Comparing Genomes with Duplications: a Computational Complexity Point of View

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

Guillaume Blin is with IGM-LabInfo, UMR CNRS 8049, Université de Marne-la-Vallée, 77454 Marne-la-Vallée Cedex 2 - France. E-mail: gblin@univ-mlv.fr

Cedric Chauve is with CGL, LaCIM et Département d'Informatique, Université du Québec À Montréal CP 8888, Succ. Centre-Ville, H3C 3P8, Montréal (QC) - Canada and Department of Mathematics, Simon Fraser University 8888 University Drive, V5A 1S6, Burnaby (BC) - Canada. E-mail: cedric.chauve@sfu.ca

Guillaume Fertin is with 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

Romeo Rizzi is with Dipartimento di Matematica e Informatica, Università di Udine - Italy. E-mail: Romeo.Rizzi@dimi.uniud.it

Stéphane Vialette is with Laboratoire de Recherche en Informatique (LRI), UMR CNRS 8623, Faculté des Sciences d'Orsay, Université Paris-Sud, 91405 Orsay - France. E-mail: vialette@lri.fr

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 genes of G_1 and genes of G_2 ; second, once this correspondence is established, it defines explicitly 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, to 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 are deleted in both genomes for each gene family, and the *matching* model, that 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 SAD number.

Index Terms

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

I. INTRODUCTION

The comparison of whole genomes from the gene order point of view has been a very active research domain since the early 90's. 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, that 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.

A first way to overcome such a limitation is to consider genomes at a higher scale than genes, e.g., *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 *non-trivial 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 in 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 II). In the second step, a permutation-based (dis)similarity measure is computed from the permutation P. The main line of research

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

following this approach seeks for 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 (resp. minimal) similarity (resp. dissimilarity) measure. Note however that there exists 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]–[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 non-trivial gene family, all but one copy in each genome are 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 to conserve more than one copy of a gene family and seeks for a maximal one-to-one correspondence between these copies [2]. Several distances have been considered under the exemplar and matching models, that are either based on minimizing the number of evolutionary events that allow to transform a genome into the other, for events like reversals[2] [1], [9]–[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]–[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

[2]The reversal model considers *signed permutations*, where each element has a sign, positive or negative, that indicates on which strand on the genome the corresponding gene is 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.

be **NP**-complete as soon as duplicates are present in genomes (see Tables I and II, in Section VI).

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 number (MAD) and Summed Adjacency Disruption number (SAD), which will be defined in Section II. We focus in Section III 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 IV and V, we prove that, under both models, both the MAD and SAD problems are **APX**-hard when genomes contain duplicates.

II. 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 first to disambiguate the data by inferring homologs, *i.e.*, a non-ambiguous 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 to 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 $\mathcal{M} = \{(i_1, j_1), (i_2, j_2), \ldots, (i\ , j\ )\}$ such that

$G_1[i'] = G_2[j']$ 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 Dist(G, a, b), is defined by Dist$(G, a, b) = |j - i|$.

Given two genomes containing duplications, a first step is thus to establish a non-ambiguous 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 $\mathcal{M} = \{(i_1, j_1), (i_2, j_2), \ldots, (i_\ell, j_\ell)\}$ between G_1 and G_2 such that (i) $G_1[i'] \neq G_1[i'_\ell]$ for all $1 \leq \ell < \ell' \leq k$ and (ii) each gene family occurs in one pair of $\mathcal{M}$. In the *matching* model, the goal is to map as many genes as possible, *i.e.*, 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), 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 conversely.

We now turn to precisely define the three similarity measures we are interested in. As mentioned before, we assume now that the two genomes are duplication-free, *i.e.*, 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, *i.e.*, $G_1 = 1\ 2\ \ldots\ n$.

a) Number of common intervals: A *common interval* between G_1 and G_2 is a substring of G_1, *i.e.*, 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 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 $\frac{n(n+1)}{2}$, *i.e.*, each possible substring of G_1 is a common interval.

b) Maximum Adjacency Disruption Number (MAD): This notion has been introduced by Sankoff and Haque [22]. The MAD number between G_1 and G_2, denoted MAD(G_1, G_2), is defined by

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

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

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 MAD$(G_1, G_2) = \max\{4, 3\} = 4$.

c) Summed Adjacency Disruption Number (SAD): This notion has also been 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 SAD(G_1, G_2), is defined by SAD$(G_1, G_2) = \sum_{1 \leq i \leq n-1} \text{Dist}(G_2, G_1[i], G_1[i+1]) + \sum_{1 \leq i \leq n-1} \text{Dist}(G_1, G_2[i], G_2[i+1])$.

Going back to our example $G_1 = 1\,2\,3\,4\,5$ and $G_2 = 1\,4\,3\,5\,2$, one obtains 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, *i.e.*, the associated optimization problem is a minimization one ; on the contrary, the number of common intervals is a *similarity* measure, *i.e.*, the associated optimization problem is a maximization one.

III. 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 (resp. MCOMI) denote the problem of computing the maximum number of common intervals in the exemplar (resp. 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 (i.e., 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 $V = \{v_1, v_2 \dots v_n\}$ and $E = \{e_1, e_2 \dots e_k\}$), 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:

$$G_1 = b_1,\ b_2 \dots b_k,\ x,\ a_1,\ C_1,\ a_2,\ C_2 \dots a_n,\ C_n,\ y,\ b_{k+n},\ b_{k+n-1} \dots b_{k+1}.$$

The a_is, the b_is, x and y are genes, while 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 $1 \leq i \leq n$, $C_i = (a_i + 1),\ (a_i + 2) \dots (a_i + 2m)$;

- for any $1 \leq i \leq n+m$, $b_i = a_n + 2m + i$;
- $x = b_{n+m} + 1$;
- $y = b_{n+m} + 2$.

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:

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

Proof: We will first prove that, for any exemplar genome G_2 obtained from G_2, any interval of size greater than or equal to 2 that contains x (resp. y) also contains y (resp. 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 . Now let us look at what other genes I could contain in G_1:

- If I contains $b_{i'}$ in G_1, since $b_{i'}$ belongs to a D'_j in G_2 , this means that I contains $b_{i'+n}$ in G_2 , and thus contains y in G_1, a contradiction.
- If I contains a_1 in G_1, I contains in particular $b_{i'+n}$ in G_2 , and thus contains y in G_1, a contradiction.

Hence, any common interval I 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 $b_{i'+n}$ in G_1, then it contains all the D'_js in G_2 , and in particular it contains all 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 $b_{i'+n-1}$ in G_2 , and thus contains $b_{i'+n}$ in G_1. We are back to the previous case.

Hence, the only common interval containing x (resp. y) is the whole genome G_1. Thus, if there are common intervals that are non-trivial, they must be, in G_1, either strictly on the left of x, strictly between x and y or strictly on the right of y. We will investigate separately 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 , any such interval would contain at least one gene in a given D'_j which occurs only in C_j 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 , a contradiction.
3) Intervals strictly between x and y in G_1: independently of the way G_2 is exem-

plarized, 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 must be taken within a given substring of the form $a_i C_i$ $(1 \leq i \leq n)$ in G_1, and the lemma is proved.

■

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. But 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 $|\Delta_i| \leq 2m-1$, this cannot happen. Hence Δ_i is not a common interval. ■

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

(1) In G_2, all the b_js have been deleted from D'_i, and in that case there are exactly two non-trivial common intervals involving D'_i.

(2) In G_2, at least one b_j has been left within D'_i, and in that case there is no non-trivial common interval involving D'_i.

Proof: By Lemma 1, we know that any non-trivial interval is composed in G_1 of elements of the sequence $a_i C_i$, for any $1 \leq i \leq n$. Hence, it is composed, in any exemplar genome G_2 obtained from G_2, of elements of the sequence $a_i D'_i$, for any $1 \leq i \leq n$.

Suppose first that all the b_js in D'_i have been deleted in our exemplar genome G_2, thus

transforming it into the exemplar subsequence D_j. By Lemma 1, we know that any non-

with those deletions, we end up with an exemplar genome G_2. In G_2, we have at least $(n-k)$ substrings of the form D'_i for which all the b_js have been deleted. Thus, by Lemma 3, we know they each imply two non-trivial 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 $\mathcal{I} = 2(n-k) + I_{\boldsymbol{T}}$ common intervals between G_1 and G_2.

($\Leftarrow$) Suppose there exists an exemplar genome G_2 obtained from G_2, and having at least $\mathcal{I} = 2(n-k) + I_{\boldsymbol{T}}$ common intervals. Then, there are at least $2(n-k)$ non-trivial common intervals. However, by Lemma 1, we know that they can only occur within the substrings $a_i\,C_i$, $1 \leq i \leq n$, in G_1, that is within the substrings $a_i\,D'_i$, $1 \leq i \leq n$, in G_2. By Lemma 3, we know that in at least $(n-k)$ such substrings, all the b_js, $1 \leq j \leq m$ have been deleted. Since G_2 is exemplar, this means that the b_js have remained in at most k substrings of the form $a_i\,D'_i$. 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, and since they are included in at most k substrings of the form $a_i\,D'_i$, we conclude that those substrings imply a Vertex Cover, of size at most 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 $occ(G_1) = 1$. 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 of Blin and Rizzi [18], in which the authors studied *conserved intervals*, a measure which 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, though 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. ■

IV. MAXIMUM ADJACENCY DISRUPTION (MAD)

Let EMAD (resp. MMAD) denote the problem of computing the minimum MAD number in the exemplar (resp. 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 (resp. MMAD) admits a $(2-\varepsilon)$-approximation algorithm, unless **P**=**NP**. This inapproximability result does not rely on the **PCP** theorem. We will also remark however, how, reconsidering the reduction proposed in view of **APX**-hardness results based on the **PCP** theorem, one can 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 (resp. 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 (resp. 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 3 positive and at most 2 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 3 occurrences within C and is called $(2,2)$-*bounded* if no variable has more than 2 positive occurrences and no more than 2 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 $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$, 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 $\mathcal{P} = \mathcal{P}_1 \cup \mathcal{P}_2$,

where $\mathcal{P}_{\mathcal{C}} = \{(y_i^j \vee y_i^{j+1 \bmod 4}), (\neg y_i^j \vee \neg y_i^{j+1 \bmod 4}) : i = 1, 2, \ldots, n,\ j = 0, 1, 2, 3\}$ and $\mathcal{P}_{\mathcal{L}} = \{P_1, P_2, \ldots, P_m\}$, 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 ℓ is the t-th positive (or the t-th 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} (resp., 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 $\mathcal{P}_{\mathcal{C}}$ 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 $t_Y(y_i^1) := t_X(y_i^3) := t_X(x_i)$ 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 $\mathcal{P}_{\mathcal{C}}$ 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 $t_Y(y_i^0), t_Y(y_i^2), t_Y(y_i^1), t_Y(y_i^3)$, we can always assume that t_Y is canonical. Indeed, at

least two extra clauses from $\mathcal{P}$ get satisfied in restoring the consistency among the variables y_j^0, y_j^1, y_j^2 and y_j^3 while, at the same time, since at most two truth values have been affected, at most two clauses from $\mathcal{P}_i$ may lose in satisfaction. In other words, we can make t canonical by a majority vote on y_j^0, y_j^1, y_j^2 and y_j^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. Once t 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 above does not need the **PCP** theorem and is all what is required in the following for proving that, for no $\varepsilon > 0$, EMAD (resp. MMAD) admits a $(2 - \varepsilon)$-approximation algorithm, unless **P**=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 (resp. 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 $(2, 2)$-bounded 3SAT which rests on algorithm A as a subroutine. The **APX**-hardness of UNIFORM-SAT then follows from the **APX**-hardness of $(2, 2)$-bounded 3SAT.

After receiving in input an instance $\langle X, \mathcal{C} \rangle$ of $(2, 2)$-bounded 3SAT, we construct the instance $\langle Y, \mathcal{P} \rangle$ of UNIFORM-SAT as described above. Assume the optimal truth assignment t_X 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, and since each clause contains at most

3 literals. Moreover, by Lemma 5, there exists a truth assignment t 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 over Y satisfying at least $(8\,n + opt)(1-\varepsilon)$ clauses in $\mathcal{P}$. And Lemma 6 (whose proof can be easily converted into a polynomial time algorithm) shows how, starting from this truth assignment t, one can obtain a truth assignment t over X such that the clauses in $\mathcal{C}$ which are satisfied under t 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 comprising only positive literals and the family $\mathcal{N} = \{N_1, N_2, \ldots, N_{m_n}\}$ of clauses comprising only negated literals. Let M be a sufficiently big positive integer that we will fix later 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 $L_1 = 2\,M + m + m_n + n - 1$ defined as follows: $G_1 = 1\ 2\ 3 \ldots L_1$. A gene at position i in G_1 with $i \leq m$ or $i \geq L_1 - m_n + 1$ is called a $*$-*gene*. Genome G_2 has length $L_2 = 2\,M + 6\,n - 1$, 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.

$$G_2 = m+1, m+2, \ldots, m+M\ , *, *, *, *, *, m+M+1, *, *, *, *, *, m+M+2, \ldots$$
$$\ldots, *, *, *, *, *, m+M+n, m+M+n+1, m+M+n+2, \ldots, m+2\,M+n-1.$$

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 $1,1$-correspondence with the n variables in X, so that the i-th 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. E.g., 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 i-th run of five consecutive $*$-genes consists, more precisely, in 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, while each gene appears at most 9 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 $t : X \mapsto \{0, 1\}$ be a satisfying truth assignment for $\langle X, \mathcal{C} \rangle$. Then there exists an exemplar subgenome G_2 of G_2 whose MAD number satisfies $\mathrm{MAD}(G_1, G_2) \leq M + m + m_n + n$.

Proof: For each clause $P_j \in \mathcal{P}$, choose a variable x_i occurring in P_j and such that $t(x_i) = 1$ (remember that t is a satisfying truth assignment) and color with red one copy of gene j occurring within the i-th 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 $t(x_i) = 0$ (again, at least one such variable must exist since t is a satisfying truth assignment) and color with red one copy of gene $(L_1 - m_n) + j = 2M + m + n - 1 + j$ occurring within the i-th run of five consecutive $*$-genes in G_2. Now, obtain G_2 from

G_2 by deleting all the $*$-genes, except those marked red. Notice that G_2 is indeed an exemplar genome on the genes $1, 2, \ldots, L_1$.

We now verify that MAD$(G_1, G_2) \leq M + m + m_n + n$

for each $i = 1, 2, \ldots, n$, it must be the case that in the i-th run of five consecutive $*$-genes in G_2, either the genes $\mathcal{P}'_j[1]$, $\mathcal{P}'_j[2]$, $\mathcal{P}'_j[3]$ have all been deleted, or the genes $\mathcal{N}'_j[1] + L_1 - m_n$ and $\mathcal{N}'_j[2] + L_1 - m_n$ have both been deleted. Consider the truth assignment $t : X \mapsto \{0, 1\}$ such that, for each $i = 1, 2, \ldots, n$, $t(x_i) = 1$ iff both $\mathcal{N}'_j[1] + L_1 - m_n$ and $\mathcal{N}'_j[2] + L_1 - m_n$ have been deleted. We claim that $t(x_i)$ 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. This copy must come from one of the runs of five consecutive $*$-genes in G_2, say from the i-th run. It follows that x_i occurs in P_j and that $t(x_i) = 1$. Similarly, for each clause $N_j \in \mathcal{N}$, we know that at least a copy of the gene $(L_1 - m_n) + j$ (i.e., of gene $2M + m + n - 1 + j$) has been retained in G_2. This copy must come from one of the runs of five consecutive $*$-genes in G_2, say from the i-th run. It follows that x_i occurs in N_j and that $t(x_i) = 0$. ■

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

Proof: We proceed as follows: we assume we are given a $(2 - \varepsilon)$-approximation algorithm A for EMAD (resp. 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 in input an instance $\langle X, \mathcal{C} \rangle$ of UNIFORM-SAT, we construct the instance $\langle G_1, G_2 \rangle$ of EMAD (resp. MMAD) as described above. If $\langle X, \mathcal{C} \rangle$ is satisfiable, then, by Lemma 7, there exists an exemplar subgenome G'_2 of G_2 such that MAD$(G_1, G'_2) \leq M + m + m_n + n$. By running algorithm A, we are hence guaranteed to find an exemplar subgenome G''_2 of G_2 such that MAD$(G_1, G''_2) \leq (M + m + m_n + n)(2 - \varepsilon) \leq 2M + 2m + 2m_n + 2n - \varepsilon M$. Now, after choosing $M \geq \frac{2p+2p_n+2n}{\varepsilon}$, we conclude that the solution G''_2 produced by algorithm A satisfies MAD$(G_1, G''_2) \leq 2M$. And Lemma 8 (whose proof can be easily converted into a polynomial time algorithm) shows how, starting

from G , 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, MAD$(G_1, G$ $) \geq 2\,M\, + n$ 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 (resp. MMAD) admits no c-approximation algorithm unless **P**=**NP**. We can get to this stronger conclusion if in the proof of Theorem 5 above we mod (we)202orem

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:

$$G_2 = n+1, n+2, \ldots, n+M, s_1^1, s_2^1, \ldots, s^1, n+M+1, s_1^2, s_2^2, \ldots, s^2, n+M+2, \ldots$$
$$\ldots, s_1^{j-1}, s_2^{j-1}, \ldots, s^{j-1}, n+M+m-1, s_1^j, s_2^j, \ldots, s^j, n+M+m.$$

The above is clearly a polynomial time reduction; besides, 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_2' of G_2 whose SAD number satisfies $\text{SAD}(G_1, G_2') \leq 2\,s\,M + 5\,M$.

Proof: For each element $v \in V$, choose a set S_j in $\mathcal{S}'$ such that $v \in S_j$, i.e., $s_j^j = v$ 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_2' from G_2 by deleting all the copies of the first n genes, except those marked with red. Notice that G_2' is indeed an exemplar genome on the genes $1, 2, \ldots, L_1$.

We now verify that $\text{SAD}(G_1, G_2') \leq 2\,s\,M + 5\,M$. Indeed,

$$\text{SAD}(G_1, G_2') =$$
$$\sum_{j=1}^{M+j+n-1} \text{Dist}(G_2', G_1[i], G_1[i+1]) +$$
$$\sum_{j=1}^{M+j+n-1} \text{Dist}(G_1, G_2'[i], G_2'[i+1])$$

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

$$\begin{aligned}
&\sum_{i=1}^{M+m+n-1} \textsf{Dist}(G_2, G_1[i], G_1[i+1]) \leq \\
&\qquad \sum_{i=1}^{n-1} \textsf{Dist}(G_2, G_1[i], G_1[i+1]) + \\
&\qquad\qquad (M+m+n) + \\
&\qquad \sum_{i=n+1}^{M+n-1} \textsf{Dist}(G_2, G_1[i], G_1[i+1]) + \\
&\qquad \sum_{i=M+n}^{M+m+n-1} \textsf{Dist}(G_2, G_1[i], G_1[i+1]) \\
&\leq n(n+m) + (M+m+n) + M + mn \\
&\leq 2\,M + 3\,mn^2 \\
&\leq 2\,M + m^2n^2 \\
&\leq 3\,M\,,
\end{aligned}$$

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

$$\begin{aligned}
&\sum_{i=1}^{M+i+n-1} \text{Dist}(G_1, G_2[i], G_2[i+1]) = \\
&\quad \sum_{i=1}^{M-1} \text{Dist}(G_1, G_2[i], G_2[i+1]) + \\
&\sum_{i=M}^{M+i+n-1} \text{Dist}(G_1, G_2[i], G_2[i+1]) = \\
&\qquad (M-1) + \\
&\quad \sum_{i=M}^{M+i+n-1} \text{Dist}(G_1, G_2[i], G_2[i+1]) \\
\leq M + &\sum_{i \in \mathcal{S}'} 1 + \sum_{i \in \mathcal{S}'} (2(M+m+n+n^2)) \\
&\leq M + m + 2\,s\,(M+m+n+n^2) \\
&\leq M + 2\,s\,M + m^2 n^2 \\
&\leq 2\,s\,M + 2\,M\,.
\end{aligned}$$

To better explain the upper bound on the term $\sum_{i=M}^{M+i+n-1} \text{Dist}(G_1, G_2[i], G_2[i+1])$ 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. (Thus, $p_0 = M$ and $p_n = M+n+m$). Clearly,

$\sum_{i=M}^{M+i+n-1} \text{Dist}(G_1, G_2[i], G_2[i+1]) = \sum_{i=1}^{i} \sum_{j=i-1}^{i-1} \text{Dist}(G_1, G_2[j], G_2[j+1])$.

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 and in G_1, whence $\sum_{j=i-1}^{i-1} \text{Dist}(G_1, G_2[j], G_2[j+1]) = 1$. Also, for each $i = 1, 2, \ldots, m$, $\text{Dist}(G_1, G_2[p_i - 1], G_2[p_{i-1} + 1]) \leq M+m+n$ and $\text{Dist}(G_1, G_2[p_i - 1], G_2[p_i]) \leq M+m+n$, since $M+m+n$ is the length of G_1. Furthermore, $p_i - p_{i-1} \leq 1 + k \leq n$, and, for each $j = 1, 2, \ldots, p_i - p_{i-1} - 2$, $\text{Dist}(G_1, G_2[p_{i-1} + j], G_2[p_{i-1} +$

$j+1]) \leq n$. ■

Lemma 10: For any exemplar subgenome G_2' of G_2 such that SAD$(G_1, G_2') < 2\,s\,M$, we can derive, from G_2' 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 $S_{j'} \in \mathcal{S}$ for which there exists a $v \in S_{j'}$, say $v = s_{i'}^{j'}$, such that, in obtaining G_2' 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'. Moreover, $|\mathcal{S}'| \leq s$ follows from SAD$(G_1, G_2') < 2\,s\,M$. Indeed, $\sum_{i'=M}^{M+n-1}$ Dist$(G_1, G_2'[i], G_2'[i+1]) \leq SAD(G_1, G_2') \leq 2\,s\,M$. However, for every i such that $S_{j'} \in \mathcal{S}'$, the genes $M+n+(i-1)$ and $M+n+i$ are not consecutive in G_2'. Let us denote with $p_{j'-1}$ and $p_{j'}$ the absolute positions of genes $M+n+(i-1)$ and $M+n+i$ within the genome G_2'. Thus, whenever $S_{j'} \in \mathcal{S}'$, then $p_{j'} > p_{j'-1} + 1$ and we have Dist$(G_1, G_2'[p_{j'-1}], G_2'[p_{j'-1}+1]) \geq M$ and Dist$(G_1, G_2'[p_{j'}-1], G_2'[p_{j'}]) \geq M$, since $G_2'[p_{j'-1}+1] \leq n$ and $G_2'[p_{j'}-1] \leq n$. It follows that $|\mathcal{S}'| \leq \frac{2\,M}{2\,M} = s$. ■

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

Proof: It is well known that 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 the number of sets (see [27]) for some $c > 0$. To be more precise, it has been proved in [27] that the instance of 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 (resp. MSAD) admits no $\left(\frac{c'}{4} \log |G_1|\right)$-approximation algorithm.

We proceed as follows: we assume we are given a $\left(\frac{c'}{4}\,\log|G_1|\right)$-approximation algorithm A for ESAD (resp. 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 (resp. MSAD) as described above. Notice that $|G_1|\leq 2\,M$ and hence $\log|G_1|\leq\log 2m^2n^2\leq 3\,(\log m+\log n)$. 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 of G_2 such that $\mathrm{SAD}(G_1,G_2)\leq 2\,opt\,M+5\,M$. By running algorithm A, we are hence guaranteed to find an exemplar subgenome G of G_2 such that $\mathrm{SAD}(G_1,G)\leq(2\,opt\,M+5\,M)\frac{c'}{4}\log|G_1|\leq\left(\frac{8}{3}\,opt\,M\right)\frac{c'}{4}\,3\,(\log m+\log n)\leq(2\,opt\,M)\,c'\,(\log m+\log n)$. 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 into a polynomial time algorithm) shows how, starting from G, one can obtain a set cover S' with $|\mathcal{S}'|\leq\frac{1}{2\,M}(2\,opt\,M)\,c'\,(\log m+\log n)=opt\,c'\,(\log m+\log n)$. ■

VI. 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 interesting to end this paper by giving an overview of 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 I, while the ones concerning the matching model are summarized in Table II.

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, 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, conserved and common intervals), similarly to the complexity results, we know that the problem is **APX**-hard even when $occ(G_1) = 1$ and $occ(G_2) = 2$, while in the others, the value of $occ(G_2)$ is either unbounded (SAD) or bounded by constant 9 (MAD).

VII. 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, Maximum Adjacency Disruption and Summed Adjacency Disruption. 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 I and II even more precise. In particular: (i) complete the cases for which no result is known or a gap exists (that is, number of reversals) ; (ii) study more deeply the complexity and approximability results with respect to parameter f ; (iii) 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," in *CompBioNets 2004: Algorithms & computational methods for biochemical and evolutionary networks*, ser. Texts in Algorithmics, vol. 3. King's Coll. Pub., London, 2004, pp. 3–16.

[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. Natl. Acad. Sci. U. S. A.*, 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 Res.*, vol. 15, no. 1, pp. 98–110, 2005.

[5] D. Sankoff, "Gene and genome duplication," *Curr. Opin. Genet. Dev.*, 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," *Mol. Biol. Evol.*, 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," in *Comparative Genomics, RECOMB 2006 International Workshop, RCG 2006*, ser. Lecture Notes in Bioinformatics, vol. 4205. Springer, Berlin, 2006, pp 24–38.

[9] D. Bryant, "The complexity of calculating exemplar distances," in *Comparative Genomics: Empirical and Analytical Approaches to Gene Order Dynamics, Map Alignment, and the Evolution of Gene Families*. Kluwer Acad. Pub., Dordrecht, 2000, pp. 207–212.

[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 Transactions on Computational Biology and Bioinformatics*, vol. 2, no. 4, pp. 302–315, 2005.

[11] K. Swenson, M. Marron, J. Earnest-DeYoung, and B. Moret, "Approximating the true evolutionary distance between two genomes," in *Proceedings of the Seventh Workshop on Algorithm Engineering and Experiments and the Second Workshop on Analytic Algorithmics and Combinatorics (ALENEX/ANALCO)*. SIAM Press, New York, 2005, pp. 121–129.

[12] N. C. Thach, "Algorithms for calculating exemplar distances," 2005, honours Year Project Report, National University of Singapore.

[13] Z. Chen, B. Fu, and B. Zhu, "The approximability of the exemplar breakpoint distance problem," in *Algorithmic Aspects in Information and Management, Second International Conference, AAIM 2006, Hong Kong, China, June 20-22, 2006, Proceedings*, ser. Lecture Notes in Comput. Sci., vol. 4041. Springer, Berlin, 2006, pp. 291–302.

[14] M. Marron, K. Swenson, and B. Moret, "Genomic distances under deletions and insertions," *Theoret. Comput. Sci.*, vol. 325, no. 3, pp. 347–360, 2004.

[15] J. Tang and B. Moret, "Phylogenetic reconstruction from gene-rearrangement data with unequal gene content," in *Algorithms and Data Structures, 8th International Workshop, WADS 2003, Ottawa, Ontario, Canada, July 30 - August 1, 2003, Proceedings*, ser. Lecture Notes in Comput. Sci., vol. 2748. Springer, Berlin, 2003, pp. 37–46.

[16] Z. Fu, X. Chen, V. Vacic, P. Nan, Y. Zhong, and J. Tang, "A parsimony approach to genome-wide ortholog assignment," in *Research in Computational Molecular Biology, 10th Annual International Conference, RECOMB 2006, Venice, Italy, April 2-5, 2006, Proceedings*, ser. Lecture Notes in Bioinformatics, vol. 3909. Springer, Berlin, 2006, pp. 578–594.

[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," in *Computing and Combinatorics, 11th Annual International Conference, COCOON 2005, Kunming, China, August 16-29, 2005, Proceedings*, ser. Lecture Notes in Comput. Sci., vol. 3595. Springer, Berlin, 2005, pp. 22–31.

[19] G. Bourque, Y. Yacef, and N. El-Mabrouk, "Maximizing synteny blocks to identify ancestral homologs," in *Comparative Genomics, RECOMB 2005 International Workshop, RCG 2005, Dublin, Ireland, September 18-20, 2005, Proceedings*, ser. Lecture Notes in Bioinformatics, vol. 3678. Springer, Berlin, 2005, pp. 21–34.

[20] Z. Chen, R. Fowler, B. Fu, and B. Zhu, "Lower bounds on the approximation of the exemplar conserved interval distance problem of genomes," in *Computing and Combinatorics, 12th Annual International Conference, COCOON 2006*, ser. Lecture Notes in Comput. Sci., vol. 4112 Springer, Berlin, 2006, pp 245–254.

[21] S. Angibaud, G. Fertin, and I. Rusu, "On the inapproximability of similarity measures between genomes containing duplicates," 2006, in Preparation.

[22] D. Sankoff and L. Haque, "Power boosts for cluster tests," in *Comparative Genomics, RECOMB 2005 International Workshop, RCG 2005, Dublin, Ireland, September 18-20, 2005, Proceedings*, ser. Lecture Notes in Bioinformatics, vol. 3678. Springer, Berlin, 2005, pp. 121–130.

[23] A. Bergeron and J. Stoye, "On the similarity of sets of permutations and its applications to genome comparison," in *Computing and Combinatorics, 9th Annual International Conference, COCOON 2003, Big Sky, MT, USA, July 25-28, 2003, Proceedings*, ser. Lecture Notes in Computer Science, vol. 2697. Springer, Berlin, 2003, pp. 68–79.

[24] M. Garey and D. Johnson, *Computers and Intractability: a guide to the theory of NP-completeness*. San Franciso: W.H. Freeman, 1979.

[25] C.H. Papadimitriou and M. Yannakakis, "Optimization, approximation, and complexity classes," *J. Comput. System Sci.*, 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 and C .Stamm and P .Widmayer, "Positioning Guards at Fixed Height above a Terrain - An

Optimum Inapproximability Result," in *6th European Symposiumal on Algorithms, ESA 98*, ser. Lecture Notes in Comput. Sci., vol. 1461 Springer, Berlin, 1998, pp 187–198.

[28] S. Angibaud, G. Fertin, I. Rusu, and S. Vialette, "How pseudo-boolean programming can help genome rearrangement distance computation," in *Comparative Genomics, RECOMB 2006 International Workshop, RCG 2006*, ser. Lecture Notes in Bioinformatics, vol. 4205. Springer, Berlin, 2006, pp 75–86.

Guillaume Blin 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 at the University of Marne-La-Vallee (FRANCE). His current research interests include computational complexity and approximation, algorithms, and bioinformatics.

Cedric Chauve Cedric Chauve obtained a PhD in computer science from the University of Bordeaux I (France) in 2000. From 2001 to 2006 he was professor in the department of computer science at University of Québec in Montréal and he is now associate professor in the department of mathematics in Simon Fraser University. His current research interest is the development of combinatorial models and effi cient algorithms for comparative genomics.

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

Romeo Rizzi Romeo Rizzi was born in 1967. He received the Laurea degree in Electronic Engineering from the Politecnico di Milano in 1991, and in 1997 received a Ph.D. in Computational Mathematics and Informatics from the University of Padova, Italy. Afterwards, he held Post-Doc and other temporary positions at research centers like CWI (Amsterdam, Holland), BRICS (Aarhus, Denmark) and IRST (Trento, Italy). In 2001, he became Assistant Professor by the University of Trento. Since 2005, he is Associate Professor by 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 scientifi c 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 Stéphane Vialette received the PhD degree in 2001 from the University Paris 7. He has been postdoctorate at Laboratoire de Génétique Moléculaire at Ecole Normale Supérieure for two years. Since 2003 he has been an assistant professor in the bioinformatics group from the computer science research laboratory (LRI) at University Paris-Sud 11. His research interests are in computational biology, algorithmics and combinatorics.

TABLE I

RESULTS CONCERNING THE EXEMPLAR MODEL.

(*****) 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 VERTEXCOVER instead of SETCOVER.

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 $occ(G_1) = 1$

TABLE II

RESULTS CONCERNING THE MATCHING MODEL.

(*****) 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 VERTEXCOVER instead of SETCOVER.

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 $occ(G_1) = 1$