245-254, 2006.

[21] S. Angibaud, G. Fertin, and I. Rusu, *On the Inapproximability of Similarity Measures between Genomes Containing Duplicates,* 2006,

[22] D. Sankoff and L. Haque, "Power Boosts for Cluster Tests," *Proc. RECOMB Int'l Workshop Comparative Genomics (RCG '05),* pp. 121-130, Sept. 2005.

[23] A. Bergeron and J. Stoye, "On the Similarity of Sets of Permutations and Its Applications to Genome Comparison," *Proc. Ninth Ann. Int'l Conf. Computing and Combinatorics (COCOON '03),* pp. 68-79, July 2003.

[24] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W.H. Freeman, 1979.

[25] C.H. Papadimitriou and M. Yannakakis, "Optimization, Approximation, and Complexity Classes," *J. Computer and System Sciences,* vol. 43, pp. 425-440, 1991.

[26] U. Feige, "A Threshold of $ln(n)$ for Approximating Set Cover," *J. ACM,* vol. 4, no. 45, pp. 634-652, 1998.

[27] S. Eidenbenz, C. Stamm, and P. Widmayer, "Positioning Guards at Fixed Height above a Terrain—An Optimum Inapproximability Result," *Proc. Sixth European Symp. Algorithms (ESA '98),* pp. 187-198, 1998.

[28] S. Angibaud, G. Fertin, I. Rusu, and S. Vialette, "How Pseudo-Boolean Programming Can Help Genome Rearrangement Distance Computation," *Proc. RECOMB 2006 Int'l Workshop Comparative Genomics (RCG '06),* pp. 75-86, 2006.

Guillaume Blin received the PhD degree in computer science from Nantes University, France, in 2005. He is an associate professor in the Gaspard Monge Institute, University of Marne-La-Vallée, France. His current research interests include computational complexity and approximation, algorithms, and bioinformatics.

Cedric Chauve received the PhD degree in computer science from the University of Bordeaux I, France, in 2000. From 2001 to 2006, he was a professor in the Department of Computer Science at the University of Québec in Montréal. He is now an associate professor in the Department of Mathematics at Simon Fraser University, Burnabay, British Columbia, Canada. His current research interest is the development of combinatorial models and efficient algorithms for comparative genomics.

Guillaume Fertin received the PhD degree in computer science from the University of Bordeaux, France, in 1999. He is a professor in the Laboratoire d'Informatique de Nantes-Atlantique (LINA) at the University of Nantes, France. His current research interests are optimization, discrete mathematics, computational complexity, and algorithms dedicated to bioinformatics.

Romeo Rizzi received the Laurea degree in electronic engineering from the Politecnico di Milano in 1991 and the PhD degree in computational mathematics and informatics from the University of Padova, Italy, in 1997. Afterward, he held postdoctoral and other temporary positions at research centers like Centrum voor Werk en Inkomen (CWI; Amsterdam), Basic Research in Computer Sciences (BRICS; Aarhus, Denmark), and Istituto per la Ricerca Scientifica e Tecnologica (IRST, Trento, Italy). In 2001, he became an assistant professor at the University of Trento. Since 2005, he has been an associate professor at the University of Udine. He has a background in operations research and his main interests are in combinatorial optimization and algorithms. He is an area editor of *4OR* and acts as a reviewer for the American Mathematical Society. He has published more than 40 articles in a broad range of scientific journals in the areas of discrete mathematics, combinatorics, and algorithms. Since 2004, he has been a trainer of the Italian team for the iOi.

Stéphane Vialette received the PhD degree in computer science from the University Paris 7 in 2001. He has been a postdoctoral researcher at the Laboratoire de Génétique Moléculaire at the Ecole Normale Supérieure for two years. Since 2003, he has been an assistant professor with the Bioinformatics Group from the Computer Science Research Laboratory (LRI) at the Université Paris-Sud 11. His research interests include computational biology, algorithmics, and combinatorics.