# Revisiting the Minimum Breakpoint Linearization Problem

Laurent Bulteau, Guillaume Fertin, Irena Rusu

# Revisiting the Minimum Breakpoint Linearization Problem

Laurent Bulteau[1,2], Guillaume Fertin[2] and Irena Rusu[2]

[1] École Normale Supérieure, 45 rue d'Ulm, 75000 Paris, France
[2] Laboratoire d'Informatique de Nantes-Atlantique (LINA), UMR CNRS 6241
Université de Nantes, 2 rue de la Houssinière, 44322 Nantes Cedex 3 - France
Laurent.Bulteau@ens.fr, {Guillaume.Fertin,Irena.Rusu}@univ-nantes.fr

**Abstract.** The gene order on a chromosome is a necessary data for most comparative genomics studies, but in many cases only partial orders can be obtained by current genetic mapping techniques. The Minimum Breakpoint Linearization Problem aims at constructing a total order from this partial knowledge, such that the breakpoint distance to a reference genome is minimized. In this paper, we first expose a flaw in two algorithms formerly known for this problem [5,3]. We then present a new modeling for this problem, and use it to design three approximation algorithms, with ratios resp. $O(\log(k)\log\log(k))$, $O(\log^2(|X|))$ and $m^2+4m-4$, where $k$ is the optimal breakpoint distance we look for, $|X|$ is upper bounded by the number of pair of genes for which the partial order is in contradiction with the reference genome, and $m$ is the number of genetic maps used to create the input partial order.

## 1 Introduction

In a number of comparative genomics algorithms, a full knowledge of the order of the genes on the chromosomes for the species under study is required. However, we have access to a limited number of fully sequenced genomes, and for other species, we only have genetic maps, in which there remains uncertainties in the gene order. Hence, the problem of inferring a total order, compatible with the partial knowledge on these genetic maps and optimizing some objective function, is a first step to study nonetheless all genomes. In the past few years, growing attention has been given to this problem, in which the objective function is an evolutionary distance to a reference genome (*e.g.* number of rearrangements [9], reversal [8,5], breakpoint [5,2,3], or common intervals [2] distance).

In this paper, we focus on the MBL problem, which aims at finding a linearization of a partial order while minimizing the breakpoint distance to a reference genome. In [5] and [3], the study of this problem uses the construction of a special graph, the adjacency-order graph, which leads to respectively a heuristic and an approximation algorithm (whose ratio depends on $m$, the number of genetic maps used to construct the studied genome). However, we have detected a flaw in this construction, which makes both above mentioned algorithms invalid on general data. Thus, in this paper, we define a new type of adjacency-order

graphs, give its construction, and show that it is effective to solve the MBL problem. This renewed approach allows us to use general graph theory results [4] to obtain new approximation algorithms for MBL. Moreover, we also achieve an $O(m^2)$-approximation, in the same spirit as was done in [3].

To describe the MBL problem, let a genome be represented by a partial order $\Pi$ over a given set $\Sigma = \{1, \ldots, n\}$ of markers. A *linearization* of $\Pi$ is a total order (or a permutation) $\pi = \pi(1) \cdot \pi(2) \cdot \ldots \cdot \pi(n)$ on $\Sigma$, such that, for all markers $i, j$, if $i <_\Pi j$, then $i <_\pi j$ (alternatively, $\pi^{-1}(i) < \pi^{-1}(j)$, or $i$ precedes $j$ in the permutation $\pi$). In that case, $\pi$ is said to be *compatible* with $\Pi$. An *interval* $I$ of $\pi$ is a list of successive values from $\pi$, that is $I = \pi(h) \cdot \pi(h+1) \cdot \ldots \cdot \pi(l)$, with $1 \leq h \leq l \leq n$, from $\pi$. For any such interval, its length $L$ is defined as $l - h + 1$. An *adjacency* in the total order $\pi$ is an interval of length $L = 2$. The *breakpoint distance* $d_B(\pi_1, \pi_2)$ between two total orders $\pi_1$ and $\pi_2$ (over the same set $\Sigma$) is defined by the total number of adjacencies in $\pi_1$ which are not adjacencies in $\pi_2$. We call $Id_n$ the identity permutation over $\Sigma = \{1, \ldots, n\}$.

The Minimum Breakpoint Linearization Problem, in its optimization formulation, can be defined as follows:

**Problem :** MBL
**Input :** A partial order $\Pi$.
**Output :** A linearization $\pi$ of $\Pi$ that minimizes $k = d_B(\pi, Id_n)$.

Fig. 1a shows an example of partial order $\Pi$, that yields four optimal linearizations (Fig. 1c) satisfying $d_B(\pi, Id_n) = 3$ for each linearization $\pi$.

It is worth noting that the input partial order is, in practice, obtained by combining a limited number $m$ of *genetic maps* [7,9]. A genetic map consists of an ordered list of blocks $B_1, B_2, \ldots, B_q$, each of which is an unordered list of markers, i.e. any two markers from the same block are incomparable. The blocks $B_1, B_2, \ldots, B_q$ induce a partial order $\Pi$ as follows: for any $a \in B_i$ and $b \in B_j$, $a <_\Pi b$ iff $i < j$. Note that it is always assumed that combining two or more genetic maps never creates conflicts.

The Minimum Breakpoint Linearization Problem, based on the genome rearrangement problem defined by Zheng and Sankoff [9], was studied independently in [2] and [5] (we note that in the latter, the problem is denoted as PBD, and deals with two partial orders instead of one partial order and one total order). In [2], Blin et al. prove that MBL is **NP**-hard and give two types of algorithms for solving MBL: (i) a heuristic and (ii) an exact, thus exponential-time, algorithm based on dynamic programming. Moreover, this last algorithm is efficient in the specific case where input genomes are created from a bounded number $m$ of gene maps, each with a bounded width. In [5], Fu and Jiang give an (independent) **NP**-hardness proof, and present the construction of the adjacency-order graph $\mathcal{G}_\Pi$ of a partial order $\Pi$ ($\Pi$ being represented as a DAG, called DAG($\Pi$)). Their central theorem, claiming that "*All the possible common adjacencies in an acyclic adjacency-order graph $\mathcal{G}_\Pi$ could always co-exist in some linearization of DAG($\Pi$)*", is used in a heuristic they provide for the problem, and is also used by Chen and Cui [3] to obtain an $\frac{m^2+m}{2}$-approximation algorithm. How-

ever, as shown in Theorem 2, the above mentioned theorem from [5] is false, and consequently both those algorithms are invalid for the general MBL Problem.

The paper is organized as follows. In Section 2, we point out the **APX**-hardness of MBL and give our counter-example to the central theorem in [5]. Section 3 is devoted to the definition of a new adjacency-order graph, which is used in Section 4 to show that solving MBL may be reduced to solving a variant of the well-known Feedback Vertex Set problem. Section 5 presents three approximation algorithms, two of which are based on state-of-the-art algorithms for the variant of Feedback Vertex Set we are interested in, whereas the third is specific to partial orders created from genomic maps, and has a ratio depending on the number $m$ of those genomic maps. Section 6 is the conclusion.

Due to space constraints, most of the proofs have been moved to the Appendix, which is available for the referees.

## 2 Revisiting Previous Works

In this section, we focus on previous works: we first show that adapting the **NP**-hardness proof for MBL from [2] leads to an **APX**-hardness proof. We then give a counterexample to Theorem 1 from [5], which implies, among others, that the approximation algorithm from [3] is invalid.

**Theorem 1** *The* MBL *problem is* **APX***-hard.*

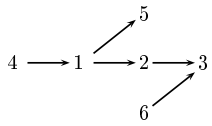
See Theorem 1 in [2], where the reduction is in fact an L-reduction from Maximum Independent Set (MIS) to MBL, if we restrict MIS to cubic graphs.

**Theorem 2** *All the adjacencies appearing in $\mathcal{G}_\Pi$ (as defined in [5]) of a DAG $\Pi$ may not always coexist in a linearization of $\Pi$, even if $\mathcal{G}_\Pi$ is acyclic.*

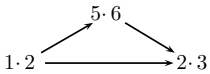
*Proof.* Consider the directed acyclic graph $DAG(\Pi)$ obtained from the partial order described in Fig. 1a. Here, there are only three possible common adjacencies between $\Pi$ and $Id_n$: $1\cdot 2$, $2\cdot 3$ and $5\cdot 6$. Following the definitions proposed in [5], there are three arcs in the adjacency-order graph $\mathcal{G}_\Pi$ (Fig. 1b): one from $1\cdot 2$ to $2\cdot 3$ (because of the common marker), one from $1\cdot 2$ to $5\cdot 6$ (because $1 < 5$ in $\Pi$), and one from $5\cdot 6$ to $2\cdot 3$ (because $6 < 3$ in $\Pi$). In that case, $\mathcal{G}_\Pi$ is acyclic; however, its three adjacencies cannot coexist in any linearization of $\Pi$ (see Fig. 1c: there can be at most two adjacencies in the same linearization). Thus Theorem 2 is proved, which contradicts Theorem 1 in [5]. □

## 3 Defining a New Adjacency-Order Graph $G_\Pi$

The direct consequence of Theorem 2 is that the adjacency-order graph defined in [5] cannot be exploited. Hence, we introduce here the construction of a new type of adjacency-order graph, whose main structural property will lead us to three different approximation algorithms.




(a) DAG($\Pi$) represents the partial order $\Pi$




(b) $\mathcal{G}_\Pi$, as defined in [5]

$4 \cdot 1 \cdot \underline{5 \cdot 6} \cdot \underline{2 \cdot 3}$
$4 \cdot \underline{1 \cdot 2} \cdot \underline{5 \cdot 6} \cdot 3$
$6 \cdot 4 \cdot \underline{1 \cdot 2 \cdot 3} \cdot 5$
$4 \cdot 6 \cdot \underline{1 \cdot 2 \cdot 3} \cdot 5$

(c) Optimal linearizations of $\Pi$

Fig. 1: Counterexample of the main theorem from [5]

This new adjacency-order graph $G_\Pi$ contains both the features of the partial order $\Pi$ and of the identity permutation $Id_n$. Some of the cycles in this graph will express the incompatibilities between the order $\Pi$ and the permutation $Id_n$. In order to count or to bound the breakpoint distance between a linearization $\pi$ of $\Pi$ and $Id_n$, one has to identify the vertices in the adjacency-order graph needed to break all these conflict-cycles, and to count or bound their number. The MBL problem thus becomes a graph theory problem, which allows us either to use existing algorithms or to build-up new algorithms based on graphs.

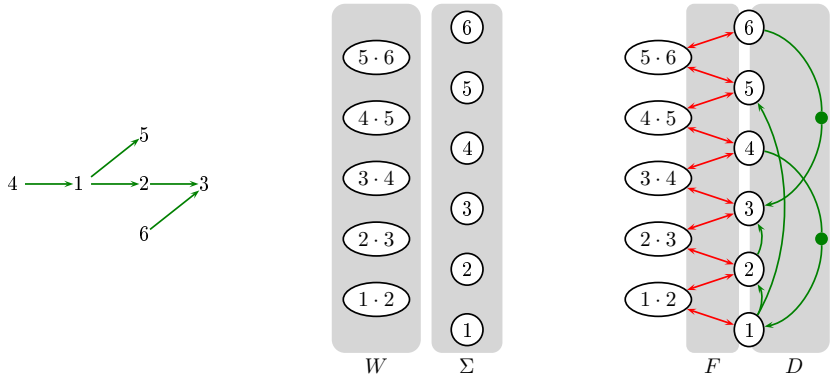
*Adjacency-order graph.* Let $\Pi = (\Sigma, D)$ be a directed acyclic graph (DAG) representing a partial order over $\Sigma = \{1, \ldots, n\}$ (see Fig. 2a), *i.e.* we write $i <_\Pi j$ iff there is a directed path from $i$ to $j$ in $\Pi$. We create a set $W$ of vertices representing the adjacencies of the identity permutation $Id_n$ by $W = \{i\cdot(i+1) \mid 1 \leq i < n\}$. Finally, let $V = \Sigma \cup W$ (Fig. 2b). Note that, in the following, we will not distinguish the vertices of $\Sigma$ and their corresponding integers (this will always be clear from the context). Moreover, the natural order $<$ over the integers is also used as an order over $\Sigma$. We now construct a set of arcs $F$ (denoted by an arrow $\rightarrow$) in the following way:

$$\begin{aligned} F = \quad & \{i\cdot(i+1) \rightarrow i \mid 1 \leq i < n\} \ \cup \ \{i\cdot(i+1) \rightarrow i+1 \mid 1 \leq i < n\} \\ \cup \ & \{i \rightarrow i\cdot(i+1) \mid 1 \leq i < n\} \ \cup \ \{i+1 \rightarrow i\cdot(i+1) \mid 1 \leq i < n\} \end{aligned}$$

Each arc in $F$ has one end in $W$ and one end in $\Sigma$. We write $E = D \cup F$ (Fig. 2c) and we define the *adjacency-order* graph $G_\Pi$ of $\Pi$ by $G_\Pi = (V, E)$.

In $G_\Pi$, the arcs of $D$ that go top-down (see Fig. 2c) intuitively show incompatibilities between the order in $\Pi$ and the order in $Id_n$. We note $X[G_\Pi]$ (or only $X$, if there is no ambiguity) the set containing them, that is $X[G_\Pi] = \{i \rightarrow j \in D \mid i > j\}$. Now, every cycle containing an arc of $X$ is called a *conflict-cycle*. In Theorem 4, we prove that the adjacencies involved in conflict-cycles are incompatible, so that we need to remove at least one adjacency from each of those cycles to obtain a linearization of $\Pi$. We also define a weight map $w[G_\Pi]$ on the vertices of $G_\Pi$, which associates 1 to each $u \in W$, and $\infty$ to $u \in \Sigma$.

*Notations.* An arc between $u$ and $v$ is written $u \rightarrow v$, or $u \rightarrow_{E'} v$ if it belongs to some subset $E'$. A *path* $P$ is a (possibly empty) sequence of arcs written $u \xrightarrow{P}{}^* v$, or $u \xrightarrow{P}{}^*_{E'} v$ if $P$ uses only arcs from $E'$. A non-empty path $Q$ is written with a $+$ sign: $u \xrightarrow{Q}{}^+ v$. A *cycle* is a non-empty path $u \xrightarrow{\mathcal{C}}{}^+ v$ with $v = u$.




(a) Original DAG $\Pi$ (b) Vertices of $G_\Pi$ (c) Arcs of $G_\Pi$

Fig. 2: Construction of an adjacency-order graph. The symmetric arcs in $F$ are represented as double arrows. The arcs in $X$ are marked with a large dot.

Given in $G_\Pi$ a path $P = v_0 \to v_1 \to \ldots \to v_\ell$, we use the following notations: $\ell(P) = \ell$ is the length of $P$, $V(P) = \{v_h \mid 0 \leq h \leq \ell\}$, $W(P) = V(P) \cap W$, $\Sigma(P) = V(P) \cap \Sigma$, $E(P) = \{v_h \to v_{h+1} \mid 0 \leq h < \ell\}$, $F(P) = E(P) \cap F$, $D(P) = E(P) \cap D$, $X(P) = E(P) \cap X$. A cycle $\mathcal{C}$ is said to be *simple* if all vertices $v_h$ are distinct (except $v_0 = v_\ell$), which implies $\ell(\mathcal{C}) = |V(\mathcal{C})| = |E(\mathcal{C})|$.

The following property gives an insight on how conflict-cycles can appear in the adjacency-order graph. (It is not, however, used in our algorithms.)

**Property 3** *Let $\mathcal{C}$ be a simple cycle with $|D(\mathcal{C})| \geq 2$. Then $\mathcal{C}$ is a conflict-cycle.*

## 4 Cutting all Conflict-Cycles in $G_\Pi$ is enough

Now that we have defined how to construct $G_\Pi$ starting from the input partial order $\Pi$, we turn to proving the main structural result of our paper: conflict-cycles contain all the conflicts between the partial order $\Pi$ and the identity permutation $Id_n$ (see Theorem 4). More precisely, when appropriate adjacencies in $Id_n$ (identified as vertices in $W$) are given up, the remaining adjacency-order graph has no conflict-cycle and this condition is necessary and sufficient to obtain a linearization of $\Pi$ that preserves all the remaining adjacencies in $Id_n$.

**Theorem 4** *Let $\Pi$ be a partial order, $G_\Pi = (V, E)$ its adjacency-order graph (with $V = \Sigma \cup W$ and $E = D \cup F$), and $W' \subseteq W$. Then there exists a total order $\pi$ over $\Sigma$, compatible with $\Pi$, and containing every adjacency from $W'$ iff $G_\Pi[W' \cup \Sigma]$ has no conflict-cycle.*

*Proof.* ($\Rightarrow$) Let $\pi$ be a linearization of $\Pi$ containing every adjacency of $W'$. The following lemma (of which the proof can be found in the appendix) will allow us to conclude by contradiction.

**Lemma 5** *Let $P = v_1 \to v_2 \to \ldots \to v_\ell$ be a path with vertices in $W' \cup \Sigma$ such that (H1) the vertices $v_i$ are pairwise distinct, (H2) $\ell \geq 2$, (H3) $v_1, v_\ell \in \Sigma$, and (H4) for any $1 \leq i < \ell$, $v_i \to v_{i+1} \in F$.*
*Let $a = \min(v_1, v_\ell)$ and $b = \max(v_1, v_\ell)$. Then the sequence $a \cdot (a+1) \cdot (a+2) \cdot \ldots \cdot b$ is an interval of $\pi$. Moreover, $\Sigma(P) = \{a, \ldots, b\}$ and $W(P) = \{c \cdot (c+1) \mid a \leq c < b\}$.*

We suppose, by contradiction, that there exists in $G_\Pi[W' \cup \Sigma]$ a cycle $\mathcal{C} = v_0 \to v_1 \to v_2 \to \ldots \to v_\ell = v_0$ containing an arc from $X$ (*e.g.*, $v_0 \to_X v_1$). Wlog, we may assume that $\mathcal{C}$ is simple (otherwise, there exists a simple subcycle of $\mathcal{C}$ that contains an arc from $X$). We distinguish two cases, depending on whether $v_0 \to_X v_1$ is the only arc in $D(\mathcal{C})$ or not.

*First case:* $v_0 \to v_1 \in X$ and for all $i$, $1 \leq i < \ell$, $v_i \to v_{i+1} \in F$ holds. In that case, we can directly use Lemma 5. Indeed, the path $v_1 \to v_2 \to \ldots \to v_\ell = v_0$ satisfies hypothesis H1 (by simplicity of $\mathcal{C}$), H2 (otherwise there would be a loop $v_0 \to v_0$ in $X$), H3 (since $v_\ell \to v_1 \in D$) and H4 (this is assumed in this first case). We also know, due to the fact that $v_0 \to_X v_1$, that $v_1 < v_\ell$. We can conclude that $v_1 \cdot v_1 + 1 \cdot \ldots \cdot v_\ell$ is an interval of $\pi$, so $v_1 <_\pi v_\ell = v_0$. This contradicts the fact that $\pi$ is compatible with $\Pi$, since $v_0 <_\Pi v_1$.

*Second case:* Let $i_0 = 0, i_1, \ldots, i_{h-1}, i_h = \ell$ be the increasing sequence of indices such that $v_{i_j} \to v_{i_j+1} \in D$ for all $j$ such that $0 \leq j < h$. Note that $h \geq 2$ and for all $j$, we have $v_{i_j} \in \Sigma$. Let us prove that for all $j < h$, the relation $v_{i_j} <_\pi v_{i_{j+1}}$ holds. The case where $i_{j+1} = i_j + 1$ is easy, since the arc $v_{i_j} \to_D v_{i_j+1}$ implies $v_{i_j} <_\Pi v_{i_{j+1}}$ (by construction of $G_\Pi$) and $v_{i_j} <_\pi v_{i_{j+1}}$ (since $\pi$ is compatible with $\Pi$). Now, assume there are several arcs between $v_{i_j}$ and $v_{i_{j+1}}$, *i.e.* $i_{j+1} = i_j + m$ with $m \geq 2$. We use Lemma 5 with the path $\mathcal{P}$ in $F$ given by $v_{i_j+1} \to v_{i_j+2} \to \ldots \to v_{i_j+m}$. Path $\mathcal{P}$ satisfies the hypotheses H1, H2, H3 and H4 of the lemma, thus one of the sequences $v_{i_j+1} \cdot (v_{i_j+1}+1) \cdot \ldots \cdot v_{i_j+m}$ and $v_{i_j+m} \cdot (v_{i_j+m}+1) \cdot \ldots \cdot v_{i_j+1}$ is an interval of $\pi$. Note that $v_{i_j}$ is a distinct vertex from $v_{i_{j+1}}$ (since $h \geq 2$), and from other vertices in the set $\Sigma(\mathcal{P})$ as well (since each of them is the source of an arc from $F$ in $\mathcal{C}$, whereas $v_{i_j}$ is the source of an arc from $D$ in $\mathcal{C}$). Consequently, $v_{i_j}$ cannot appear in either of the intervals $v_{i_j+1} \cdot (v_{i_j+1}+1) \cdot \ldots \cdot v_{i_j+m}$ and $v_{i_j+m} \cdot (v_{i_j+m}+1) \cdot \ldots \cdot v_{i_j+1}$ of $\pi$. As $v_{i_j}$ precedes $v_{i_j+1}$ in $\Pi$ (and thus in $\pi$), we have $v_{i_j} <_\pi v_{i'}$ for all $i' \in [i_j+1, i_j+m]$, and particularly, $v_{i_j} <_\pi v_{i_{j+1}}$.

In conclusion, we have $v_{i_j} <_\pi v_{i_{j+1}}$ for all $j < h$ and $v_{i_h} = v_{i_0}$, a contradiction since there can be no cycle in the relation $<_\pi$. Hence, the subgraph $G_\Pi[W' \cup \Sigma]$ does not contain any conflict-cycle.

($\Leftarrow$) *(constructive proof)* We use the following method to construct a linearization $\pi$ of $\Pi$ containing all adjacencies of $W'$, where the subgraph $G' = G_\Pi[W' \cup \Sigma]$, is assumed to contain no conflict-cycle. We denote by $V_1, \ldots V_k$ the strongly connected components of $G'$, ordered by topological order (i.e., if $u, v \in V_i$, there exists a path from $u$ to $v$ ; moreover, if $u \in V_i$ and $v \in V_j$ and there exists a path $u \to^* v$ in $G'$, then $i \leq j$). We sort the elements of each set $V_i \cap \Sigma$ in ascending order of integers, and obtain a sequence $\mu_i$. The concate-

nation $\mu_1 \cdot \mu_2 \cdot \ldots$ gives $\pi$, a total order over $\Sigma$. We now check that $\pi$ contains every adjacency in $W'$ and is compatible with $\Pi$.

Let $a \cdot (a+1) \in W'$. Vertices $a$ and $a+1$ are in the same strong connected component $V_i$, because of the arcs $a \leftrightarrow a \cdot (a+1) \leftrightarrow a+1$. Those two elements are obviously consecutive in the corresponding $\mu_i$, and appear as an adjacency in $\pi$. By contradiction, assume now that there exist two distinct elements $a, b \in \Sigma$ such that $a <_\pi b$ and $b <_\Pi a$. We denote by $i$ and $j$ the indices such that $a \in V_i$ and $b \in V_j$. Since $a <_\pi b$, we have $i \leq j$, and since $b <_\Pi a$, there exists a path $b \overset{P_1}{\rightarrow}{}^+_D a$ in $(\Sigma, D)$. Therefore, in $G'$, we have $i \geq j$. We thus deduce that $i = j$, and therefore $a$ and $b$ share the same strong connected component. This means that there also exists a path $P_2$ from $a$ to $b$ in $G'$. Hence, we have a cycle $b \overset{P_1}{\rightarrow}{}^+ a \overset{P_2}{\rightarrow}{}^+ b$, which cannot be a conflict-cycle, thus those paths do not use any arc from $X$. The latter is in particular true along $P_1$, which implies $b < a$, since each arc $u \rightarrow v$ in $D - X$ is such that $u < v$. On the other hand, $a$ appears before $b$ in $\pi$, and therefore in $\mu_i$, so $a < b$, a contradiction. Finally $\pi$ is a feasible solution for MBL($\Pi$), with at least $|W'|$ common adjacencies with the identity permutation $Id_n$. □

Since all vertices in $W - W'$ count for unconserved adjacencies (and thus define $d_B(\pi, Id_n)$), from Theorem 4 we directly get the following corollary.

**Corollary 6** *The value $k$ of an optimal solution of* MBL*($\Pi$) is the minimum number of vertices one needs to delete in $W$ to remove all conflict-cycles from $G_\Pi$.*

## 5 Three Approximation Algorithms for MBL

### 5.1 Two Approximation Algorithms based on Subset-FVS

Our previous result implies that we have reduced the problem MBL to a generalization of the well studied Feedback Vertex Set (FVS) problem, where only the conflict-cycles must be cut. In order to solve MBL, we use a (more general) variant of FVS, named Subset-FVS and studied by Even et al. [4], whose definition is the following:

**Problem :** Subset-FVS
**Input :** A directed graph $G = (V, E)$, a set $Y \subseteq V \cup E$, a weight map $w : V \rightarrow \mathbb{R}$.
**Output :** A set $V'' \subseteq V$ of minimum weight such that, with $V' = V - V''$, no cycle in $G[V']$ uses a vertex or an arc from $Y$.

In our paper, we are only interested in the restriction of Subset-FVS on adjacency-order graphs, where $Y$ is the set of top-down arcs and $w$ is such that only vertices in $W$ can be deleted:

**Problem :** AOG-Subset-FVS
**Input :** An adjacency-order graph $G_\Pi$, $Y = X[G_\Pi]$, $w = w[G_\Pi]$
**Output :** A set $W''$ solution of Subset-FVS$(G_\Pi, Y, w)$

We note that any algorithm for Subset-FVS is also valid for AOG-Subset-FVS. Two approximation algorithms are given in [4] for Subset-FVS. The first

**Algorithm 1** $O(\log^2(|X|))$- and $O(\log(k)\log\log(k))$-approximation for MBL

**Input:** A directed acyclic graph $\Pi = (\Sigma, D)$
1. Create $G_\Pi = (V, E)$ the adjacency-order graph of $\Pi$;
2. $W'' \leftarrow$ AOG-Subset-FVS$(G_\Pi, X[G_\Pi], w[G_\Pi])$;
3. $W' \leftarrow W - W''$;
4. $(V_1, V_2, \ldots, V_h) \leftarrow$ SCC-sort$(G_\Pi[W' \cup \Sigma])$;
5. **For** $i \leftarrow 1$ **to** $h$;
6. $\quad \mu_i \leftarrow \text{sort}(V_i \cap \Sigma)$;
7. $\pi \leftarrow \mu_1 \cdot \mu_2 \cdot \ldots \cdot \mu_h$;
8. **return** $\pi$;

one achieves an approximation ratio of $O(\log^2|Y|)$, while the second algorithm achieves a ratio of $O(\min(\log(\tau^*)\log\log(\tau^*), \log(n)\log\log(n)))$, where $\tau^*$ is the value of the optimal *fractional* solution for the corresponding linear programming problem (thus $\tau^*$ is upper bounded by the optimal solution of Subset-FVS).

We use those approximation algorithms to solve MBL (see Algorithm 1). We denote by SCC-sort() an algorithm that decomposes a graph into its strong connected components, and then topological sorts these components. Also, let sort() be an algorithm that sorts a set of integers according to the increasing order of its elements. Algorithm 1 is derived from the constructive proof of Theorem 4, and its correctness follows from Theorem 4 itself.

Depending on the algorithm used for AOG-Subset-FVS, Algorithm 1 can be either an exponential-time exact algorithm, an $O(\log^2|X|)$-approximation or an $O(\log(k)\log\log(k))$-approximation (where $|X|$ is the number of arcs $u \to v$ in $\Pi = (\Sigma, D)$ with $u > v$, and $k$ the optimal value of our problem). Note that the two latter ratios are incomparable, since we may have $|X| \approx nk$ or $k \approx n|X|$ (see for instance Fig. 5a and Fig. 5b in the appendix).

### 5.2 An $(m^2 + 4m - 4)$-approximation Algorithm

In this section, we assume that the partial order $\Pi$ is generated from $m$ gene maps. Recall that a gene map is a totally ordered sequence of blocks, each of which is an unordered set of markers. We exploit this supplementary information to obtain an $(m^2 + 4m - 4)$-approximation algorithm for AOG-Subset-FVS, and therefore a new approximation algorithm, having the same ratio, for MBL. Before giving the algorithm, we first introduce a few definitions: a path $u \xrightarrow{R}{}^*_D v$ in $(\Sigma, D)$ is said to be a *shortcut* of a conflict-cycle $\mathcal{C}$ (see Fig. 3), if:

- $u, v \in \Sigma(\mathcal{C})$ (we write $P$ and $Q$ the paths such that $\mathcal{C} = u \xrightarrow{P}{}^+ v \xrightarrow{Q}{}^+ u$),
- cycle $\mathcal{C}' = u \xrightarrow{P}{}^+ v \xrightarrow{R}{}^*_D u$ is a conflict-cycle,
- $W(Q) \neq \emptyset$ (using the shortcut removes at least one adjacency).

We say that a conflict-cycle is *minimal* if it has no shortcut. With the following property, we ensure that removing minimal conflict-cycles is enough to remove all conflict-cycles.

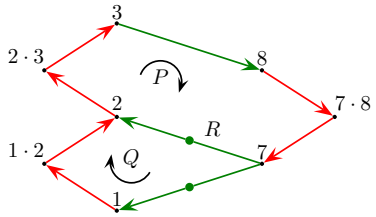



Fig. 3: Cycle $\mathcal{C} = 2 \xrightarrow{P} {}^*7 \xrightarrow{Q} {}^*2$ is a conflict-cycle (it contains $7 \to 1 \in X$). The length-1 path $R$ forms a shortcut for $\mathcal{C}$ (with $\mathcal{C}' = 2 \xrightarrow{P} {}^*7 \xrightarrow{R} {}^*2$, $\mathcal{C}'$ is a conflict-cycle, and $W(Q) = \{1 \cdot 2\}$). So $\mathcal{C}'$ is the only minimal conflict-cycle.

**Algorithm 2** $(m^2 + 4m - 4)$-approximation for AOG-Subset-FVS

**Input:** An adjacency-order graph $G_\Pi = (V, E), X[G_\Pi], w[G_\Pi]$
1. $W'' \leftarrow \emptyset$;
2. **while** there exists a minimal conflict-cycle $\mathcal{C}$ in $G_\Pi[V - W'']$;
3. $\quad L \leftarrow$ the set of low joints of $\mathcal{C}$;
4. $\quad W'' \leftarrow W'' \cup \{e^F \ : \ e \in L\}$;
5. **return** $W''$;

**Property 7** *If an adjacency-order graph contains a conflict-cycle, it also contains a minimal conflict-cycle.*

In a cycle $\mathcal{C}$, we say that a vertex $e \in \Sigma(\mathcal{C})$ is a *low joint*, if in this cycle, $u$ is adjacent to both an arc of $D$ and an arc of $F$ linking it to $e \cdot (e+1)$. Formally, there exist vertices $e^D \in \Sigma(\mathcal{C})$ and $e^F = u \cdot (u+1) \in W(\mathcal{C})$ such that one of the paths $e^D \to_D e \to_F e^F$ or $e^F \to_F e \to_D e^D$ appears in $\mathcal{C}$.

Algorithm 2 is an $(m^2+4m-4)$-approximation for AOG-Subset-FVS. Used as a subroutine in Algorithm 1, it gives us an $(m^2+4m-4)$-approximation for the MBL problem (see Corollary 10). Due to space constraints, the approximation ratio analysis for this Algorithm 2 is only briefly summarized: the first step is to bound the number of low joints that can appear in some minimal conflict-cycle by $m$ (although two markers corresponding to different low joints can appear in the same gene map). Then, we show that for any adjacency $w \in W$, at most $2m$ minimal conflict-cycles using $w$ can appear during step 2. of Algorithm 2. Hence we can bound the number of vertices deleted by Algorithm 2 for a given vertex $w \in W$ by $2m^2$. A tighter analysis gives us the following result.

**Lemma 8** *Let $w \in W$ and let $\mathbb{C}$ be the set of all cycles considered during step 2. of Algorithm 2 going via $w$. Then the cardinality of the set of low joints in cycles of $\mathbb{C}$ is upper bounded by $m^2 + 4m - 4$.*

**Theorem 9** *Algorithm 2 is an $(m^2+4m-4)$-approximation of* AOG-Subset-FVS*, where $m$ is the number of gene maps used to create the input graph.*

*Proof.* Correctness of Algorithm 2 follows from Corollary 6, since Algorithm 2 removes at least one vertex from each conflict-cycle. Let $W^o = \{w_1^o, \ldots, w_k^o\}$ be an optimal solution of size $k$. For each $w_i^o$, Algorithm 2 deletes at most $m^2 + 4m - 4$ adjacencies of $W$ (by Lemma 8). Since every cycle considered by the algorithm goes through some $w_i^o$, the total size of the output solution is at most $k(m^2 + 4m - 4)$. □

**Corollary 10** *Using Algorithm 2 as an approximation for* AOG-Subset-FVS *in Algorithm 1 yields an* $(m^2 + 4m - 4)$*-approximation for the* MBL *problem.*

## 6 Conclusion

In this paper, we revisited the MBL problem with the aim of providing correct algorithms in replacement of those proposed in [5,3], which were based on an inaccurate statement. We proposed a new graph $G_{\Pi}$ to represent the conflicts between the given partial order $\Pi$ and the reference genome $Id_n$, we characterized the cycles containing these conflicts, we showed how the MBL problem reduces to solving the AOG-Subset-FVS problem in $G_{\Pi}$, and we proposed three approximation algorithms. These algorithms allow us to approach a given, practical instance of MBL from different viewpoints, by choosing the appropriate algorithm depending on the data at hand (i.e., whether the instance is created from few gene maps) and on the parameter evaluation ($k$ and $|X|$). We also pointed out that MBL is **APX**-hard ; following this line, it would be interesting to know whether there exists a constant-ratio approximation algorithm for MBL (which would classify MBL as **APX**-complete). Another challenging question is whether MBL is Fixed-Parameter Tractable, notably when the parameter is the number $m$ of gene maps that were used to construct the partial order $\Pi$.

## References


1. P. Alimonti and V. Kann. Hardness of approximating problems on cubic graphs. In G. C. Bongiovanni, D. P. Bovet, and G. Di Battista, editors, *CIAC*, volume 1203 of *LNCS*, pages 288–298. Springer, 1997.
2. G. Blin, E. Blais, D. Hermelin, P. Guillon, M. Blanchette, and N. El-Mabrouk. Gene maps linearization using genomic rearrangement distances. *Journal of Computational Biology*, 14(4):394–407, 2007.
3. X. Chen and Y. Cui. An approximation algorithm for the minimum breakpoint linearization problem. *IEEE/ACM Trans. Comput. Biology Bioinform.*, 6(3):401–409, 2009.
4. G. Even, J. Naor, B. Schieber, and M. Sudan. Approximating minimum feedback sets and multi-cuts in directed graphs. In E. Balas and J. Clausen, editors, *IPCO*, volume 920 of *Lecture Notes in Computer Science*, pages 14–28. Springer, 1995.
5. Z. Fu and T. Jiang. Computing the breakpoint distance between partially ordered genomes. *J. Bioinformatics and Computational Biology*, 5(5):1087–1101, 2007.
6. C. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *J. Comput. Syst. Sci.*, 43(3):425–440, 1991.

7. I.V. Yap, D. Schneider, J. Kleinberg, D. Matthews, S. Cartinhourb, and S.R. McCouch. A graph-theoretic approach to comparing and integrating genetic, physical and sequence-based maps. *Genetics*, 165(4):2235–2247, 2003.
8. C. Zheng, A. Lenert, and D. Sankoff. Reversal distance for partially ordered genomes. In *ISMB (Supplement of Bioinformatics)*, pages 502–508, 2005.
9. C. Zheng and D. Sankoff. Genome rearrangements with partially ordered chromosomes. In L. Wang, editor, *COCOON*, volume 3595 of *Lecture Notes in Computer Science*, pages 52–62. Springer, 2005.

## Appendix for the Referees

### 6.1 Proof details for Theorem 1 (APX-hardness)

**Theorem 1** *The* MBL *problem is* **APX***-hard.*

*Proof.* In [2], Blin et al. show the **NP**-hardness of MBL by reducing it from the Maximum Independent Set problem. The polynomial reduction they use, which they called PO-construction, leads them to the following theorem.

**Theorem** [2] A connected graph $G = (V, E)$ admits an independent set $V' \subseteq V$ such that $|V'| \geq k$ iff there exists a linearization $\pi'$ of $\Pi$ such that $d_B(\pi', \pi) \leq (3n + 2) - k$, where $\pi$ and $\Pi$ result from a PO-construction of $G$.

The PO-construction given in [2] is in fact an L-reduction [6], if we consider the restriction of Maximum Independent Set on cubic graphs, i.e. graphs for which every vertex has degree 3. We write $\alpha(G)$ the maximum value of an independent set of $G$, and $\mathrm{OPT}_{MBL}(\Pi, \pi)$ the optimal value for MBL, with input $\pi$ and $\Pi$. It is known that, for all cubic graphs $G$ with $n$ vertices, $\alpha(G) \geq \frac{n}{4}$. So $\mathrm{OPT}_{MBL}(\Pi, \pi)$, where $\pi$ and $\Pi$ result from the PO-construction of a cubic graph $G$, is at most $3n + 2 - \alpha(G) \leq 11\alpha(G) + 2$: this gives the first inequality of the L-reduction. Concerning the second inequality, we again assume that $\pi$ and $\Pi$ are obtained from a PO-construction of a cubic graph $G$. If we have a linearization $\pi'$ of $\Pi$ such that $d_B(\pi', \pi) = k'$, then we can find an independent set of $G$ of size $k$ with $k \geq 3n+2-k'$. Then $|k-\alpha(G)| \leq -(3n+2-k')+\alpha(G) = |k'-\mathrm{OPT}_{MBL}(\Pi, \pi)|$. Finally, since the restriction of Maximum Independent Set on cubic graphs is an **APX**-hard problem [1], we conclude that MBL is **APX**-hard as well. □

### 6.2 Proof of Property 3

The proof of this property makes uses of the following lemma, which will also be used in the analysis of algorithm 2.

**Lemma 11** *Let $\mathcal{C}$ be a (not necessarily simple) cycle of $G_\Pi$. Let $c \in \Sigma$, such that there exists $a, b \in \Sigma(\mathcal{C})$ with $a \leq c < b$. Then one of the following propositions is true:*

*(i) $\mathcal{C}$ contains an arc $u \rightarrow_X v$ with $v \leq c < u$*
*(ii) $\mathcal{C}$ contains both arcs $c + 1 \rightarrow_F c\cdot(c + 1)$ and $c\cdot(c + 1) \rightarrow_F c$*

*Proof.* Define $c^+ = \{d \mid d > c\} \cup \{d\cdot(d + 1) \mid d > c\}$ and $c^- = \{d \mid d \leq c\} \cup \{d\cdot(d+1) \mid d < c\}$. Then $c^+ \cup \{c\cdot(c+1)\} \cup c^-$ is a partition of $V$. We show that when proposition (i) is false, proposition (ii) is necessarily true. Assume that proposition (i) is false. Since $\mathcal{C}$ contains vertices in both $c^+ \cup \{c\cdot(c + 1)\}$ and $c^-$ (resp. $b$ and $a$), it thus contains an arc $u \rightarrow v$ with $u \in c^+ \cup \{c\cdot(c + 1)\}$ and $v \in c^-$. We must have $u \rightarrow v \in F$, otherwise $u \rightarrow v \in D$ implies $u \rightarrow v \in X$ (since $u > v$), and proposition (i) would be true, a contradiction. Necessarily

$u = c\cdot(c+1)$ and $v = c$ (there is no arc in $F$ going out of $c^+$ into $c^-$). So $\mathcal{C}$ contains the arc $c\cdot(c+1) \to c$. Using the same argument, we can show that there is an arc $u' \to v'$ in $\mathcal{C}$ with $u' \in c^+$ and $v' \in \{c\cdot(c+1)\} \cup c^-$. Since $u' \to v'$ cannot be in $X$ (since proposition (i) is false) nor in $D - X$ (these arcs go from $c^-$ to $c^+$), then it is in $F$, and we can only have $u' = c+1$ and $v' = c\cdot(c+1)$. So $\mathcal{C}$ also uses the arc $c\cdot(c+1) \to_F c$, and thus proposition (ii) is true. □

**Property 3** *Let $\mathcal{C}$ be a simple cycle with $|D(\mathcal{C})| \geq 2$. Then $\mathcal{C}$ is a conflict-cycle.*

*Proof.* Let $a = \min(\Sigma(\mathcal{C}))$, $b = \max(\Sigma(\mathcal{C}))$, and $u \to_D v$ and $u' \to_D v'$ two arcs of $D$ appearing in $\mathcal{C}$. By contradiction, we suppose $\mathcal{C}$ is not a conflict-cycle (i.e., $\mathcal{C}$ contains no arc of $X$). For each $c \in \Sigma$ with $a \leq c < b$, we can use Lemma 11: only proposition (ii) can be true for each $c$, and there is an arc of $F$ going out of $c+1$ and another going into $c$ appearing in cycle $\mathcal{C}$. Since this cycle is simple, we have $u \neq c+1$ and $v \neq c$ for all $a \leq c < b$ (and similarly, $u' \neq c+1$ and $v' \neq c$). By definition of $a$ and $b$, $u, u', v, v'$ cannot be out of the interval $\{a, \ldots, b\}$, so $u = u' = a$ and $v = v' = b$. It implies that the simple cycle $\mathcal{C}$ uses twice the same arc $a \to_D b$, a contradiction. □

### 6.3 Proof of Lemma 5

**Lemma 5** *Let $P = v_1 \to v_2 \to \ldots \to v_\ell$ be a path with vertices in $W' \cup \Sigma$ such that (H1) the vertices $v_i$ are pairwise distinct, (H2) $\ell \geq 2$, (H3) $v_1, v_\ell \in \Sigma$, and (H4) for any $1 \leq i < \ell$, $v_i \to v_{i+1} \in F$.*
*Let $a = \min(v_1, v_\ell)$ and $b = \max(v_1, v_\ell)$. Then the sequence $a\cdot(a+1)\cdot(a+2)\cdot\ldots\cdot b$ is an interval of $\pi$. Moreover, $\Sigma(P) = \{a, \ldots, b\}$ and $W(P) = \{c\cdot(c+1) \mid a \leq c < b\}$.*

*Proof.* Using H4, we can consider only the bipartite graph $(W' \cup \Sigma, F)$. With H3 we obtain that $\ell$ is odd, and $v_{2i-1} \in \Sigma$ $(1 \leq i \leq \frac{\ell+1}{2})$ whereas $v_{2i} \in W'$ $(1 \leq i \leq \frac{\ell-1}{2})$. Moreover, H2 implies that $\ell \geq 3$. The proof is by induction on $\ell$:

*For $\ell = 3$:* Since $v_1 \to_F v_2$ and $v_1 \in \Sigma$, there are two possible cases: (i) $v_2 = v_1\cdot(v_1+1)$ and (ii) $v_2 = (v_1-1)\cdot v_1$.

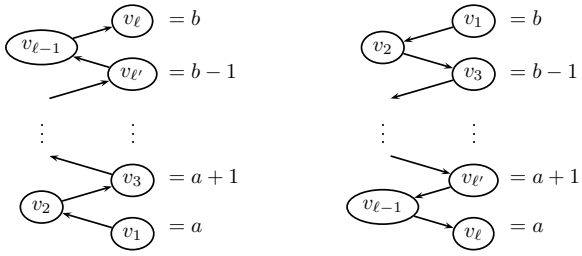
(i) $v_2 = v_1\cdot(v_1+1)$. Then $v_3 = v_1 + 1$ (since $v_2 \to_F v_3$ and $v_3 \neq v_1$, due to H1). In this case, $a = v_1$, $b = a+1$ and $a\cdot b$ is an interval of $\pi$ (indeed, it corresponds to the adjacency $v_2$ which belongs to $W'$).

(ii) $v_2 = (v_1-1)\cdot v_1$. Likewise, $v_3 = v_1 - 1$, $a = v_3$, $b = a+1$, and $a\cdot b$ is an interval of $\pi$.

Finally, in both cases we have $\Sigma(P) = \{v_1, v_3\} = \{a, b\}$ and $W(P) = \{v_2\} = \{a\cdot(a+1)\}$: the lemma is true for $\ell = 3$.

*For $\ell = \ell' + 2$, $\ell' \geq 3$ :* By induction, the lemma is true for the path $P' = v_1 \to \ldots \to v_{\ell'}$. Again, we need to consider two different cases (see Fig. 4a and 4b): (i) $v_1 < v_{\ell'}$ and (ii) $v_1 > v_{\ell'}$.

(i) In this case, $v_{\ell'-1} = (v_{\ell'}-1)\cdot v_{\ell'}$, and by H1, $v_{\ell-1} = v_{\ell'+1} = v_{\ell'}\cdot(v_{\ell'}+1)$, and $v_\ell = v_{\ell'+2} = v_{\ell'} + 1$. If we write $a' = v_1$ and $b' = v_{\ell'}$, then by induction




(a) (b)

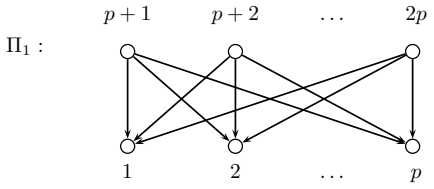
Fig. 4: Two types of paths are possible in $(W' \cup \Sigma, F)$.

the sequence $a' \cdot a'+1 \dots b'$ is an interval of $\pi$. And the sequence $b' \cdot b'+1$ is also an interval of $\pi$, since it corresponds to vertex $v_{\ell'+1} \in W'$. So the full sequence $a' \cdot a'+1 \dots b' \cdot b'+1$ is an interval of $\pi$. This proves the first part of the lemma, since $a = v_1 = a'$ and $b = v_\ell = b'+1$. The second part is also true with $\Sigma(P) = \Sigma(P') \cup \{b\}$ and $W(P) = W(P') \cup \{(b-1) \cdot b\}$.

(ii) This case is symmetric to the previous one, with $a = a'-1 = v_{\ell'}-1$, $b = b' = v_1$ and $v_\ell = a$. We link together the sequences $(a'-1) \cdot a'$ and $a' \cdot (a'+1) \dots b'$ to prove that $a' \cdot (a'+1) \dots b'$ is an interval of $\pi$. Moreover, $\Sigma(P) = \Sigma(P') \cup \{a\}$ and $W(P) = W(P') \cup \{a \cdot (a+1)\}$.

□

### 6.4 Examples for comparing approximation ratios for Algorithm 1




(a) With $\Pi_1$: $|X| = p^2 \ >> \ k = 1$

$\Pi_2 : \ 1 \to 3 \to \dots \to (2p-1) \to 2 \to 4 \to \dots \to (2p)$

(b) With $\Pi_2$: $|X| = 1 \ << \ k = 2p-1$

Fig. 5: Comparing $|X|$ (number of arcs $u \to v$ with $v < u$) to $k$ (optimal breakpoint distance to the identity)

### 6.5 Properties on minimal conflict-cycles

**Property 12** *A minimal conflict-cycle is necessarily simple.*

*Proof.* By contradiction, if a conflict-cycle $\mathcal{C}$ is not simple, then there exists a vertex $u$ used twice in it: $\mathcal{C} = u \xrightarrow{P}{}^{+} u \xrightarrow{Q}{}^{+} u$, and one of $P$ and $Q$, say $P$, uses at least one arc from $X$. We can assume that $u \in \Sigma$ (if $u = i\cdot(i+1) \in F$, $\mathcal{C}$ uses twice at least one vertex in $\{i, i+1\}$). Then the empty path $R$ from $u$ to $u$ is a shortcut of $\mathcal{C}$: the cycle $u \xrightarrow{P}{}^{+} u$ is a conflict-cycle, and $W(Q) \neq \emptyset$ (since $(\Sigma, D)$ is acyclic, and $Q$ is a cycle, $Q$ contains an arc of $F$, hence a vertex of $W$). So $\mathcal{C}$ is not minimal. □

**Property 7** *If an adjacency-order graph contains a conflict-cycle, it also contains a minimal conflict-cycle.*

*Proof.* Take a non-minimal conflict-cycle $\mathcal{C}$. If $\mathcal{C}$ is not simple, there exists a conflict-cycle $\mathcal{C}$' such that $\ell(\mathcal{C}') < \ell(\mathcal{C})$ and $|W(\mathcal{C}')| \leq |W(\mathcal{C})|$ (see Property 12). If $\mathcal{C}$ is simple, we use the shortcut to create a conflict-cycle $\mathcal{C}$' with $|W(\mathcal{C}')| < |W(\mathcal{C}')|$. Applied recursively, this process necessarily ends with a minimal conflict-cycle ($\ell(\mathcal{C})$ and $|W(\mathcal{C})|$ must remain positive integers). □

### 6.6 Analysis of Algorithm 2: definitions and first properties

For this approximation algorithm, we assume that $\Pi$ is created from a limited number $m$ of gene maps (see Section 1). From these gene maps, we can deduce two properties: (1) if there is an arc between $u$ and $v$ in $\Pi$, then $u$ and $v$ appear in consecutive blocks of the same gene map, and (2) if $u$ and $v$ appear in the same gene map, but in different blocks, then there exists a path $u \rightarrow^{+} v$ or $v \rightarrow^{+} u$ in $\Pi$.

In a cycle $\mathcal{C}$, we call *joint* a vertex $e \in \Sigma$ whose incident arcs in $\mathcal{C}$ belong to $D(\mathcal{C})$ and to $F(\mathcal{C})$. Alternatively, there exist vertices $e^D$ and $e^F$ such that

*i.* either both arcs $e^D \rightarrow_D e$, $e \rightarrow_F e^F$ appear in $\mathcal{C}$, or
*ii.* both arcs $e^F \rightarrow_F e$, $e \rightarrow_D e^D$ appear in $\mathcal{C}$.

Consequently, two types of joints are identified: a joint is of type *(i.)* (resp. *(ii.)*) if it marks the end of a subpath with arcs from $D$ (resp. from $F$) and the beginning of a subpath with arcs from $F$ (resp. from $D$). In both cases, $e^F = e\cdot(e+1)$ or $e^F = (e-1)\cdot e$. Hence, $e$ is a *low joint* if $e^F = e\cdot(e+1)$ (intuitively, $e^F$ is at the bottom of a subpath with arcs from $F$).

Given a vertex $w \in W(\mathcal{C})$, we say that $e$ is the *low joint associated to* $w$ in $\mathcal{C}$, if the cycle $\mathcal{C}$ uses one of the paths $w \rightarrow_F^* e$ or $e \rightarrow_F^* w$ ($e$ is either the first low joint after $w$ in $\mathcal{C}$, or the last one before $w$).

For each $u \in \Sigma$, we denote $I(u) \subset \{1, \ldots, m\}$ the numbers of the gene maps in which $u$ appears ($I(u) \neq \emptyset$). For each arc $u \rightarrow_D v$ of $D$, we denote $\eta(u \rightarrow_D v)$ the number of a gene map in which $u$ and $v$ appear in consecutive blocks. Then

$\eta(u \rightarrow_D v) \in I(u) \cap I(v)$. Given a cycle $\mathcal{C}$, we extend the notation $\eta$ to each of its joints $e$: $\eta(e) = \eta(e^D \rightarrow e)$ if $e$ is of type *(i)*; and $\eta(e) = \eta(e \rightarrow e^D)$ otherwise.

We also extend $I$ to paths: given a path $P = u_0 \rightarrow u_1 \rightarrow \ldots \rightarrow u_\ell$, we write

$$I(P) = \bigcup_{\substack{0 \leq i \leq \ell \\ u_i \in \Sigma}} I(u_i)$$

**Lemma 13** *Let $e \rightarrow f$ be an arc of $D$, and let $u \in \Sigma$ such that $\eta(e \rightarrow f) \in I(u)$. Then one of the paths $e \rightarrow^* u$ or $u \rightarrow^* f$ appears in the graph $(\Sigma, D)$.*

*Proof.* The three markers $u$, $e$ and $f$ appear in the same gene map numbered $\eta(e \rightarrow f)$. If $u$ appears in a block strictly following $e$, then there exists a path $e \rightarrow^* u$ in $\Pi$, and thus in the graph $(\Sigma, D)$. Else, since the block of $e$ strictly precedes the block of $f$, $u$ also appears strictly before $f$ in this gene map, so there exists a path $u \rightarrow^* f$ in $\Pi$(and thus in the graph $(\Sigma, D)$). □

### 6.7 Bounding the number of joints in a minimal conflict-circle

**Lemma 14** *Let $\mathcal{C}$ be a minimal conflict-cycle where three vertices $u, e, f \in \Sigma(\mathcal{C})$ are such that (see Fig. 6):*

- $\mathcal{C} = u \overset{P_1}{\rightarrow}{}^+ e \rightarrow_D f \overset{P_2}{\rightarrow}{}^+ u$
- *Each of the paths $P_1$ and $P_2$ uses at least one vertex from $W$ and at least one arc from $D$.*

*Then $\eta(e \rightarrow_D f) \notin I(u)$.*

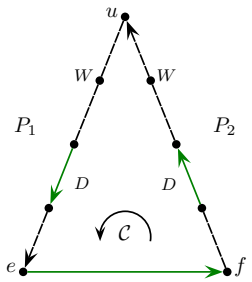



Fig. 6: A cycle satisfying the conditions of Lemma 14 (dotted lines represent paths)

*Proof.* By contradiction, assume that $\eta(e \rightarrow_D f) \in I(u)$. Then (by Lemma 13) there exists a path $R$ in $D$ connecting either $e$ to $u$ or $u$ to $f$. In the first case, we write $P = P_1$ and $Q = e \rightarrow_D f \overset{P_2}{\rightarrow}{}^+ u$, and in the second, $P = P_2$ and

$Q = u \xrightarrow{P_1} {}^+e \rightarrow_D f$, so that there exists a cycle $\mathcal{C}' = u \xrightarrow{P} {}^+e \xrightarrow{R} {}^*u$ (resp., $\mathcal{C}' = f \xrightarrow{P} {}^+u \xrightarrow{R} {}^*f$). Since $\mathcal{C}$ is a minimal conflict-cycle then $R$ cannot be a shortcut, and with $W(Q)$ not being empty, cycle $\mathcal{C}'$ cannot be a conflict-cycle. Hence, no arc in $D(\mathcal{C}')$ appears in $X$.

Let $a = \min(\Sigma(\mathcal{C}'))$ and $b = \max(\Sigma(\mathcal{C}'))$. For all $c \in \{a, \dots, b-1\}$, we can apply Lemma 11 on $\mathcal{C}$', where only proposition (ii) can be true. Hence, $\mathcal{C}$' contains every arc $c+1 \rightarrow_F c$· $(c+1)$, $c$· $(c+1) \rightarrow_F c$ between $b$ and $a$. Moreover, none of these arcs can appear in $R$, so they all come from path $P$.

But $P$ also contains at least one arc from $D$: let $a' \rightarrow_D b'$ be such an arc. By definition of $a$ and $b$, we have $(a', b') \in \{a, \dots, b\}$. The path $P$ is part of a simple cycle, so it can enter and leave only once each vertex. This implies $a' \notin \{a+1, \dots b\}$ and $b' \notin \{a, \dots, b-1\}$, so $a' = a$ and $b' = b$. Thus $P$ contains every arc of the cycle $a \rightarrow_D b \rightarrow_F^* a$, which contradicts the fact that $\mathcal{C}$ is simple. □

**Lemma 15** *Let $\mathcal{C}$ be a minimal conflict-cycle, with $\lambda \geq 5$ joints. Let $e$ and $f$ be two non consecutive joints of $\mathcal{C}$. Then $\eta(e) \neq \eta(f)$.*

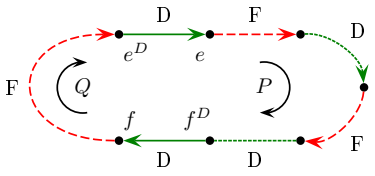



Fig. 7: Illustration of Lemma 15, with $\lambda = 6$. Here $e_s = e^D$, $e_t = e$, $f_s = f^D$, $f_t = t$.

*Proof.* We write $e_s, e_t, f_s, f_t$ the four vertices of $\Sigma$ such that $\{e_s, e_t\} = \{e, e^D\}$, $\{f_s, f_t\} = \{f, f^D\}$, and $\mathcal{C}$ uses both arcs $e_s \rightarrow e_t$ and $f_s \rightarrow f_t$ (see Fig. 7). We write $P$ and $Q$ the paths such that:

$$\mathcal{C} = e_s \rightarrow e_t \xrightarrow{P} {}^* f_s \rightarrow f_t \xrightarrow{Q} {}^* e_s.$$

Since $e$ and $f$ are not consecutive joints, both paths $P$ and $Q$ use an arc from $F$ (and a vertex from $W$). Moreover, since there are at least five joints, one of $P$ and $Q$ has a joint as an inner vertex. Wlog, let this path be $P$. Thus $P$ uses an arc from $D$.

We denote by $Q'$ the path $f_s \rightarrow f_t \xrightarrow{Q} {}^* e_s$, which also contains an arc from $D$. Hence the cycle $\mathcal{C} = f_s \xrightarrow{Q'} {}^* e_s \rightarrow e_t \xrightarrow{P} {}^* f_s$ satisfies the conditions of Lemma 14: $\eta(e) \notin I(f_s)$. This proves the lemma, since $\eta(f) \in I(f_s)$. □

**Lemma 16** *Let $\mathcal{C}$ be a minimal conflict-cycle with $\lambda$ joints. Let $P$ be a path in $F$ between two consecutive joints (e.g. $e_2$ and $e_3$) in $\mathcal{C}$, and $P'$ be the path $P$ where both ends are removed. Then $\lambda \leq 2(m - |I(P')|) + 4$.*

*Proof.* If $\lambda \leq 4$, the lemma is proved, so we can assume there are at least five joints.

Let $u \in \Sigma(P')$. Vertex $u$ appears between joints $e_2$ and $e_3$, so we can use Lemma 14 with $u$, $e_i$ and $e_i^D$ for all $i \notin \{1, 2, 3, 4\}$. Indeed, paths $P_1$ and $P_2$ contain respectively $u \rightarrow_F^+ e_3 \rightarrow_D^+ e_4$ and $e_1 \rightarrow_D^+ e_2 \rightarrow_F^+ u$, so they both use an arc from $D$, an arc from $F$, and a vertex from $W$. Hence, for all $i \in \{5, \ldots, \lambda\}$, we have $\forall u \in \Sigma(P')$, $\eta(e_i) \notin I(u)$, so $\eta(e_i) \notin I(P')$.

By Lemma 15, the cycle $\mathcal{C}$ cannot have more than two joints with the same value of $\eta$, since three joints cannot be pairwise consecutive when $\lambda \geq 5$. Thus we have $\lambda - 4 \leq 2|\{1, \ldots, m\} - I(P')| = 2(m - |I(P')|)$. □

### 6.8 Bounding the number of cycles considered in Algorithm 2

**Lemma 17** *Let $w = v \cdot (v+1) \in W$, $\mathcal{C}_1$ and $\mathcal{C}_2$ two cycles being considered during step 2. of Algorithm 2 ($\mathcal{C}_1$ being considered before $\mathcal{C}_2$), such that $w \in W(\mathcal{C}_1) \cap W(\mathcal{C}_2)$. Denote by $a$ the low joint associated to $w$ in $\mathcal{C}_1$, $b$ the one associated to $w$ in $\mathcal{C}_2$. Then either $\eta(a) \neq \eta(b)$, or $\eta(a) = \eta(b)$, $a$ is of type* ii*, and $a^D$ and $b$ appear in the same block of the gene map $\eta(a)$.*

*Proof.* Vertices $a$ and $b$ are low joints associated to $w = v \cdot (v+1)$, so $a \leq v$ and $b \leq v$. Vertex $a^F = a \cdot (a+1)$ when $\mathcal{C}_1$ is being considered so it cannot appear in $\mathcal{C}_2$. Thus $a < b$, and consequently $b^F$ appears in the path $a \rightarrow_F^* w$ or $w \rightarrow_F^* a$. In $\mathcal{C}_2$, we write $P$ the path linking $w$ and $b$ in $F$, and $Q_2$ the path linking $b^D$ and $w$ in $F \cup D$ (see Fig. 8a). Depending on the type of $b$, we have

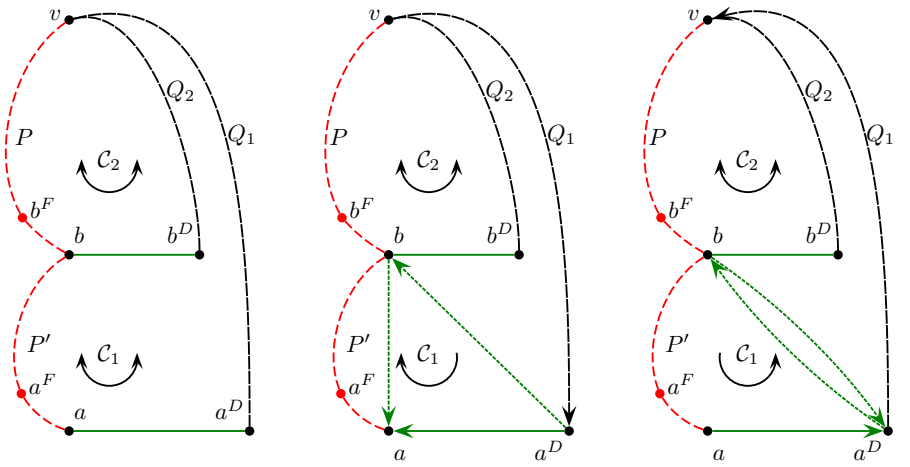
$$i\ \ \mathcal{C}_2 = w \xrightarrow{Q_2}{}^* b^D \rightarrow_D b \xrightarrow{P}{}^* w, \text{ or}$$
$$ii\ \ \mathcal{C}_2 = w \xrightarrow{P}{}^* b \rightarrow_D b^D \xrightarrow{Q_2}{}^* w.$$

In $\mathcal{C}_1$, we write $P'$ the path linking $b$ and $a$ in $F$, and $Q_1$ the path linking $a^D$ and $w$ in $F \cup D$. Thus, depending on the type of $a$, we have

$$i\ \ \mathcal{C}_2 = w \xrightarrow{Q_1}{}^* a^D \rightarrow_D a \xrightarrow{P'}{}^* b \xrightarrow{P}{}^* w, \text{ or}$$
$$ii\ \ \mathcal{C}_2 = w \xrightarrow{P}{}^* b \xrightarrow{P'}{}^* a \rightarrow_D a^D \xrightarrow{Q_2}{}^* w.$$

Note that $P$ can be followed in different ways in $\mathcal{C}_1$ and $\mathcal{C}_2$. We do not distinguish this case in the notations, since paths in $F$ can always be followed in both ways.

We suppose, by contradiction, that $\eta(a) = \eta(b)$.




(a) Decomposition of $\mathcal{C}_1$ and $\mathcal{C}_2$ into arc and paths.

(b) First case (joint $a$ is of type $i$)

(c) Second case (joint $a$ is of type $ii$)

Fig. 8: Path decomposition of $\mathcal{C}_1$ and $\mathcal{C}_2$ and case study for Lemma 17.

*First case: $a$ is a joint of type (i.)* We use Lemma 13 with $b$ and $a^D \to a$: there exists a path $R$ in $(\Sigma, D)$ from $b$ to $a$ or from $a^D$ to $b$ (see Fig. 8b). If $R = b \to^* a$, we define $\mathcal{C}' = a \xrightarrow{P'}{}^* b \xrightarrow{R}{}^* a$, then $\mathcal{C}$' is a conflict-cycle (by Lemma 11: $\mathcal{C}$' cannot contain $a^F \to a$ but visits the vertices $a$ and $a+1$: hence it contains an arc from $X$). Moreover, $W(\mathcal{C}') \subsetneq W(\mathcal{C}_1)$, hence $\mathcal{C}_1$ cannot be a minimal conflict-cycle. Similarly, if $R = a^D \to^* b$, then $\mathcal{C}' = b \xrightarrow{P}{}^* w \xrightarrow{Q_1}{}^* a^D \xrightarrow{R}{}^* b$ is a conflict-cycle and $\mathcal{C}_1$ is not a minimal conflict-cycle. Thus this first case is a contradiction to the fact that $\eta(a) = \eta(b)$.

*Second case: $a$ is a joint of type (ii.)* We distinguish three sub-cases 2.1, 2.2 and 2.3 where, respectively, the path $b \xrightarrow{R}{}^* a^D$ exists in $(\Sigma, D)$, the path $a^D \xrightarrow{R}{}^* b$ exists in $(\Sigma, D)$, and $a^D$ and $b$ are incomparable in $D$ (see Fig. 8b).

2.1 $R = b \to^* a^D$. We consider $\mathcal{C}' = b \xrightarrow{R}{}^* a^D \xrightarrow{Q_1}{}^* w \xrightarrow{P}{}^* b$. If $Q_1$ contains an arc from $X$, $\mathcal{C}$' is a conflict-cycle. Otherwise, the only marked arc is $\mathcal{C}_1$ is necessarily $a \to_X a^D$, so $a > a^D$. Hence the path $R$ in $D$ uses at least one marked arc (since $b > a > a^D$), so $\mathcal{C}'$ is again a conflict-cycle. In both cases, it contradicts the fact that $\mathcal{C}_1$ is a minimal conflict-cycle.

2.2 $R = a^D \to^* b$. We consider $\mathcal{C}' = b \xrightarrow{P'}{}^* a \to a^D \xrightarrow{R}{}^* b$. If $a^D < a$, then $a \to a^D$ is in $X$, so $\mathcal{C}$' is a conflict-cycle. Otherwise, since $\mathcal{C}_1$ is simple, $a^D > b > a$, $R$ contains an arc from $X$, and $\mathcal{C}$' is also a conflict-cycle. Again, $\mathcal{C}_1$ cannot be a minimal conflict-cycle.

2.3 $a^D$ and $b$ are incomparable in $(\Sigma, D)$. Since they appear in the same gene map numbered $\eta(a) = \eta(b)$, they appear in the same block of this map.

□

**Lemma 18** *Let $w = v \cdot (v+1) \in W$, $\mathcal{C}_1$, $\mathcal{C}_2$ and $\mathcal{C}_3$ three cycles being considered during step 2. of Algorithm 2 (in this order), such that $w \in W(\mathcal{C}_1) \cap W(\mathcal{C}_2) \cap W(\mathcal{C}_3)$. Denote respectively by $a$, $b$ and $c$ the low joints associated to $w$ in $\mathcal{C}_1$, $\mathcal{C}_2$ and $\mathcal{C}_3$. Then we cannot have $\eta(a) = \eta(b) = \eta(c)$*

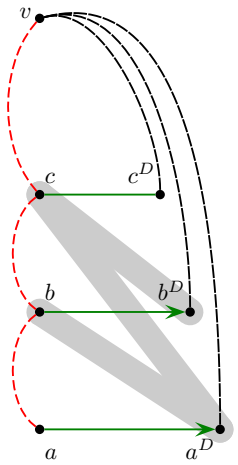



Fig. 9: Illustration of Proof of Lemma 18. Vertices appearing in the same block of the gene map $\eta$ are marked in gray.

*Proof.* Assume that $\eta = \eta(a) = \eta(b) = \eta(c)$. Then we use Lemma 17 with $(\mathcal{C}_1, \mathcal{C}_2)$, $(\mathcal{C}_1, \mathcal{C}_3)$ and $(\mathcal{C}_2, \mathcal{C}_3)$ successively, thus (see Fig. 9) :

- $a^D$ and $b$ appear in the same block of gene map $\eta$,
- $a^D$ and $c$ appear in the same block of gene map $\eta$,
- $b^D$ and $c$ appear in the same block of gene map $\eta$.

So, $b$ and $b^D$ both come from the same block of gene map $\eta$, which contradicts $\eta(b) = \eta$ (in the gene map $\eta(b)$, $b$ and $b^D$ appear in consecutive blocks). □

### 6.9 Proof of Lemma 8

**Lemma 8** *Let $w \in W$ and $\mathbb{C}$ the set of all cycles considered during step 2. of Algorithm 2 going via $w$. Then the cardinality of the set of low joints in cycles of $\mathbb{C}$ is bounded by $m^2 + 4m - 4$.*

*Proof.* We write $w = v{\cdot}\,(v+1) \in W$, and $\mathbb{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_q\}$ the set of the $q$ cycles considered, in this order, by Algorithm 2. In each cycle $\mathcal{C}_h$, $w$ can be associated to a low joint $v_h$ and to the corresponding deleted vertex $w_h = v_h^F = v_h{\cdot}\,(v_h+1)$. We write $P_h$ the path $w_h \rightarrow_F^* w$ or $w \rightarrow_F^* w_h$ in $\mathcal{C}_h$, and $\lambda_h$ the number of joints of $\mathcal{C}_h$. Thus $\frac{\lambda_h}{2}$ is the number of low joints (and the number of deleted vertices) in this cycle.

Since $w_h$ is deleted while $\mathcal{C}_h$ is being considered, for all $h' > h$, $w_h \notin W(\mathcal{C}_{h'})$, and $v_h < v_{h'} \leq v$ (indeed, $\forall u \in \{v_{h'}, \ldots, v\}$, the vertex $u{\cdot}\,(u+1)$ belongs to $W(\mathcal{C}_{h'})$, and $v_h, v_{h'} \leq v$). Finally, the path $P_h$ uses each $v_{h'}$, $h' > h$.

Consider now the list $(\eta(v_{h+1}), \eta(v_{h+2}), \ldots, \eta(v_q))$. Using Lemma 18, the same value cannot appear more than twice in this list, so it contains at least $\left\lceil \frac{q-h}{2} \right\rceil$ different values. The first consequence is $q \leq 2m$ (with $h = 0$, all values are in a set of size $m$). And since for all $h' \in \{h+1, \ldots, q\}$, $\eta(v_{h'}) \in I(P_h)$, we have $|I(P_h)| \geq \left\lceil \frac{q-h}{2} \right\rceil$.

Using Lemma 16 for all $h$, we obtain $\lambda_h \leq 2(m - |I(P_h)|) + 4$. We can thus bound the number of low joints in $\mathcal{C}_h$:

$$\frac{\lambda_h}{2} \leq \frac{2(m - |I(P_h)|) + 4}{2} = m - |I(P_h)| + 2 \leq m - \left\lceil \frac{q-h}{2} \right\rceil + 2$$

By Lemma 15, we also have that, for $m \geq 2$, $\lambda_h \leq 2m$. Let $L_w$ be the number of low joints in the cycles going via $w$:

$$\begin{aligned}
L_w &= \sum_{h=1}^{q} \frac{\lambda_h}{2} \\
&\leq \sum_{h=1}^{q} \min\left(m, m - \left\lceil \frac{q-h}{2} \right\rceil + 2\right) \\
&\leq \sum_{d=0}^{q-1} \min\left(m, m - \left\lceil \frac{d}{2} \right\rceil + 2\right) \\
&\leq \left(\sum_{d=0}^{2m-1} m - \left\lceil \frac{d}{2} \right\rceil + 2\right) - 4 \\
&\leq \left(\sum_{i=0}^{m-1} (m-i+2) + (m-i+1)\right) - 4 \\
&\leq 2m^2 - m(m-1) + 3m - 4 \\
&\leq m^2 + 4m - 4
\end{aligned}$$

thus $L_w \leq m^2 + 4m - 4$, which proves Lemma 8. □

### 6.10 Complexity analysis of Algorithms 1 and 2

The running time of Algorithm 1 is $O(n + |D|)$, plus the running time of the AOG-Subset-FVS subroutine (each 'sort' operation in step 6 can in fact be

done in linear time, since each set $V_i \cap \Sigma$ can be written $\{a, a+1, \ldots, b\}$: we only need to look for the minimum and maximum of this set).

Algorithm 2 can be executed in time $O(n^3)$. This can be done by first precomputing the transitive closure of $(\Sigma, D)$ (while keeping a mark on paths using an arc in $X$). Then we look for minimal conflict-cycles in the following way: for each vertex $u$, temporarily join all vertices $v$ such that $u \rightarrow_X v$ into a single vertex $v_0$, and compute a shortest path from $v_0$ to $u$ (in time $O(n^2)$): this gives a conflict-cycle $u \rightarrow_X v_0 \stackrel{P}{\rightarrow}{}^* u$; or if there is no such path, continue with the next vertex $u$. For all pairs of vertices $(x, y)$ in the cycle, test if the conflict-cycle can be reduced by using a path in $(\Sigma, D)$ (the $O(n^2)$ tests can each be done in constant time, using the precomputed transitive closure of $(\Sigma, D)$). If so, continue with the reduced cycle, until we reach a minimal conflict-cycle. Then, if we write $\ell$ the number of minimal conflict-cycles considered by the algorithm, the total complexity of Algorithm 2 is $O(n^3 + (\ell + n)(n^2 + n^2)) = O(n^3)$ since $\ell < n$ (we delete at least one vertex in $F$ for each cycle).