Genome rearrangement by reversals and insertions/deletions of contiguous segments

Nadia El-Mabrouk[1]

Département d'informatique et de recherche opérationnelle, Université de Montréal,
CP 6128 succursale Centre-Ville, Montréal, Québec, H3C 3J7.
mabrouk@iro.umontreal.ca

Abstract. Analysis of genome rearrangements allows to compare molecular data from species that diverged a very long time ago. Results and complexities are tightly related to the type of data and genome-level mutations considered. For sorted and signed data, Hannenhalli and Pevzner (HP) developed the first polynomial algorithm in the field. This algorithm solves the problem of sorting by reversals. In this paper, we show how to extend the HP approach to include insertions and deletions, allowing to compare genomes containing different genes. We provide an exact algorithm for the asymmetric case, as applies in organellar genome evolution, and a heuristic for the symmetric case, with bounds and a diagnostic for determining whether the output is optimal.

1 Introduction

The polynomial algorithm of Hannenhalli and Pevzner (hereafter "HP") for sorting by reversals [2] was a breakthough for the formal analysis of evolutionary genome rearrangement. And they were able to extend their approach to include the analysis of interchromosomal translocations [3]. The applicability of this theory, however, has been limited by the difficulty in extending it to other rearrangement mechanisms, such as gene duplication, or the transposition, duplication or insertion of genomic fragments. In this paper, I show how to extend the HP approach to include insertions and deletions, providing an exact algorithm for the asymmetric case, as applies in organellar genome evolution, and a heuristic for the symmetric case, with bounds and a diagnostic for determining whether the output is optimal.

This work will be situated in the context of circular genomes, as is most frequently the case for single chromosomal genomes such as organelles and prokaryotes. We assume that the direction of transcription is known. Formally, given a set $\mathcal{A}$ of genes, a genome is a circular string of signed ($+$ or $-$) elements of $\mathcal{A}$, no gene appearing more than once. A **gene segment** is a substring, a subsequence of consecutive genes in a genome. For a string $X = x_1 x_2 \cdots x_r$, denote by $-X$ the **reverse** string $-x_r - x_{r-1} \cdots - x_1$. A **reversal** (or **inversion**) transforms some proper substring of a genome into its reverse.

Let G and H be two genomes with some genes in common, and others specific to each genome. Write $\mathcal{A}$ for the set of genes in both G and H, and $\mathcal{A}_G$ and $\mathcal{A}_H$ for those in G and H only respectively. Our problem is to find the minimum number of reversals, insertions and deletions of gene segments necessary to transform G to H, with the restriction that deletions and insertions are not applyed on genes present in both genomes G and H. Although such manipulations my be meaningful in some circonstances, in this paper we constrain insertions/deletions to apply on genes present in $\mathcal{A}_G$ and $\mathcal{A}_H$ only.

2 The Hannenhalli-Pevzner theory.

Given a set $\mathcal{A}$ of genes and two circular genomes G and H such that each gene of $\mathcal{A}$ appears exactly once in each genome, how can we find the minimum number of reversals necessary to transform G into H? The HP solution is an exact polynomial algorithm, based on a bicoloured cycle graph constructed from G and H. This graph will be the focus of our extension to the problem to include segment insertion and deletion. We first sketch the HP method.

The HP bicoloured **cycle graph** $\mathcal{G}$ of G with respect to H is as follows. If gene x_i in genome $G = x_1 \cdots x_n$ has positive sign, replace it by the pair $x_i^t x_i^h$, and if it is negative, by $x_i^h x_i^t$. Then the vertices of $\mathcal{G}$ are just the x^t and the x^h for all x in $\mathcal{A}$. Without confusion we can also apply the terminology "vertices of G" to mean elements like x^t or x^h. Any two vertices which are adjacent in G, other than x_i^t and x_i^h from the same x, are connected by a black edge, and any two adjacent in H, by a gray edge. Each vertex is incident to exactly one black and one gray edge, so that there is a unique decomposition of $\mathcal{G}$ into c disjoint cycles of alternating edge colours. By the **size of a cycle** we mean the number of black edges it contains. Note that c is maximized when $G = H$, in which case each cycle is of size 1 and $c = |\mathcal{A}|$.

Let e and f be two black edges of a cycle C of length $t > 1$. Think of both e and f as being oriented clockwise in the graph. Each one of them induces an orientation of C. If the orientation induced by e and f coincide, we say that they are **convergent**; otherwise we say that they are **divergent**.

A reversal is determined by the two points where it "cuts" the current genome, which correspond each to a black edge. We say that a **reversal ρ is determined by the two black edges** $e = (a, b)$ and $f = (c, d)$ if ρ reverses the segment of G bounded by vertices b and c. The **reversal determined by a gray edge** g is the reversal determined by the two black edges adjacent to g.

Let ρ be a reversal determined by the two black edges e and f, and $\Delta(c)$ the increase in the number of cycles of the graph after the reversal ρ.

1. If e and f belong to different cycles, then $\Delta(c) = -1$.
2. If e and f belong to the same cycle and converge, then $\Delta(c) = 0$.
3. If e and f belong to the same cycle and diverge, then $\Delta(c) = 1$.

The goal is to perform as many reversals of the third type as possible. A key concept in the method is the **unoriented components**. A gray edge g in a cycle of size > 1 is **oriented** if the two black edges adjacent to g diverge. Otherwise, the edge is **unoriented**. A cycle C is **oriented** if it contains at least one oriented gray edge. Otherwise it is **unoriented**. Two cycles containing gray edges that "cross", e.g., gene a adjacent to gene c in Cycle 1, gene b adjacent to gene d in Cycle 2 in H, but ordered a, b, c, d in G, are **connected**. A (connected) **component** of $\mathcal{G}$ is a subset of the cycles, built recursively from one cycle, at each step adding all the remaining cycles connected to any of those already in the construction. An **oriented component** has at least one oriented cycle and thus at least one oriented gray edge. Otherwise it is an **unoriented component**.

We say that component U **separates** two components U' and U'' if any edge we tried to draw from a vertex of U' to one of U'' would cut a gray edge of U. A **hurdle** is an unoriented component which does not separate any pair of components. We further distinguish between two types of hurdles. A hurdle $\mathcal{H}$ is a **superhurdle** if the elimination of $\mathcal{H}$ from $\mathcal{G}$ transforms a non-hurdle $\mathcal{H}'$ into a hurdle, and is a **simple hurdle** if not. A cycle graph $\mathcal{G}$ is a **fortress** if it contains an odd number of hurdles, and they are all superhurdles.

Two hurdles $\mathcal{H}_1$ and $\mathcal{H}_2$ are **consecutive** in $\mathcal{G}$ if there is no other hurdle $\mathcal{H}$ in $\mathcal{G}$ situated between $\mathcal{H}_1$ and $\mathcal{H}_2$ (clockwise or counterclockwise).

HP showed that the minimum number of reversals necessary to transform G into H is: $\quad I(G,H) = b(G,H) - c(G,H) + h(G,H) + fr(G,H)$ $\;$ (**1**), where $b(G,H)$ is the number of black edges, $c(G,H)$ is the number of cycles (of size ≥ 1), $h(G,H)$ is the number of hurdles of $\mathcal{G}$, and $fr(G,H)$ is 1 if $\mathcal{G}$ is a fortress and 0 otherwise.

The HP algorithm for transforming G into H with a minimum of reversals is described in Figure 1. Note that $h(G,H)$ changes value during the execution of the algorithm. In [2] it is shown that at each step, a reversal satisfying the conditions of the algorithm can be found.

Remark 1 : A reversal determined by two divergent black edges in a single unoriented cycle C transforms C into an oriented cycle C'; the total number of cycles of the graph stays the same. A reversal determined by two black edges from two different unoriented cycles C_1 and C_2 transforms C_1 and C_2 into a single oriented cycle C; the total number of cycles in the graph decreases by 1.

We extend the HP method to take into account deletions in Section 3, insertions in Section 4, and deletions plus insertions in Section 5. For reasons of space, we omit proofs in this abstract.

3 Deletions

Returning to the problem at hand, we require the minimum number of **rearrangement operations**, i.e., reversals, insertions and deletions, necessary to transform genome G into genome H. We first consider the case where there are no insertions. All genes in H are also in G, i.e. $\mathcal{A}_H = \emptyset$. The problem is then to transform G into H with a minimum of reversals and deletions. Given that the genes of $\mathcal{A}_G$ are destined to be deleted, their identities and signs are irrelevant, and could be replaced with any symbols different from those used for the genes in $\mathcal{A}$. For any subsequence of vertices of G of form $S = u_1 u_2 \cdots u_{p-1} u_p$, where

Algorithm HP:
While $\mathcal{G}$ contains a cycle of size > 1
do
  If $\mathcal{G}$ contains an oriented connected component $\mathcal{CC}$
    Choose an oriented gray edge g of $\mathcal{CC}$ such that the reversal determined by g does not create a new unoriented connected component;
    Execute the reversal determined by g;
  Otherwise
    If $h(G, H)$ is even {hurdle merging}
      Consider two appropriate (not consecutive) hurdles $\mathcal{H}_1$ and $\mathcal{H}_2$;
      Arbitrarily choose a black edge e in $\mathcal{H}_1$ and a black edge f in $\mathcal{H}_2$, and execute the reversal determined by e and f;
    Otherwise
      If $h(G, H)$ is odd and there is a simple hurdle $\mathcal{H}$ {hurdle cutting}
        Execute the reversal determined by two arbitrarily chosen edges e and f belonging to a single cycle in $\mathcal{H}$;
        Otherwise {fortress, hurdle merging}
          If $\mathcal{G}$ is a fortress with more than three superhurdles
            Consider two not consecutive hurdles $\mathcal{H}_1$ and $\mathcal{H}_2$;
            Arbitrarily choose a black edge e in $\mathcal{H}_1$ and a black edge, f in $\mathcal{H}_2$ and execute the corresponding reversal;
          Otherwise {3-fortress, hurdle merging}
            Consider any two hurdles $\mathcal{H}_1$ and $\mathcal{H}_2$;
            Arbitrarily choose a back edge e in $\mathcal{H}_1$ and a black edge f in $\mathcal{H}_2$, and execute the corresponding reversal;

Fig. 1. The HP algorithm for sorting genomes by reversals.

u_1, u_p correspond to two genes of $\mathcal{A}$, and for all i, $2 \leq i \leq p-1$, u_i corresponds to a gene of $\mathcal{A}_G$, we replace S by the subsequence $S' = u_1 \delta(u_1) u_p$.

Example 1 *Consider sets* $\mathcal{A} = \{a, b, c, d, e, f\}$, $\mathcal{A}_G = \{g, h, i, j, k, l\}$, *and the circular genome* G *(i.e. the first term follows the last one):* $+g\ +a\ -h\ +b\ +c\ +i\ +d\ +l\ +e\ -j\ +k\ +f$. *Then* G *can be rewritten:* $\delta(f^h)\ +a\ \delta(a^h)\ +b\ +c\ \delta(c^h)\ +d\ \delta(d^h)\ +e\ \delta(e^h)\ +f$.

We represent genomes G and H by the bicoloured cycle graph $\mathcal{G}(G, H)$, where $\mathbf{V}$ is the set of vertices, B is the set of black edges and D is the set of gray edges, these three sets defined as follows :

- $\mathbf{V} = \{x^s\}_{x \in \mathcal{A}}^{s \in \{h,t\}}$.
- The black edges pertain to genome G. There are two sorts of black edges: **direct** black edges link two adjacent vertices in G, and **indirect** black edges link two vertices separated by a δ. For an indirect black edge $e = (a, b)$, $\delta(a)$ is the **label of e**.
- Gray edges link adjacent vertices in H.

An **indirect cycle** is one containing at least one indirect edge, and a hurdle containing at least one indirect cycle is an **indirect hurdle**, otherwise it is a **direct hurdle**.

Example 2 *Consider the genome G of example 1 and the genome $H = +a$ $+d$ $+c$ $+b$ $+f$ $+e$. The graph $\mathcal{G}(G, H)$ corresponding to these two genomes is depicted in Figure 2. This graph is made up of two cycles, C_1 and C_2, both indirect. These two cycles are unoriented and form a hurdle.*

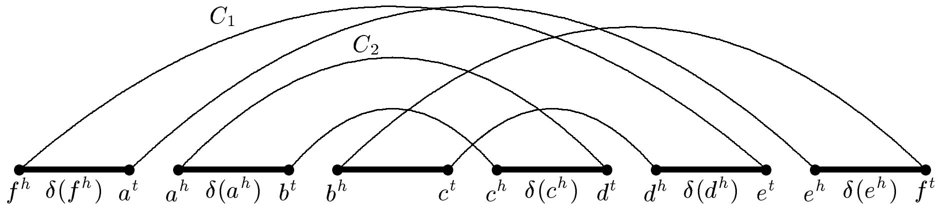

Fig. 2. The black edges with a δ are indirect, i.e., there are genes of $\mathcal{A}_G$ between the two vertices which determine the edge.

The **reduced genome** $\overline{G}$ induced by G is obtained by deleting from G all the genes in $\mathcal{A}_G$. The cycle graph $\mathcal{G}(\overline{G}, H)$ associated with genomes $\overline{G}$ and H is obtained from $\mathcal{G}(G, H)$ by considering all the black edges of B to be direct.

For a reduced genome, the reversal acting on the two black edges $e = (a, b)$ and $f = (c, d)$ reverses the segment of the genome $\overline{G}$ bounded by vertices b and c. For a genome G containing genes to be deleted, the indirect edges determine, not an adjacency in the genome G, but an interval containing only genes to be deleted. We thus have to define what we mean by "the reversal acting on two black edges", of which at least one is indirect.

Let $e = (a, b)$ and $f = (c, d)$ be two indirect edges. The interval $[b, c]$ designates the interval of G bounded by the two vertices b and c; the interval $[b, \delta(c)]$ designates the interval of G bounded on the left by b and on the right by the element of $\mathcal{A}_G$ adjacent to d.

Definition 1. *Consider two black edges $e = (a, b)$ and $f = (c, d)$. The* **reversal** ρ **determined by** e **and** f *is defined as follows. If e and f are not both indirect, ρ reverses segment $[b, c]$. If e and f are indirect, ρ reverses segment $[b, \delta(c)]$ (see Figure 3.a).*

Consider an oriented gray edge $g = (b, d)$ adjacent to two black edges $e = (a, b)$ and $f = (c, d)$. The **reversal** ρ **determined by** g *reverses segment $[b, c]$ if f is direct, and reverses segment $[b, \delta(c)]$ if f is indirect (see Figure 3.b).*

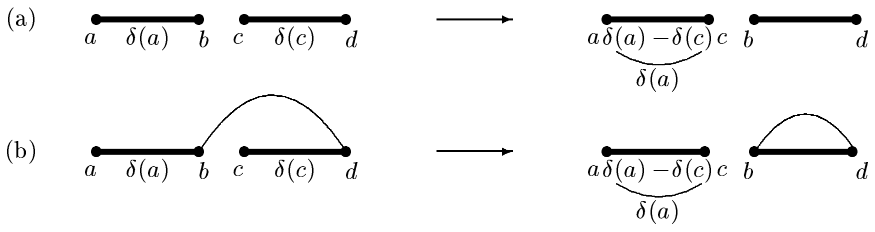

Fig. 3. (a) Reversal determined by two indirect black edges. (b) Reversal determined by a gray edge adjacent to 2 indirect black edges. This creates a direct cycle of size 1.

The HP algorithm can be generalized to graph containing direct and indirect edges by making use of the above definition of a reversal determined by two black edges, or by a gray edge. After each reversal, for each vertex a, $\delta(a)$ determines the string of genes of $\mathcal{A}_G$ following a in the resulting genome. These strings of genes change throughout the execution of the algorithm.

Lemma 1 *A reversal determined by an oriented gray edge of a cycle C transforms C into two cycles C_1 and C_2, at least one of which is of size 1. Suppose this is true of C_1. Then the black edge of C_1 is direct (Figure 3.b).*

Corollary 1 *An oriented cycle C of size n is transformed by the HP algorithm, in $n-1$ reversals, into n cycles of size 1. If C is direct, the n cycles are all direct. If not, only one of these cycles is indirect.*

Corollary 1 states that for each oriented cycle, the HP algorithm gathers all the genes to delete from C into a single segment. At the end of the algorithm, a single deletion is required for each oriented cycle. Now consider the solution for unoriented cycles forming hurdles. The merging of two hurdles $\mathcal{H}_1$ and $\mathcal{H}_2$ is achieved by combining two unoriented cycles C_1 and C_2 one from each of the two hurdles. This gets rid of the two hurdles, and creates an oriented cycle. This cycle is then solved by HP as described above. So that there are as few deletions as possible, the strategy is to carry out as many merging as possible of pairs of indirect cycles. This requires careful consideration of hurdle choice, and cycle choice for each hurdle, within the HP algorithm. Merging will occur only when the total number of hurdles is even. To keep as many indirect hurdles as possible, if the initial number of hurdles is odd and some hurdle must be cut, we will prefer to do this to a direct hurdle. In summary:

hurdle merging
  Choose two non-consecutive hurdles $\mathcal{H}_1$ and $\mathcal{H}_2$, indirect if possible, in a certain way;
  Choose e in $\mathcal{H}_1$ and f in $\mathcal{H}_2$, indirect if possible;
hurdle cutting
  Choose a simple hurdle $\mathcal{H}$, direct if possible;
  Perform the reversal determined by any 2 black edges from the same cycle of $\mathcal{H}$.

It remains to specify precisely how to choose hurdles to merge. Let $\mathcal{SH}$ be the set of hurdles of $\mathcal{G}(G,H)$, and let $h(G,H) = |\mathcal{SH}|$. We subdivide $\mathcal{SH}$ into subsets $I_1, D_1, \cdots I_p, D_p$, where each I_i is a maximal set made up of consecutive indirect hurdles, bounded at either end by maximal sets D_{i-1} and D_{i+1} of consecutive direct hurdles. The order within the sets I_i and D_i is arbitrary. We set $I_{p+1} = I_1$, and $I_0 = I_p$ (and similarly for D_{p+1} and D_0). The sets I_i, for $1 \leq i \leq p$, are called direct. The order in which hurdle merging occurs is given by the procedure "hurdle-merging" described in Figure 4.

From now on, when we refer to the HP algorithm, this choice procedure will be assumed. Let $\alpha(G,H)$ be the number of merging during procedure hurdle-merging.

Procedure hurdle-merging
For all i, $1 \leq i \leq p$, do
  While there are at least three hurdles in I_i do
    Perform the merging of any two non-consecutive hurdles in I_i;
If $p > 1$
  Perform the merging of 2 arbitrary hurdles belonging to different direct sets of size 1;
  Let $I'_1, \cdots, I'_r$ be the direct sets of size 2 remaining at this step;
  For all j, $1 \leq j \leq r$, do
    Perform the merging of any hurdle of I'_j with any hurdle of I'_{j+1};
Otherwise
  If $p = 1$ and there are no direct hurdles
    Perform the merging of the two remaining hurdles of I_1.

Fig. 4. A suitable order for hurdle merging

Lemma 2 *Procedure hurdle-merging is correct. Moreover, if* $p = 1$, $h(G, H)$ *is even and there is at least one direct hurdle, then* $\alpha(G, H) = \frac{h(G,H)}{2} - 1$. *Otherwise,* $\alpha(\mathcal{G}) = \left\lfloor \frac{h(G,H)}{2} \right\rfloor$.

Lemma 3 *Let* $\alpha(P)$ *be the number of merging of pairs of indirect hurdles of* $\mathcal{G}(G, H)$*, according to any order* P*. Then* $\alpha(G, H) \geq \alpha(P)$.

Let $\mathcal{G}_1(G, H)$ be the graph containing only size 1 cycles obtained after applying the HP algorithm to the graph $\mathcal{G}(G, H)$. Consider procedure deletion, described as follow, applied to $\mathcal{G}_1(G, H)$:

Procedure deletion:
For each indirect edge $e = (a, b)$ of $\mathcal{G}_1(G, H)$
  Delete the gene segment between a and b, symbolized by $\delta(a)$;

We will call the HP algorithm augmented by Procedure deletion the **Reversal-deletion algorithm**. It is clear that this algorithm transforms the genome G into the genome H. Denote by $ci(G, H)$ the number of indirect cycles of $\mathcal{G}(G, H)$, and $c(G, H) = ci(G, H) - \alpha(G, H)$.

Theorem 1. *Let* $I(\overline{G}, H)$ *be the minimum number of reversals necessary to transform* $\overline{G}$ *into* H*. The number of reversals and deletions performed by the Reversal-deletion algorithm is:* $ID(G, H) = I(\overline{G}, H) + c(G, H)$ *where* $I(\overline{G}, H)$ *is the number of reversals, and* $c(G, H)$ *is the number of deletions. Moreover,* $ID(G, H)$ *is the minimum number of rearrangement operations necessary to transform* G *to* H.

4 Insertions

The case $\mathcal{A}_G = \emptyset$, where the set of genes of G is a subset of the genes of H, i.e. the problem of transforming G into H with a minimum of reversals and insertions, can be solved by the reversals-deletions analysis where G takes on the role of H, and vice versa. Each deletion in the H-to-G solution becomes an insertion in the G-to-H. Each reversal in one direction corresponds to a reversal of the same segment in the opposite direction. Nevertheless, we will examine the case of insertions in order to introduce some concepts for the subsequent analysis.

Consider then the problem of transforming H to G by a minimum number of reversals and deletions. Let $\mathcal{G}(H,G)$ be the corresponding graph. We will not, however, replace the genes of H to be deleted by the δ parameters. Let $ID(H,G) = I(\overline{H},G) + c(H,G)$ be the minimum number of reversals and deletions necessary to transform H into G, where $I(\overline{H},G)$ is the number of reversals and $c(H,G)$ the coresponding number of deletions. Let $\mathcal{G}_1$ be the graph containing only cycles of size 1 obtained by applying the HP algorithm to the graph $\mathcal{G}(H,G)$. The number of indirect cycles of this graph is $c(H,G)$. Each indirect black edge corresponds to an adjacency in G, and is labeled by a sequence of genes of $\mathcal{A}_H$. Consider the genome G^c obtained from G as follows: for each pair of adjacent vertices a and b in G, if (a,b) is an indirect black edge of $\mathcal{G}_1$ labeled by the sequence of genes $y_1 \cdots y_p$ de $\mathcal{A}_H$, then insert this sequence of genes between a and b in G. We call G^c the **filled genome of** G relative to H.

Lemma 4 *The minimum number of reversals and insertions necessary to transform G into H is $II(G,H) = I(\overline{H},G) + c(H,G)$. Moreover, $I(\overline{H},G)$ is the number of reversals, and $c(H,G)$ is the number of insertions.*

Example 3 *Consider the sets $\mathcal{A} = \{a,b,c,d,e,f,g\}$, $\mathcal{A}_H = \{y_1,y_2,y_3,y_4\}$, and the two genomes $G = +a\ +b\ +c\ +d\ +e\ +f\ +g$ and $H = +y_1\ +d\ +y_2\ +f\ +b\ +y_3\ +c\ +e\ +y_4\ +a\ +g$. Figure 5 depicts the graph $\mathcal{G}(H,G)$.*

Given that $\mathcal{G}(H,G)$ contains 7 black edges, 3 cycles and 1 hurdle, $I(\overline{H},G) = 7-3+1 = 5$. The HP algorithm does 5 reversals, and produce the following filled genome: $G^c = +y_1\ -y_2\ -y_4\ +a\ +b\ +y_3\ +c\ +d\ +e\ +f\ +g$. Two insertions are necessary to transform G into G^c.

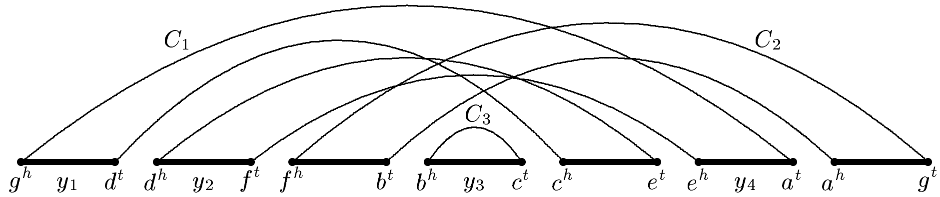

Fig. 5. $\mathcal{G}(H,G)$ contains 7 black edges, 3 cycles, of which 2 are indirect, and 1 hurdle.

5 Insertions and deletions

Consider now the general case where $\mathcal{A}_G \neq \emptyset$ and $\mathcal{A}_H \neq \emptyset$. Let $\overline{G}$ be the reduced genome of G, and $\overline{H}$ the reduced genome of H.

Lemma 5 *Let $IDI(G,H)$ be the minimum number of rearrangement operations necessary to transform G into H. Then $IDI(G,H) \geq I(\overline{G},\overline{H}) + c(H,\overline{G}) + c(G,\overline{H})$.*

Let $\overline{G}^c$ be the filled genome of $\overline{G}$ relative to H. Given that $\overline{G}^c$ and H contain the same set of vertices, the edges of $\mathcal{G}(\overline{G}^c,H)$ must all be direct.

We must now take into account the vertices of $\mathcal{A}_G$. Let a and b be two adjacent vertices in $\overline{G}$, separated in G by a segment $X(a)$ of vertices of $\mathcal{A}_G$. If a and b are separated in $\overline{G}^c$ by a segment $Y(a)$ of vertices of $\mathcal{A}_H$, then we must place $X(a)$ before or after $Y(a)$, so that we require as few deletions as possible to transform $\overline{G}^c$ into H. By G^c we denote the genome obtained by appropriately adding the vertices of $\mathcal{A}_G$ to $\overline{G}^c$. The graph $\mathcal{G}(G^c, H)$ is obtained from $\mathcal{G}(\overline{G}^c, H)$ by transforming certain black edges of $\mathcal{G}(\overline{G}^c, H)$ into indirect edges.

The procedure **construct-G^c** constructs the graph $\mathcal{G}(G^c, H)$ corresponding to a genome G^c, and consequently, determines the genome G^c, based on $\mathcal{G}(\overline{G}^c, H)$. This graph is constructed so that the corresponding parameter $c(G^c, H)$ is minimal. We write $X(a)$ for the gene segment of $\mathcal{A}_G$ which follows vertex a in G, if there is one. At each step of the construction, we consider a cycle (or a hurdle) of the graph obtained to be indirect if it contains indirect edges (constructed during previous steps). In addition, note that when an edge of the graph is labeled, it becomes indirect.

Procedure construct-G^c:

For each black edge (a, b) of $\mathcal{G}(\overline{G}^c, H)$ such that a is succeeded in G by a segment $X(a)$ do
  If (a, b) is a (direct) black edge of $\mathcal{G}(\overline{G}^c, H)$
    Transform (a, b) into indirect edge by labeling it by $X(a)$;
  Otherwise, let $y_1 \cdots y_p$ be the sequence of vertices of $\mathcal{A}_H$ situated between a and b in $\overline{G}^c$
    If one of the two black edges (a, y_1) or (y_p, b) belongs to an indirect cycle,
      Consider one such edge and label it by $X(a)$;
    Otherwise
      If one of the black edges (a, y_1) or (y_p, b) belongs to an indirect hurdle
        Consider one such edge and label it by $X(a)$;
      Otherwise
        Label by $X(a)$ either one of the two edges;

After having obtained the genome G^c from G by $c(H, \overline{G})$ suitable insertions, the transformation of G^c into H is obtained by the Reversal-deletion algorithm applied to the graph $\mathcal{G}(G^c, H)$. This algorithm does $I(\overline{G}, \overline{H})$ reversals and $c(G^c, H)$ deletions.

We summarize the transformation from G to H as follows:

1. Apply the HP algorithm to the graph $\mathcal{G}(H, \overline{G})$. The final graph obtained contains $c(H, \overline{G})$ indirect edges.
2. Do the $c(H, \overline{G})$ insertions deducible from the indirect edges, and transform $\overline{G}$ to $\overline{G}^c$.
3. Fill the genome $\overline{G}^c$ to obtain genome G^c, using Procedure construct-G^c.
4. Apply the Reversal-deletion algorithm to the graph $\mathcal{G}(G^c, H)$. The algorithm does $I(\overline{G}, \overline{H})$ reversals and $c(G^c, H)$ deletions.

Proposition 1. *The number of rearrangement operations occurring during the application of the method is:* $IDI_1(G, H) = I(\overline{G}, \overline{H}) + c(H, \overline{G}) + c(G^c, H)$, *where*

$I(\overline{G},\overline{H})$ is the number of reversals, $c(H,\overline{G})$ is the number of insertions and $c(G^c,H)$ is the number of deletions.

Moreover, the number of reversals and insertions is minimal, and if $c(G^c,H) = c(G,\overline{H})$, then the total number of rearrangement operations is minimal.

Example 4 *Consider the sets $\mathcal{A} = \{a,b,c,d,e,f,g\}$, $\mathcal{A}_G = \{x_1,x_2,x_3,x_4,x_5\}$, $\mathcal{A}_H = \{y_1,y_2,y_3,y_4\}$, and the two genomes $G = +x_1\ +a\ +x_2\ +b\ +c+x_3\ +d\ +x_4\ +e\ +x_5\ +f\ +g$, $H = +y_1\ +d\ +y_2\ +f\ +b\ +y_3\ +c\ +e\ +y_4\ +a\ +g$.*

Given that $\overline{G}$ and H are the two genomes in Example 3, Figure 6 represents graph $\mathcal{G}(H,\overline{G})$. Two insertions are necessary to transform $\overline{G}$ into $\overline{G}^c = +y_1\ -y_2\ -y_4+a\ +b\ +y_3\ +c\ +d\ +e\ +f\ +g$.

Figure 6 depicts $\mathcal{G}(G^c,H)$, which contains two indirect cycle C_1 and C_2, and a single hurdle. Therefore $c(G^c,H) = 2$. Moreover, since $I(\overline{G},\overline{H}) = 5$ (Example 3), 5 reversals followed by 2 deletions are necessary to convert G^c to H. The total number of rearrangement operations used to transform G to H is thus $2+5+2=9$.

On the other hand, the construction of graph $\mathcal{G}(G,\overline{H})$ reveals that $c(G,\overline{H}) = 2$. Nine is thus the minimal number of rearrangement operations necessary to transform G into H.

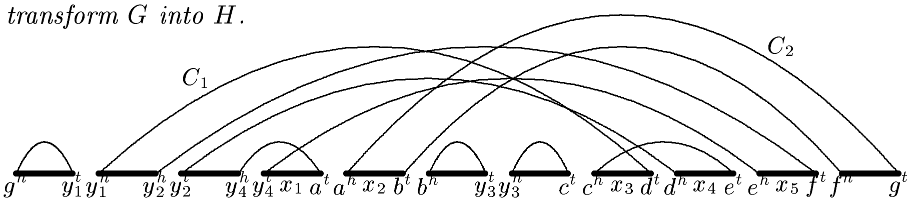

Fig. 6. The graph $\mathcal{G}(G^c,H)$.

Theorem 2. *Let $ci(G,\overline{H})$ be the number of indirect edges of $\mathcal{G}(G,\overline{H})$, and let $IDI_1(G,H)$ be the number of rearrangement operations obtained by the method described above. Then:*

$$I(\overline{G},\overline{H}) + c(H,\overline{G}) + c(G,\overline{H}) \leq IDI_1(G,H) \leq I(\overline{G},\overline{H}) + c(H,\overline{G}) + ci(G,\overline{H})$$

Remark 2 : A better solution will sometimes be found by calculating the minimum number of operations necessary to transform H to G. Thus it is a good idea to always consider
$\min(IDI_1(G,H), IDI_1(H,G))$.

The time complexity of our method depends on the complexity of the HP algorithm. If we require only the minimum number of reversals necessary to transform a genome G into a genome H, both of size n, it suffices to construct the cycle graph of G and H, and to count the number of cycles, super-hurdles and simples hurdles within it. As presented in [2], the cycle graph is constructed in $O(n^2)$ time. The complexity becomes $O(n^4)$ if we want to produce an explicit sequence of reversals which transform G to H. The HP algorithm was sped up by Berman and Hannenhalli [1]. The complexities they report are expressed in terms of $\alpha(n)$, the inverse of Ackerman's function. The construction of cycle graph is $O(n\alpha(n))$ and the actual production of the reversals is $O(n^2\alpha(n))$. Kaplan, Shamir and Tarjan [4] further improved on this complexity by simplifying the HP algorithm. The production of a minimal sequence of reversals becomes $O(n^2)$. We will assume this complexity for the HP algorithm in analyzing our extension to deletion and insertion.

Theorem 3. *Consider* $n = \max(|\mathcal{A}| + |\mathcal{A}_G|, |\mathcal{A}| + |\mathcal{A}_H|)$. *The transformation of* G *to* H *by the above method is of time complexity* $O(n^2)$.

Remark 3 : The determination of the parameter $c(G^c, H)$ necessitates the construction of genome G^c. To do this, the HP algorithm must be applied to graph $\mathcal{G}(H, \overline{G})$. The complexity of our method thus remains $O(n^2)$, even if we are only looking for the minimum number of rearrangement operations, and not the actual operations.

6 An application to the comparison of mitochondrial genomes

To test our method on some realistic data of moderate size, we selected three mitochondrial genomes from GOBASE archives[5]. *Reclimonas americana* is thought to most resemble the ancestral mitochondrion, and other mitochondria to have evolved through deletion of segments or individual genes from the set of genes found in *Reclinomonas*. *Allomyces macrogynus* is a fungus and *Tetrahymena pyriformis* an unrelated protist. To simplify the problem somewhat, we discarded all tRNA genes from the gene orders, since the homological correspondences among tRNAs is not always clear in mitochondria. This left 66 genes for *Reclinomonas*, 17 for *Allomyces* and 23 for *Tetrahymena*.

Reclinomonas contains 49 genes in addition to those in *Allomyces*, and these genes are scattered among 10 segments in the genome. The transformation of the *Reclinomonas* genome to the *Allomyces* one by our algorithm requires 14 reversals but only three segment deletions.

Reclinomonas contains 43 genes in addition to those in *Tetrahymena*, and these genes are scattered among 11 segments in the genome. The transformation of the *Reclinomonas* genome to the *Tetrahymena* one by our algorithm requires 19 reversals but only two segment deletions.

Allomyces and *Tetrahymena* have 12 genes in common. The five supplementary genes in *Allomyces* are each in a separate segment, and the 11 supplementary genes in *Tetrahymena* are found in 6 segments. Converting the *Tetrahymena* genome into the *Allomyces* one by our algorithm requires 9 reversals, 2 insertions et 3 deletions. Moreover, the construction of the graph $\mathcal{G}(G, \overline{H})$ reveals that this number of operations is minimal.

7 Conclusion

The approach taken in this paper could readily be extended to a distance defined on multichromosomal genomes, where the rearrangement operations include reciprocal translocation, chromosome fusion and chromosome fission as well as reversal, insertion and deletion. The Hannenhalli and Pevzner approach to translocation and reversal distance [3] makes use of the same kind of cycle graph that we exploit in the present research.

Acknowledgments: Research supported in part by a grant from the Natural Science and Engineering Research Council of Canada. The author is a Scholar in the Evolutionary Biology Program of the Canadian Institute for Advanced Research.

References

1. Berman, P. and Hannenhalli, S.: Fast sorting by reversal. In Hirschberg, D. and Myers, G. (eds), *Combinatorial Pattern Matching. 7th Annual Symposium. Lecture Notes in Computer Science* 1075. Springer, New York, pp. 168-185.
2. Hannenhalli, S., Pevzner, P.A.: Transforming cabbage into turnip. (polynomial algorithm for sorting signed permutations by reversals). In: Proceedings of the 27th Annual ACM-SIAM Symposium on the Theory of Computing (1995) 178–189.
3. Hannenhalli, S., Pevzner, P.A.: Transforming men into mice (polynomial algorithm for genomic distance problem). In: Proceedings of the IEEE 36th Annual Symposium on Foundations of Computer Science (1995) 581–592.
4. Kaplan, H., Shamir R., Tarjan R.E.: Faster and simpler algorithm for sorting signed permutations by reversals. In: Proceedings of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms. ACM, New York, pp 344-351.
5. Korab-Laskowska M, Rioux P, Brossard N, Littlejohn TG, Gray MW, Lang BF and Burger G (1998). The Organelle Genome Database Project (GOBASE). *Nucleic Acids Res* 26: 139-146