An algorithmic view on multi-related-segments: A unifying model for approximate common interval

Xiao Yang, Florian Sikora, Guillaume Blin, Sylvie Hamel, Roméo Rizzi, Srinivas Aluru

An Algorithmic View on Multi-related-segments: a unifying model for approximate common interval

Xiao Yang[1], Florian Sikora[2,5], Guillaume Blin[2], Sylvie Hamel[3], Romeo Rizzi[4], and Srinivas Aluru[6]

[1] GSAP, Broad Institute of MIT & Harvard, USA, xiaoyang@broadinstitute.org
[2] Université Paris-Est, LIGM, UMR 8049 - France, {sikora,gblin}@univ-mlv.fr
[3] DIRO - Université de Montréal - QC - Canada, hamelsyl@iro.umontreal.ca
[4] DIMI - Università di Udine - Udine - Italy, romeo.rizzi@uniud.it
[5] Lehrstuhl für Bioinformatik, Friedrich-Schiller-Universität Jena - Germany
[6] DECE, Iowa State University, USA, aluru@iastate.edu

Abstract. A set of genes that are proximately located on multiple chromosomes often implies their origin from the same ancestral genomic segment or their involvement in the same biological process. Among the numerous studies devoted to model and infer these gene sets, the recently introduced APPROXIMATE COMMON INTERVAL (ACI) models capture gene loss events in addition to the gene insertion, duplication and inversion events already incorporated by earlier models. However, the computational tractability of the corresponding problems remains open in most of the cases. In this contribution, we propose an algorithmic study of a unifying model for ACI, namely MULTI-RELATED-SEGMENTS, and demonstrate that capturing gene losses induces intractability in many cases.

1 Introduction

The genetic blueprint of an organism is encoded in a set of DNA sequences, known as chromosomes. During evolution, some subsequences of a chromosome diverged while others, known as *genes*, were conserved among different organisms. A chromosome is typically represented as a sequence of genes, then evolution is described as a series of discrete events: gene insertion, loss, duplication and inversion. One of the most important goals in comparative genomics is to identify a set of genes that are in proximate locations on multiple chromosomes and their actual chromosomal occurrences. Indeed, preservation of gene co-locality tends to indicate that the corresponding genes either form a functional unit (*e.g.*, operons) or result from speciation or duplication events [12]. In the literature, the former is termed "*gene cluster*" [3], whereas the latter is known as "*synteny*" [22]. Both were extensively studied during the past decade, and numerous models and algorithms were proposed to define and identify them. From an algorithmic point of view, we present a unified model to capture approximate common intervals and provide tractability results in association with evolutionary events.

2 Gene Proximity: Properties and Models

Modeling gene proximity based on biological intuition is known to be difficult, but some key properties have been raised by Hoberman and Durand [12]. We present a formalization of these properties by developing the notion of MULTI-RELATED-SEGMENTS [20,21], meanwhile, show that some of them are inadequately captured by existing models. We consider here related algorithmic aspects.

2.1 Key Properties of Gene Proximity

Observing the co-occurrence of a gene set $\mathcal{A}$ in different chromosomal segments indicates the common origin of these segments. Genes in $\mathcal{A}$ are referred as *ancestral genes*. Naturally, these segments of interest are subject to evolutionary constraints. The first crucial constraint consists in *evidence of any gene of interest as being ancestral*. This property is usually related to observing a minimum β occurrences of such a gene among the segments, thereby reducing the possibility of misinterpreting what is in fact a chance occurrence. Secondly, each segment *contributes sufficiently* to the ancestral gene set. More formally, each segment contains at least ϵ_m different ancestral genes. Then, consider gene loss and insertion events that may have occurred, such an segment may not necessarily contain all ancestral genes while each may pick up genes independently. To constrain the frequency of these events so that the signal of common origin can still be detected, *local* and *global ancestral gene density* constraints apply. The former is captured by allowing at most α interleaving genes between two consecutive ancestral genes, while the latter is captured by allowing a maximum ϵ_l gene losses in each segment and a maximum ϵ_t total gene losses among all segments.

2.2 Existing Models

Consider k chromosomes, each represented as a permutation over a given gene set $\mathcal{A}$. A CONSERVED SEGMENT [14] consists of a set of genes that occur consecutively in the same order on every input chromosome. Once the constraint of the preserved ordering is removed, it leads to the COMMON INTERVAL (CI) model definition [19]. If the unordered pair of the first and the last genes of a CI is the same on each chromosome, this CI is moreover called *conserved* [5]. Furthermore, if we relax the constraint that genes in a CI have to be consecutive in each chromosomal occurrence – namely, two genes belonging to a CI can be interleaved by a bounded number of genes not belonging to it – the definition of GENE-TEAMS (GT) model [4] follows. The GT model is of higher biological relevance since it in addition captures gene insertions. The aforementioned models can

be applied to strings to account for gene duplications, but the number of resulting gene sets complying the model may increase exponentially. More recently, APPROXIMATE COMMON INTERVAL (ACI) models were introduced [1,17,7,13], where not all ancestral genes need to occur in every segment. Among these, MEDIAN GENE CLUSTER (MGC) model [7] is the most recent formulation, but the complexity of this model remains open.

3 Multi-related segments model

We now present MULTI-RELATED-SEGMENTS (MRS) model [20,21], which is defined as consisting of a set of segments of interest, each evolved from an ancestral segment with gene set $\mathcal{A}$ via gene insertion, loss, duplication, and inversion events. Formally, a MRS is defined as follows. To ensure *evidence of being ancestral genes*, each gene in $\mathcal{A}$ occurs in at least β (≥ 2) segments. Each segment of interest has to contain at least ϵ_m different ancestral genes and is maximal (*i.e.*, not extendable by including surrounding genes) – thus, imposing a constraint on the *minimum contribution to* $\mathcal{A}$. Similar to the GT model, the *local ancestral gene density* is constrained allowing at most α non-ancestral genes between any two consecutive ancestral genes. To control *global ancestral gene density*, we require each segment to induce no more than ϵ_l gene losses and the total number of gene losses of all segments to be lower than ϵ_t. Then, given a set of chromosomes and parameters $\alpha, \beta, \epsilon_m, \epsilon_l$ and ϵ_t, the general problem is to identify all MRS.

The formulation of MRS captures existing models and holds a better biological intuition. MRS corresponds to a CI when $\beta = k$, $\epsilon_m = |\mathcal{A}|$ and $\alpha = 0$, and to a GT when $\alpha \geq 0$. Compared with these two models, MRS further captures gene loss events. Note that this aspect was already considered in the MGC model [7]. Nevertheless, there are several major differences. Firstly, MRS captures the same origin of more than two segments in the absence of strong pairwise similarity information, such as differential gene loss [18] and uber-operon [8] – which is not the case for MGC due to the requirement that segments pairwisely share some common genes. Secondly, the minimum evidence of a gene being ancestral is more flexible in MRS by requiring β occurrences of any ancestral gene – which has to be at least $\frac{k}{2}$ in MGC. Finally, the local ancestral gene density is not required in MGC – which is, as explained in [12], crucial.

From an algorithmic point of view, regarding all above mentioned models, complexity increases when chromosomes are delineated as strings rather than permutations: the problem is still tractable when considering CI [2,9] but folds into the hardness as soon as conserved CI is considered [6]. The GT model, which captures gene insertions, duplications and

inversions, is polynomial on permutations [4] but exponential over strings [16]. Considering the complexity of ACI models, which further captures gene losses, an algorithm with $O(kn^3 + occ)$ run time over strings was proposed [1] where *occ* is bounded by the number of substrings of the genomes. In this paper, we further investigate the complexity of ACI models, by considering from an algorithmic point of view the problem of MRS inference. Since known algorithmic results are available in previous modeling of gene duplications, insertions and inversions, our focus is on deriving if the problem is tractable when trying to model gene losses.

4 Complexity Analysis of MRS

One has to note that, in the following, we will set the numerical parameters of the model to specific values. This just consists in an algorithmic trick for ease of proof. We first consider the case when the ancestral gene set $\mathcal{A}$ is *a priori* known. The problem, termed LOCATEMRS, then corresponds to locate, given k chromosomes $\mathcal{S} = \{S_1, S_2, \ldots, S_k\}$ represented as strings, a feasible MRS originating from $\mathcal{A}$. We prove that this problem is **NP**-hard even in the restricted case where $|\mathcal{CS}(S_i)| = |S_i|$ and meanwhile, at most one substring per S_i can belong to the resulting MRS, for every $S_i \in \mathcal{S}$. It follows that LOCATEMRS problem is **NP**-hard. Next, we prove LOCATEMRS to be fixed-parameter tractable (FPT) [10], and provide an efficient dynamic programming solution. Then, we prove that the optimization problem to identify all MRS is hard to approximate. Finally, we show that with the removal of the maximum number of gene loss constraint and the maximum number of substrings per input sequence constraint, a polynomial algorithm can be derived. Due to space constraint, some proofs are deferred to the full version of the paper.

4.1 Identify A MRS Given $\mathcal{A}$

Let us consider that no gene insertion is allowed ($\alpha = 0$), $\beta \geq 2$ and $\epsilon_t = \epsilon_l = \infty$. Then, by definition, any MRS consists of substrings involving only genes in $\mathcal{A}$. Thus, each input chromosome can be pre-processed in order to remove any gene not belonging to $\mathcal{A}$, resulting in a sequence of substrings. One may then filter out any substring that does not respect the *minimum contribution to* $\mathcal{A}$ criterion (*i.e.*, using ϵ_m). Any remaining substring will be referred as *of interest*. Finally, since in the MRS definition, we are looking for maximal substrings (*i.e.*, not extendable by including surrounding genes), any substring of interest will be either kept or fully removed in the solution.

Definition 1. LOCATEMRS*: Given a character set* $\mathcal{A}$*, a string set* $\mathcal{S} = \{S_1^1, S_1^2, \ldots, S_2^1, S_2^2, \ldots, S_k^1, S_k^2 \ldots\}$ *where* S_j^i *corresponds to the* i^{th} *sub-*

$$\mathcal{S}_{\mathcal{T}} \begin{cases} T_1 = \mathbf{u_1 x_1 u_2 x_2 u_3 x_3} & - \; v_1 w_1 v_2 w_2 v_3 w_3 z_3 \\ T_2 = v_2 x_2 v_3 x_3 u_4 x_4 & - \; \mathbf{u_2 w_2 u_3 w_3 z_3 v_4 w_4 z_4} \\ T_3 = w_3 x_3 v_4 x_4 u_5 x_5 & - \; \mathbf{u_3 v_3 z_3 u_4 w_4 z_4 v_5 w_5 z_5} \\ T_4 = \mathbf{w_4 x_4 v_5 x_5 u_6 x_6} & - \; u_4 v_4 z_4 u_5 w_5 z_5 v_6 w_6 \\ T_5 = v_1 x_1 w_5 x_5 v_6 x_6 & - \; \mathbf{u_1 w_1 u_5 v_5 z_5 u_6 w_6} \end{cases}$$

$$\mathcal{S}_{\mathcal{X}} \begin{cases} S_{1,1} = \mathbf{x_1} & S_{2,1} = \mathbf{x_2} & S_{6,1} = \mathbf{x_6} \\ S_{1,2} = u_1 - \mathbf{w_1} & S_{2,2} = u_2 - \mathbf{w_2} & S_{6,2} = u_6 - \mathbf{w_6} \\ S_{1,3} = u_1 - \mathbf{v_1} & S_{2,3} = u_2 - \mathbf{v_2} & S_{6,3} = u_6 - \mathbf{v_6} \\ S_{1,4} = w_1 - \mathbf{v_1} & S_{2,4} = w_2 - \mathbf{v_2} & S_{6,4} = w_6 - \mathbf{v_6} \\ S_{3,1} = \mathbf{x_3} & S_{4,1} = \mathbf{x_4} & S_{5,1} = \mathbf{x_5} \\ S_{3,2} = u_3 - \mathbf{v_3} & S_{4,2} = \mathbf{u_4} - v_4 & S_{5,2} = \mathbf{u_5} - v_5 \\ S_{3,3} = \mathbf{w_3} - v_3 & S_{4,3} = w_4 - \mathbf{v_4} & S_{5,3} = \mathbf{w_5} - v_5 \end{cases}$$

Fig. 1. Illustration of the construction on the following instance of X3C: $\mathcal{X} = \{1,2,3,4,5,6\}$ and $\mathcal{T} = \{(\mathbf{1},\mathbf{2},\mathbf{3}),(2,3,4),(3,4,5),(\mathbf{4},\mathbf{5},\mathbf{6}),(1,5,6)\}$. A correspondance between the solutions of the problems is highlighted in bold.

string of interest of S_j (i.e., j^{th} chromosome), find a subsets $\mathcal{S}' \subseteq \mathcal{S}$ corresponding to a Mrs*, such that $\mathcal{A} = \bigcup_{S \in \mathcal{S}'} \mathcal{CS}(S)$ and each character of $\mathcal{A}$ appears in at least two elements of $\mathcal{S}'$, and $\forall S_i^a, S_j^b \in \mathcal{S}'$, $i \neq j$.*

We will prove that LocateMrs is hard but fixed-parameter tractable. We first consider that $|S| = |\mathcal{CS}(S)|$ for any $S \in \mathcal{S}$ (*i.e.,* S is a permutation). Note that this problem is in **NP**. Indeed, given a subset $\mathcal{S}'$ of $\mathcal{S}$, one can check in polynomial time that each character of $\mathcal{A}$ appears in at least two elements of $\mathcal{S}'$ and that no more than one substring S_j^i of any S_j belongs to $\mathcal{S}'$. To prove that this problem is moreover **NP**-hard, we provide a polynomial reduction from the **NP**-complete problem Exact Cover by 3-sets (X3C) [11]: Given a finite set $\mathcal{X} = \{x_1, \ldots, x_{|\mathcal{X}|}\}$ and a family $\mathcal{T} = \{t_1, \ldots, t_{|\mathcal{T}|}\}$ of triples over $\mathcal{X}$, is there a subfamily $\mathcal{T}' \subseteq \mathcal{T}$ such that every $x_i \in \mathcal{X}$ is contained in exactly one element of $\mathcal{T}'$?

X3C problem is hard even in the special case where each element of $\mathcal{X}$ appears at most three times in $\mathcal{T}$ [11]. Then, it is sufficient to consider the case where each element appears either two or three times. Indeed, any triple containing some element that occurs only once has to be part of any solution and can be removed from further consideration. According to the problem definition, a solution corresponds to a selection of one among the at most three occurrences of any element of $\mathcal{X}$. Without loss of generality, we fix the triple order in $\mathcal{T}$.

Let us now provide the construction from any instance $(\mathcal{X}, \mathcal{T})$ of X3C problem (an example is given in Figure 1). For each element $x_i \in \mathcal{X}$, let x_i, u_i, v_i, w_i and z_i be some characters. The set $\mathcal{S}$ will be built on

$|\mathcal{T}|$ sequences, which represent the triples of $\mathcal{T}$, and four (resp. three) additional sequences, which represent any element of $\mathcal{X}$ occurring twice (resp. three times) in $\mathcal{T}$. Let $\mathcal{S}_\mathcal{T} = \{T_1, \dots T_{|\mathcal{T}|}\}$ (resp. $\mathcal{S}_\mathcal{X}$) be the set of sequences representing the triples (resp. the elements of $\mathcal{X}$). Moreover, we use the symbol "–" to separate the non-adjacent substrings in a given string, *e.g.*, $S = S^1 - S^2 - S^3$. Note that, the order of the characters in these substrings is not important according to the definition of Mrs. Let us first construct $\mathcal{S}_\mathcal{T}$ as follows. First, for each element $x_i \in \mathcal{X}$ occurring twice in $\mathcal{T}$, concatenate $u_i x_i$ (resp. $v_i x_i$) to T_j^1 (initially empty) and $v_i w_i$ (resp. $u_i w_i$) to T_j^2 (initially empty) if the first (resp. second) occurrence of x_i appears in the j^{th} triple of $\mathcal{T}$. Second, for each element $x_i \in \mathcal{X}$ occurring three times in $\mathcal{T}$, concatenate $u_i x_i$ (resp. $v_i x_i$ and $w_i x_i$) to T_j^1 and $v_i w_i z_i$ (resp. $u_i w_i z_i$ and $u_i v_i z_i$) to T_j^2 if the first (resp. second and third) occurrence of x_i appears in the j^{th} triple of $\mathcal{T}$. Let us now construct the set $\mathcal{S}_\mathcal{X}$. For each element $x_i \in \mathcal{X}$ occurring two times in $\mathcal{T}$, add the following four sequences to $\mathcal{S}_\mathcal{X}$: $S_{i,1} = x_i$, $S_{i,2} = u_i - w_i$, $S_{i,3} = u_i - v_i$, $S_{i,4} = w_i - v_i$. And, for each element $x_i \in \mathcal{X}$ occurring three times in $\mathcal{T}$, add the following three sequences to $\mathcal{S}_\mathcal{X}$: $S_{i,1} = x_i$, $S_{i,2} = u_i - v_i$, $S_{i,3} = w_i - v_i$. We finally define $\mathcal{A}$ to be the set of all characters used in the construction.

Lemma 1. *There exists a solution $\mathcal{T}' \subseteq \mathcal{T}$ to* X3C *problem over $(\mathcal{X}, \mathcal{T})$ if and only if in the corresponding built instance $(\mathcal{A}, \mathcal{S})$ of* LocateMrs *there exists a subset $\mathcal{S}' \subseteq \mathcal{S}$ corresponding to a* Mrs.

Correctness of Lemma 1 implies the following result.

Theorem 1. LocateMrs *problem is* **NP***-complete even in the special case where none of the input strings contains duplicated characters and at most one substring S_j^i of every S_j can belong to any solution $\mathcal{S}'$.*

We now prove that LocateMrs belongs to the class of the fixed-parameter tractable (FPT) problems [10]. In other words, it can be solved efficiently by an algorithm exponential only with respect to a fixed parameter – $|\mathcal{A}|$ in our case – while polynomial in the size of the input.

Theorem 2. LocateMrs *problem is Fixed-Parameter Tractable in $|\mathcal{A}|$*

To show this, we provide a dynamic programming solution. According to LocateMrs definition, one has to select exactly one substring of interest among all of them in each sequence S_j. A naive algorithm may try all such combinations and check for each if any character appears in at least two substrings. Such an algorithm has an exponential running time. We

will prove that by using an efficient dynamic programming strategy, one may hold the exponential factor in the size of the ancestral gene set. Note that one does not need to compute the exact number of times each character occurs but only to ensure that it occurs in at least two substrings in the solution. According to this remark, consider a fixed ordering of characters $(a_1, a_2, \ldots, a_{|\mathcal{A}|})$ of $\mathcal{A}$, we compute after adding substring S to the current solution a vector $\mathcal{C} = (c_1, c_2, \ldots, c_{|\mathcal{A}|})$, where $c_i \in \{0, 1, 2\}$ denotes respectively that a_i is not contained, contained in one, or contained in at least two substrings. For example, consider $\mathcal{A} = \{1, 2, 3, 4, 5\}$ and current solution $\mathcal{S}' = \{124\}$, one may derive a vector $\mathcal{C} = (2, 1, 0, 2, 1)$ after adding substring "1445" to $\mathcal{S}'$. The main property of this representation is that, given $\mathcal{A}$, there are only $3^{|\mathcal{A}|}$ possible vectors. Further, let $\mu(x)$ and $\chi_S(x)$ denote, respectively, the position of x in the fixed ordering of $\mathcal{A}$ and the boolean function indicating whether x occurs in S. We define a boolean dynamic table D indexed by the last substring added and the vector $\mathcal{C}$ for the current solution. The main recursion[1] is defined as follows:

$$D(S_j^i, (c_1, \ldots c_{|\mathcal{A}|})) \;=\; \begin{cases} 1 - \text{if } \exists i', j' < j \text{ s.t. } D(S_{j'}^{i'}, (c'_1, \ldots c'_{|\mathcal{A}|})) = 1 \\ \quad \text{and } \forall 1 \leq l \leq |\mathcal{A}|, \chi_{S_j^i}(x) + c'_l = \min\{2, c_l\} \\ \quad \text{where } \mu(x) = l \\ 0 - \text{otherwise} \end{cases}$$

Algorithm 1 LocateMrs($\mathcal{A}$, $\mathcal{S} = \{S_1^1, S_1^2, \ldots, S_2^1, S_2^2, \ldots, S_k^1, S_k^2, \ldots\}$)

```
1: Initialize all entries of D to 0
2: for each S_1^i ∈ S do D(S_1^i, (c_1, ..., c_|A|)) = 1 where ∀x ∈ S_1^i, c_μ(x) = χ_{S_1^i}(x) done
3: for j = 2 to k do
4:     for each S_j^i ∈ S do Fill out D(S_j^i, (c_1, ..., c_|A|)) done
5: end for
6: for each S_k^i ∈ S do
7:     if D(S_k^i, (2, ..., 2)) = 1 then return True end if
8: end for
9: return False
```

Given this function, one can apply Algorithm 1. The algorithm computes, for each sequence S_j the possible character set solution induced by any combination of substrings of interest from sequences $S_{j'}$ with $j' < j$. Therefore, any entry $D(S_j^i, (2, \ldots, 2)) = 1$ corresponds to a Mrs being found. One may, using a simple back-tracking technic, rebuild one optimal solution. Let us now prove the time complexity of this algorithm. In order to fill out D, one has to compute $|\mathcal{S}| \times 3^{|\mathcal{A}|}$ entries. Indeed, there are at most $|\mathcal{S}|$ different substrings and $3^{|\mathcal{A}|}$ possible character sets. The main

[1] The base case is made in Algorithm 1

recursion needs, for each entry, to browse at most $|\mathcal{S}| \times 3^{|\mathcal{A}|}$ other entries of D. This leads to an overall $O((|\mathcal{S}| \times 3^{|\mathcal{A}|})^2)$ running time algorithm. Hence, the problem is FPT with respect to $|\mathcal{A}|$.

4.2 Identify All MRS When $\mathcal{A}$ Is Unknown

We will prove that finding all MRS problem is hard even in the special case where none of the sequences contains duplicated characters and in any solution $\mathcal{S}'$, for any sequence S_j at most one substring $S_j^i \in \mathcal{S}'$ (*i.e.,* $\alpha = 0$, $\beta \geq 2$, $\epsilon_t = \epsilon_l = \infty$).

First, note that the problem is in **NP** since given a subset $\mathcal{S}'$ of $\mathcal{S}$, one can polynomially check that each element of $\mathcal{A}$ appears in at least two substrings and no more than one substring of any sequence belongs to $\mathcal{S}'$. To prove that this problem is moreover **NP**-hard, we provide a polynomial reduction from the **NP**-complete problem X3C [21] based on a slight modification of the reduction of the previous section. Indeed, if one replaces each of the separations "–" between substrings of interest by a unique character appearing only once in $\mathcal{S}$, then by definition, those added characters will never be part of a MRS since any character should appear at least twice in a MRS. Due to the unextandability property of MRS, one should be able to find neither a smaller nor a bigger substring of interest in each sequence than in the LOCATEMRS formulation. The rest of the proof still holds, leading to the following theorem.

Theorem 3. *Finding a* MRS *problem is* **NP***-complete even in the special case where none of the sequences contains duplicated characters and in any solution $\mathcal{S}'$, at most one substring from each S_j belongs to $\mathcal{S}'$.*

Let us then consider the optimization version of the problem (Definition 2) where one wants to find a MRS induced by the maximum unknown ancestral gene set (in other words, one constrains the minimum size of $\mathcal{A}$), and at the same time, at most one substring of each S_j can belong to the MRS.

Definition 2. MAXMRS*: Given a set of k strings $\mathcal{S} = \{S_1, \dots S_k\}$, find any possible $(\mathcal{A}, \mathcal{S}')$ where $\mathcal{S}' = \{S'_1, S'_2, \dots S'_k : S'_i$ is a substring of $S_i\}$, $\mathcal{A} = \bigcup_{i=1}^{k} \mathcal{CS}(S'_i)$, and $|\mathcal{A}|$ is maximum.*

We will demonstrate that this optimization problem is hard to approximate. Meanwhile, we show that the inapproximability of this problem may stem from forbidding more than one substring per input chromosome, the relaxation of which leads to polynomiality.

In the following, we consider that $\beta \geq 2$, $\alpha = 0$, $\epsilon_m = 1$, and $\epsilon_t = \epsilon_l = \infty$. We prove the inapproximability of MAXMRS below by proposing

a reduction from the MINIMUM SET COVER (MINSC) problem: Given a family $\mathcal{F}$ of subsets of a finite universe $\mathcal{U}$, find a set cover $\mathcal{F}'$ for $\mathcal{U}$ – that is a subfamily $\mathcal{F}' \subseteq \mathcal{F}$ whose union is $\mathcal{U}$ – of the minimum cardinality.

Since any character appearing once in an input string will not be part of a MRS, we use the symbol "–" to denote any such character. The presence of symbol "–" will induce, in MAXMRS problem, that characters appearing before and after it in any input string cannot be part of the same solution. Given any instance $(\mathcal{F},\mathcal{U})$ of MINSC, where $\mathcal{U} = \{u_1, \dots u_n\}$ and $\mathcal{F} = \{F_i : F_i = \{u_i^1, u_i^2, \dots u_i^{n_i}\}, 1 \leq i \leq m\}$, we define a set of strings $\mathcal{S} = \{S_0, \dots, S_{k=2m}\}$ with $S_0 = u_1 \dots u_n$, $S_i = S_i^1$ – $S_i^2 = u_i^1 u_i^2 \dots u_i^{n_i}$ – v_i and $S_{m+i} = v_i$, for $1 \leq i \leq m$.

Lemma 2. *If there exists a cover $\mathcal{F}' \subseteq \mathcal{F}$ for $\mathcal{U}$ (i.e. $\mathcal{U} = \bigcup_{F \in \mathcal{F}'} F$) then there exists a solution $(\mathcal{A}, \mathcal{S}')$ (i.e. a* MRS*) for the built up instance $\mathcal{S}$ of* MAXMRS *such that $|\mathcal{A}| = n + m - |\mathcal{F}'|$.*

Lemma 3. *Given a solution $(\mathcal{A}, \mathcal{S}')$ for a built up instance $\mathcal{S}$ of* MAXMRS*, we can construct in polynomial-time a cover $\mathcal{F}' \subseteq \mathcal{F}$ for $\mathcal{U}$, such that $|\mathcal{F}'| \leq m - (|\mathcal{A}| - n)$.*

Proof. We first define a polynomial-time subroutine that transforms any solution $(\mathcal{A}, \mathcal{S}')$ to an equally good solution where $\mathcal{CS}(S_0) \subseteq \mathcal{A}$. For any character of S_0 not belonging to $\mathcal{A}$ – say u_j, add to $\mathcal{S}'$ one substring S_i^1 that was not in $\mathcal{S}'$ but contains u_j, meanwhile, remove from $\mathcal{S}'$ correspondingly two substrings S_i^2 and S_{m+i}. Every such replacement operation will change a given v_i by u_j in $\mathcal{A}$ without decreasing the cardinality of $\mathcal{A}$ (*i.e.* an equally good solution). Once this subroutine has been applied to $(\mathcal{A}, \mathcal{S}')$, one can build a cover $\mathcal{F}' = \{F_i : S_i^2 \notin \mathcal{S}'\}$. The subroutine guarantees that all elements of S_0 belong to $\mathcal{F}'$ – a cover for $\mathcal{U}$. Clearly, $|\mathcal{A}| - n$ corresponds to the number of v_js belonging to $\mathcal{A}$. Considering $S_1, S_2 ... S_m$ (where S_j^2s appear), there exist at most $m - (|\mathcal{A}| - n)$ strings such that $S_j^2 \notin \mathcal{S}'$; inducing that $|\mathcal{F}'| \leq m - (|\mathcal{A}| - n)$. □

Theorem 4. MAXMRS *is* **APX***-hard even in the special case where, for every input string S_i, $|\mathcal{CS}(S_i)| = |S_i|$.*

Proof. Consider MINIMUM 3-SETCOVER-3 (MIN3SC-3), a subproblem of MINSC, where the size of any set in $\mathcal{F}$ is bounded by 3 as well as the number of times each character of $\mathcal{U}$ occurs in $\mathcal{F}$. We will prove the theorem by contradiction, assuming that MAXMRS admits a Polynomial-Time Approximation Scheme (PTAS), *i.e.* one would be able to find an approximation algorithm leading to an approximate solution $(\mathcal{A}_{APX}, \mathcal{S}_{APX})$, which compared with the optimal solution $(\mathcal{A}_{OPT}, \mathcal{S}_{OPT})$, induces $|\mathcal{A}_{APX}| \geq$

$(1-\epsilon)\cdot|\mathcal{A}_{OPT}|$ for a parameter $\epsilon > 0$. Accordingly, under the same assumption, we will prove that MIN3SC-3 also admits a PTAS, *i.e.* one would be able to find an approximation algorithm leading to an approximate solution $\mathcal{F}_{APX}$, which compared with the optimal solution $\mathcal{F}_{OPT}$, induces $|\mathcal{F}_{APX}| \leq (1+\gamma)\cdot|\mathcal{F}_{OPT}|$ for a parameter $\gamma > 0$ – a contradiction to the fact that MIN3SC-3 is **APX**-hard [15].

Since each character of $\mathcal{U}$ occurs at most three times in $\mathcal{F}$, the size of the ground set used to build $\mathcal{F}$ is at most $3n$, leading to $m \leq 3n$. Moreover, any cover $\mathcal{F}' \subseteq \mathcal{F}$ of $\mathcal{U}$ is at least of size $\frac{n}{3}$ since $\mathcal{F}$ is composed of sets of size at most three. Hence, $\frac{n}{3} \leq |\mathcal{F}_{OPT}|$ and consequently, $m \leq 9\cdot|\mathcal{F}_{OPT}|$.

If we have an approximate solution $(\mathcal{A}_{APX}, \mathcal{S}_{APX})$, then

By Lemma 3,	$\|\mathcal{F}_{APX}\|$	$\leq m - (\|\mathcal{A}_{APX}\| - n)$
By assumption,	$m - (\|\mathcal{A}_{APX}\| - n)$	$\leq m - ((1-\epsilon)\cdot\|\mathcal{A}_{OPT}\| - n)$
By Lemma 2,	$\|\mathcal{A}_{OPT}\|$	$= n + m - \|\mathcal{F}_{OPT}\|$
Which leads to,	$\|\mathcal{F}_{APX}\|$	$\leq \epsilon\cdot n + \epsilon\cdot m + (1-\epsilon)\cdot\|\mathcal{F}_{OPT}\|$
$(m \leq 3n \leq 9\|\mathcal{F}_{OPT}\|)$		$\leq 12\epsilon\cdot\|\mathcal{F}_{OPT}\| + (1-\epsilon)\cdot\|\mathcal{F}_{OPT}\|$
Finally,		$\leq (1+11\epsilon)\cdot\|\mathcal{F}_{OPT}\|$ □

We now prove the following result on restricted instances.

Theorem 5. *If one restricts neither the maximum number of gene losses per substring of interest, nor the maximum number of substrings of interest per chromosome, and if every input sequence contains no duplicated characters, finding all the* MRS *becomes a polynomial task.*

Proof. Consider a graph $G = (V, E)$ obtained from $\mathcal{S}$ in such a way that a vertex is assigned to every character in each string $S_i \in \mathcal{S}$ and a red (resp. blue) colored edge is created between any two adjacent characters (resp. any two vertices representing identical characters). Given this representation, the notion of character set naturally extends to any subgraph $G[V']$ of G as the set of represented characters by V'. Our method consists in an iterative procedure which stops when none of the following operations can be applied anymore. The results will consist of a set of connected components, each corresponding to a MRS. The first operation consists in removing from V any vertex which is only incident to red colored edges. This polynomial operation results in the removal of genes not appearing twice in a candidate connected component. The second operation gets rid of candidates not fulfilling the minimum contribution to the ancestral gene set by pruning any red edge-induced subgraph G' such that $|\mathcal{CS}(G')| < \epsilon_m$. This operation can be done in linear time by browsing any connected component. Once none of these operations can be done anymore, it is easy to see that each remaining connected component corresponds to a MRS. □

References

1. A. Amir, L. Gasieniec, and R. Shalom. Improved approximate common interval. *Inf. Process. Lett.*, 103(4):142–149, 2007.
2. A. Bergeron, C. Chauve, F. de Montgolfier, and M. Raffinot. Computing Common Intervals of K Permutations, with Applications to Modular Decomposition of Graphs. In *ESA*, volume 3669 of *LNCS*, pages 779–790, 2005.
3. A. Bergeron, C. Chauve, and Y. Gingras. *Bioinformatics Algorithms: Techniques and Applications*, chapter 8, pages 177–202. Wiley & Sons, Inc, 2008.
4. A. Bergeron, S. Corteel, and M. Raffinot. The algorithmic of gene teams. In *WABI*, volume 2452 of *LNCS*, pages 464–476, 2002.
5. A. Bergeron and J. Stoye. On the similarity of sets of permutations and its applications to genome comparison. *J Comput Biol*, 13(7):1340–1354, 2006.
6. G. Blin, C. Chauve, G. Fertin, R. Rizzi, and S. Vialette. Comparing genomes with duplications: a computational complexity point of view. *ACM TCBB*, 4(4):523–534, 2007.
7. S. Böcker, K. Jahn, J. Mixtacki, and J. Stoye. Computation of median gene clusters. *J Comput Biol*, 16(8):1085–1099, 2009.
8. D. Che, G. Li, F. Mao, H. Wu, and Y. Xu. Detecting uber-operons in prokaryotic genomes. *Nucleic Acids Res*, 34(8):2418–2427, 2006.
9. G. Didier, T. Schmidt, J. Stoye, and D. Tsur. Character sets of strings. *JDA*, 5(2):330–340, 2007.
10. R. Downey and M. Fellows. *Parameterized Complexity*. Springer-Verlag, 1999.
11. M. Garey and D. Johnson. *Computers and Intractability: A guide to the theory of NP-completeness*. W.H. Freeman, 1979.
12. R. Hoberman and D. Durand. The incompatible desiderata of gene cluster properties. In *Recomb-CG*, volume 3678 of *LNCS*, pages 73–87, 2005.
13. K. Jahn. Efficient computation of approximate gene clusters based on reference occurrences. *Journal of Computational Biology*, 18(9):1255–1274, 2011.
14. J. H. Nadeau and B. A. Taylor. Lengths of chromosomal segments conserved since divergence of man and mouse. *Proc Natl Acad Sci U S A*, 81(3):814–818, 1984.
15. C. Papadimitriou and M. Yannakakis. Optimization, approximation and complexity classes. *J. Comput. System Sci.*, 43:425–440, 1991.
16. S. Pasek, A. Bergeron, J. Risler, A. Louis, E. Ollivier, and M. Raffinot. Identification of genomic features using microsyntenies of domains: domain teams. *Genome Res*, 15(6):867–874, 2005.
17. S. Rahmann and G. W. Klau. Integer Linear Programs for Discovering Approximate Gene Clusters. In *WABI*, volume 4175 of *LNCS*, pages 298–309, 2006.
18. C. Simillion, K. Vandepoele, and Y. V. de Peer. Recent developments in computational approaches for uncovering genomic homology. *Bioessays*, 26(11):1225–1235, 2004.
19. T. Uno and M. Yagiura. Fast algorithms to enumerate all common intervals of two permutations. *Algorithmica*, 26(2):290–309, 2000.
20. X. Yang and S. Aluru. A Unified Model for Multi-genome Synteny and Gene Cluster Inference. Technical report, Iowa State University, 2009.
21. X. Yang and S. Aluru. An improved model for gene cluster inference. In *BiCob'2010*, pages 190 – 195, 2010.
22. X. Yang and S. Aluru. *Algorithms in Computational Molecular Biology: Techniques, Approaches and Applications*, chapter 32, pages 725–747. Wiley & Sons, Inc, 2011.

Appendix

Missing proofs for main results

Lemma 1. *There exists a solution $\mathcal{T}' \subseteq \mathcal{T}$ to* X3C *problem over $(\mathcal{X}, \mathcal{T})$ if and only if in the corresponding built instance $(\mathcal{A}, \mathcal{S})$ of* LocateMrs *there exists a subset $\mathcal{S}' \subseteq \mathcal{S}$ corresponding to a* Mrs*.*

Proof. ($\Rightarrow$) Suppose such $\mathcal{T}' \subseteq \mathcal{T}$ exists for the instance $(\mathcal{X}, \mathcal{T})$. Let us prove that we can compute in polynomial time an $\mathcal{S}' \subseteq \mathcal{S}$ corresponding to a Mrs in the corresponding built instance of LocateMrs. For any triple $t_q \in \mathcal{T}$, if $t_q \in \mathcal{T}'$ then add T_q^1 to $\mathcal{S}'$; add T_q^2 otherwise. Moreover, for $1 \leq i \leq |\mathcal{X}|$, add $S_{i,1}$ to $\mathcal{S}'$. For any element $x_i \in \mathcal{X}$ occuring two times in $\mathcal{T}$, if the first triple containing x_i is in $\mathcal{T}'$ then add $S_{i,2}^2, S_{i,3}^2, S_{i,4}^2$ to $\mathcal{S}'$; add $S_{i,2}^1, S_{i,3}^1, S_{i,4}^1$ to $\mathcal{S}'$ otherwise. For any element $x_i \in \mathcal{X}$ occuring three times in $\mathcal{T}$, if the first triple containing x_i is in $\mathcal{T}'$ then add $S_{i,2}^2, S_{i,3}^1$ to $\mathcal{S}'$; if the second triple containing x_i is in $\mathcal{T}'$ then add $S_{i,2}^1, S_{i,3}^1$ to $\mathcal{S}'$; add $S_{i,2}^1, S_{i,3}^2$ to $\mathcal{S}'$ otherwise (see Figure 1). Let us now prove that $\mathcal{S}'$ is indeed a Mrs, *i.e.*, $\mathcal{A} = \bigcup_{S \in \mathcal{S}'} \mathcal{CS}(S)$ and any element of $\mathcal{A}$ appears at least twice in $\mathcal{S}'$. First note that, by definition, in $\mathcal{T}'$, there is exactly one triple containing any element of $\mathcal{X}$. Thus, by construction, there is exactly one substring per sequence in $\mathcal{S}'$. Let us consider any element $x_i \in \mathcal{X}$ which occurs twice in $\mathcal{T}$ – say in triples t_q and $t_{q'}$ for the first and respectively second occurrences of x_i. Then, since $\mathcal{T}'$ is an exact cover, there exists exactly one among t_q and $t_{q'}$ in $\mathcal{T}'$ – say t_q. Therefore, by construction, $\{T_q^1, T_{q'}^2, S_{i,1}, S_{i,2}^2, S_{i,3}^2, S_{i,4}^2\} \subseteq \mathcal{S}'$ (the reader may, for example, consider x_1 in Figure 1). Consequently, a) two occurrences of x_i appear in $\mathcal{S}'$ (one from $S_{i,1}$ and one from T_q^1), b) two occurrences of u_i appear in $\mathcal{S}'$ (one from T_q^1 and one from $T_{q'}^2$), c) two occurrences of v_i appear in $\mathcal{S}'$ (one from $S_{i,3}$ and one from $S_{i,4}$), and d) two occurrences of w_i appear in $\mathcal{S}'$ (one from $S_{i,2}$ and one from $T_{q'}^2$). With a similar reasoning, one can check that any of x_i, u_i, v_i, w_i appears twice when $t_{q'} \in \mathcal{T}'$. Let us now consider each element $x_i \in \mathcal{X}$ that occurs three times in $\mathcal{T}$ – say in triples t_q, $t_{q'}$ and $t_{q''}$ for the first (resp. second and third) occurrence of x_i. Then, since $\mathcal{T}'$ is an exact cover, there exists exactly one among t_q, $t_{q'}$ and $t_{q''}$ in $\mathcal{T}'$ – say t_q. Therefore, by construction, $\{T_q^1, T_{q'}^2, T_{q''}^2, S_{i,1}, S_{i,2}^2, S_{i,3}^1\} \subseteq \mathcal{S}'$ (the reader may, for example, consider x_3 in Figure 1). Consequently, a) two occurrences of x_i appear in $\mathcal{S}'$ (one from $S_{i,1}$ and one from T_q^1), b) three occurrences of u_i appear in $\mathcal{S}'$ (one from T_q^1, one from $T_{q'}^2$ and one from $T_{q''}^2$), c) two occurrences of v_i appear in $\mathcal{S}'$ (one from $T_{q''}^2$ and one from $S_{i,2}$), d) two occurrences of w_i appear in $\mathcal{S}'$ (one from $S_{i,3}$ and one from

$T^2_{q'}$), and e) two occurrences of z_i appear in $\mathcal{S}'$ (one from $T^2_{q'}$ and one from $T^2_{q''}$). With a similar reasoning, one can check that any of x_i, u_i, v_i, w_i, z_i appears at least twice when $t_{q'} \in \mathcal{T}'$ or $t_{q''} \in \mathcal{T}'$. We have completed the proof that each element of $\mathcal{A}$ appears at least twice in $\mathcal{S}'$; inducing that $\mathcal{S}'$ is indeed a MRS.

($\Leftarrow$) Suppose now that such a set $\mathcal{S}' \subseteq \mathcal{S}$ exists for the corresponding built instance of LOCATEMRS $(\mathcal{A}, \mathcal{S})$. We will prove that we can compute in polynomial time a $\mathcal{T}' \subseteq \mathcal{T}$ corresponding to a solution for the instance $(\mathcal{X}, \mathcal{T})$. For any $1 \leq i \leq |\mathcal{T}|$, $t_i \in \mathcal{T}'$ if $T^1_i \subseteq \mathcal{S}'$. Let us now prove that $\mathcal{T}'$ is indeed an exact cover of $\mathcal{T}$. Note that, any solution $\mathcal{S}'$ is a subset of $\mathcal{S}$ since we are looking for unextandable substrings (inducing that any substring of interest is either fully kept or removed). Moreover, recall that we consider here the special case of the problem where in any solution $\mathcal{S}'$, for any string S_j, at most one substring $S^i_j \in \mathcal{S}'$. Let us first prove that given two triples t_q and $t_{q'}$ both containing one of the two occurrences of $x_i \in \mathcal{X}$ (*i.e.* the case where x_i appears twice in $\mathcal{X}$), then, exactly one of the substrings T^1_q and $T^1_{q'}$ belongs to $\mathcal{S}'$ (the reader may, for example, consider x_1 in Figure 1). By contradiction, suppose this is not the case, *i.e.* $\{T^1_q, T^1_{q'}\} \subseteq \mathcal{S}'$. Then, in $\mathcal{S}'$, there are already one occurrence of u_i, one occurrence of v_i and two occurrences of x_i. Note that the set of elements related to variable x_i, *i.e.*, $\{x_i, u_i, v_i, w_i\}$, appears in the following sequences $\{T_q, T_{q'}, S_{i,1}, S_{i,2}, S_{i,3}, S_{i,4}\}$. Since, by definition, in $\mathcal{S}'$, every element should appear at least twice, using exactly one of the substrings of each of these sequences, one should be able to obtain another u_i, another v_i and two w_i. Unfortunately, this is not possible in any combination of the corresponding substrings excluding, by hypothesis, T^2_q and $T^2_{q'}$; a contradiction. Now consider the three triples t_q, $t_{q'}$ and $t_{q''}$, each containing one of the three occurrences of $x_i \in \mathcal{X}$ (*i.e.* the case where x_i appears three times in $\mathcal{X}$), then, exactly one among the substrings T^1_q, $T^1_{q'}$ and $T^1_{q''}$ belongs to $\mathcal{S}'$ (the reader may, for example, consider x_3 in Figure 1). By construction, the occurrences of z_i belong to T^2_q, $T^2_{q'}$ and $T^2_{q''}$. In order to obtain at least two occurrences of z_i in $\mathcal{S}'$, at least two of the substrings among $\{T^2_q, T^2_{q'}, T^2_{q''}\}$ should be in $\mathcal{S}'$. Moreover, since the occurrences of x_i belong to $\{T^1_q, T^1_{q'}, T^1_{q''}, S_{i,1}\}$, exactly one of $\{T^1_q, T^1_{q'}, T^1_{q''}\}$ should be in $\mathcal{S}'$. We have proved that for each element x_i occuring twice (resp. three times) in $\mathcal{T}$, exactly one of the triples containing x_i is kept in $\mathcal{T}'$. Thus, $\mathcal{T}'$ is indeed an exact cover of $\mathcal{T}$. □

Lemma 2. *If there exists a cover $\mathcal{F}' \subseteq \mathcal{F}$ for $\mathcal{U}$ (i.e., $\mathcal{U} = \bigcup_{F \in \mathcal{F}'} F$) then there exists a solution $(\mathcal{A}, \mathcal{S}')$ (i.e., a* MRS*) for the built up instance $\mathcal{S}$ of* MAXMRS *such that $|\mathcal{A}| = n + m - |\mathcal{F}'|$.*

Proof. Consider the following solution $\mathcal{S}' = \{S_0\} \bigcup \{S_i^1 \in \mathcal{S} : F_i \in \mathcal{F}'\} \bigcup \{S_i^2, S_{m+i} \in \mathcal{S} : F_i \notin \mathcal{F}'\}$. By definition, $\mathcal{S}'$ is indeed a MRS since it cannot be extended (it is made of one unextandable substring of each sequence) and any element appears twice. Moreover, its character set $\mathcal{A} = \mathcal{U} \bigcup \{v_i : F_i \notin \mathcal{F}'\}$ which is of size $n + m - |\mathcal{F}'|$. □