Perfect Sorting by Reversals

Marie-France Sagot, Eric Tannier

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Perfect Sorting by Reversals

Marie-France Sagot — Eric Tannier

N° 5540

Février 2005

Thème BIO

ISSN 0249-6399 ISRN INRIA/RR--5540--FR+ENG

Perfect Sorting by Reversals

Marie-France Sagot* , Eric Tannier*

Thème BIO — Systèmes biologiques
Projet Helix

Rapport de recherche n° 5540 — Février 2005 — 13 pages

Résumé : En bioinformatique, l'ordre des gènes est souvent modélisé par des permutations signées. Un problème classique pour la comparaison des génomes est la détection des segments conservés, c'est à dire les groupes de gènes situés sur une même région chromosomique dans plusieurs espèces, ce qui semble indiquer qu'ils sont restés groupés au cours de l'évolution. Un second problème très étudié est la reconstitution des événements évolutifs qui transforment une permutation en une autre. Plusieurs travaux commencent à mélanger les principes de segments conservés et réarrangements, et la possible incompatibilité des résultats a plusieurs fois été remarquée. Dans un article récent, Bérard, Bergeron et Chauve posent le problème de l'existence d'un algorithme polynomial qui déterminerait s'il existe un scénario minimum d'inversion qui transformerait un génome en un autre tout en conservant les groupes de gènes communs. Nous donnons ici cet algorithme polynomial, ce qui généralise les résultats de l'article mentionné.

Mots-clés : Tri par inversions, réarrangements dans les génomes, segments conservés de gènes, intervalles communs, bioinformatique

* INRIA Rhône-Alpes

Perfect Sorting by Reversals

Abstract: In computational biology, gene order data is often modelled as signed permutations. A classical problem in genome comparison is to detect conserved segments in a permutation, that is, genes that are co-localised in several species, indicating that they remained grouped during evolution. A second largely studied problem related to gene order data is to compute a minimum scenario of reversals that transforms a signed permutation into another. Several studies began to mix the two problems, and it was observed that their results are not always compatible: often parsimonious scenarios of reversals break conserved segments. In a recent study, Bérard, Bergeron and Chauve stated as an open question whether it was possible to design a polynomial time algorithm to decide if there exists a minimum scenario of reversals that transforms a genome into another while keeping the clusters of co-localised genes together. In this paper, we give this polynomial algorithm, and thus generalise the theoretical result of the aforementioned paper.

Key-words: Reversal Sorting, Genome Rearrangements, Conserved Segments, Common Intervals, Computational Biology

1 Introduction

In computational biology, it is commonly accepted, using a parsimony argument, that if a group of homologous genes (that is genes having a common ancestry) is co-localised in two different species, then they were probably together in the common ancestor and were not later separated during evolution. The detection of such conserved clusters of homologous genes, also called *conserved segments*, has already been the subject of several algorithmic studies (see for instance [1, 9, 10]).

In the theory of rearrangements, applying the parsimony principle means minimising the number of events in a reconstruction of possible evolutionary events between species. The algorithmics related to the rearrangements theory has also been intensively studied. The main results have been obtained on the problem of sorting by reversals [8, 4], which is a common event in evolution. The problem in this case concerns finding an optimal scenario of reversals, that is a shortest sequence of reversals that transforms one genome into the other.

A drawback of the methods developed so far for finding such parsimonious scenarios is that they do not respect the principle of conserved segments : despite the promising title of a former paper, *Common Intervals and Sorting by Reversals : A Marriage of Necessity* [3], it has indeed been noticed several times that in the case of reversals, the two criteria are not always compatible. A minimum rearrangement scenario may break conserved segments and then put them back together later again. A few studies [5, 2, 7] began to mix the two principles. We go further in this direction, thus answering an open question mentioned in [2]. The question concerned the possibility of designing a polynomial time algorithm to decide whether there exists a minimum scenario of reversals that transforms a genome into another while keeping the clusters of co-localised genes together. In this paper, we give this polynomial algorithm, and thus generalise the theoretical result of the aforementioned paper.

This study has also several other motivations. Indeed, a second drawback of the algorithms for obtaining optimal scenarios of rearrangements is the huge number of solutions they provide. Trying to add criteria coming from constraints on conserved segments could be a solution to discriminate more plausible solutions.

Moreover, in another study on the global alignment of mammalian genomes [6], the authors use conserved segments to cluster markers along a genome, and then apply rearrangement algorithms on the clustered data. To cope with the possible incompatibility between the two principles (of conserved segments and of a most parsimonious scenario), they have to use an arbitrary threshold for deciding whether a rearrangement is inside or outside a conserved segment : below this threshold, a rearrangement is ignored and data are clustered, and above the threshold, it is considered as a *bona-fide* rearrangement. The authors of [6] put as an open problem to get rid of this artificial parameter. Knowing how to cluster genes during the execution of a rearrangement algorithm would be a more natural way to answer this question.

This study may be seen as a step towards a new way of detecting conserved segments : an intuitive criterion indicating that a segment is conserved is that it has not been "too"

rearranged during evolution. Dealing with conserved segments inside the rearrangement theory could lead to better detection principles and algorithms.

We describe the usual model for dealing with gene order and orientation in the next section. In Section 3, we recall some basic facts about the structure of conserved segments stated in [9], as well as a padding operation described in [8] and adapted here to conserved segments. Finally, we give our main result and algorithm in Section 4.

2 Chromosomes as Signed Permutations

2.1 Generalities

Genome rearrangements such as reversals may change the order of the genes in a genome, and also the direction of transcription. We identify the genes with the integers $1, \ldots, n$, with a plus or minus sign to indicate their orientation. The order and orientation of genomic markers will be represented by a *signed permutation* of $\{1, \ldots, n\}$, that is, by a bijective function π over $[-n, n] \setminus \{0\}$ such that $\pi_{-i} = -\pi_i$, where $\pi_i = \pi(i)$.

To simplify exposition, we adopt the usual extension which consists in adding $\pi_0 = 0$, and $\pi_{n+1} = n + 1$ to the permutation. We therefore often define a signed permutation by writing $(0\ \pi_1\ \ldots\ \pi_n\ n+1)$. The *identity permutation* $(0\ 1\ \ldots\ n+1)$ is denoted by Id.

For all $i \in \{0, \ldots, n\}$, the pair $\pi_i \pi_{i+1}$ is called a *point* of π, and more precisely an *adjacency* if $\pi_i + 1 = \pi_{i+1}$ and a *breakpoint* otherwise. The number of points of a permutation π is denoted by $p(\pi)$, and the number of its breakpoints by $b(\pi)$.

The *reversal* of the interval $[i, j] \subseteq [1, n]$ $(i \leq j)$ is the signed permutation $\rho_{i,j} = (0\ \ldots\ i-1\ -j\ \ldots\ -i\ j+1\ \ldots\ n+1)$. Note that $\pi \cdot \rho_{i,j}$ is the permutation obtained from π by reversing the order and flipping the signs of the elements in the interval $[i, j]$:

$$\pi \cdot \rho_{i,j} = (\pi_0\ \ldots\ \pi_{i-1}\ \ -\pi_j\ \ldots\ -\pi_i\ \ \pi_{j+1}\ \ldots\ \pi_{n+1})$$

If $\rho_1, \ldots, \rho_k$ is a sequence of reversals, we say that it *sorts* a permutation π if $\pi \cdot \rho_1 \cdots \rho_k = Id$. The length of a shortest sequence of reversals that sorts π is called the *reversal distance* of π, and is denoted by $d(\pi)$. A shortest sequence of reversals sorting π is called a *parsimonious* sequence.

A *segment* of a permutation π is a set $\{|\pi_a|, \ldots, |\pi_b|\}$, with $1 \leq a < b \leq n$. The numbers π_a and π_b are the *endpoints* or *extremities* of the segment. Two segments are said to *overlap* if they intersect but one is not contained in the other. A reversal $\rho_{i,j}$ *breaks* $\{|\pi_a|, \ldots, |\pi_b|\}$ if $[i, j]$ and $[a, b]$ overlap. A sequence of reversals *breaks* a segment S if at least one reversal of the sequence breaks S.

2.2 The breakpoint graph

The breakpoint graph is a usual tool for dealing with signed permutations. It is present in almost every study on sorting by reversals. We use it intensively in the proofs of correctness of our method.

The *breakpoint graph* $BG(\pi)$ of a permutation π is a graph with vertex set V defined as follows : for each integer i in $\{1,\ldots,n\}$, let i^- and i^+ be two vertices in V ; add to V the two vertices 0^+ and $(n+1)^-$. Observe that all vertex labels are non negative numbers, but for simplicity and to avoid having to use absolute values, we may later refer to vertex $(-i)^+$ (or $(-i)^-$) : this is the same as vertex i^+ (i^-).

The breakpoint graph of a signed permutation has sometimes been called the *diagram of desire and reality* due to the edge set E of $BG(\pi)$, which is the union of two perfect matchings of V, denoted by R, the *reality edges* and D, the *desire edges* :

- D contains the edges $i^+(i+1)^-$ for all $i \in \{0,\ldots,n\}$;
- R contains an edge for all $i \in \{0,\ldots,n\}$, from π_i^+ if π_i is non negative, and from π_i^- otherwise, to π_{i+1}^- if π_{i+1} is non negative, and to π_{i+1}^+ otherwise.

Reality edges define the permutation π (what you have), and desire edges define Id (what you want to have).

To avoid case checking, in the notation of an edge, the mention of the exponent $+$ or $-$ may be omitted. For instance, $\pi_i\pi_{i+1}$ is a reality edge, indicating nothing as concerns the signs of π_i and π_{i+1}.

It is easy to check that every vertex of $BG(\pi)$ has degree two (it has one incident edge in R and one in D), so the breakpoint graph is a set of disjoint cycles. By the cycles of a permutation π, we mean the cycles of $BG(\pi)$. The number of cycles of π is denoted by $c(\pi)$.

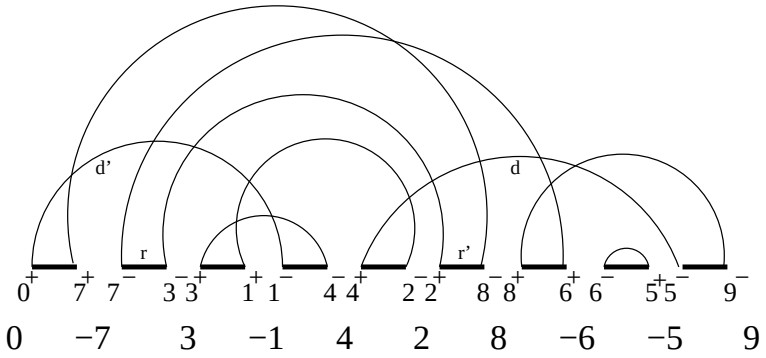

FIG. 1 – The breakpoint graph of the permutation $(0\ -7\ 3\ -1\ 4\ 2\ 8\ -6\ -5\ 9)$. Reality edges are represented in bold, and desire edges are represented by thin lines.

2.3 Conserved segments

Let π be a signed permutation of $\{1,\ldots,n\}$, and $S = \{|\pi_a|,\ldots,|\pi_b|\}$ a segment of π, for $[a,b] \subseteq [1,n]$. Let $m = \min_{i\in[a,b]} |\pi_i|$ and $M = \max_{i\in[a,b]} |\pi_i|$.

The segment S is said to be *oriented* if there exist $i,j \in [a,b]$, such that π_i and π_j have different signs, and *unoriented* otherwise.

The segment S is said to be *sorted* if for all $i \in [a, b-1]$, the point $\pi_i\pi_{i+1}$ is an adjacency. A sorted segment is always unoriented. It is sorted *positively* if $\pi_a > 0$ and *negatively* if $\pi_a < 0$. In $\pi = (0\ -7\ \ 3\ -1\ \ 4\ \ 2\ \ 8\ -6\ -5\ \ 9)$, $\{6, 5\}$ is sorted negatively.

The segment S is said to be *conserved* if $M - m = b - a$. In $\pi = (0\ -7\ \ 3\ -1\ \ 4\ \ 2\ \ 8\ -6\ -5\ \ 9)$, $\{3, 1, 4, 2\}$ is conserved.

The segment S is said to be *isolated* if it is conserved, and either $\pi_a = m$ and $\pi_b = M$, or $\pi_a = -M$ and $\pi_b = -m$. We require in addition that the first and last point of the segment are breakpoints, and not adjacencies. In $\pi = (0\ -7\ \ -4\ \ 2\ \ -3\ \ -1\ \ 8\ -6\ -5\ \ 9)$, the segment $\{4, 2, 3, 1\}$ is isolated.

The segment S is said to be *strongly conserved* if there exists a parsimonious sequence of reversals which does not break S.

A strongly conserved segment is conserved, but the converse is not true. For example, in $(0\ -2\ -3\ \ 1\ \ 4)$, $\{2, 3\}$ is conserved but any parsimonious scenario breaks it.

An isolated segment is not always strongly conserved. However, the permutations for which this is not the case are rare and irrelevant for our study, as we shall see in Section 2.4.

According to [2], we say that a sorting sequence of reversals is *perfect* if it breaks no conserved segment. If a permutation has a perfect parsimonious scenario, then all its conserved segments are strongly conserved. The converse is however not true : for example, in $(0\ -3\ \ 4\ -1\ \ 2\ \ 5)$, both $\{1, 2\}$ and $\{3, 4\}$ are strongly conserved, but any parsimonious sequence of reversals breaks one of them.

Perfect sorting sequences of minimum size have been studied in [7]. It is proved that given a permutation and a subset $\mathcal{S}$ of its conserved segments, it is NP-hard to compute the minimum scenario that does not break the segments of $\mathcal{S}$. The problem of finding a perfect sequence of reversals of minimum length is still open, to our knowledge. In [7], the following easy but fundamental lemma is presented.

Lemma 1 *If a sequence of reversals sorts a permutation and does not break a segment S, then there exists a sorting sequence of same size (with the same reversals), in which all the reversals contained in S (they sort S) are before all other reversals (they sort outside S).*

The parsimonious scenario such that the smallest possible number of reversals break some conserved segments is evoked in [2], but not solved. The authors study a special class of permutations for which there exists a perfect parsimonious scenario, and the question is asked whether it is possible to decide in polynomial time, given a permutation, if there is a perfect parsimonious scenario that sorts it. In Section 4, we give this algorithm. Before that, we still need some preliminaries.

2.4 Sorting by Reversals

The main result about sorting by reversals is a theorem of Hannenhalli and Pevzner [8], which yielded the first polynomial algorithm to find a parsimonious sequence of reversals sorting any signed permutation.

We mention here a weaker version of this theorem, to avoid introducing notions which are useless for our purpose. One of the consequences of the general version of Hannenhalli and Pevzer's theorem is that it is possible to characterise the permutations for which all parsimonious sequences of reversals have to break some isolated segment. According to the standard vocabulary, they are the permutations that need a "hurdle merging". They can be characterised in this way.

Lemma 2 *[8] A permutation has an isolated segment which is not strongly conserved if and only if it has at least three unoriented isolated non sorted segments A, B, C, such that either $A \subset B \subset C$, or $A \subset B$ and $C \cap B = \emptyset$.*

We call such permutations *fools*. Such permutations will obviously never have a perfect parsimonious scenario. We therefore start by assuming that the permutations we treat are not fools. It is easy to decide in linear time if a permutation is a fool or not (see for example [4]). We denote by $u(\pi)$ the number of unoriented isolated segments in a permutation π.

Theorem 1 *[8] Let π be a permutation but not a fool. Then $d(\pi) = p(\pi) - c(\pi) + u(\pi)$.*

This means that any reversal in a parsimonious scenario increases the number of cycles of the permutation ($p(\pi) = n+1$ does not change after a reversal), except one (the first one) for each unoriented isolated segment. Each isolated segment is sorted separately (by definition, they do not overlap), and independently from the rest of the permutation. If it is unoriented, any reversal which does not break conserved segments, and does not decrease $c(\pi)$ is a good candidate for a perfect parsimonious scenario. After its application, the isolated segment is oriented. It is therefore enough to design our algorithm for oriented isolated segments, and to try every possibility at the first step for unoriented ones.

3 The structure of conserved segments

As mentioned in the previous section, two isolated segments cannot overlap, and so it is easy to identify an inclusion-wise minimal one, and then treat it separately. This is not immediately the case in general for conserved segments. However, a nice structure studied in [9] and an operation on permutations first described in [8] will allow to consider conserved segments in the same way as isolated segments, that is, to identify an inclusion-wise minimal conserved segment, "isolate" it, and sort it apart from the rest of the permutation when it is possible.

3.1 Minimal conserved segments

We recall basic facts about the structure described in [9] that are useful for our purpose. The first one immediately follows from the definition of a conserved segment.

Lemma 3 *If two conserved segments C and D intersect, then $C \cap D$ and $C \cup D$ are conserved segments.*

A conserved segment is called *irreducible* if it is not the union of two overlapping conserved segments. A *chain* is a sequence of irreducible segments $S_1, \ldots, S_k$, such that S_i and S_{i+1} overlap for all i. The elements of a maximal chain form a conserved segment, which is called a *basic* segment. The following lemma follows from the structure of irreducible segments studied in [9].

Lemma 4 *The family of basic segments of a permutation is nested, in the sense that two basic segments are disjoint or one is contained in the other. The intersection of two overlapping irreducible segments is either a singleton, or is a basic segment.*

The nested structure of basic segments allows to consider inclusion-wise minimal basic segments (not containing any other basic segment). The particular properties of minimal basic segments follow from the previous lemma.

Lemma 5 *If $S = \{|\pi_a|, \ldots, |\pi_b|\}$ is an inclusion-wise minimal basic segment, then either S is irreducible, and none of its proper subsets is conserved, or every proper subset of S is conserved, and $|\pi_a|, \ldots, |\pi_b|$ is an increasing or decreasing sequence.*

Now that we have drawn the possibility to identify a minimal conserved segment with particular properties, we are going to try to "isolate" it.

3.2 Padding a permutation

Isolating a conserved segment is done by an operation called *padding* that is described in [8]. It consists in transforming a permutation into a simpler one, with equivalent properties. We show that it can be used to deal with conserved segments as well. To begin with, we want to be able to add elements to a permutation without changing the existing indices. To do so, we deal with "generalised" signed permutations in the sense that the permutations will be bijective functions from a set of indices to a set of values, both being ordered sets of reals instead of integer numbers. For example, $(0 \quad 3.5 \quad -3 \quad 1 \quad 2 \quad 4)$ is a generalised permutation, where $\pi_0 = 0$, $\pi_{0.5} = 3.5$, $\pi_1 = -3$, $\pi_2 = 1$, $\pi_3 = 2$, $\pi_4 = 4$.

A *padding* of a permutation π consists in adding an index k such that $i < k < i + 1$, for some existing index i, and its image through π, such that $j < |\pi_k| < j + 1$, for some existing value j (π_k may be positive or negative). For example, if $\pi = (0 \quad -3 \quad 1 \quad 2 \quad 4)$ is a signed permutation over $\{0, 1, 2, 3, 4\}$, $\pi' = (0 \quad 3.5 \quad -3 \quad 1 \quad 2 \quad 4)$, is a padding of π with $0 < k < 1$ and $3 < |\pi_k| < 4$. The resulting generalised permutation π' has the same breakpoint graph as π, except that the two edges $r = \pi_i\pi_{i+1}$ and $d = j(j + 1)$ are now each split into two edges r_1, r_2 and d_1, d_2. An example of padding is shown in Figure 2.

A padding is said to be *safe* if the resulting permutation π' has one cycle more than π, and no new unoriented isolated segment, that is, according to Theorem 1, if $d(\pi') = d(\pi)$. Any sequence sorting π' also sorts π (just ignore the added element). If the transformation is safe, a parsimonious scenario for π' will therefore provide a parsimonious scenario for π. By extension, a sequence of paddings is *safe* if the resulting permutation π' satisfies $d(\pi') = d(\pi)$.

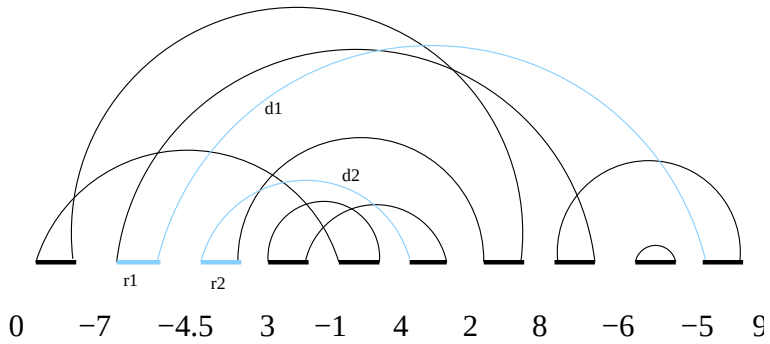

FIG. 2 – A padding of the permutation $(0\ -7\ 3\ -1\ 4\ 2\ 8\ -6\ -5\ 9)$. The index 1.5 is added, with value -4.5. The breakpoint graph is unchanged, except for the two edges r, d (see Figure 1) that have been deleted and replaced by r_1, r_2 and d_1, d_2. The four resulting edges are drawn in gray. The resulting permutation has one cycle more than the original one and there is no unoriented isolated segment, so the padding is safe.

Let $S = \{|\pi_a|, \ldots, |\pi_b|\}$ ($[a, b] \subseteq [1, n]$) be a conserved segment, $M = \max_{i\in[a,b]} |\pi_i|$, and $m = \min_{i\in[a,b]} |\pi_i|$. We say that S has a *positive padding* if it is safe to pad it with an index k_1, such that $a - 1 < k_1 < a$, and $m - 1 < \pi_{k_1} < m$, and then with an index k_2, such that $b < k_2 < b + 1$, and $M < \pi_{k_2} < M + 1$. We say that S has a *negative padding* if it is safe to pad it with an index k_1, such that $a - 1 < k_1 < a$, and $M < -\pi_{k_1} < M + 1$, and then with an index k_2, such that $b < k_2 < b + 1$, and $m - 1 < -\pi_{k_2} < m$.

For example, in Figure 3, there is a negative padding of the conserved segment $\{3, 1, 4, 2\}$ in the permutation $(0\ -7\ 3\ -1\ 4\ 2\ 8\ -6\ -5\ 9)$. There is no positive padding of the same segment, as shown later in Figure 4.

Note that if the segment S is isolated, then it has a padding, positive if $\pi_a = m$, negative if $\pi_a = -M$. After a padding of a segment S, positive or negative, $S \cup \{\pi_{k_1}, \pi_{k_2}\}$ is isolated. The number of cycles of the breakpoint graph has to increase by two if the padding is safe.

Lemma 6 *A segment is strongly conserved if and only if it has a safe padding (positive or negative).*

Proof. If a segment S is strongly conserved in a permutation π, then let $\rho_1, \ldots, \rho_k$ be a parsimonious sequence sorting π without breaking S, and say $\rho_1, \ldots, \rho_l$ are the reversals contained in S (from Lemma 1, they may be positioned first in the sequence). Then in $\pi \cdot \rho_1 \cdots \rho_l$, S is a sorted segment, let us say it is sorted positively (resp. negatively). At this point, it is obvious that a positive (resp. negative) padding is safe (it creates adjacencies). Consider the same padding with the original permutation π. This padding is safe because the sequence $\rho_1, \ldots, \rho_k$ sorts the resulting permutation π', thus $d(\pi) = d(\pi') = k$.

Conversely, if there is a safe padding of S, call π' the obtained permutation, π'_{k_1} and π'_{k_2} the two added values. After the padding, $S \cup \{\pi'_{k_1}, \pi'_{k_2}\}$ is an isolated oriented segment

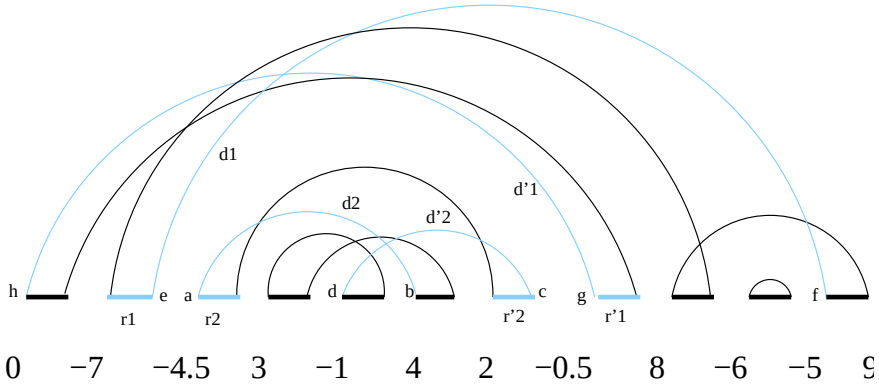

FIG. 3 – A negative padding of segment $\{3, 1, 4, 2\}$ in the permutation $(0\ -7\ 3\ -1\ 4\ 2\ 8\ -6\ -5\ 9)$. Note that there are 3 cycles in the breakpoint graph, and no unoriented isolated segment. The segment $\{4.5, 3, 1, 4, 2, 0.5\}$ is isolated, but oriented. There was only one cycle in $(0\ -7\ 3\ -1\ 4\ 2\ 8\ -6\ -5\ 9)$, so the padding is safe.

of π'. From Lemma 2, there is a parsimonious sequence sorting π' which does not break $S \cup \{\pi'_{k_1}, \pi'_{k_2}\}$ because there is no new unoriented isolated segment, so the permutation is not a fool. The same sequence sorts π, it is parsimonious because the padding is safe, and it does not break S. Then S is strongly conserved. □

We now have the possibility to identify a minimal conserved segment, and test whether it is strongly conserved or not. The remaining difficulty is to choose between a positive and a negative padding when both are possible.

4 Perfect parsimonious sequences of reversals

As noticed in [7], the main difficulty in finding perfect sequences of reversals of minimum length (among all perfect sequences) is that it is sometimes impossible to decide whether to sort a particular segment positively or negatively. We shall see that in the case of parsimonious scenarios, this choice is constrained by the data.

We denote by $d_+(S)$ the minimum number of reversals needed to sort a conserved segment S positively, and $d_-(S)$ the minimum number of reversals needed to sort it negatively. Of course, $|d_+(S) - d_-(S)| \leq 1$, because if it is sorted in one direction, then one reversal is sufficient to have it sorted in the other. If $d_+(S) = d_-(S)$, the segment S is called *neutral*.

If there is a perfect parsimonious scenario, any conserved segment is strongly conserved, so any segment has a safe padding from Lemma 6. However, the converse is not true. The possibility of designing a simple algorithm to decide the existence of a perfect parsimonious scenario is given by the following lemma.

Lemma 7 *If a segment is neutral, then it cannot have both a positive and a negative padding.*

Proof. Let $[a,b] \subseteq [1,n]$, such that $S = \{|\pi_a|, \ldots, |\pi_b|\}$ is a neutral conserved segment. Let $M = \max_{i\in[a,b]} |\pi_i|$, and $m = \min_{i\in[a,b]} |\pi_i|$. Suppose S has a negative padding, call π^- the resulting permutation. This means it is safe to pad S with an index k_1, such that $a-1 < k_1 < a$, and $M < -\pi^-_{k_1} < M+1$, and an index k_2, such that $b < k_2 < b+1$, and $m-1 < -\pi^-_{k_2} < m$. Let r and d be the reality and desire edges deleted after the padding with the index k_1, and r' and d' the reality and desire edges deleted after the padding with the index k_2 (see Figure 1 for an example). Suppose S has also a positive padding, and call π^+ the resulting permutation. This means that it is safe to pad S with the index k_1, with $m-1 < \pi^+_{k_1} < m$, and the index k_2, with $M < \pi^+_{k_2} < M+1$. The deleted edges are the same ones, except that r and d' are deleted after the padding with k_1, and r' and d are deleted after the padding with k_2.

As both paddings are safe, the number of cycles has to increase for each padding operation. We therefore have that r and d belong to the same cycle of π, as well as r' and d', r and d', r' and d. So r, d, r' and d' all belong to the same cycle in the breakpoint graph of π. Observe that because S is conserved, r, d, r' and d' are the only edges that have one extremity with a label inside S, and the other outside S.

In π^-, d and d' are both replaced by two edges, say d_1, d_2, and d'_1, d'_2. One edge among the two has its extremities with labels inside S, and the other outside S. As in Figure 3, say $d_2 = ab$, and $d'_2 = cd$ have their extremities with labels inside S.

In π^+, the edges which replace d and d' and have their extremities with labels inside S are ad and cb (see Figure 4 for an example).

Recall that d, d', r, r' are the only edges affected by the paddings, the remaining of the breakpoint graph is unchanged. So if ab and cd belong to different cycles in π^-, then ad and cb belong to the same cycle in π^+, and vice-versa. In this case however, the segment S would not be neutral as $d_-(S) \neq d_+(S)$ from Theorem 1. Since S is neutral, the edges ab and cd have to belong to the same cycle both in π^- and in π^+. It is the case in the example of Figures 3 (for π^-) and 4 (for π^+).

We now repeat the argument for the edges that have their extremities with labels outside S, that is d_1, d'_1. In π^-, let $d_1 = ef$ and $d'_1 = gh$. Then in π^+, $d_1 = eh$ and $d'_1 = gf$. If d_1 and d'_1 are in different cycles in π^-, they are in the same cycle in π^+, and vice-versa. We therefore have that either $c(\pi^-) = c(\pi) + 1$, or $c(\pi^+) = c(\pi) + 1$, and one the paddings is not safe, because in this case, either $d(\pi^+) > d(\pi)$, or $d(\pi^-) > d(\pi)$. This is what happens in Figure 4, where $d(\pi^+) > d(\pi^-) = d(\pi)$.

As a consequence, a neutral segment cannot have both a positive and negative padding. □

The principle of the algorithm follows immediately.

Theorem 2 *Given a permutation π, it is possible to design in polynomial time a perfect parsimonious sequence of reversals sorting π if one exists.*

Proof. We apply the usual techniques to sort oriented isolated segments by reversals. We do not describe this in detail. One can see for instance [11] for a fast method to do that.

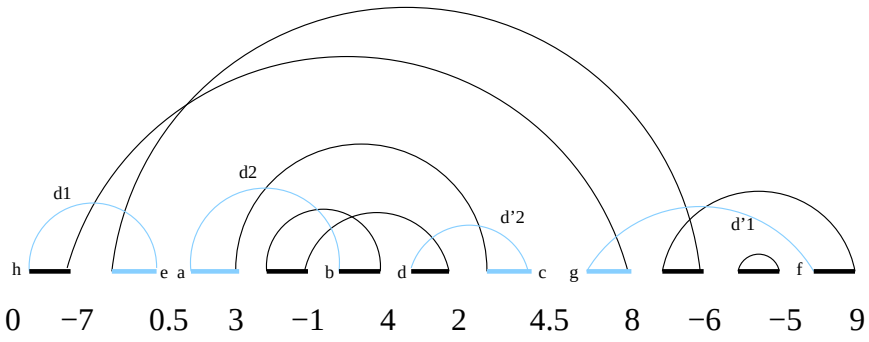

FIG. 4 – An attempt of a positive padding of the segment $\{3,1,4,2\}$ in the permutation $(0\ -7\ 3\ -1\ 4\ 2\ 8\ -6\ -5\ 9)$. The graph is almost the same as for the negative padding. However edges are changed from ab and cd to ad and bc, so in one case there are two cycles outside the segment while in the other case there is only one such cycle. This explains why both paddings are impossible.

Let π be an arbitrary permutation. We first check if it is not a fool (see Lemma 2). If it is, there is no perfect parsimonious sorting sequence.

We now treat each isolated segment separately, starting with the inclusion-wise minimal ones, up to the whole permutation, or stopping when a contradiction is found. This is the only way of obtaining a perfect parsimonious solution, from Theorem 1.

- *Sorting an oriented isolated segment*

Suppose first that the considered isolated segment I (possibly $\{0,\ldots,n+1\}$) is oriented. Let now S be any inclusion-wise minimal basic conserved segment inside I. We try both paddings of S to see if it is strongly conserved. If S is not strongly conserved, then there is no perfect parsimonious sequence. If both paddings exist, then by Lemma 7, the segment S is not neutral. In this case, we choose the positive padding if $d_+(S) < d_-(S)$, and the negative one otherwise. If $d_+(S) = d_-(S)$, the choice of the padding is constrained. In every case, the permutation is padded with two values π_{k_1} and π_{k_2}, and $S \cup \{\pi_{k_1}, \pi_{k_2}\}$ is now isolated. By Lemma 5, either no proper subset of S is conserved, or all its proper subsets are conserved. We examine the two cases separately.

In the first case, we sort S with, for example, the method of [11], for isolated segments. This preserves all conserved segments because there is none inside S, and the way S is sorted does not affect the remaining of the permutation.

In the second case, the only allowed reversals are the reversals of singletons (the reversals of the whole interval is not an admitted operation, since otherwise the opposite padding is chosen). The reversals of singletons never overlap, so they may be applied in any order. If the padding is positive, we reverse all negative numbers, and if the padding is negative, we reverse all the positive numbers. At the end, the segment S is sorted.

We apply the same method to all the basic conserved segments, starting with inclusion-wise minimal ones, up to the segment I itself.

- *Sorting an unoriented isolated segment*

Let us suppose now that I is unoriented. The first reversal will make it oriented, and then the aforementioned method ("Sorting an oriented isolated segment") is applied. To orient I, we choose a reversal that does not break any conserved segment, nor decreases $c(\pi)$. Every such reversal has to be tried. There are at most $|I|^2$ of them, and they are applied only at the first step, so this operation yields a polynomial algorithm.

- *Conclusion*

We have seen how to sort an isolated segment. All are sorted the same way, and separately. If there is a perfect parsimonious sequence sorting π, then the algorithm produces it, because the way an isolated segment is sorted never affects the permutation outside the segment.

This method therefore decides if there is a perfect parsimonious sequence sorting π in polynomial time. □

Références

[1] Beal M.-P., Bergeron, A., Corteel, S., Raffinot, M., "An Algorithmic View of Gene Teams", *Theor. Comput. Sci.* 320(2-3) :395-418, 2004.

[2] Bérard S., Bergeron A. and Chauve C., "Conserved structures in evolution scenarios", 2nd RECOMB Comparative Genomics Satellite Workshop, to appear in *Lecture Notes in BioInfomatics*, 2004.

[3] Bergeron A., Heber S. and Stoye J., "Common Intervals and Sorting by Reversals : A Marriage of Necessity", *Bioinformatics*, 1 :1-10, 2002.

[4] Bergeron A., Mixtacki J., Stoye J., "The inversion distance problem", in *Mathematics of evolution and phylogeny* (O. Gascuel Ed.) Oxford University Press, 2005.

[5] Bergeron, A., Stoye, J., "On the Similarity of Sets of Permutations and its Applications to Genome Comparison", COCOON'03 Proceedings, *Lecture Notes in Computer Science*, vol. 2697, 68-79, 2003.

[6] Bourque G., Pevzner P. and Tesler G., "Reconstructing the genomic architecture of ancestral Mammals", *Genome Research* 14 :507-516, 2004.

[7] Figeac M. and Varré J.-S., "Sorting by reversals with common intervals", Proceedings of WABI 2004, *Lecture Notes in Computer Science*, vol. 3240, 26-37, 2004.

[8] Hannenhalli S. and Pevzner P. , "Transforming cabbage into turnip (polynomial algorithm for sorting signed permutations by reversals", *Proceedings of the 27th ACM Symposium on Theory of Computing*, 178-189, 1995.

[9] Heber S. and Stoye J., "Finding all Common Intervals of k Permutations", Proceedings of CPM 2001, *Lecture Notes in Computer Science*, vol. 2089, 207-218, 2001.

[10] Heber S. and Stoye J., "Algorithms for Finding Gene Clusters", Proceedings of WABI 2001, *Lecture Notes in Computer Science*, vol. 2149, 252-263, 2001.

[11] Tannier E. Bergeron A. and Sagot M.-F., "Advances on Sorting by Reversals", to appear in *Discrete Applied Mathematics*, 2005.

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399