Computational Biology
Lecture 16: Genome rearrangements, sorting by reversals
Saad Mneimneh

When comparing genomes in different species, a piece of chromosome in one species can be moved or copied to a different location in another species. Therefore, we talk about genome rearragements.

So far we used alignment algorithms to compare two sequences, possibly coming from different species, but alignments do not capture all genome rearragements. An alignment of two sequences determines a series of shifts (in one direction) of a sequence to bring it close to the other sequence. For instance, the following rearragement cannot be captured by an alignment:

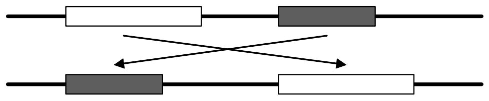


Figure 1: Rearrangement

Finding all the "good" local alignments of two sequences could capture the notion of rearragements (we will see later how to find all maximal matches between two sequences); however, these local alignments do not explain the mechanism by which one genome is transformed into the other.

Biologically speaking, a genome is obtained from another by a number of a special kind of rearrangements called *reversals*. A reversal consists of reversing a piece of the genome. After a number of reversals, the genome is transformed into another one. Since nature chooses always the easy path, we seek the minimum number of reversals that transform one genome into the other. Therefore, the distance between two genomes will be the minimum number of reversals needed to transform one into the other.

Here's an example of genome rearrangement of two related plants, alfafa and garden pea:

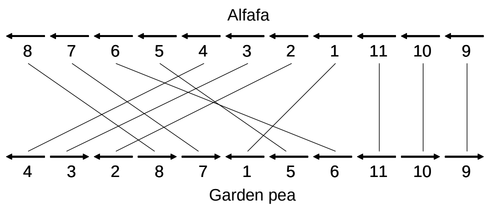


Figure 2: Rearrangements of alfafa and pea

In the rearrangement above, each block, possibly containing more than one gene, is transcribed as a unit. Blocks with the same number are homologous. The orientation of a block denotes the strand where transcription occurs. Homologous blocks in different species might have different orientations because of reversals. A reversal occurs on a number of contiguous blocks and results in reversing their order and orientation. Below we show the minimum number of reversals needed to transform alfafa into pea (a box indicated where the reversal occurs):

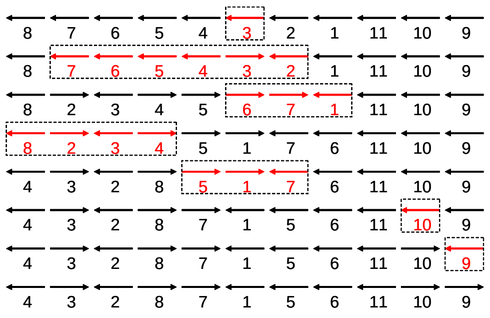


Figure 3: Transforming alfafa into pea

Formalism: signed permutations

A signed permutation α over the set $L = 1, 2, ..., n$ is a permutation of L such that $\alpha(i) = +a$ or $-a$, where $a \in L$.

For example, $\alpha = (+3, -2, -1, +4, -5)$ is a signed permutation over $L = \{1, 2, 3, 4, 5\}$.

A reversal $\rho = [i, j]$, $i \leq j$, on a signed permutation α is a signed permutation α' such that:

$$\alpha' = \alpha[i, j] = (\alpha(1), ..., \alpha(i-1), -\alpha(j), ..., -\alpha(i), \alpha(j+1), ..., \alpha(n))$$

Note how the definition of signed permutations and reversals capture the notion of a reversal of contiguous blocks in a genome. The problem that we are trying to solve here is:

Sorting by Reversals: Given two signed permutations α and β, find the minimum number of reversals $\rho_1, ..., \rho_t$ that will transform α into β, i.e.

$$\alpha\rho_1\rho_2...\rho_t = \beta$$

We define the reversal distance $d_\beta(\alpha) = t$. (note that $d_\beta(\alpha) = d_\alpha(\beta)$). We will ommit the subscript β when implicit or not particularly important.

The reason we call the problem sorting by reversals is that β can be the identity permutation without any loss of generality, i.e. $\beta = (+1, +2, ..., +n)$ (it is just a matter of labeling blocks in increasing order and assuming one orientation for all blocks in the target genomes represented as β).

Now for the problem above to make sense, it should be possible to transform α into β by using reversals only. So it that always possible? The answer is affirmative. Consider the following algorithm: Find j such that $\alpha(j) = \beta(1)$. Apply a reversal $[1, j]$ on α. Now $|\alpha(1)| = |\beta(1)|$. If $\alpha(1) = -\beta(1)$, apply a reversal $[1, 1]$. Now $\alpha(1) = \beta(1)$. Consider the signed permutations $\alpha' = (\alpha(2), ..., \alpha(n))$ and $\beta' = (\beta(2), ..., \beta(n))$ and proceed in the same way. Eventually we will transform α into β.

Our approach to develop an algorithm for transforming α into β using the minimum number of reversals is the following:

We first find a lower bound on $d_\beta(\alpha)$. The first bound we find will not be tight. We then refine the bound to obtain a new bound that is very tight.

Next, we determine a set of *safe* reversals. A safe reversal ρ has the property that $d_\beta(\alpha\rho) < d_\beta(\alpha)$. In other words, by applying a safe reversal, we decrease $d_\beta(\alpha)$ and bring α closer to β.

Finally, we develop an algorithm that applies only safe reversals until α is transformed into β. We show that an algorithm that applies only safe reversals cannot do that for more than a number of times equal to our lower bound, hence proving that our algorithm uses a minimum number of reversals.

Breakpoints: a first lower bound

Let α be a signed permutation. We define the extended version of α as the following signed permutation:

$$(\alpha(0), \alpha(1), ..., \alpha(n), \alpha(n+1)) = (0, \alpha(1), ..., \alpha(n), n+1)$$

Breakpoint: If (x, y) appear in (extended) α but neither (x, y) nor $(-y, -x)$ appear in (extended) β, then (x, y) is a breakpoint of α with respect to β.

Example: $\alpha = (-2, -3, +1, +6, -5, -4)$ and $\beta = (+1, +2, +3, +4, +5, +6)$. Then the extended versions of α and β are respectively $(0, -2, -3, +1, +6, -5, -4, +7)$ and $\beta = (0, +1, +2, +3, +4, +5, +6, +7)$. The breakpoints of α with respect to β are: $(0, -2)$, $(-2, -3)$, $(-3, +1)$, $(+1, +6)$, $(+6, -5)$, $(-4, +7)$. Note that $(-5, -4)$ is not a breakpoint since $(4, 5)$ appears in β.

Why breakpoints are important? Intuitively, if (x, y) is a breakpoint, then in order to transform α into β, some reversal must separate between x and y. Therefore, the number of breakpoints is an indication to how many reversals are required.

Let $b_\beta(\alpha)$ be the number of breakpoints of α with respect to β. Note that $b_\beta(\beta) = 0$. We will ommit the subscript β when implicit or not particularly important.

A simple observation is that a reversal $\rho = [i, j]$ can reduce the number of breakpoints by at most two since it separates between $\alpha(i-1)$ and $\alpha(i))$ and between $\alpha(j)$ and $\alpha(j+1))$. This simple observation can be expressed as follows: $b(\alpha) - b(\alpha\rho) \leq 2$. Note that a reversal ρ that decreases the number of breakpoints by two is not necessarily a one that makes progress, i.e. it is possible that $d(\alpha\rho) \neq d(\alpha) - 1$.

Let ρ_1, ..., ρ_t be such that $\alpha\rho_1 ... \rho_t = \beta$. Then,

$$\begin{array}{ll} b(\alpha) - b(\alpha\rho_1) & \leq 2 \\ b(\alpha\rho_1) - b(\alpha\rho_1\rho_2) & \leq 2 \\ \vdots & \\ b(\alpha\rho_1 ... \rho_{t-1}) - b(\alpha\rho_1 ... \rho_t) & \leq 2 \end{array}$$

This implies that $b(\alpha) - b(\alpha\rho_1 ... \rho_t) \leq 2t$. But $b(\alpha\rho_1 ... \rho_t) = b(\beta) = 0$. Therefore $b(\alpha) \leq 2t$. But $d(\alpha) \geq t$; therefore, $d(\alpha) \geq \frac{b(\alpha)}{2}$. This lower bound is not very tight. We will refine it.

Reality and Desire Diagram: a better lower bound

We repreresnt each element of the signed permutation as a tuple: $+a$ is represented as $(-a+a)$ and $-a$ is represented as $(+a-a)$. We will also consider then extended version of a permutation, i.e. having 0 and $n+1$ at the left and right extermities respectively. For example, the signed permutation $\alpha = (+3, -2, -1, +4, -5)$ is represented as a sequence of adjacent tuples as follows.

$$\alpha = 0\ (-3+3)\ (-2+2)\ (+1-1)\ (-4+4)\ (+5-5)\ 6$$

We represent α by a graph as follows:

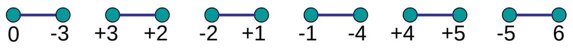


Figure 4: α as a graph

Similarly, $\beta = (+1, +2, +3, +4, +5)$ is represented as the following graph:

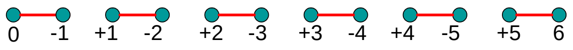


Figure 5: β as a graph

The advantage of this redundant representation of the elements of a permutation is that now we can represent α and β on the same (multi) graph as follows:

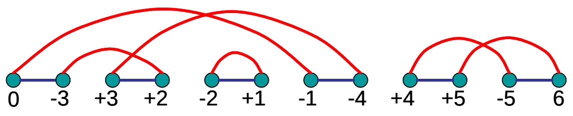


Figure 6: α and β as one graph

We call the above "Reality and Desire Diagtam" of α with respect to β and we denote it by $RD_\beta(\alpha)$. Again, we will ommit the subscript β when implicit or not particularly important. The reason we call the above graph a "diagram" is because a graph can be drawn in an arbitrary way, but we will always draw $RD(\alpha)$ in a special. Vertices are on the line and their order is the same order as the elements of α. α is represented by horizontal edges (on the line) and they will be called reality edges. β is represented by the top edges (always above the line) and they will be called desire edges. An alternative is to close the two end points of the line (0 and $n+1$) together making a circle and putting the reality edges on the circumference of the circle and the desire edges inside the circle:

Let us look at the properties of $RD(\alpha)$ as a graph:

- (1) each vertex has one reality edge and one desire edge that are incident to it
- (1) => (2) the connected components are alternating cycles (edges alternate between reality and desire)

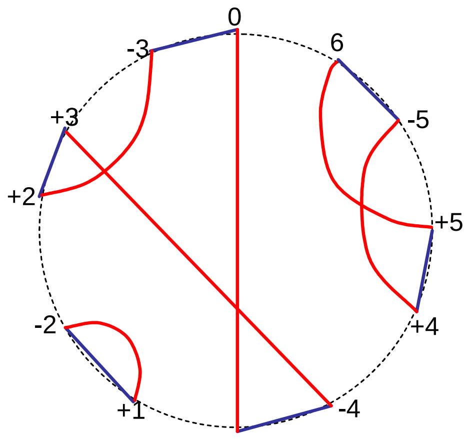


Figure 7: Reality and Desire diagram $RD_\beta(\alpha)$

- (2) => (3) the maximum number of components (cycles) is $n+1$ (each cycle in that case will have a length of 2).
- (4) each cycle of length 2 corresponds to a non-breakpoint
- (4) => (5) if $c_\beta(\alpha)$ is the number of cycles in $RD_\beta(\alpha)$, then $c_\beta(\beta) = n+1$ and β is the only permutation with that property

So in a way, we can see that the process of transforming α into β as a process of transforming $RD_\beta(\alpha)$ into a graph with as many cycles as possible (i.e. $n+1$). That does not necessarily mean that every reversal that increases the number of cycles is good. However, a very natural question now is how does a reversal effect the cycles in $RD(\alpha)$?

Let us look first at how a reversal affects $RD(\alpha)$. A reversal is characterized by the two points were it cuts the permutatation, each defined by a reality edge.

Let ρ be a reversal defined by two reality edges (s,t) and (u,v), the $RD(\alpha\rho)$ differs from $RD(\alpha)$ as follows:

- Reality edges (s,t) and (u,v) are replaced by (s,u) and (t,v)
- Desire edges remain unchanged
- Vertices $u, ..., t$ are reversed

The following figure illustrates the effect of a reversal on $RD(\alpha)$.

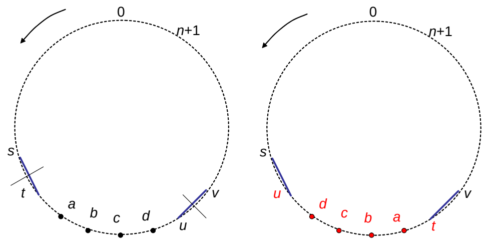


Figure 8: Effect of reversal on $RD_\beta(\alpha)$

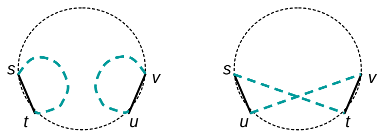
The following result tells us how the number of cycles is affected by a reversal. We need an important definition before stating the result. Two reality edges on the same cycle converge iff they are traversed in the same direction (clockwise or counterclockwise) on the cycle. For example, in Figure 7, edges $(+3,+2)$ and $(-1,-4)$ converge, and edges $(0,-3)$ and $(+3,+2)$ diverge.

Let ρ be a reversal defined by two reality edges e and f, then:

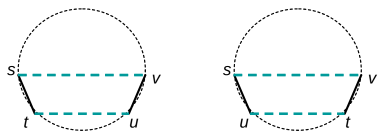
- If e and f belong to different cycles, $c(\alpha\rho) = c(\alpha) - 1$
- If e and f belong to the same cycle and converge, $c(\alpha\rho) = c(\alpha)$

- If e and f belong to the same cycle and diverge, $c(\alpha\rho) = c(\alpha) + 1$

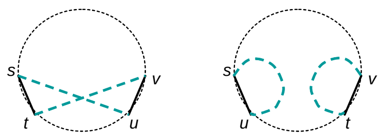
Proof: Illustrated below for each case:



case I



case II



case III

Therefore, $c(\alpha\rho) - c(\alpha) \leq 1$ for any reversal ρ. This gives us another lower bound:

Let ρ_1, ..., ρ_t be such that $\alpha\rho_1 ... \rho_t = \beta$. Then,

$$\begin{array}{ll} c(\alpha\rho_1) - c(\alpha) & \leq 1 \\ c(\alpha\rho_1\rho_2) - c(\alpha\rho_1) & \leq 1 \\ \vdots & \\ c(\alpha\rho_1 ... \rho_t) - c(\alpha\rho_1 ... \rho_{t-1}) & \leq 1 \end{array}$$

This implies that $c(\alpha\rho_1 ... \rho_t) - c(\alpha) \leq t$. But $c(\alpha\rho_1 ... \rho_t) = c(\beta) = n + 1$. Therefore $n + 1 - c(\alpha) \leq t$. But $d(\alpha) \geq t$; therefore, $d(\alpha) \geq n + 1 - c(\alpha)$. This lower bound is very good, unlike the one previously derived. For most signed permutations, it comes very close to the actual distance. The reason it does not always work will be explained below.

Components in $RD_\beta(\alpha)$

We will explain how a number of cycles in $RD(\alpha)$ can be put together to form a component. We will define several properties of components and obtain a very tight lower bound based on these properties.

First we define some simple properties of cycles. A cycle is *good* iff it has two reality edges that diverge. A cycle is *bad* iff all its reality edges converge

A *proper* cycle is a cycle with at least 4 edges. Note that we need not worry about non-proper cycles (2 edges only), they represent the non-breakpoints (reality = desire) and should remain unchanged.

Assuming the special way of drawing $RD(\alpha)$, two cycle *interleave* iff a desire edge of one crosses a desire edge of another. For instance, in the figure below, A and E interleave, B and C interleave, C and D interleave, and B and D interleave.

We define components as follows: construct the interleaving graph where vertices are cycles, and two vertices are connected by an edge iff their two corresponding cycles interleave. The connected components of the interleaving graph (each an interleaving set of cycles) are the components of $RD(\alpha)$. For instance, in the figure below, the components are $\{F\}$, $\{A, E\}$, and $\{B, C, D\}$.

A component is *good* iff it contains at least one good cycle. Otherwise, the component is *bad*. For instnace, in the figure below, $\{F\}$ is a good component because the cycle F is good, $\{A, E\}$ is a bad component because neither A nor E is a good cycle, and finally, $\{B, C, D\}$ is a good component because the cycle C is good.

Why do we have this characterization of good and bad components? Well, consider a bad component. By definition it does not contain a good cycle. Therefore, it does not contain a cycle that has two divergent edges. As a result, any reversal defined by two reality edges in that component does not increase the number of cycles, and hence does not make progress towards the desired permutation β. Moreover, by the definition of a component, any reversal defined by two reality edges in one component cannot affect another component (non of the cycles of the first interleave with any cycle of the second).

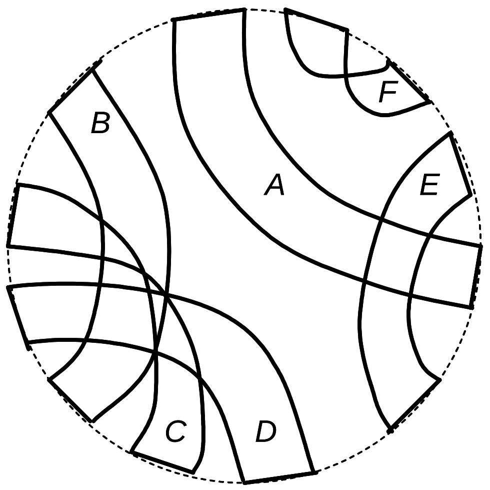


Figure 9: Interleaving and components

Therefore, a bad component will remain until

- a reversal defined by two reality edges in that component is performed, or
- a reversal defined by two reality edges in different components (i.e. in different cycles) is performed

In either cases, the reversal does not increase the number of cycles and hence does not make progress, and the lower bound $n + 1 - c(\alpha)$ will not be achieved.

However, not all bad components will necessarily contribute to a "bad" reversal as argued above. The reason being that a bad component can be "twisted" by a reversal and transformed into a good component. For instance, if a bad component B separates two other bad components A and C, and a reversal ρ defined by two edges $e \in A$ and $f \in C$ is performed (it decreases the number of cycles by 1), ρ will reverse the order of some vertices in B causing some bad cycles of B to become good and hence making B a good component (by the same argument a good component can be turned into a bad one). The two extra reversals ρ and the one needed to increase the number of cycles back to where it was, can be associated with A and C. B did not contribute to any bad reversal. Below we present a characterization of the bad components that really contribute in exceeding the lower bound $n + 1 - c(\alpha)$.

We divide bad components into *hurdles* and non hurdles. We further divide hurdles into super hurdles and simple hurdles.

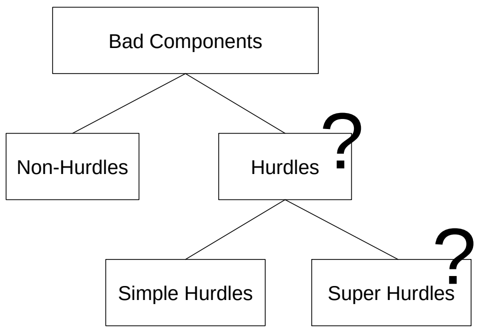


Figure 10: Classification of bad components

As argued above, separation is an important concept. We say that a component B separates components A and C if all chords in $RD(\alpha)$ that link a vertex in A to a vetex in C cross a desire edge of B (assuming the special way of drawing $RD(\alpha)$).

A argued abovem a reversal defined by two edges in different components A and C will result in every component B that separate A and C being twisted. A bad component becomes good when twisted, but a good component can remain good or become bad upon twisting. Therefore, we will resort to twisting only when there are no good components around.

A bad component is a *hurdle* iff it does not separate any two other bad components. For instance, in the figure below, components A, C, D, and F are hurdles.

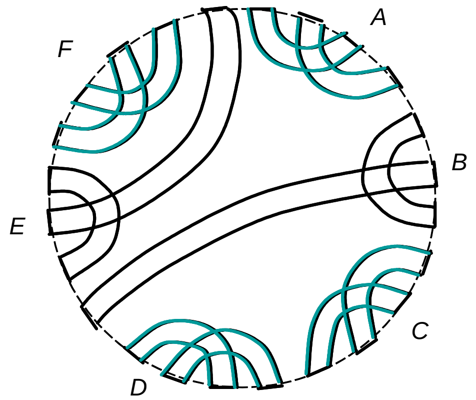


Figure 11: Hurdles

A super hurdle is a hurdle A such that its removal will cause a non-hurdle B to become a hurdle, we say A protects B. Otherwise, a hurdle is just a simple hurdle. For instance, in the figure above, component F is a super hurdle because it protects E. All other hurdles are simple hurdles.

Finally, a signed permutation α is called a fortress iff $RD(\alpha)$ has an odd number of hurdles and all of them are super hurdles. The smallest fortress has three super hurdles as illustrated below (note $RD(\alpha)$ has to have at least 48 vertices, i.e. $n + 1 = 48/2$ and α has at least 23 elements).

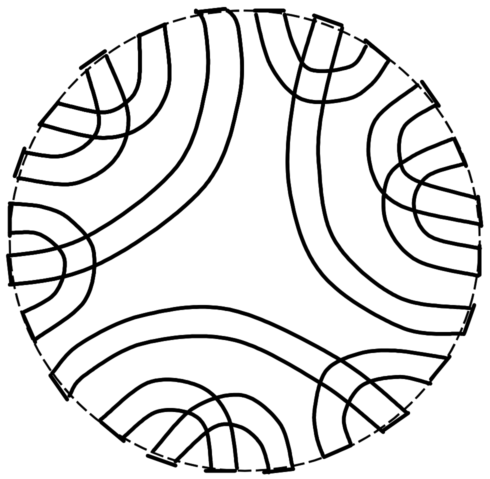


Figure 12: Fortress

We will see later that $d_\beta(\alpha) = n + 1 - c_\beta(\alpha) + h_\beta(\alpha)$ if α is not a fortress, where $h_\beta(\alpha)$ is the number of hurdles. If α is a fortress, then $d_\beta(\alpha) = n + 1 - c_\beta(\alpha) + h_\beta(\alpha) + 1$.

References

Setubal J., Meidanis J., Introduction to Computational Molecular Biology, Chapter 7.
Pevzner P., Computational Molecular Biology, Chapter 10.