UNIVERSITY OF CALIFORNIA, SAN DIEGO

DCJ-Indel Sorting

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Mathematics

by

Phillip E. C. Compeau III

Committee in charge:

Pavel A. Pevzner, Chair
Professor Vineet Bafna
Professor Guershon Harel
Professor Sergei Kosakovsky Pond
Professor Glenn Tesler

2014

The dissertation of Phillip E. C. Compeau III is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

_____

_____

_____

_____

_____

Chair

University of California, San Diego

2014

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

Thank you to Pavel Pevzner for serving as my advisor and to the other members of my thesis committee for helpful comments.

VITA

2008	B. S. *magna cum laude* with High Honors in Mathematics, Davidson College
2009	Master of Advanced Study in Mathematics, Cambridge University
2010	M. A. in Pure Mathematics, University of California, San Diego
2009-2014	Graduate Teaching Assistant, Research Assistant, Associate Instructor, University of California, San Diego
2014	Ph. D. in Mathematics, University of California, San Diego

PUBLICATIONS

Phillip E. C. Compeau, "A Generalized View of DCJ-Indel Sorting", *Lecture Notes in Computer Science*, 2014.

Phillip Compeau and Pavel Pevzner, *Bioinformatics Algorithms: An Active Learning Approach*, Active Learning Publishers, 2014.

Phillip E. C. Compeau, "DCJ-Indel Sorting Revisited", *Algorithms for Molecular Biology*, 8, 2013.

Phillip E. C. Compeau, "A Simplified View of DCJ-Indel Distance, *Lecture Notes in Computer Science*, 7534, 2012.

Phillip E. C. Compeau, Pavel A. Pevzner, and Glenn Tesler, "How to apply de Bruijn graphs to genome assembly", *Nature Biotechnology*, 29, 2011.

Phillip Compeau and Pavel A. Pevzner, "Genome Reconstruction: A Puzzle with a Billion Pieces", *Bioinformatics for Biologists*, Cambridge University Press, 2011.

Phillip Compeau, "Girth of Pancake Graphs", *Discrete Applied Mathematics*, 159, 2011.

Phillip Compeau, "Cycles in Pancake Graphs". Undergraduate Honors Thesis, Davidson College. 2008.

ABSTRACT OF THE DISSERTATION

DCJ-Indel Sorting

by

Phillip E. C. Compeau III

Doctor of Philosophy in Mathematics

University of California, San Diego, 2014

Pavel A. Pevzner, Chair

Fill in abstract later.

Chapter 1

Are There Fragile Regions in the Human Genome?

1.1 Of Mice and Men

> *"I have further been told," said the cat, "that you can also transform yourself into the smallest of animals, for example, a rat or a mouse. But I can scarcely believe that. I must admit to you that I think it would be quite impossible."*
>
> *"Impossible!" cried the ogre. "You shall see!"*
>
> *He immediately changed himself into a mouse and began to run about the floor. As soon as the cat saw this, he fell upon him and ate him up.*

1.1.1 How different are the human and mouse genomes?

When Charles Perrault described the transformation of an ogre into a mouse in "Puss in Boots", he could hardly have anticipated that three centuries later, research would show that the human and mouse genomes are surprisingly similar. Nearly every human gene has a mouse counterpart, although mice greatly outperform us when it comes to the olfactory genes responsible for smell. We are essentially mice without tails — we even have the genes needed to make a tail, but these genes have been "silenced" during our evolution. We started with a fairy tale question: "How can an ogre transform into a mouse?" Since we share most of the same genes with mice, we now ask a question about

mammalian evolution: "What evolutionary forces have transformed the genome of the human-mouse ancestor into the present-day human and mouse genomes?"

If a precocious child had grown out of reading fairy tales and wanted to learn about how the human and mouse genomes differ, then here is what we would tell her. You can cut the 23 human chromosomes into 280 pieces, shuffle these DNA fragments, and then glue the pieces together in a new order to form the 20 mouse chromosomes. The truth, however, is that evolution has not employed a single dramatic cut-and-paste operation; instead, it applies smaller changes known as **genome rearrangements**, which will be our focus in this chapter.

Unfortunately, our bioinformatics time machine won't take us more than a few centuries into the past. If it did, we could travel 75 million years back in time, watching humans slowly change into a small, furry animal that lived with dinosaurs. Then, we could travel back to the present, watching how this animal evolved into the mouse. In this chapter, we hope to understand the genome rearrangements that have separated the human and mouse genomes without having to revamp our time machine.

1.1.2 Synteny blocks

To simplify genome comparison, we will first focus on the X chromosome, which is one of the two sex-determining chromosomes in mammals and has retained nearly all its genes throughout mammalian evolution (see **DETOUR: Why is the Gene Content of Mammalian X Chromosomes So Conserved?**). PAGE 53 We can therefore view the X chromosome as a "mini-genome" when comparing mice to humans, since this chromosome's genes have not jumped around onto different chromosomes (and vice-versa). Figure 1.1 illustrates that the mouse and human X chromosomes can be divided into only 11 segments that are arranged differently in the two species.

Figure 1.1 offers a compact representation of how the human and mouse X chromosomes differ, but what does it really mean? It turns out not only that most

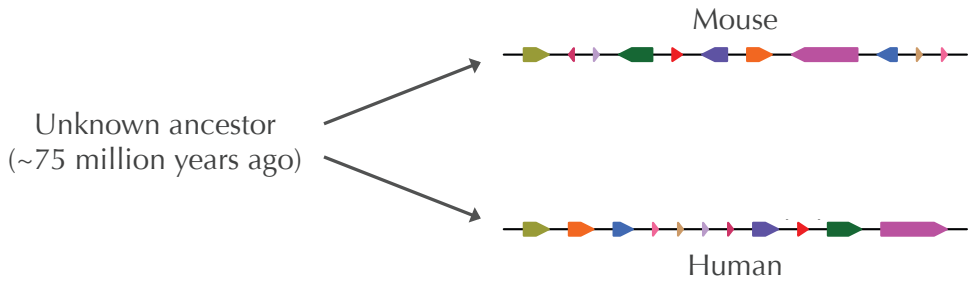


FIGURE 1.1: Mouse and human X chromosomes represented as 11 colored, directed segments (synteny blocks).

human genes have mouse counterparts, but also that hundreds of similar genes often line up one after another in the same order in the two species genomes. Each of the 11 colored segments in Figure 1.1 represents such a procession of similar genes and is called a **synteny block**. Later, we will explain how to construct synteny blocks and what the left and right **directions** of the blocks signify.

Synteny blocks simplify the comparison of the mouse and human X chromosomes from 150 million base pairs to only 11 units. This simplification is analogous to comparing two similar photographs. If we compare the images one pixel at a time, we may be overwhelmed by the scale of the problem; instead, we need to zoom out in order to notice higher-level patterns. It is no accident that biologists use the term "resolution" to discuss the level at which genomes are analyzed.

1.1.3 Reversals

You have probably been wondering how the genome changes when it undergoes a genome rearrangement. Genome rearrangements were discovered 90 years ago when Alfred Sturtevant was studying fruit fly mutants with scarlet- and peach-colored eyes as well as abnormally shaped deltoid wings. Sturtevant determined the genes coding for these traits, called **scarlet**, **peach**, and **delta**, and he was amazed to find that the arrangement of these genes in *Drosophila melanogaster* (**scarlet**, **peach**, **delta**) differed from their arrangement in *Drosophila*

simulans (**scarlet**, **delta**, **peach**). He immediately conjectured that the chromosomal segment containing **peach** and **delta** must have been flipped around (see **DETOUR: Discovery of Genome Rearrangements**). Sturtevant had witnessed the most common form of genome rearrangement, called a **reversal**, which flips around an interval of a chromosome and inverts the directions of any synteny blocks within the interval.

PAGE 53

Figure 1.2 shows a series of 7 reversals transforming the mouse X chromosome into the human X chromosome. If this scenario is correct, then the X chromosome of the ancestor of humans and mice must be represented by one of the intermediate synteny block orderings. Unfortunately, this series of 7 reversals offers only one of 1070 different 7-step scenarios transforming the mouse X chromosome into the human X chromosome. We have no clue which scenario is correct, or even whether the correct scenario had exactly 7 reversals.

STOP and Think: Can you convert the mouse X chromosome into the human X chromosome using only 6 reversals?

Regardless of how many reversals separate the human and mouse X chromosomes, you can see that reversals must be rare genomic events. Indeed, genome rearrangements typically cause the death or sterility of the mutated organism, thus preventing it from passing the rearrangement on to the next generation. However, a tiny fraction of genome rearrangements may have a positive effect on survival and propagate through a species as the result of natural selection. When a population becomes isolated from the rest of its species for long enough, the work of rearrangements can even create a new species.

1.1.4 Rearrangement hotspots

Geology provides a thought-provoking analogy for thinking about genome evolution. You might like to think of genome rearrangements as "genomic earthquakes" that dramatically change the chromosomal architecture of an organism. Genome rearrangements contrast with much more frequent point mutations,

which work slowly and are analogous to "genomic erosion".

You can visualize a reversal as breaking the genome on both sides of a chromosomal interval, flipping the interval, and then gluing the resulting segments in a new order. Keeping in mind that earthquakes occur more frequently in specific locations on Earth, we wonder if a similar principle holds for reversals — are they happening over and over again in specific genomic regions? A fundamental question in chromosome evolution studies is whether the **breakpoints** of reversals (i.e., the the ends of the inverted intervals) occur along "fault lines" called **rearrangement hotspots**. If such hotspots exist in the human genome, we want to locate them and determine how they might relate to genetic disorders, which are often attributable to rearrangements.

Of course, we should rigorously define what we mean by a "rearrangement hotspot". Re-examining the 7-reversal scenario changing the mouse X chromosome into the human X chromosome in Figure 1.2, we record the endpoints of each reversal using vertical segments. Regions affected by multiple reversals are indicated by multiple vertical segments in the human X chromosome. For example, the region adjacent to the pointed side of block 3 in Figure 1.2 is used as an endpoint of both the 4th and 5th reversals. As a result, we have placed two vertical lines between blocks 3 and 4 in the human X chromosome. However, just because we showed two breakpoints in this region does not imply that this region is a rearrangement hotspot, since the reversals in Figure 1.2 represent just one possible evolutionary scenario. Because the true rearrangement scenario is unknown, it is not immediately clear how we could determine whether rearrangement hotspots exist.

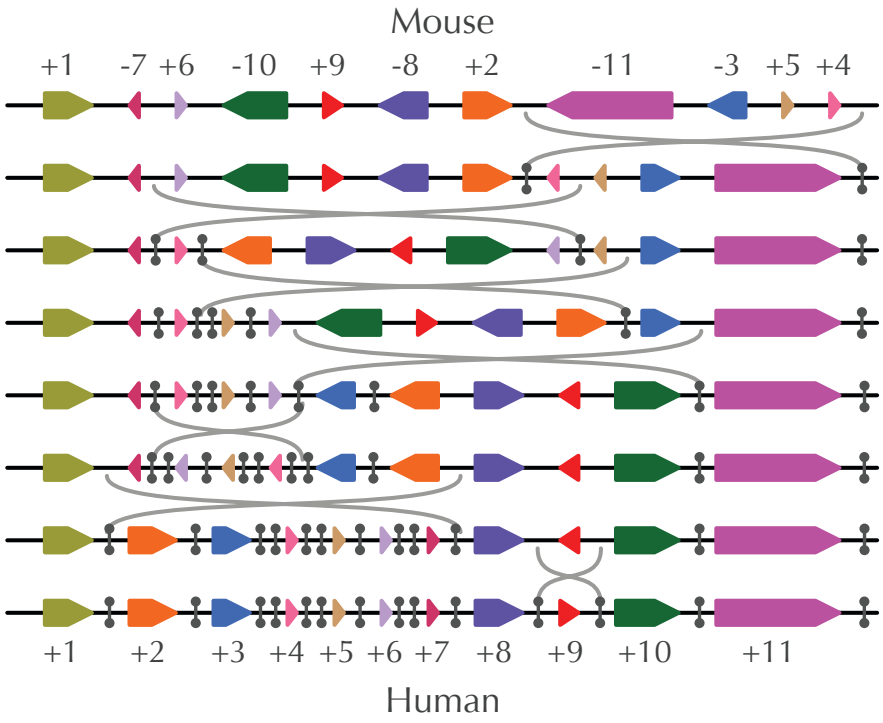


FIGURE 1.2: Transforming the mouse X chromosome into the human X chromosome with 7 reversals. Each synteny block is uniquely colored and labeled with an integer between 1 and 11; the positive or negative sign of each integer indicates the synteny block's direction (pointing right or left, respectively). Two short vertical segments delineate the endpoints of the inverted interval in each reversal. Suppose that this evolutionary scenario is correct and that, say, the 5th synteny block arrangement from the top presents the true ancestral arrangement. Then the first 4 reversals happened on the evolutionary path from mice to the human-mouse common ancestor (traveling backward in time), and the final 3 reversals happened on the evolutionary path from the common ancestor to humans (traveling forward in time). In this chapter, we are not interested in reconstructing the ancestral genome and thus are not concerned with whether a certain reversal travels backward or forward in time.

1.2 The Random Breakage Model of Chromosome Evolution

In 1973, Susumu Ohno proposed the **Random Breakage Model** of chromosome evolution. This hypothesis states that the breakpoints of rearrangements are selected randomly, implying that rearrangement hotspots in mammalian genomes do not exist. Yet this model lacked supporting evidence when it was introduced. After all, how could we possibly determine whether rearrangement hotspots exist without knowing the exact sequence of rearrangements separating two species?

STOP and Think: Consider the following questions.

1. Say that a series of random reversals result in one huge synteny block covering 90% of the genome in addition to 99 tiny synteny blocks covering the remaining 10% of the genome. Should we be surprised?
2. What if random reversals result in 100 synteny blocks of roughly the same length? Should we be surprised?

The idea that we wish to impress on you in the preceding questions is that we can test the Random Breakage Model by analyzing the distribution of synteny block lengths. For example, the lengths of the human-mouse synteny blocks on the X chromosome vary widely, with the largest block (block 11 in Figure 1.2) taking up nearly 25% of the entire length of the X chromosome. Is this variation in synteny block length consistent with the Random Breakage Model?

In 1984, Joseph Nadeau and Benjamin Taylor asked what the expected lengths of synteny blocks should be after N reversals occurring at random locations in the genome. If we rule out the unlikely event that two random reversals cut the chromosome in exactly the same position, then N random reversals cut the chromosome in $2N$ locations and produce $2N + 1$ synteny blocks. Figure 1.3 (top) depicts the result of a computational experiment in

which 320 random reversals are applied to a simulated chromosome consisting of 25,000 genes, producing $2 \cdot 320 + 1 = 641$ synteny blocks. The average synteny block size is $25{,}000/641 \approx 34$ genes, but this does not mean that all synteny blocks should have approximately 34 genes. If we select random locations for breakpoints, then some blocks may have only a few genes, whereas other blocks may contain over a hundred. The point is that regardless of how many times we run this simulation, the resulting distributions of synteny block lengths will be similar. Figure 1.3 (bottom) averages the results of 100 such simulations and illustrates that the distribution of synteny block lengths can be approximated by a curve corresponding to an **exponential distribution** (see DETOUR: The Exponential Distribution). The exponential distribution predicts that we should observe about seven blocks having 34 genes and one or two much larger blocks having 100 genes.

PAGE 54

What happens when we look at the histogram for the real human and mouse synteny blocks? When Nadeau and Taylor constructed this histogram for the limited genetic data available in 1984, they observed that the lengths of blocks fit the exponential distribution well. In the 1990s, more accurate synteny block data fit the exponential distribution even better (Figure 1.4). Case closed — even though we don't know the exact rearrangements causing our genome to evolve over the last 75 million years, these rearrangements must have followed the Random Breakage Model!

STOP and Think: Do you agree with the logic behind this argument?

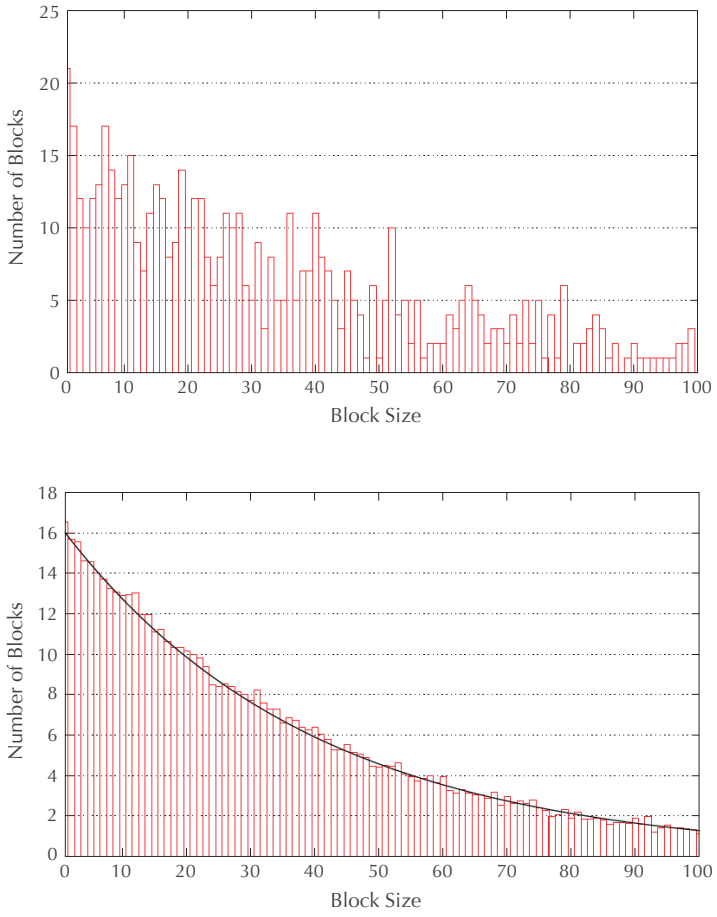


FIGURE 1.3: (Top) A histogram showing the number of blocks of each size (for a simulated genome with 25,000 genes after 320 randomly chosen reversals). Blocks having more than 100 genes are not shown. (Bottom) An average histogram of synteny block lengths for 100 simulations, fitted by the exponential distribution.

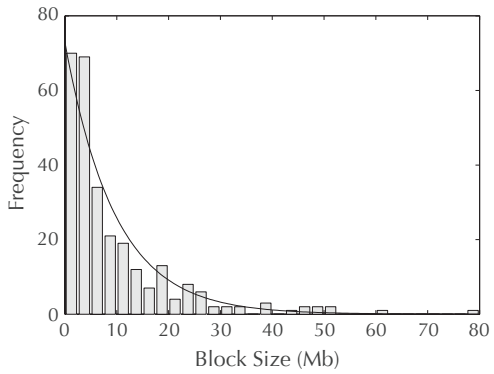


FIGURE 1.4: Histogram of human-mouse synteny block lengths (only synteny blocks longer than 1 million nucleotides are shown). The histogram is fitted by an exponential distribution.

1.3 Sorting by Reversals

We now have evidence in favor of the Random Breakage Model, but this evidence is far from conclusive. To test this model, let's start building a mathematical model for rearrangement analysis. We will therefore return to a problem that we hinted at the introduction, which is finding the minimum number of reversals that could transform the mouse X chromosome into the human X chromosome.

STOP and Think: From a biological perspective, why do you think we want to find the minimum possible number of reversals?

We ask for the minimum number of reversals in accordance with the principle of **Occam's razor**. This maxim states that when presented with some quandary, we should explain it using the simplest hypothesis that is consistent with what we already know. In this case, it seems most reasonable that evolution would take the "shortest path" connecting two species, i.e., the most **parsimonious** evolutionary scenario. Evolution may not always take the shortest path, but even when it does not, the number of steps in the true evolutionary scenario often comes close to the number of steps in the most parsimonious scenario. How, then, can we find the length of this shortest path?

Genome rearrangement studies typically ignore the lengths of synteny blocks and represent chromosomes by **signed permutations**. Each block is labeled by a number, which is assigned a positive/negative sign depending on the block's direction. The number of elements in a signed permutation is its **length**. As you can see from Figure 1.2, the human and mouse X chromosomes can be represented by the following signed permutations of length 11:

$$\textbf{Mouse: } (+1\ \ -7\ \ +6\ \ -10\ \ +9\ \ -8\ \ +2\ \ -11\ \ -3\ \ \ +5\ \ \ +4)$$
$$\textbf{Human: } (+1\ \ +2\ \ +3\ \ \ +4\ \ +5\ \ +6\ \ +7\ \ \ +8\ \ +9\ \ +10\ \ +11)$$

In the rest of the chapter, we will refer to signed permutations as **permutations** for short. Because we assume that each synteny block is unique, we do not allow

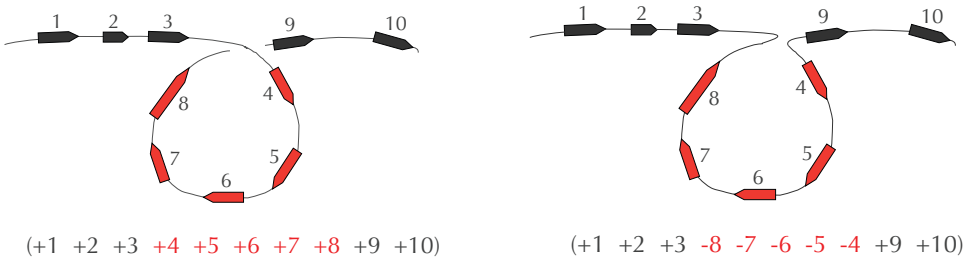


FIGURE 1.5: A cartoon illustrating how a reversal breaks a chromosome in two places and inverts the segment between the two breakpoints. Note that the reversal changes the sign of each element within the permutation's inverted segment.

repeated numbers in permutations (e.g., $(+1\ -2\ +3\ +2)$ is not a permutation).

EXERCISE BREAK: How many permutations of length n are there?

We can model reversals by inverting the elements within an interval of a permutation, then switching the signs of any elements within the inverted interval. For example, the cartoon in Figure 1.5 illustrates how a reversal changes the permutation $(+1\ +2\ +3\ \mathbf{+4\ +5\ +6\ +7\ +8}\ +9\ +10)$ into $(+1\ +2\ +3\ \mathbf{-8\ -7\ -6\ -5\ -4}\ +9\ +10)$. This reversal can be viewed as first breaking the permutation between +3 and +4 as well as between +8 and +9:

$$(+1\ +2\ +3)(\mathbf{+4\ +5\ +6\ +7\ +8})(+9\ +10)$$

It then inverts this middle segment:

$$(+1\ +2\ +3)(\mathbf{-8\ -7\ -6\ -5\ -4})(+9\ +10)$$

and finally glues the three segments back together to form a new permutation:

$$(+1\ +2\ +3\ \mathbf{-8\ -7\ -6\ -5\ -4}\ +9\ +10)$$

EXERCISE BREAK: How many different reversals can be applied to a permutation of length n?

We define the **reversal distance** between permutations P and Q, denoted $d_{\text{rev}}(P, Q)$, as the minimum number of reversals required to transform P into Q.

Reversal Distance Problem:
Calculate the reversal distance between two permutations.

Input: Two permutations of equal length.
Output: The reversal distance between these permutations.

We represented the human X chromosome by $(+1\ +2\ +3\ +4\ +5\ +6\ +7\ +8\ +9\ +10\ +11)$; this permutation, in which blocks are ordered from smallest to largest with positive directions, is called the **identity permutation**. The reason why we used the identity permutation of length 11 to represent the human X chromosome is that when comparing two genomes, we can label the synteny blocks in one of the genomes however we like. The block labeling for which the human X chromosome is the identity permutation automatically induces the representation of the mouse chromosome as $(+1\ -7\ +6\ -10\ +9\ -8\ +2\ -11\ -3\ +5\ +4)$. Of course, we could have instead encoded the mouse X chromosome as the identity permutation, which would have induced the encoding of the human X chromosome as $(+1\ +7\ -9\ +11\ +10\ +3\ -2\ -6\ +5\ -4\ -8)$ (Figure 1.6).

STOP and Think: Is the reversal distance between $(+1\ +2\ +3\ +4\ +5\ +6\ +7\ +8\ +9\ +10\ +11)$ and $(+1\ -7\ +6\ -10\ +9\ -8\ +2\ -11\ -3\ +5\ +4)$ equal to the reversal distance between $(+1\ +2\ +3\ +4\ +5\ +6\ +7\ +8\ +9\ +10\ +11)$ and $(+1\ +7\ -9\ +11\ +10\ +3\ -2\ -6\ +5\ -4\ -8)$? In other words, does the specific labeling of synteny blocks affect the reversal distance between the two chromosomes?

Because we have the freedom to label synteny blocks however we like, we will consider an offshoot of the Reversal Distance Problem in which permutation Q is the identity permutation $(+1\ +2\ \ldots +n)$. This computational problem is called **sorting by reversals**, and we denote the minimum number of reversals

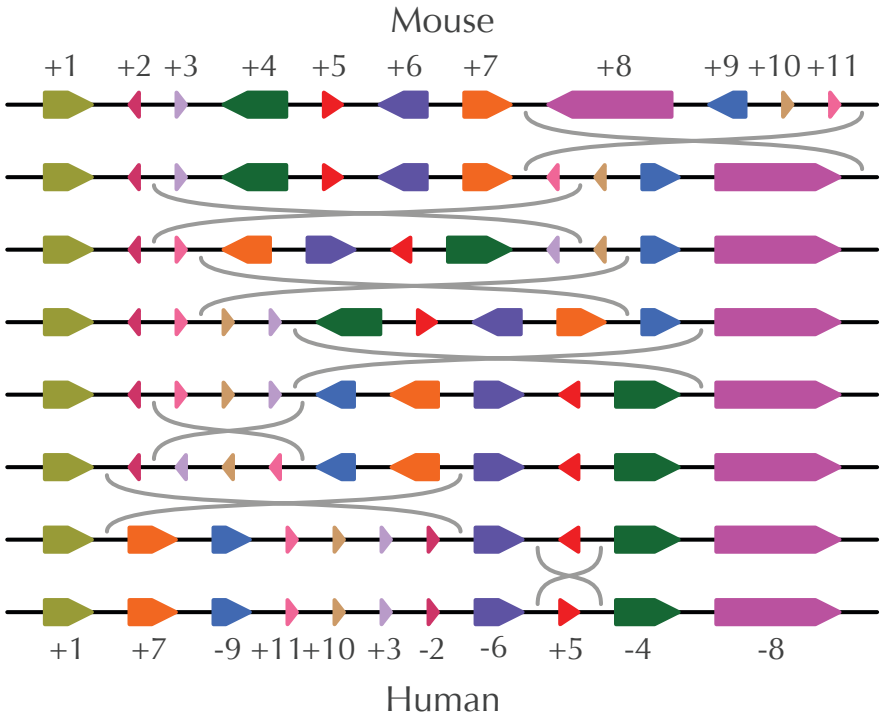


FIGURE 1.6: Encoding the mouse X chromosome as the identity permutation implies encoding the human X chromosome as $(+1\ +7\ -9\ +11\ +10\ +3\ -2\ -6\ +5\ -4\ -8)$.

required to sort P into the identity permutation as $d_{\text{rev}}(P)$. The history of sorting by reversals is founded in a culinary application and involves two celebrities (see **DETOUR: Bill Gates and David X. Cohen Flip Pancakes**).

PAGE 55

Sorting by Reversals Problem:
Compute the reversal distance between a permutation and the identity permutation.

Input: A permutation P.
Output: The reversal distance $d_{\text{rev}}(P)$.

Here is a sorting of $(+2\ -4\ -3\ +5\ -8\ -7\ -6\ +1)$ using five reversals, with the inverted interval at each step shown in red:

$$
\begin{array}{l}
(+2\ \mathbf{-4}\ \mathbf{-3}\ +5\ -8\ -7\ -6\ +1) \\
(+2\ +3\ +4\ +5\ \mathbf{-8}\ \mathbf{-7}\ \mathbf{-6}\ +1) \\
(+2\ +3\ +4\ +5\ +6\ +7\ +8\ \mathbf{+1}) \\
(\mathbf{+2}\ \mathbf{+3}\ \mathbf{+4}\ \mathbf{+5}\ \mathbf{+6}\ \mathbf{+7}\ \mathbf{+8}\ -1) \\
(\mathbf{-8}\ \mathbf{-7}\ \mathbf{-6}\ \mathbf{-5}\ \mathbf{-4}\ \mathbf{-3}\ \mathbf{-2}\ \mathbf{-1}) \\
(+1\ +2\ +3\ +4\ +5\ +6\ +7\ +8)
\end{array}
$$

STOP and Think: Can you sort this permutation using fewer reversals?

Here is a faster sorting:

$$
\begin{array}{l}
(+2\ \mathbf{-4}\ \mathbf{-3}\ +5\ -8\ -7\ -6\ +1) \\
(\mathbf{+2}\ \mathbf{+3}\ \mathbf{+4}\ \mathbf{+5}\ -8\ -7\ -6\ +1) \\
(-5\ -4\ -3\ -2\ \mathbf{-8}\ \mathbf{-7}\ \mathbf{-6}\ \mathbf{+1}) \\
(\mathbf{-5}\ \mathbf{-4}\ \mathbf{-3}\ \mathbf{-2}\ \mathbf{-1}\ +6\ +7\ +8) \\
(+1\ +2\ +3\ +4\ +5\ +6\ +7\ +8)
\end{array}
$$

STOP and Think: Consider the following questions.

1. Is it possible to sort this permutation even faster?
2. During sorting by reversals, the intermediate permutations in the example above are getting more and more "ordered". Can you come up with a quantitative measure of how ordered a permutation is?

1.4 A Greedy Algorithm for Sorting by Reversals

Let's see if we can design a greedy heuristic to approximate $d_{\text{rev}}(P)$. The simplest idea is to first fix $+1$ in the first position, then fix $+2$ in the second position, and so on. For example, element 1 is already in the correct position and has the correct sign in the mouse X chromosome, but element 2 is not in the

correct position. We can keep element 1 fixed and move element 2 to the correct position by applying a single reversal.

$$(+1\ -7\ +6\ -10\ +9\ -8\ +2\ -11\ -3\ +5\ +4)$$
$$(+1\ -2\ +8\ -9\ +10\ -6\ +7\ -11\ -3\ +5\ +4)$$

One more reversal flips element 2 around so that it has the correct sign:

$$(+1\ -2\ +8\ -9\ +10\ -6\ +7\ -11\ -3\ +5\ +4)$$
$$(+1\ +2\ +8\ -9\ +10\ -6\ +7\ -11\ -3\ +5\ +4)$$

By iterating, we can successively move larger and larger elements to their correct positions in the identity permutation by following the reversals below. The inverted interval of each reversal is still shown in red, and elements that have been placed in the correct position are shown in blue.

$$(+1\ -7\ +6\ -10\ +9\ -8\ +2\ -11\ -3\ +5\ +4)$$
$$(+1\ -2\ +8\ -9\ +10\ -6\ +7\ -11\ -3\ +5\ +4)$$
$$(+1\ +2\ +8\ -9\ +10\ -6\ +7\ -11\ -3\ +5\ +4)$$
$$(+1\ +2\ +3\ +11\ -7\ +6\ -10\ +9\ -8\ +5\ +4)$$
$$(+1\ +2\ +3\ -4\ -5\ +8\ -9\ +10\ -6\ +7\ -11)$$
$$(+1\ +2\ +3\ +4\ -5\ +8\ -9\ +10\ -6\ +7\ -11)$$
$$(+1\ +2\ +3\ +4\ +5\ +8\ -9\ +10\ -6\ +7\ -11)$$
$$(+1\ +2\ +3\ +4\ +5\ +6\ -10\ +9\ -8\ +7\ -11)$$
$$(+1\ +2\ +3\ +4\ +5\ +6\ -7\ +8\ -9\ +10\ -11)$$
$$(+1\ +2\ +3\ +4\ +5\ +6\ +7\ +8\ -9\ +10\ -11)$$
$$(+1\ +2\ +3\ +4\ +5\ +6\ +7\ +8\ +9\ +10\ -11)$$
$$(+1\ +2\ +3\ +4\ +5\ +6\ +7\ +8\ +9\ +10\ +11)$$

This example motivates a greedy heuristic called **GREEDYSORTING**. We say that element k in permutation $P = (p_1 \ldots p_n)$ is **sorted** if $p_k = +k$ and **unsorted** otherwise. For every unsorted element k that is located outside the first k positions in P, there exists a single reversal, called the ***k*-sorting reversal**, which fixes the first $k-1$ elements of P and moves element k to the k-th position. For example, in the sorting of the mouse X chromosome shown above, the 2-sorting reversal transforms $(+1\ -7\ +6\ -10\ +9\ -8\ +2\ -11\ -3\ +5\ +4)$ into $(+1\ -2$

$+8\ -9\ +10\ -6\ +7\ -11\ -3\ +5\ +4)$. In this case, one additional reversal flipping -2 around was needed to sort element 2.

We now give the pseudocode for **GREEDYSORTING**, which applies k-sorting reversals for increasing values of k. Here, $|P|$ refers to the length of permutation P.

```
GREEDYSORTING(P)
    approxReversalDistance ← 0
    for k ← 1 to |P|
        if element k is not sorted
            apply the k-sorting reversal to P
            approxReversalDistance ← approxReversalDistance + 1
            if k-th element of P is −k
                apply the reversal flipping the k-th element of P
                approxReversalDistance ← approxReversalDistance + 1
    return approxReversalDistance
```

In the case of the mouse X chromosome, **GREEDYSORTING** requires 11 reversals, but we already know that this permutation can be sorted with 7 reversals, which causes us to wonder: how good of a heuristic is **GREEDYSORTING**?

EXERCISE BREAK: What is the largest number of reversals **GREEDYSORTING** could ever require to sort a permutation of length n?

Consider the permutation $(-6\ +1\ +2\ +3\ +4\ +5)$. You can verify that the greedy heuristic requires ten steps to sort this permutation, and yet it can be sorted using just two reversals!

$$(\mathbf{-6}\ \mathbf{+1}\ \mathbf{+2}\ \mathbf{+3}\ \mathbf{+4}\ \mathbf{+5})$$
$$(\mathbf{-5}\ \mathbf{-4}\ \mathbf{-3}\ \mathbf{-2}\ \mathbf{-1}\ +6)$$
$$(+1\ +2\ +3\ +4\ +5\ +6)$$

This example demonstrates that **GREEDYSORTING** provides a poor approximation for the reversal distance.

STOP and Think: Can you find a *lower* bound on $d_{\text{rev}}(P)$? For example, can you show that the mouse permutation $(+1\ -7\ +6\ -10\ +9\ -8\ +2\ -11\ -3\ +5\ +4)$ cannot be sorted with fewer than 7 reversals?

1.5 Breakpoints

1.5.1 What are breakpoints?

Consider the sorting by reversals shown in Figure 1.7. We would like to quantify how each subsequent permutation is moving closer to the identity as we apply subsequent reversals. Consider the first reversal; at the right endpoint of the inverted interval, it changes the consecutive elements $(-11\ +13)$ into the much more desirable $(+12\ +13)$. Less obvious is the work of the fourth reversal, which places -11 immediately left of -10 so that in the next step, the consecutive elements $(-11\ -10)$ can be part of an inverted interval, creating the desirable consecutive elements $(+10\ +11)$.

	BREAKPOINTS(P)
$\mid +3\ +4\ \ +5 \mid \mathbf{-12} \mid\ \mathbf{-8}\ \ \mathbf{-7}\ \ \mathbf{-6} \mid\ \mathbf{+1}\ \ \mathbf{+2} \mid \mathbf{+10} \mid\ \mathbf{+9} \mid \mathbf{-11} \mid +13$	8
$\mid \mathbf{+3}\ \mathbf{+4}\ \ \mathbf{+5} \mid \mathbf{+11} \mid\ \mathbf{-9} \mid \mathbf{-10} \mid\ \mathbf{-2}\ \ \mathbf{-1} \mid +6\ \ +7\ \ +8 \mid +12\ +13$	7
$+1\ +2 \mid \mathbf{+10} \mid\ \mathbf{+9} \mid \mathbf{-11} \mid\ \mathbf{-5}\ \ \mathbf{-4}\ \ \mathbf{-3} \mid +6\ \ +7\ \ +8 \mid +12\ +13$	6
$+1\ +2\ \ +3\ \ +4\ \ +5 \mid \mathbf{+11} \mid\ \mathbf{-9} \mid -10 \mid +6\ \ +7\ \ +8 \mid +12\ +13$	5
$+1\ +2\ \ +3\ \ +4\ \ +5 \mid\ +9 \mid \mathbf{-11}\ \ \mathbf{-10} \mid \mathbf{+6}\ \ \mathbf{+7}\ \ \mathbf{+8} \mid +12\ +13$	4
$+1\ +2\ \ +3\ \ +4\ \ +5 \mid\ \mathbf{+9} \mid\ \mathbf{-8}\ \ \mathbf{-7}\ \ \mathbf{-6} \mid +10\ +11\ +12\ +13$	3
$+1\ +2\ \ +3\ \ +4\ \ +5\ \ +6\ \ +7\ \ +8 \mid \mathbf{-9} \mid +10\ +11\ +12\ +13$	2
$+1\ +2\ \ +3\ \ +4\ \ +5\ \ +6\ \ +7\ \ +8\ \ +9\ +10\ +11\ +12\ +13$	0

FIGURE 1.7: A sorting by reversals. The inverted interval of each reversal is shown in red, while breakpoints in each permutation are marked by vertical segments.

The intuition that we are trying to build is that consecutive elements like

$(+12\ +13)$ are desirable because they appear in the same order as in the identity permutation. However, consecutive elements like $(-11\ -10)$ are also desirable, since these elements can be later inverted into the correct order. The pairs $(+12\ +13)$ and $(-11\ -10)$ have something in common; the second element is equal to one more than the first element. We therefore say that consecutive elements $(p_i\ p_{i+1})$ in permutation $P = (p_1 \ldots p_n)$ form an **adjacency** if $p_{i+1} - p_i$ is equal to 1. By definition, for any positive element $k < n$, both $(k\ k+1)$ and $(-(k+1)\ -k)$ are adjacencies. If $p_{i+1} - p_i$ is not equal to 1, then we say that $(p_i\ p_{i+1})$ is a **breakpoint**.

We can think about a breakpoint intuitively as a pair of consecutive elements that are "out of order" compared to the identity permutation $(+1\ +2\ \ldots +n)$. For example, the pair $(+5\ -12)$ is a breakpoint because +5 and -12 are not neighbors in the identity permutation. Similarly, $(-12\ -8)$, $(-6\ +1)$, $(+2\ +10)$, $(+9\ -11)$, and $(-11\ +13)$ are clearly out of order. But $(+10\ +9)$ is also a breakpoint (even though it is formed by consecutive integers) since its signs are out of order compared to the identity permutation (and $9 - 10 \neq 1$).

We will further represent the beginning and end of permutation P by adding 0 to the left of the first element and $n+1$ to the right of the last element:

$$(\mathbf{0}\ p_1 \ldots p_n\ \mathbf{(n+1)})$$

As a result, there are $n+1$ pairs of consecutive elements:

$$(\mathbf{0}\ p_1),\ (p_1\ p_2),\ (p_2\ p_3),\ \ldots,\ (p_{n-1}\ p_n),\ (p_n\ \mathbf{(n+1)})$$

We use ADJACENCIES(P) and BREAKPOINTS(P) to denote the number of adjacencies and breakpoints of permutation P, respectively. Figure 1.7 illustrates how the number of breakpoints changes during sorting by reversals (note that 0 and $n+1$ are placeholders and cannot be affected by a reversal).

1.5.2 Counting breakpoints

Because any pair of consecutive elements of a permutation form either a breakpoint or adjacency, we have the following identity for any permutation P of length n:

$$\text{ADJACENCIES}(P) + \text{BREAKPOINTS}(P) = n + 1.$$

This formula implies that a permutation on n elements may have up to $n+1$ adjacencies.

STOP and Think: How many permutations on n elements have $n+1$ adjacencies?

You can verify that the identity permutation $(+1\ +2 \ldots +n)$ is the only permutation for which all consecutive elements are adjacencies, meaning that it has no breakpoints. Note also that the permutation $(-n\ -(n-1) \ldots -2\ -1)$ has adjacencies for every consecutive pair of elements except for the two breakpoints $(0\ -n)$ and $(-1\ (n+1))$ at the ends of the permutation.

EXERCISE BREAK: How many permutations of length n have exactly $n-1$ adjacencies?

Number of Breakpoints Problem:
Find the number of breakpoints in a permutation.

Input: A permutation P.
Output: The number of breakpoints in P.

STOP and Think: We defined a breakpoint between an arbitrary permutation and the identity permutation. Generalize the notion of a breakpoint between two arbitrary permutations, and design a linear-time algorithm for computing this number.

1.5.3 Sorting by reversals as breakpoint elimination

The reversals in Figure 1.7 reduce the number of breakpoints from 8 to 0. Note that the permutation becomes more and more "ordered" after every reversal as the number of breakpoints reduces at each step. You can therefore think of sorting by reversals as the process of breakpoint elimination — reducing the number of breakpoints in a permutation P from $\text{BREAKPOINTS}(P)$ to 0.

STOP and Think: What is the maximum number of breakpoints that can be eliminated by a single reversal?

Consider the first reversal in Figure 1.7, which reduces the number of breakpoints from 8 to 7. On either side of the inverted interval, breakpoints and adjacencies certainly do not change; for example, the breakpoint $(0\ +3)$ and the adjacency $(+13\ +14)$ remain the same. Also note that every breakpoint within the inverted interval of a reversal remains a breakpoint after the reversal. In other words, if $(p_i\ p_{i+1})$ formed a breakpoint within the span of a reversal, i.e.,

$$p_{i+1} - p_i \neq 1,$$

then these consecutive elements will remain a breakpoint after the reversal changes them into $(-p_{i+1}\ -p_i)$:

$$-p_i - (-p_{i+1}) = p_{i+1} - p_i \neq 1.$$

For example, there are five breakpoints within the span of the following reversal on the permutation $(0\ +3\ +4\ +5\ \mathbf{-12\ -8\ -7\ -6\ +1\ +2\ +10\ +9\ -11}\ +13\ +14\ 15)$:

$$\mathbf{(-12\ -8)} \qquad \mathbf{(-6\ +1)} \qquad \mathbf{(+2\ +10)} \qquad \mathbf{(+10\ +9)} \qquad \mathbf{(+9\ -11)}$$

After the reversal, these breakpoints become the following five breakpoints:

$$\mathbf{(+11\ -9)} \qquad \mathbf{(-9\ -10)} \qquad \mathbf{(-10\ -2)} \qquad \mathbf{(-1\ +6)} \qquad \mathbf{(+8\ +12)}$$

Since all breakpoints inside and outside the span of a reversal remain breakpoints after a reversal, the only breakpoints that could be eliminated by a reversal are the two breakpoints located on the boundaries of the inverted interval. The breakpoints on the boundaries of the first reversal in Figure 1.7 are $(+5\ \mathbf{-12})$ and $(\mathbf{-11}\ +13)$; the reversal converts them into a breakpoint $(+5\ +11)$ and an adjacency $(+12\ +13)$, thus reducing the number of breakpoints by 1.

STOP and Think: Can the permutation $(+3\ +4\ +5\ -12\ -8\ -7\ -6\ +1\ +2\ +10\ +9\ -11\ +13\ +14)$, which has 8 breakpoints, be sorted with 3 reversals?

A reversal can eliminate at most two breakpoints, so two reversals can eliminate at most four breakpoints, three reversals can eliminate at most six breakpoints, and so on. This reasoning establishes the following theorem.

Breakpoint Theorem: *The reversal distance $d_{rev}(P)$ is always greater than or equal to* BREAKPOINTS$(P)/2$.

It would be nice if we could *always* find a reversal that eliminates two breakpoints from a permutation, as this would imply a simple greedy algorithm for optimal sorting by reversals. Unfortunately, this is not the case; for a simple example, consider the permutation $P = (+2\ +1)$, which has three breakpoints. However, you can verify that there are no reversals reducing the number of breakpoints in P.

EXERCISE BREAK: How many permutations of length n have the property that no reversal applied to P decreases BREAKPOINTS(P)?

It turns out that every permutation of length n can be sorted using at most $n+1$ reversals and that the permutation $(+n\ +(n-1)\ \ldots +1)$ requires $n+1$ reversals to sort. Since this permutation has $n+1$ breakpoints, there is a large gap between the lower bound of $(n+1)/2$ provided by the Breakpoint Theorem and the reversal distance.

You will soon see that the idea of breakpoints will help us return to our original aim of testing the Random Breakage Model. For now, we would like to

move from permutations, which can only model single chromosomes, to a more general multichromosomal model. You may be surprised that we are moving to a seemingly more difficult model before resolving the unichromosomal case, which is already difficult. However, it turns out that our new multichromosomal model will be easier to analyze!

1.6 Rearrangements in Tumor Genomes

As we move toward a more robust model for genome comparison, we need to incorporate rearrangements that move genes from one chromosome to another. Indeed, the genes from a single human chromosome usually have their counterparts distributed over many mouse chromosomes (and vice-versa). We hope that there is a nagging voice in your head, wondering: *How can a genome rearrangement affect multiple chromosomes?*

Although multichromosomal rearrangements have occurred during species evolution over millions of years, we can witness them during a much narrower time frame in cancer cells, which exhibit many chromosomal aberrations. Some of these mutations have no direct effect on tumor development, but many types of tumors display recurrent rearrangements that trigger tumor growth by disrupting genes or altering gene regulation. By studying these rearrangements, we can identify genes that are important for tumor growth, leading to improved cancer diagnostics and therapeutics.

Figure 1.8 presents a rearrangement involving human chromosomes 9 and 22 in **chronic myeloid leukemia (CML)**. In this type of rearrangement, called a **translocation**, two intervals of DNA are excised from the end of chromosomes 9 and 22 and then reattached on opposite chromosomes. One of the rearranged chromosomes is called the **Philadelphia chromosome**. This chromosome fuses together two genes called ABL and BCR that normally have nothing to do with each other. However, when joined on the Philadelphia chromosome, these two genes create a single **chimeric gene** coding for the **ABL-BCR fusion protein**,

which has been implicated in the development of CML.

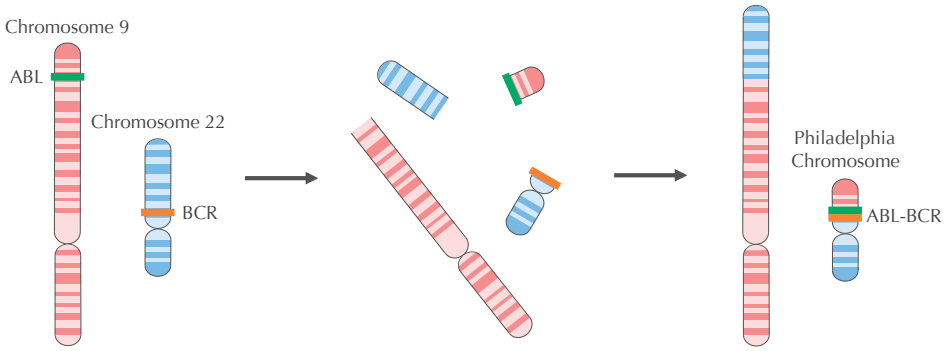


FIGURE 1.8: The Philadelphia chromosome is formed by a translocation affecting chromosomes 9 and 22. It fuses together the ABL and BCR genes, forming a chimeric gene that can trigger CML.

Once scientists understood the root cause of CML, they started searching for a compound inhibiting ABL-BCR, which resulted in the introduction of a drug called **Gleevec** in 2001. Gleevec is a **targeted therapy** against CML that inhibits cancer cells but does not affect normal cells and has shown great clinical results. However, since it targets only the ABL-BCR fusion protein, Gleevec works for CML (and very few other cancers) but does not treat most other cancers. Nevertheless, the introduction of Gleevec has bolstered researchers' hopes that the search for specific rearrangements in other types of cancer may produce additional specialized cancer therapies.

1.7 From Unichromosomal to Multichromosomal Genomes

1.7.1 Translocations, fusions, and fissions

To model translocations, we represent a multichromosomal genome with k chromosomes as a permutation that has been partitioned into k pieces. For example, the genome $(+1\ +2\ +3\ +4\ +5\ +6)(+7\ +8\ +9\ +10\ +11)$ is made up of the two chromosomes $(+1\ +2\ +3\ +4\ +5\ +6)$ and $(+7\ +8\ +9\ +10\ +11)$. A

translocation exchanges segments of different chromosomes, e.g., a translocation of (+1 +2 +3 +4 +5 +6) and (+7 +8 +9 +10 +11) may result in the chromosomes (+1 +2 +3 +4 +9 +10 +11) and (+7 +8 +5 +6). You can think about a translocation as breaking each of the two chromosomes

(+1 +2 +3 +4 +5 +6) (+7 +8 +9 +10 +11)

into two parts:

(+1 +2 +3 +4) (+5 +6) (+7 +8) (+9 +10 +11)

and then gluing the resulting segments into two new chromosomes:

(+1 +2 +3 +4+9 +10 +11) (+7 +8 +5 +6)

Rearrangements in multichromosomal genomes are not limited to reversals and translocations. They also include chromosome **fusions**, which merge two chromosomes into a single chromosome, as well as **fissions**, which break a single chromosome into two chromosomes. For example, (+1 +2 +3 +4 +5 +6) and (+7 +8 +9 +10 +11) can be fused into the single chromosome (+1 +2 +3 +4 +5 +6 +7 +8 +9 +10 +11); a subsequent fission of this chromosome could result in the two chromosomes (+1 +2 +3 +4) and (+5 +6 +7 +8 +9 +10 +11). Five million years ago, shortly after the human and chimpanzee lineages split, a fusion of two chromosomes (called 2A and 2B) in one of our ancestors created human chromosome 2 and reduced our chromosome count from 24 to 23.

STOP and Think: *A priori*, it could just as easily be the case that the human-chimpanzee ancestor had an intact chromosome 2, and that a fission split these two chromosomes into chimpanzee chromosomes 2A and 2B. How would you choose between the two scenarios? Hint: gorillas and orangutans, like chimpanzees, also have 24 chromosomes.

1.7.2 From a permutation to a graph

We will henceforth assume that all chromosomes in a genome are circular. This assumption represents a slight distortion of biological reality, as mammalian chromosomes are linear. However, circularizing a linear chromosome by joining its endpoints will simplify the subsequent analysis without affecting our conclusions.

We now have a multichromosomal genomic model, along with four types of rearrangements (reversals, translocations, fusions, and fissions) that can transform one genome into another. To model genomes with circular chromosomes, we will use a graph. First represent each synteny block by a directed black edge indicating its direction, and then link black edges corresponding to adjacent synteny blocks with a colored edge. Figure 1.9 shows each circular chromosome as an **alternating cycle** of red and black edges. In this model, the human genome can be represented using 280 human-mouse synteny blocks spread over 23 alternating cycles.

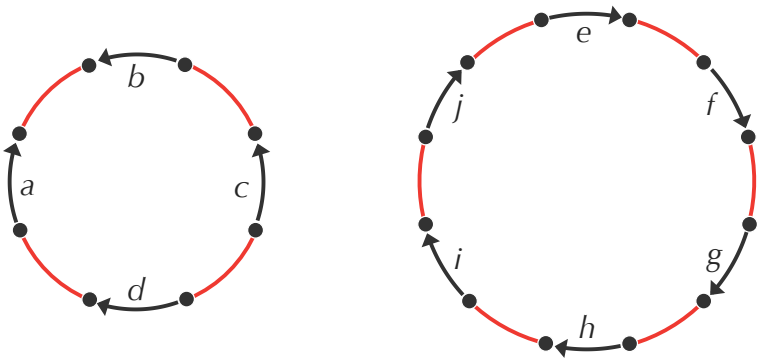
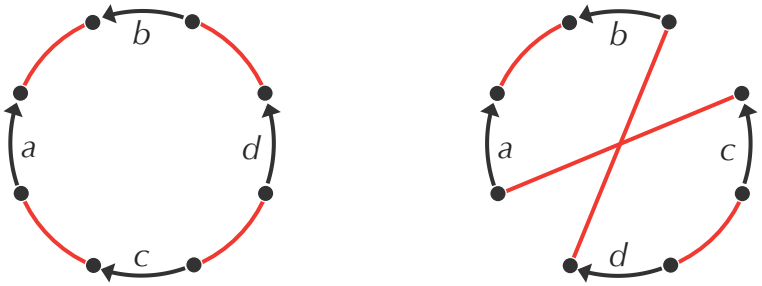


FIGURE 1.9: A genome with two circular chromosomes, $(+a\ -b\ -c\ +d)$ and $(+e\ +f\ +g\ +h\ +i\ +j)$. Black directed edges represent synteny blocks, and red undirected edges connect adjacent synteny blocks. A circular chromosome with n elements can be written in $2n$ different ways; the chromosome on the left can be written as $(+a\ -b\ -c\ +d)$, $(-b\ -c\ +d\ +a)$, $(-c\ +d\ +a\ -b)$, $(+d\ +a\ -b\ -c)$, $(-a\ -d\ +c\ +b)$ $(-d\ +c\ +b\ -a)$, $(+c\ +b\ -a\ -d)$, and $(+b\ -a\ -d\ +c)$.

STOP and Think: Let P and Q be genomes consisting of linear chromosomes, and let P^* and Q^* be the circularized versions of these genomes. Can you convert a given series of reversals/translocations/fusions/fissions transforming P into Q into a series of rearrangements transforming P^* into Q^*? What about the reverse operation — can you convert a series of rearrangements transforming P^* into Q^* into a series of rearrangements transforming P into Q?



1.7.3 2-breaks

We now focus on one of the chromosomes in a multi-chromosomal genome and consider a reversal transforming the circular chromosome $P = (+a\ -b\ -c\ +d)$ into $Q = (+a\ -b\ -d\ +c)$. We can draw Q in a variety of ways, depending on how we choose to arrange its black edges. Figure 1.10 shows two such equivalent representations.

FIGURE 1.10: Two equivalent drawings of the circular permutation $Q = (+a\ -b\ -d\ +c)$.

Although the first drawing of Q in Figure 1.10 is its most natural representation, we will use the second representation because its black edges are arranged around the circle in exactly the same order as they appear in the natural representation of $P = (+a\ -b\ -c\ +d)$. As illustrated in Figure 1.11, keeping the black edges fixed allows us to visualize the effect of the reversal. As you can see, the reversal deletes ("breaks") two red edges in P (connecting b to c and d to a) and replaces them with two new red edges (connecting b to d and c to a).

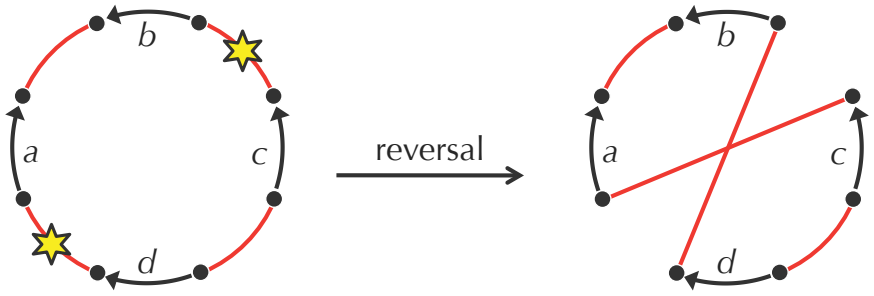


FIGURE 1.11: A reversal transforms $P = (+a\ -b\ -c\ +d)$ into $Q = (+a\ -b\ -d\ +c)$. We have arranged the black edges of Q so that they have the same orientation and position as the black edges in the natural representation of P. The reversal can be viewed as deleting the two red edges labeled by stars and replacing them with two new red edges on the same four nodes.

Figure 1.12 illustrates a fission of genome $P = (+a\ -b\ -c\ +d)$ into $Q = (+a\ -b)(-c\ +d)$; reversing this operation corresponds to a fusion of the two chromosomes of Q to yield P. Both the fusion and the fission operations, like the reversal, correspond to deleting two edges in one genome and replacing them with two new edges in the other genome.

A translocation involving two linear chromosomes can also be mimicked by circularizing these chromosomes and then replacing two red edges with two different red edges, as shown in Figure 1.13. We have therefore found a common theme uniting the four different types of rearrangements. They all can be viewed as breaking two red edges of the genome graph and replacing them with two new colored edges on the same four nodes. For this reason, we define the general operation on the genome graph in which two red edges are replaced with two new red edges on the same four nodes as a **2-break**.

We would like to find a shortest sequence of 2-breaks transforming genome P into genome Q, and we refer to the number of operations in this shortest sequence as the **2-break distance** between P and Q, denoted $d(P, Q)$.

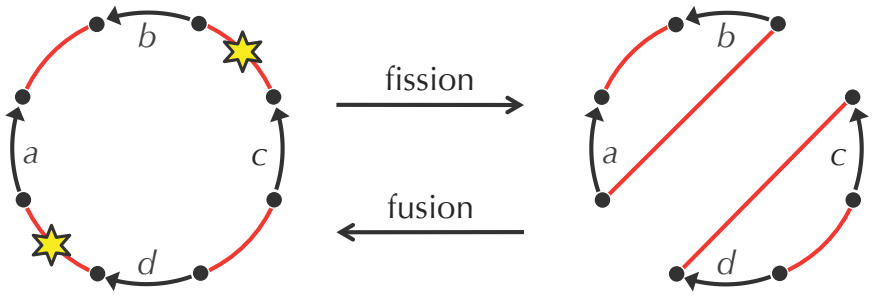


FIGURE 1.12: A fission of the single chromosome $P = (+a\ -b\ -c\ +d)$ into the genome $Q = (+a\ -b)(-c\ +d)$. We have again arranged the black edges of Q so that they have the same position and orientation as in the natural representation of P. The inverse operation is a fusion, transforming the two chromosomes of Q into a single chromosome by breaking two red edges of Q and replacing them with two other edges.

2-Break Distance Problem:
Find the 2-break distance between two genomes.

Input: Two genomes with circular chromosomes on the same set of synteny blocks.
Output: The 2-break distance between these genomes.

To compute the 2-break distance, we will return to the notion of breakpoints to construct a graph for comparing two genomes.

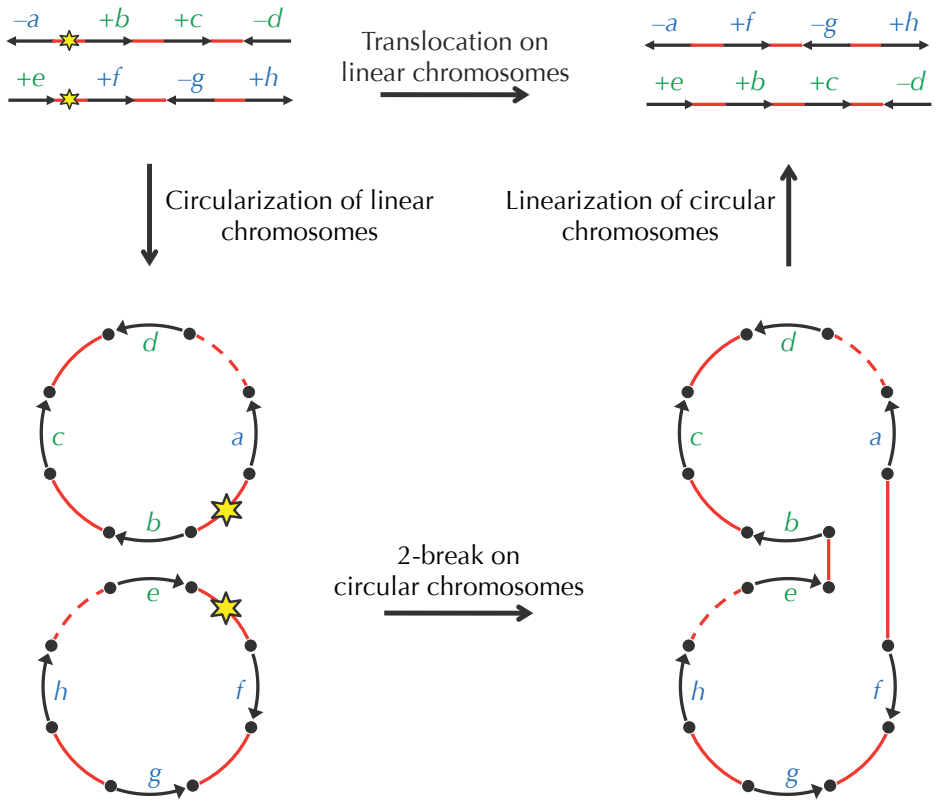


FIGURE 1.13: A translocation of linear chromosomes $(-a\ +b\ +c\ -d)$ and $(+e\ +f\ -g\ +h)$ transforms them into linear chromosomes $(-a\ +f\ -g\ +h)$ and $(+e\ +b\ +c\ -d)$. This translocation can also be accomplished by first circularizing the chromosomes, then applying a 2-break to the new chromosomes, and finally converting the resulting circular chromosomes into two linear chromosomes.

1.8 Breakpoint Graphs

Consider the genomes $P = (+a\ -b\ -c\ +d)$ and $Q = (+a\ +c\ +b\ -d)$ (Figure 1.14, left). Note that we have used red for the colored edges of P and blue for the colored edges of Q. As before, we rearrange the black edges of Q so that they are arranged exactly as in P (Figure 1.14, middle). If we superimpose the graphs of P and Q, then we obtain the tri-colored **breakpoint graph** BREAKPOINTGRAPH(P, Q) (Figure 1.14, right).

Note that the red and black edges in the breakpoint graph form genome P, and the blue and black edges form genome Q. Moreover, the red and blue

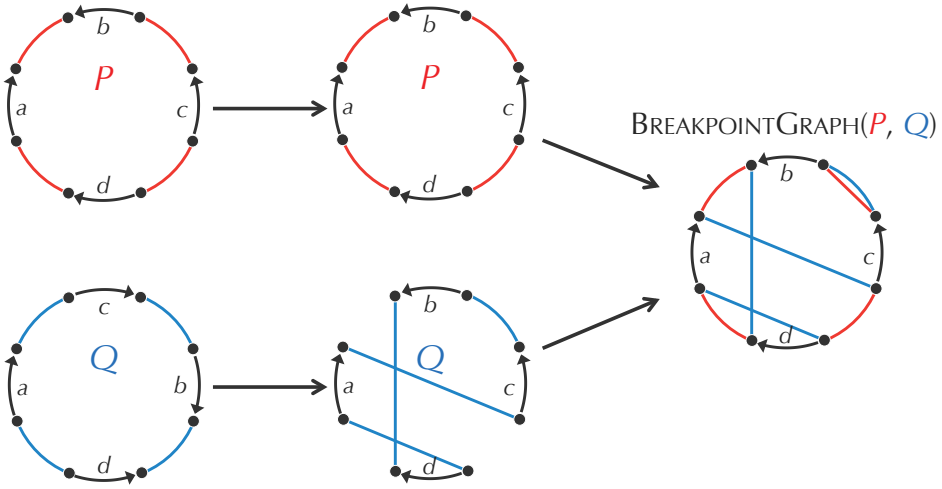


FIGURE 1.14: (Left) A red-black genome $P = (+a\ -b\ -c\ +d)$ and a blue-black genome $Q = (+a\ +c\ +b\ -d)$. (Middle) Rearranging the black edges of Q so that they are arranged the same as in P. (Right) The breakpoint graph BREAKPOINTGRAPH(P, Q), formed by superimposing the graphs of P and Q.

edges in the breakpoint graph form a collection of red-blue alternating cycles.

STOP and Think: Prove that the red and blue edges in any breakpoint graph form alternating cycles. Hint: How many red and blue edges meet at each node of the breakpoint graph?

We denote the number of red-blue alternating cycles in BREAKPOINTGRAPH(P, Q) as CYCLES(P, Q). For $P = (+a\ -b\ -c\ +d)$ and $Q = (+a\ +c\ +b\ -d)$, CYCLES(P, Q) = 2, as shown in Figure 1.15 (left). In what follows, we will be focusing on the red-blue alternating cycles in breakpoint graphs and often omit the black edges.

Although Figure 1.14 illustrates the construction of the breakpoint graph for single-chromosomal genomes, the breakpoint graph can be constructed for genomes with multiple chromosomes in exactly the same way (Figure 1.16).

STOP and Think: Given genome P, which genome Q maximizes CYCLES(P, Q)?

We denote the number of synteny blocks shared by genomes P and Q as BLOCKS(P, Q).

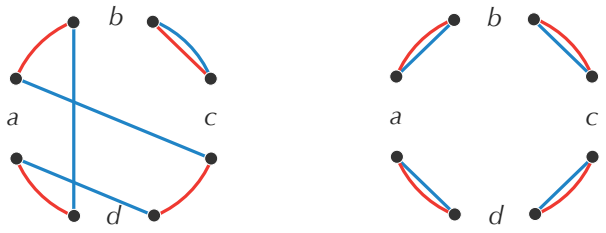


FIGURE 1.15: (Left) The red-blue alternating cycles in BREAKPOINTGRAPH(P, Q) for $P = (+a\ -b\ -c\ +d)$ and $Q = (+a\ +c\ +b\ -d)$. (Right) The trivial breakpoint graph BREAKPOINTGRAPH(P, P), formed by two copies of the genome $P = (+a\ -b\ -c\ +d)$. The breakpoint graph of *any* genome with itself consists only of trivial (i.e., length 2) alternating cycles.

As shown in Figure 1.15 (right), when P and Q are identical, their breakpoint graph consists of BLOCKS(P, Q) cycles of length 2, each containing one red and one blue edge. We refer to cycles of length 2 as **trivial cycles** and the breakpoint graph formed by identical genomes as the **trivial breakpoint graph**.

You are likely wondering how the breakpoint graph is useful. We can view a 2-break transforming P into P' as an operation on BREAKPOINTGRAPH(P, Q) that yields BREAKPOINTGRAPH(P', Q) (Figure 1.17).

By extension, we can view a series of 2-breaks transforming P into Q as a series of 2-breaks transforming BREAKPOINTGRAPH(P, Q) into BREAKPOINTGRAPH(Q, Q), the trivial breakpoint graph (Figure 1.18). Figure 1.19 illustrates a transformation of a breakpoint graph with CYCLES(P, Q) $= 2$ into a trivial breakpoint graph with CYCLES(Q, Q) $= 4$ using two 2-breaks.

Since every transformation of P into Q transforms BREAKPOINTGRAPH(P, Q) into the trivial breakpoint graph BREAKPOINTGRAPH(Q, Q), any sorting by 2-breaks increases the number of red-blue cycles by

$$\text{CYCLES}(Q, Q) - \text{CYCLES}(P, Q) = \text{BLOCKS}(P, Q) - \text{CYCLES}(P, Q).$$

STOP and Think: How much can each individual 2-break contribute to this increase? In other words, if P' is obtained from P by a 2-break, how much bigger can CYCLES(P', Q) be than CYCLES(P, Q)?

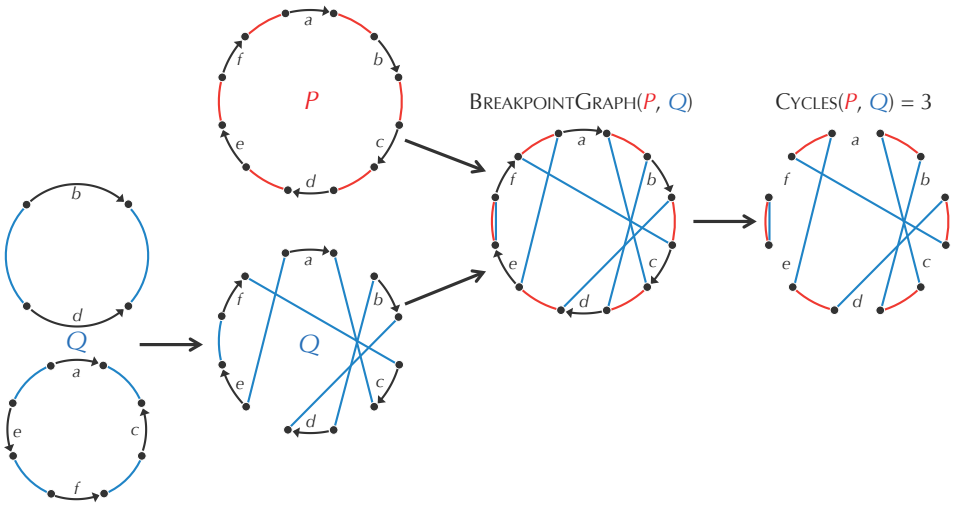


FIGURE 1.16: The construction of BREAKPOINTGRAPH(P, Q) for the unichromosomal genome $P = (+a\ +b\ +c\ +d\ +e\ +f)$ and the two-chromosome genome $Q = (+a\ -c\ -f\ -e)(+b\ -d)$. At the bottom, to illustrate the construction of the breakpoint graph, we first rearrange the black edges of Q so that they are drawn the same as in P.

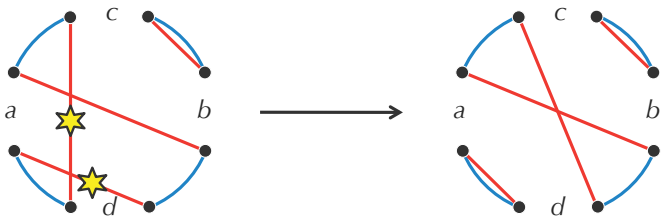


FIGURE 1.17: A 2-break transforming genome P into genome P' also transforms BREAKPOINTGRAPH(P, Q) into BREAKPOINTGRAPH(P', Q) for any permutation Q.

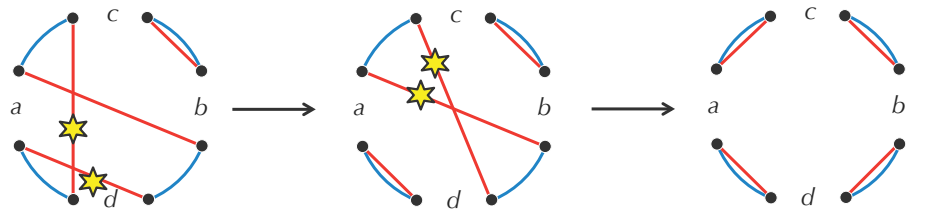


FIGURE 1.18: Every 2-break transformation of P into Q corresponds to a transformation of BREAKPOINTGRAPH(P, Q) into BREAKPOINTGRAPH(Q, Q). In the example shown, the number of red-blue cycles in the graph increases from CYCLES$(P, Q) = 2$ to BREAKPOINTGRAPH(Q, Q) = BLOCKS$(Q, Q) = 4$.

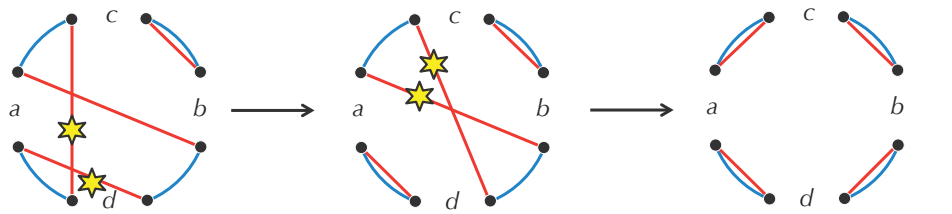


FIGURE 1.19: The transformation $P \rightarrow P' \rightarrow Q$ induces a transformation of the breakpoint graph BREAKPOINTGRAPH(P, Q) with 2 alternating cycles into the trivial breakpoint graph. Stars indicate red edges that are replaced in a 2-break.

1.9 Computing the 2-Break Distance

The Breakpoint Theorem stated that a reversal applied to a linear chromosome P can reduce $\text{BREAKPOINTS}(P)$ by at most 2. We now prove that a 2-break applied to a multichromosomal genome P can increase $\text{CYCLES}(P, Q)$ by at most 1, i.e., for any 2-break transforming P into P', and for any genome Q, $\text{CYCLES}(P', Q)$ cannot exceed $\text{CYCLES}(P, Q) + 1$.

Cycle Theorem: *For genomes P and Q, any 2-break applied to P can increase $\text{CYCLES}(P, Q)$ by at most 1.*

Proof. Figure 1.20 presents three cases that illustrate how a 2-break applied to P can affect the breakpoint graph. Each 2-break affects two red edges that either belong to the same cycle or to two different cycles in $\text{BREAKPOINTGRAPH}(P, Q)$. In the former case, the 2-break either does not change $\text{CYCLES}(P, Q)$, or it increases it by 1. In the latter case, it decreases $\text{CYCLES}(P, Q)$ by 1. □

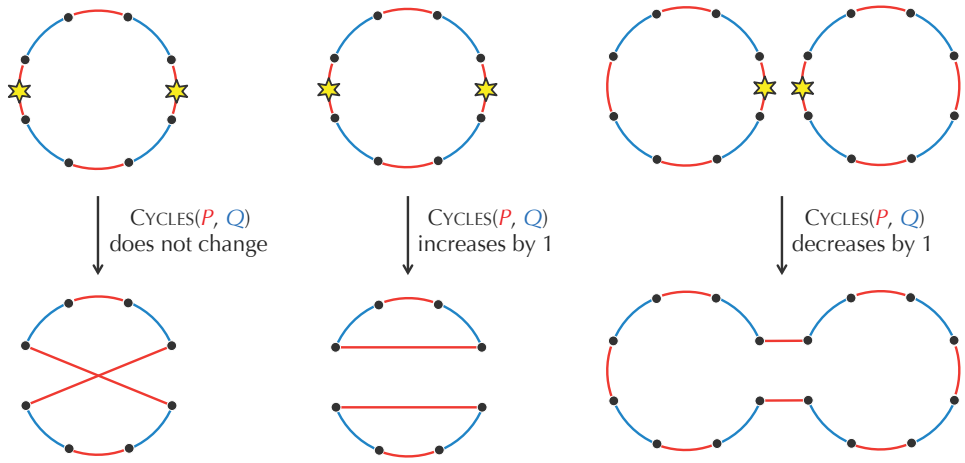


FIGURE 1.20: Three cases illustrating how a 2-break can affect the breakpoint graph.

Although the preceding proof is short and intuitive, it is not a formal proof, but rather an invitation to examine Figure 1.20. If you are interested in a more rigorous mathematical argument, please read the next proof.

Proof. A 2-break adds 2 new red edges and thus forms at most **2** new cycles (containing two new red edges) in $\text{BREAKPOINTGRAPH}(P, Q)$. At the same time,

it breaks 2 red edges and thus removes at least **1** old cycle (containing two old edges) from BREAKPOINTGRAPH(P, Q). Thus, the number of red-blue cycles in the breakpoint graph increases by at most $\mathbf{2} - \mathbf{1} = 1$, implying that CYCLES$(P, Q)$ increases by at most 1. □

Recall that there are permutations for which the number of breakpoints cannot be reduced, a fact that defeated our hopes for a greedy algorithm for sorting by reversals that reduces the number of breakpoints at each step. In the case of 2-breaks (on genomes with circular chromosomes), we now know that each 2-break can increase CYCLES(P, Q) by at most 1. But is it *always* possible to find a 2-break that increases CYCLES(P, Q) by 1? The answer, perhaps surprisingly, is yes.

2-Break Distance Theorem: *The 2-break distance between genomes P and Q is equal to* BLOCKS(P, Q) − CYCLES(P, Q).

Proof. Recall that every sorting by 2-breaks must increase the number of alternating cycles by BLOCKS(P, Q) − CYCLES(P, Q). The Cycle Theorem implies that each 2-break increases the number of cycles in the breakpoint graph by at most 1. This immediately implies in turn that $d(P, Q)$ is *at least* BLOCKS(P, Q) − CYCLES(P, Q). If P is not equal to Q, there must be a non-trivial cycle in BLOCKS(P, Q), i.e., a cycle with more than 2 edges. As shown in Figure 1.20 (middle), any non-trivial cycle in the breakpoint graph can be split into two cycles by a 2-break, implying that we can always find a 2-break increasing the number of red-blue cycles by 1. Therefore, $d(P, Q)$ is equal to BLOCKS(P, Q) − CYCLES(P, Q). □

Armed with this theorem, you should be ready to design an algorithm solving the 2-Break Distance Problem. Furthermore, having proved the formula $d(P, Q) =$ BLOCKS(P, Q) − CYCLES(P, Q) for the 2-break distance between genomes with multiple circular chromosomes, we wonder whether we can find an analogous formula for the reversal distance between single linear chromosomes.

STOP and Think: Compute the 2-break distance between the circularized human and mouse X chromosomes. Can you transform a series of 2-breaks for circularized chromosomes into a series of reversals sorting the linear X chromosomes?

Perhaps surprisingly, a fast algorithm for sorting permutations by reversals does exist, yielding an exact formula for the reversal distance! Although this sorting algorithm relies on the notion of breakpoints, it is unfortunately too complicated to present here (see DETOUR: Similar Problems with Different Fates).

The breakpoint graph constructed on the 280 human-mouse synteny blocks contains 35 alternating cycles, so that the 2-break distance between these genomes is $280 - 35 = 245$. Again, we don't know exactly how many 2-breaks happened in the last 75 million years, but we are certain that there were *at least* 245 steps. Remember this fact, since it will prove important in the next section.

1.10 Rearrangement Hotspots in the Human Genome

1.10.1 The Random Breakage Model meets the 2-Break Distance Theorem

You have probably anticipated from the beginning of the chapter that we would eventually argue against the Random Breakage Model. But it may still be unclear to you how the 2-break distance could possibly be used to do so.

Rearrangement Hotspots Theorem: *There are rearrangement hotspots in the human genome.*

Proof. Recall that if the Random Breakage Model is correct, then N reversals applied to a linear chromosome will produce approximately $2N + 1$ synteny blocks, since the probability is very low that two nearby locations in the genome will be used as the breakage point of more than one reversal. Similarly, N random 2-breaks applied to circular chromosomes will produce $2N$ synteny blocks.

Since there are 280 human-mouse synteny blocks, there must have been approximately $280/2 = 140$ 2-breaks on the evolutionary path between humans and mice. However, the 2-Break Distance Theorem tells us that there were at least 245 2-breaks on this evolutionary path.

STOP and Think: Is $245 \approx 140$?

Since 245 is much larger than 140, we have arrived at a contradiction, implying that one of our assumptions is incorrect! But the only assumption we made in this proof was "*If the Random Breakage Model is correct...*" Thus, this assumption must have been wrong. □

This argument, which is not a mathematical proof, is nevertheless logically solid. It offers an example of a **proof by contradiction**, in which we begin by assuming the statement that we intend to disprove and then demonstrate how this assumption leads to a contradiction. As a result of the Rearrangement Hotspots Theorem, we conclude that there was breakpoint reuse on the human-mouse evolutionary path. This breakpoint reuse was extensive, as quantified by the large ratio between the actual 2-break distance and what the 2-break distance would have been under the Random Breakage Model ($245/140 = 1.75$).

Of course, our arguments need to be made statistically sound in order to ensure that the discrepancy between the Random Breakage Model's prediction and the 2-break distance is significant. After all, even though genomes are large, there is still a small chance that randomly chosen 2-breaks might occasionally break a genome more than once in a small interval. Unfortunately, the necessary statistical analysis is beyond the scope of this chapter.

1.10.2 The Fragile Breakage Model

But wait — what about Nadeau and Taylor's argument in favor of the Random Breakage Model? We certainly cannot ignore that the lengths of the human-mouse synteny blocks resemble an exponential distribution.

STOP and Think: Can you find anything wrong with Nadeau and Taylor's logic?

The Nadeau and Taylor argument in favor of the Random Breakage Model exemplifies a classic logic fallacy. It is true that if breakage is random, then the histogram of synteny block lengths should follow the exponential distribution. But it is a completely different statement to conclude that just because synteny block lengths follow the exponential distribution, breakage must have been random. The distribution of synteny block lengths certainly provides support for the Random Breakage Model, but it does not prove that it is correct.

Nevertheless, any alternative hypothesis we put forth for the Random Breakage Model must account for the observation that the distribution of synteny block lengths for the human and mouse genomes is approximately exponential.

STOP and Think: Can you propose a different model of chromosome evolution that explains rearrangement hotspots and is consistent with the exponential distribution of synteny block lengths?

The contradiction of the Random Breakage Model led to an alternative **Fragile Breakage Model** of chromosome evolution, which was proposed in 2003. This model states that every mammalian genome is a mosaic of long solid regions, which are rarely affected by rearrangements, as well as short **fragile regions** that serve as rearrangement hotspots and that account only for a small fraction of the genome. For humans and mice, these fragile regions make up approximately 3% of the genome.

If we once again follow Occam's razor, then the most reasonable way to allow for exponentially distributed synteny block lengths is if the fragile regions themselves are distributed randomly in the genome. Indeed, *randomly* selecting breakpoints within *randomly* distributed fragile regions is not unlike randomly selecting the endpoints of a rearrangement throughout the entire genome. Yet although we now have a model that fits our observations, many questions re-

main. For example, it is unclear where fragile regions are located, or what causes genomic fragility in the first place.

STOP and Think: Consider the following statement: "The exponential distribution of synteny block lengths and extensive breakpoint re-use imply that the Fragile Breakage Model must be true." Is this argument logically sound?

The point we are driving at by asking the preceding question is that we will never be able to prove a scientific theory like the Fragile Breakage Model in the same way that we have proved one of the mathematical theorems in this chapter. In fact, many biological theories are based on arguments that a mathematician would view as fallacious; the logical framework used in biology is quite different from that used in mathematics. To take an historical example, neither Darwin nor anyone else has ever proved that evolution by natural selection is the only — or even the most likely — explanation for how life on Earth evolved!

We have already given many reasons to biology professors to send us to Biology 101 boot camp, but now we will probably be rounded up and imprisoned with Intelligent Design proponents. However, not even Darwinism is unassailable; in the 20th Century, this theory was revised into Neo-Darwinism, and there is little doubt that it will continue to evolve.

1.11 Epilogue: Synteny Block Construction

Throughout our discussion of genome rearrangements, we assumed that we were given synteny blocks in advance. In this section, we will explain how to construct synteny blocks from genomic sequences.

1.11.1 Genomic dot-plots

Biologists sometimes visualize repeated k-mers within a string as a collection of points in the plane; a point with coordinates (x, y) represents identical k-mers occurring at positions x and y in the string. The top panels in Figure 1.21 present two of these **genomic dot plots**. Of course, since DNA is double-stranded, we should expand the notion of repeated k-mers to account for repeats occurring in the complementary strand. In the bottom left panel of Figure 1.21, blue points (x, y) indicate that the k-mers starting at positions x and y of the string are reverse complementary.

1.11.2 Finding shared k-mers

Recalling that a synteny block is defined by many similar genes occurring in the same order in two genomes, let's first find the positions of all k-mers that are shared by the human and mouse X chromosomes. If we choose k to be sufficiently large (e.g., $k = 30$), then it is rather unlikely that shared k-mers represent spurious similarities. A more likely explanation is that they come from related genes (or shared repeats) in the human and mouse genomes.

Formally, we say that a k-mer is **shared** by two genomes if either the k-mer or its reverse complement appears in each genome. Below are four pairs of 3-mers (shown in bold) that are shared by `AAACTCATC` and `TTTCAAATC`; note that the second pair of 3-mers are reverse complements of each other.

```
    0              0                4              6
    AAACTCATC      AAACTCATC        AAACTCATC      AAACTCATC
TTTCAAATC          TTTCAAATC          TTTCAAATC      TTTCAAATC
    4              0                2              6
```

We can further generalize the genomic dot plot to analyze the shared k-mer content of two genomes. We color the point (x, y) red if the two genomes share a k-mer at respective positions x and y; we color (x, y) blue if the two genomes have reverse complementary k-mers at these starting positions. See Figure 1.21

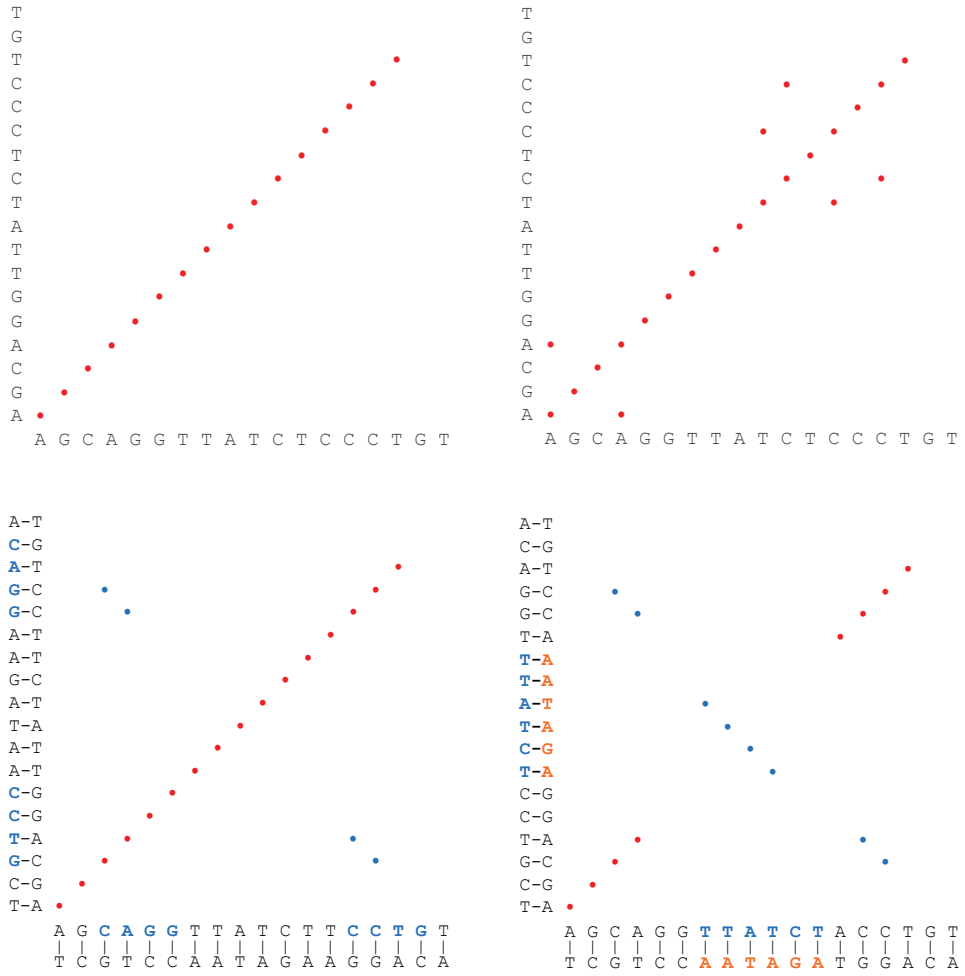


FIGURE 1.21: A visualization of repeated k-mers within the string AGCAGGTTATCTCCCTGT for $k = 3$ (top left) and $k = 2$ (top right). (Bottom left) We add blue points to the plot shown in in the upper left corner to indicate reverse complementary k-mers. For example, CCT and AGG are reverse complementary 3-mers in AGCAGGTTATCTTCCTGT. (Bottom right): Genomic dot-plot showing shared 3-mers between AGCAGG**TTATCT**CCCTGT and AGCAGG**AGATAA**CCCTGT. The latter sequence resulted from the former sequence by a reversal of the segment **TTATCT**. Each point (x, y) corresponds to a k-mer shared by the two genomes. Red points indicate identical shared k-mers, whereas blue points indicate reverse complementary k-mers. Note that the dot-plot has four "noisy" blue points in the diagram: two in the upper left corner, and two in the bottom right corner. You will also notice that red dots can be connected into line segments with slope 1 and blue dots can be connected into line segments with slope -1. The resulting three synteny blocks (**AGCAGG**, **TTATCT**, and **CCCTGT**) correspond to three diagonals (each formed by four points) in the dot-plot.

(bottom right).

EXERCISE BREAK: Find all shared 2-mers of `AAACTCATC` and `TTTCAAATC`.

Shared *k*-mers Problem:

Given two strings, find all their shared k-mers.

Input: An integer k and two strings.

Output: All k-mers shared by these strings, in the form of ordered pairs (x, y).

Since downloading long human and mouse chromosomes is time-consuming, we will instead solve the Shared k-mers Problem for the bacteria *E. coli* and *S. enterica*, which we have already encountered in previous chapters.

EXERCISE BREAK: Answer the following questions regarding counting shared k-mers.

1. Compute the expected number of 30-mers shared by two random strings, each a billion nucleotides long.
2. How many shared 30-mers do the *E. coli* and *S. enterica* genomes have?

In previous chapters, we have worked with the genomes for the bacteria *E. coli* and *S. enterica*, each of which is about 5 million nucleotides long. It can be shown that the expected number of shared 30-mers between two random 5 million nucleotide-long sequences is approximately $2 \cdot (5 \cdot 10^6)^2 / 4^{30} \approx 1/20,000$.

Yet solving the Shared k-mers Problem for *E. coli* and *S. enterica* yields over 200,000 pairs (x, y) corresponding to shared 30-mers. The surprisingly large number of shared 30-mers indicates that *E. coli* and *S. enterica* are close

relatives that have retained many similar genes inherited from their common ancestor. However, these genes may be arranged in a different order in the two species: how can we infer synteny blocks from these genomes' shared k-mers? The genomic dot-plot plot for *E. coli* and *S. enterica* is shown in Figure 1.22.

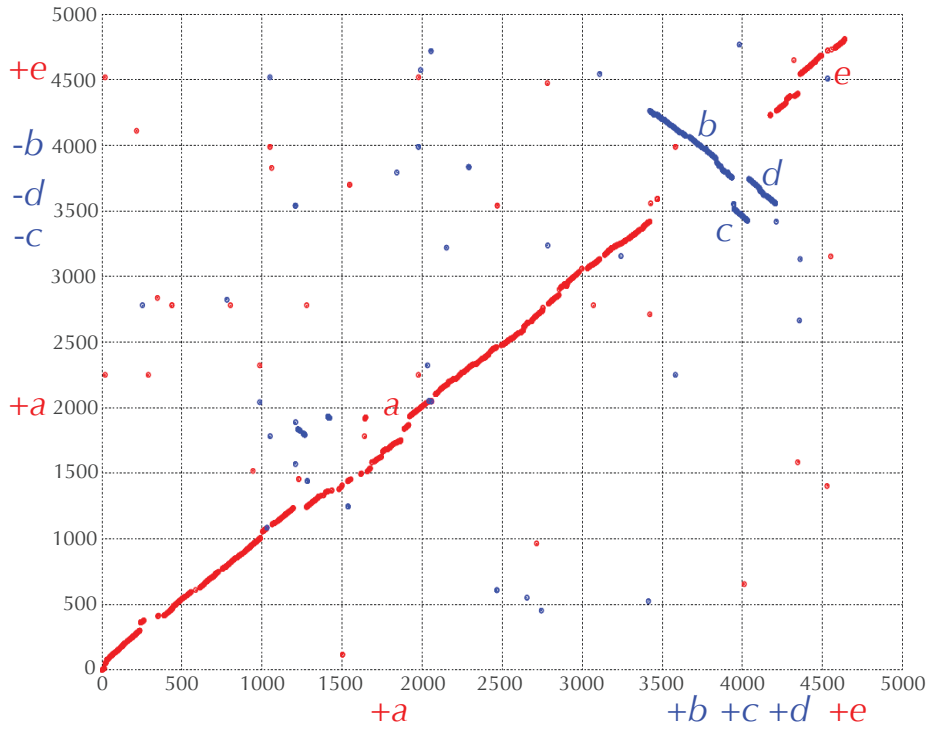


FIGURE 1.22: Genomic dot-plot of *E. coli* (horizontal axis) and *S. enterica* (vertical axis) for $k = 30$. Each point (x, y) corresponds to a k-mer shared by the two genomes. Red points indicate identical shared k-mers, whereas blue points indicate reverse complementary k-mers. Each axis is measured in kilobases (thousands of base pairs).

STOP and Think: Can you see the synteny blocks in the genomic dot-plot in Figure 1.22?

1.11.3 From shared *k*-mers to synteny blocks

The genomic dot-plot in Figure 1.22 indicates five regions of similarity in the form of points that clump together into approximately diagonal segments. These segments are labeled by a, b, c, d, and e according to the order in which they appear in the *E. coli* genome; we ignore smaller diagonals such as the short blue diagonal starting around position 1.3 million in *E. coli* and around position 1.9 million in *S. enterica*. For example, while a corresponds to a long diagonal segment of slope 1 that covers approximately the first 3.5 million positions in both genomes, b corresponds to a shorter diagonal segment of slope -1 that starts shortly before position 3.5 million in *E. coli* and shortly after position 4 million in *S. enterica*. Although b appears small in Figure 1.22, don't be fooled by the scale of the figure; b is over 100,000 nucleotides long and contains nearly 100 genes.

The segments a, b, c, d, and e give us the synteny blocks that we have been looking for. If we project these blocks onto the x- and y-axes, then the ordering of blocks on each axis corresponds to the ordering of synteny blocks in the respective bacterium. The ordering of synteny blocks in *E. coli* (plotted on the x-axis) is $(+a\ +b\ +c\ +d\ +e)$, and the ordering in *S. enterica* (y-axis) is $(+a\ -c\ -d\ -b\ +e)$. Note that the blue letters in *S. enterica* are assigned a negative sign because these blocks were constructed from reverse complementary k-mers. Figure 1.22 also illustrates what the directions of blocks are — they respectively correspond to diagonals in the dot-plot with slope 1 (blocks with a "+" sign) and slope -1 (blocks with a "-" sign).

We have therefore represented two bacterial genomes using just five synteny blocks. Of course, this simplification required us to throw out some noisy points in the genome plot, corresponding to tiny regions of similarity that did not surpass a threshold length in order to be considered synteny blocks.

We are now ready to construct the 11 human-mouse synteny blocks originally presented in Figure 1.1 (page 3), but since the human and mouse X chromosomes are rather long, we will instead provide you with all positions (x, y) where they share significant similarities. Figure 1.23 (top left) presents the resulting genomic dot-plot for the human and mouse X chromosomes, where each dot rep-

resents a long similar region rather than a shared k-mer. Our eyes immediately find 11 diagonals in this plot corresponding to the human-mouse X chromosome synteny blocks — problem solved! We state this problem as the Finding Synteny Blocks Problem.

Finding Synteny Blocks Problem:
Find diagonals in the genomic dot-plot.

Input: A set of points *DotPlot* in the plane.
Output: A set of diagonals in *DotPlot* representing synteny blocks.

Unfortunately, it remains unclear how to write a program to do what our eyes found to be so easy; we hope you have already noticed that the Finding Synteny Blocks Problem is not a well-formulated computational problem. As we have mentioned, the diagonals in Figure 1.23 (top left) are not perfect. Moreover, there are many gaps within diagonals that cannot be seen by the human eye but will become apparent if we zoom into the genome plot. It is thus absolutely unclear what method the human brain is using to transform the dots into the 11 diagonals in the genomic dot-plot.

STOP and Think: How can we translate the brain's tendency to construct the diagonals that you see in Figure 1.23 (top left) into an algorithm that a computer can understand?

1.11.4 Synteny blocks as connected components in graphs

The reason why you can easily see the synteny blocks in a genomic dot-plot is that your brain is good at **clustering** nearby points in an image. To mimic this process with a computer, we therefore need a precise notion of clustering. Given a set of points *DotPlot* in the plane as well as a parameter *maxDistance*, we

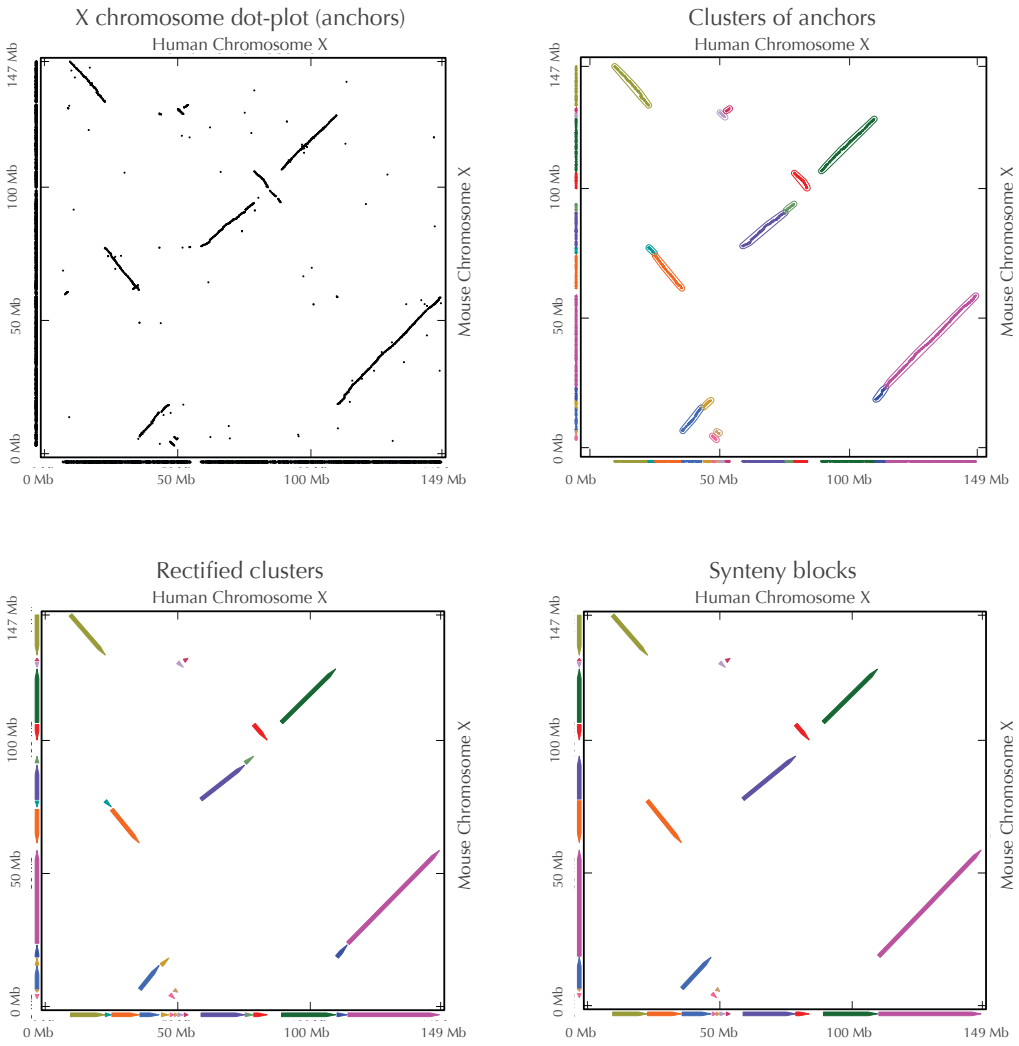


FIGURE 1.23: From local similarities to synteny blocks. (Top left) The genomic dot-plot for the human and mouse X chromosomes, representing all positions (x, y) where they share significant similarities. In contrast with Figure 1.22, we do not distinguish between red and blue dots. (Top right) Clusters (connected components) of points in the genomic dot-plot are formed by constructing the synteny graph. (Bottom left) Rectified clusters from the synteny graph transform each cluster into an exact diagonal of slope ± 1. (Bottom right) Aggregated synteny blocks. Projection of the synteny blocks to the x-and y-axes results in the arrangements of synteny blocks in the respective human and mouse genomes $(+1\ +2\ +3\ +4\ +5\ +6\ +7\ +8\ +9\ +10\ +11)$ and $(+1\ -7\ +6\ -10\ +9\ -8\ +2\ -11\ -3\ +5\ +4)$.

will construct the (undirected) **synteny graph** SYNTENYGRAPH(*DotPlot*, *maxDistance*) by connecting two points in *DotPlot* with an edge if the distance between them does not exceed *maxDistance*.

Every graph can be divided into disjoint connected subgraphs called **connected components**. The connected components in SYNTENYGRAPH(*DotPlot*, *maxDistance*) represent candidate synteny blocks between the two genomes (Figure 1.24). When we construct the synteny graph for the human and mouse X chromosomes, we find a huge number of small connected components (the exact number depends on our choice of the *maxDistance* parameter). However, we will ignore these small connected components, since they may represent spurious similarities. We thus introduce a parameter called *minSize* representing the minimum number of points in a connected component that we will consider as forming a synteny block. Our goal is to return all connected components having at least *minSize* nodes.

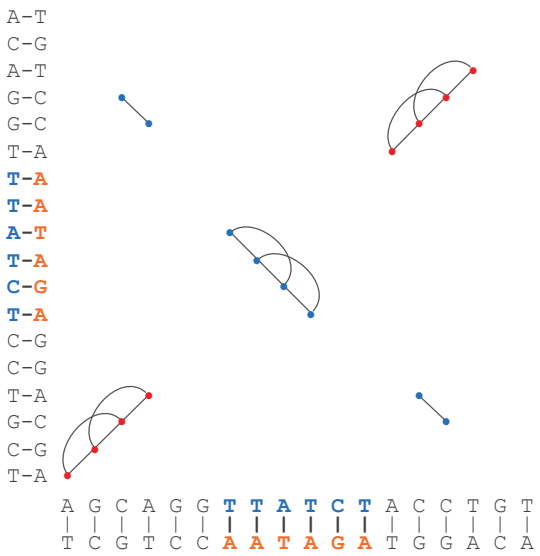


FIGURE 1.24: The graph SYNTENYGRAPH(*DotPlot*, 4) constructed from the genomic dot-plot of AGCAGG**TTATCT**CCCTGT and AGCAGG**AGATAA**CCCTGT for $k = 3$. Note that the three synteny blocks (all of which have four nodes) correspond to diagonals in the genomic dot-plot. We ignore the two smaller, noisy synteny blocks.

SYNTENYBLOCKS(*DotPlot*, *maxDistance*, *minSize*)
construct SYNTENYGRAPH(*DotPlot*, *maxDistance*)
find the connected components in SYNTENYGRAPH(*DotPlot*, *maxDistance*)
output connected components of at least *minSize* nodes as candidate synteny blocks

As Figure 1.23 (top right) illustrates, **SYNTENYBLOCKS** has a tendency to partition a single diagonal (as perceived by the human eye) into multiple diagonals due to gaps that exceed the parameter *maxDistance*. However, this partitioning is not a problem, since the broken diagonals can be combined later into a single (aggregated) synteny block.

STOP and Think: We have defined synteny blocks as large connected components in SYNTENYGRAPH(*DotPlot*, *maxDistance*) but have not described how to determine where these synteny blocks are located in the original genomes. Using Figure 1.23 as a hint, design an algorithm for finding this information.

You should now be ready to solve the challenge problem and discover that the choice of parameters is one of the dark secrets of bioinformatics research.

CHALLENGE PROBLEM: Construct the synteny blocks for the human and mouse X chromosomes and compute the 2-break distance between the circularized human and mouse X chromosomes using the synteny blocks that you constructed. How does this distance change depending on the parameters *maxDistance* and *minSize*?

1.12 Open Problem: Can Rearrangements Shed Light on Bacterial Evolution?

Although there exist efficient algorithms for analyzing *pairwise* genome rearrangements, constructing rearrangement scenarios for *multiple* genomes remains an open problem. For example, we now know how to find a most parsimonious rearrangement scenario transforming the mouse X chromosome into the human X chromosome. However, the problem of finding a most parsimonious rearrangement scenario for the human, mouse and rat X chromosomes (let alone for their entire genomes) is a more difficult problem. The difficulties further amplify when we attempt to reconstruct a rearrangement history for dozens of mammalian genomes. To address this challenge, we start from the simpler (but still unsolved) case of bacterial genomes.

Let *Tree* be a tree (i.e., a connected acyclic undirected graph) with nodes labeled by some genomes. In the case of bacterial genomes, we assume that every node (genome) is labeled by a circular permutation on n elements. Given an edge e connecting nodes v and w in *Tree*, we define DISTANCE(v, w) as the 2-break distance between genomes v and w. The **tree distance** DISTANCE$(Tree)$ is the sum

$$\sum_{\text{all edges } (v,w) \text{ in } Tree} \text{DISTANCE}(v, w).$$

Given a set of genomes $P_1, \ldots, P_n$ and an evolutionary tree *Tree* with n leaves labeled by $P_1, \ldots, P_n$, the Ancestral Genome Reconstruction Problem attempts to reconstruct genomes at the internal nodes of the tree such that DISTANCE$(Tree)$ is minimized across all possible reconstructions of genomes at internal nodes.

Ancestral Genome Reconstruction Problem:
Given a tree with leaves labeled by genomes, reconstruct ancestral genomes that minimize the tree distance.

Input: A tree *Tree* with each leaf labeled by a genome.
Output: Genomes *AncestralGenomes* assigned to the internal nodes of *Tree* such that DISTANCE(*Tree*) is minimized across all possible choices of *AncestralGenomes*.

In the case when *Tree* is not given, we need to infer it from the genomes.

Multiple Genome Rearrangement Problem:
Given a set of genomes, reconstruct a tree with leaves labeled by these genomes and minimum tree distance.

Input: A set of genomes.
Output: A tree *Tree* with leaves labeled by these genomes and internal nodes labeled by (unknown) genomes *AncestralGenomes* such that DISTANCE(*Tree*) is minimal among all possible choices of *Tree* and *AncestralGenomes*.

While many algorithms have been proposed for the Multiple Genome Rearrangement Problem, they have mainly been applied to analyze mammalian evolution (see [MRR$^+$08a] and [AP09] for some examples). However, there have been hardly any applications of the Multiple Genome Rearrangement Problem for analyzing bacterial evolution. The fact that bacterial genomes are approximately 1000 times smaller than mammalian genomes does not make this problem 1000 times easier. In fact, there are unique challenges and opportunities in bacterial

evolutionary research.

Consider 100 genomes from three closely related bacterial genera, *Salmonella*, *Shigella*, and *Escherichia*, whose various species are responsible for dysentery, typhoid fever, and a variety of foodborne illnesses. After you construct synteny blocks shared by all these genomes, you will see that there are relatively few (usually fewer than 10) rearrangements between every pair of genomes. However, solving the Multiple Genome Rearrangement Problem even in the case of closely related genomes presents a formidable challenge, and nobody has been able to construct a rearrangement scenario for more than a couple dozen — let alone 100! — species yet.

After you solve this puzzle, you will be able to address the question of whether there are fragile regions in bacterial genomes. Answering this question for a pair of bacterial genomes, like we did for the human and mouse genomes, may not be possible because there are typically fewer than 10 rearrangements between them. But answering this question for 100 bacterial genomes may be possible if we witness the same breakage occurring independently on many branches of the evolutionary tree. However, you will need to develop algorithms to analyze fragile regions in multiple (rather than pairwise) genomes.

After you construct the evolutionary tree, you will also be in a position to analyze the question of what triggers rearrangements. While many authors have discussed the causes of fragility, this question remains open, with no shortage of hypotheses. [ZB09] demonstrated that many rearrangements are flanked by **matching duplications**, a pair of long similar regions located within a pair of breakpoint regions corresponding to a rearrangement event. However, they limited their study to mammalian evolution, and it remains unclear what triggers rearrangements in bacteria; can you answer this question?

1.13 Detours

1.13.1 Why is the gene content of mammalian X chromosomes so conserved?

While mammalian X chromosomes are enriched in genes related to sexual reproduction, most of the approximately 1000 genes on the X chromosome have nothing to do with gender. Ideally, they should be expressed (i.e., transcribed and eventually translated) in roughly the same quantities in females and males. But since females have two X chromosomes and males have only one, it would seem that all the genes on the X chromosome should have twice the expression level in females. This imbalance would lead to a problem in the complex cellular system of checks and balances underlying gene expression.

The need to balance gene expression in males and females led to the evolution of special mechanisms of **dosage compensation**, or the inactivation of one X chromosome in females to equalize gene expression between the sexes. Because of dosage compensation, the gene content of the X chromosome is highly conserved between mammalian species because if a gene jumps off the X chromosome, then its expression may double, thus creating a genetic imbalance.

1.13.2 Discovery of genome rearrangements

Genetic maps, which show the positions of genes along chromosomes, were used by Thomas Hunt Morgan's lab at Columbia as early as 1913. An amazing thing happened in 1921 when Morgan's student Alfred Sturtevant created genetic maps for two different species of *Drosophila*. It was clear just by looking at the maps that an entire genomic interval had been inverted in one species as compared to another! The only reasonable explanation was that a reversal had flipped this chromosomal interval around. Sturtevant posited that this had happened when the chromosome became tangled on itself and formed a loop.

Another breakthrough occurred with the discovery that the salivary

glands of *Drosophila* contain **polytene cells**. In normal cellular division, each daughter cell receives one copy of the genome. However, in the nuclei of polytene cells, DNA replication occurs repeatedly in the absence of cell division. The resulting chromosomes then knit themselves together into much larger "superchromosomes" called **polytene chromosomes**.

Polytene chromosomes serve a practical purpose for the fruit fly, which uses the extra DNA to boost the production of gene transcripts, producing lots of sticky saliva. But the human value of polytene chromosomes is perhaps greater. When Sturtevant and his collaborator, Theodosius Dobzhansky, looked at polytene chromosomes under a microscope, they were able to witness the work of rearrangements firsthand in tangled mutant chromosomes. In 1938, Sturtevant and Dobzhansky published a milestone paper with an evolutionary tree presenting a rearrangement scenario with 17 reversals for various species of *Drosophila*. Their drawing was the first evolutionary tree in history to be constructed based on molecular data.

1.13.3 The exponential distribution

A **Bernoulli trial** is a random experiment with two possible outcomes, "success" (having probability p) and "failure" (having probability $1 - p$). The **geometric distribution** is the probability distribution underlying the random variable X representing the number of Bernoulli trials needed to obtain the first success:

$$\Pr(X = k) = (1 - p)^{k-1} p.$$

A **Poisson process** is a continuous-time stochastic process counting the number of events in a given time interval, if we assume that the events occur independently at a constant rate. For example, the Poisson process offers a good model of time points for passengers arriving to a large train station. If we assume that the number of passengers arriving during a very small time interval ϵ is $\lambda \cdot \epsilon$ (where λ is a constant), then we are interested in the probability $F(X)$ that nobody will arrive to the station during a time interval X. The

exponential distribution describes the time between events in a Poisson process.

STOP and Think: Do you see any similarities between the Poisson process and the Bernoulli trials or between the exponential and geometric distributions?

The exponential distribution is merely the continuous analogue of the geometric distribution. More precisely, the Poisson process is characterized by a **rate parameter** λ, such that the number of events k in the time interval $[X, X + \epsilon]$ follows the **Poisson probability distribution**:

$$e^{-\lambda \cdot \epsilon}(\lambda \cdot \epsilon)^k / k!$$

The **probability density function** of the exponential distribution is $\lambda e^{-\lambda \cdot X}$ (compare with the geometric distribution shown in Figure 1.25).

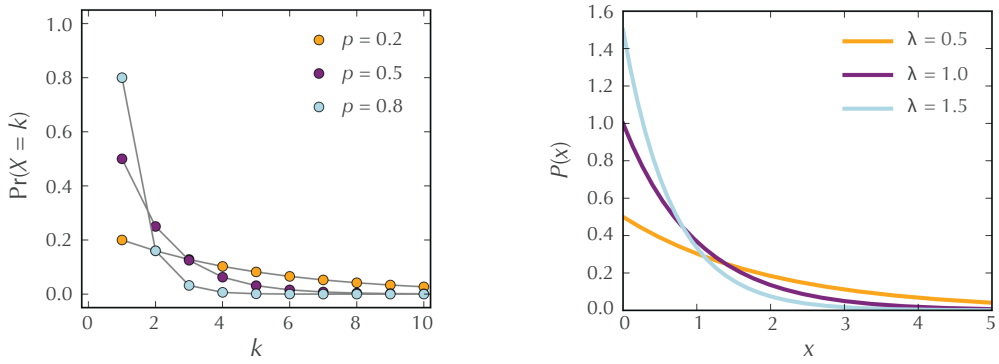


FIGURE 1.25: The probability density functions of the geometric (left) and exponential (right) distributions, each provided for three different parameter values. Courtesy Skbkekas (Wikipedia user).

1.13.4 Bill Gates and David X. Cohen flip pancakes

Before biologists faced genome rearrangement problems, mathematicians posed the **Pancake Flipping Problem**, arising from the following hypothetical waiter's conundrum.

> *The chef in our place is sloppy, and when he prepares a stack of pancakes they come out all different sizes. Therefore, when I deliver them to a customer, on the way to a table I rearrange them (so that the smallest winds up on top, and so on, down to the largest at the bottom) by grabbing several from the top and flipping them over, repeating this (varying the number I flip) as many times as necessary. If there are n pancakes, what is the maximum number of flips that I will ever have to use to rearrange them?*

Formally, a **prefix reversal** is a reversal that flips a prefix, or initial interval, of a permutation. The **Pancake Flipping Problem** corresponds to sorting unsigned permutations by prefix reversals. For example, the series of prefix reversals shown below ignores signs and represents the sorting of an **unsigned permutation**, (1 7 6 10 9 8 2 11 3 5 4), into the **identity unsigned permutation**, (1 2 3 4 5 6 7 8 9 10 11). The inverted interval is shown in red, and sorted intervals at the end of the permutation are shown in blue.

(**1** **7** **6** **10** **9** **8** **2** **11** 3 5 4)
(**11** **2** **8** **9** **10** **6** **7** **1** **3** **5** **4**)
(**4** **5** **3** **1** **7** **6** **10** 9 8 2 **11**)
(**10** **6** **7** **1** **3** **5** **4** **9** **8** **2** **11**)
(**2** **8** **9** 4 5 3 1 7 6 **10** **11**)
(**9** **8** **2** **4** **5** **3** **1** **7** **6** **10** **11**)
(**6** **7** 1 3 5 4 2 **8** **9** **10** **11**)
(**7** **6** **1** **3** **5** **4** **2** **8** **9** **10** **11**)
(**2** **4** **5** 3 1 **6** **7** **8** **9** **10** **11**)
(**5** **4** **2** **3** **1** **6** **7** **8** **9** **10** **11**)
(**1** **3** 2 **4** **5** **6** **7** **8** **9** **10** **11**)
(**3** **1** **2** **4** **5** **6** **7** **8** **9** **10** **11**)
(**2** **1** **3** **4** **5** **6** **7** **8** **9** **10** **11**)
(**1** **2** **3** **4** **5** **6** **7** **8** **9** **10** **11**)

When we instead desire a minimum series of prefix reversals sorting a *signed* permutation, the problem is called the **Burnt Pancake Flipping Problem** (each pancake is "burnt" on one side, giving it two possible orientations).

STOP and Think: Prove that every unsigned permutation of length n can be sorted using at most $2 \cdot (n-1)$ prefix reversals. Prove that every signed permutation of length n can be sorted using at most $3 \cdot (n-1)+1$ prefix reversals.

Bill Gates, an undergraduate student at Harvard in the mid-1970s, and Christos Papadimitriou, a professor at Harvard in the mid-1970s, made the first attempt to solve the Pancake Flipping Problem and proved that any permutation of length n can be sorted with at most $5/3 \cdot (n+1)$ prefix reversals, a result that would not be improved for three decades. David X. Cohen worked on the Burnt Pancake Flipping Problem at Berkeley before he left computer science to become a writer for *The Simpsons* and eventually producer of *Futurama*. Along with Manuel Blum, he demonstrated that the Burnt Pancake Flipping Problem can be solved with at most $2 \cdot (n-1)$ prefix reversals.

1.13.5 Similar problems with different fates

In the main text, we defined the breakpoint graph for circular chromosomes, but this structure can easily be extended to linear chromosomes. Figure 1.26 depicts the human and the mouse X chromosomes as alternating red-black and blue-black paths (1st and 2nd panels). These two paths are superimposed in the 3rd panel to form the breakpoint graph, which has 5 alternating red-blue cycles.

STOP and Think: Prove the following analogue of the Cycle Theorem for permutations: Given permutations P and Q, any reversal applied to P can increase $\text{CYCLES}(P, Q)$ by at most 1.

While the number of trivial cycles is equal to $\text{BLOCKS}(Q, Q)$ in the identity breakpoint graph of a circular permutation, the trivial breakpoint graph of a linear permutation has $\text{BLOCKS}(Q, Q)+1$ trivial cycles. Since the Cycle Theorem holds for linear permutations, perhaps the reversal distance $d_{\text{rev}}(P, Q)$ is equal to $\text{BLOCKS}(P, Q)+1-\text{CYCLES}(P, Q)$ for linear chromosomes? After all, for the

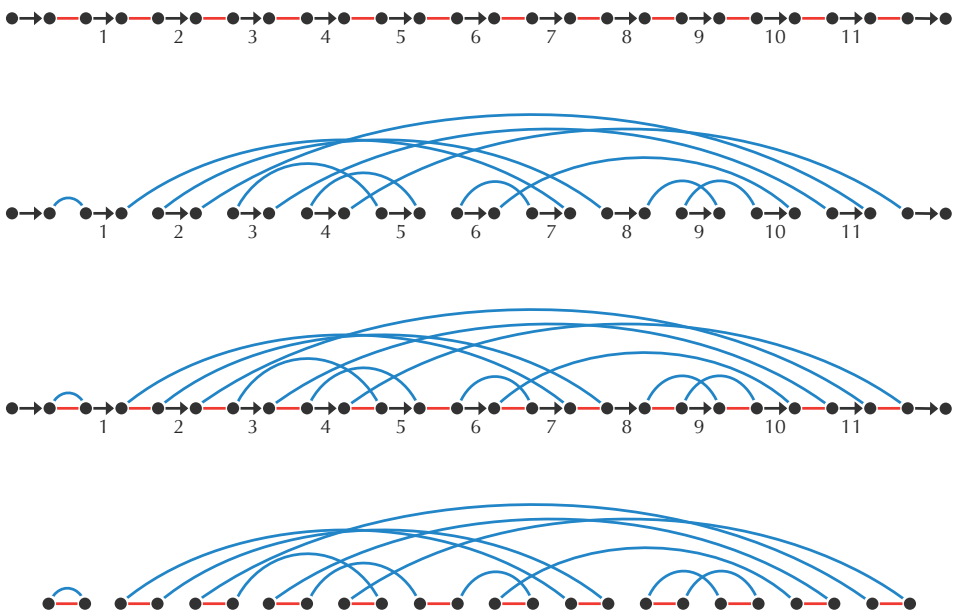


FIGURE 1.26: (1st panel) An alternating path of red and black edges representing the human X chromosome $(+1\ +2\ +3\ +4\ +5\ +6\ +7\ +8\ +9\ +10\ +11)$. (2nd panel) An alternating path of blue and black edges representing the mouse X chromosome $(+1\ -7\ +6\ -10\ +9\ -8\ +2\ -11\ -3\ +5\ +4)$. (3rd panel) The breakpoint graph of the mouse and human X chromosomes is obtained by superimposing red-black and blue-black paths from the first two panels. (4th panel) To highlight the five alternating red-blue cycles in the breakpoint graph, black edges are removed.

human and mouse X chromosomes, $\text{BLOCKS}(P, Q) + 1 - \text{CYCLES}(P, Q)$ is equal to $11 + 1 - 5 = 7$, which we already know to be the reversal distance between the human and mouse X chromosomes.

STOP and Think: Can you modify the proof of the 2-Break Distance Theorem to prove that $d_{\text{rev}}(P, Q) = \text{BLOCKS}(P, Q) + 1 - \text{CYCLES}(P, Q)$ for linear permutations P and Q?

You can verify that the pesky permutation $P = (+2\ +1)$ does not satisfy the condition $d_{\text{rev}}(P, I) = \text{BLOCKS}(P, I) + 1 - \text{CYCLES}(P, I)$, where I is the identity permutation, thus making it unlikely that we will be able to develop a simple

algorithm for the computation of reversal distance.

However, the lower bound $d_{\text{rev}}(P, Q) \geq \text{BLOCKS}(P, Q) + 1 - \text{CYCLES}(P, Q)$ approximates the reversal distance between linear permutations extremely well. This intriguing performance raised the question of whether this bound is close to an exact formula. In 1999, Hannenhalli and Pevzner found this formula by defining two special types of breakpoint graph structures called "hurdles" and "fortresses". Denoting the number of hurdles and fortresses in $\text{BREAKPOINTGRAPH}(P, Q)$ by $\text{HURDLES}(P, Q)$ and $\text{FORTRESSES}(P, Q)$, respectively, they proved that the reversal distance $d_{\text{rev}}(P, Q)$ is given by

$$\text{BLOCKS}(P, Q) + 1 - \text{CYCLES}(P, Q) + \text{HURDLES}(P, Q) + \text{FORTRESSES}(P, Q).$$

Using this formula, they developed a polynomial algorithm for computing $d_{\text{rev}}(P, Q)$.

1.14 Bibliography Notes

Alfred Sturtevant was the first to discover rearrangements while comparing gene orders in fruit flies ([Stu21]). Together with Theodosius Dobzhansky, Sturtevant pioneered the analysis of genome rearrangements in molecular biology, publishing a milestone paper that presented a rearrangement scenario for many fruit fly species ([SD36]). The Random Breakage Model was proposed by [Ohn73], further developed by [NT84], and refuted by [PT03b].

The notion of the breakpoint graph described in this chapter was proposed by [BP96a]. The polynomial algorithm for sorting by reversals was developed by [HP99]. The synteny block construction algorithm presented in this chapter was described by [PT03a]. The 2-break operation was introduced in [YAF05a] under the name of "double cut and join".

The first algorithmic analysis of the Pancake Flipping problem was described by [GP79a]. The first algorithmic analysis of the Burnt Pancake Flipping problem was described by [CB95].

The Multiple Genome Rearrangement problem was addressed by [MRR+08a] and [AP09]. [ZB09] observed that matching duplications may trigger genome rearrangements.

Chapter 2

DCJ-Indel Sorting Revisited

2.1 Abstract

Background: The introduction of the double cut and join operation (DCJ) caused a flurry of research into the study of multichromosomal rearrangements. However, little of this work has incorporated indels (i.e., insertions and deletions of chromosomes and chromosomal intervals) into the calculation of genomic distance functions, with the exception of Braga et al., who provided a linear time algorithm for the problem of DCJ-indel sorting. Although their algorithm only takes linear time, its derivation is lengthy and depends on a large number of possible cases.

Results: We note the simple idea that a deletion of a chromosomal interval can be viewed as a DCJ that creates a new circular chromosome. This framework will allow us to amortize indels as DCJs, which in turn permits the application of the classical breakpoint graph to obtain a simplified indel model that still solves the problem of DCJ-indel sorting in linear time via a more concise formulation that relies on the simpler problem of DCJ sorting. Furthermore, we can extend this result to fully characterize the solution space of DCJ-indel sorting.

Conclusion: Encoding indels as DCJ operations offers a new insight into why the problem of DCJ-indel sorting is not ultimately any more difficult than that of sorting by DCJs alone. There is still room for research in this area, most

notably the problem of sorting when the cost of indels is allowed to vary with respect to the cost of a DCJ and we demand a minimum cost transformation of one genome into another.

Keywords

Genome rearrangements, DCJ, indels, sorting, solution space

Background

In the simplest terms, DNA may mutate in two fundamentally different ways. On the one hand, single-nucleotide polymorphisms alter the base at a single position of the nucleic acid polymer; on the other hand, huge mutations called chromosomal rearrangements can move around, duplicate, insert, or delete huge blocks of DNA, often from one chromosome to another.

Chromosomal rearrangements were first observed by Dobzhansky and Sturtevant in 1938 ([DS38]), but extensive efforts to quantify their study did not take off until the early 1990s. In the last two decades, a number of discrete genomic models have been proposed and studied (see [FLR$^+$09] for an overview of the combinatorics of genome rearrangements).

Having selected a genomic model and a collection of genome operations to consider, the standard algorithmic problem is the computation of the *distance* between two genomes Π and Γ, or the minimum number of allowable operations required to transform Π into Γ; the more difficult problem of *sorting* demands the operations themselves. The first historical example of such a discrete genomic distance is the *prefix reversal distance* for permutations (which model the order of genes along a single linear chromosome), introduced in [Har75] and bounded in [GP79b, HS97, CFM$^+$09]. The computation of prefix reversal distance has been proposed to be NP-Hard (see [BFRnt]).

More recent research has moved past permutations and toward multichromosomal genomic models that incorporate both linear and circular chromosomes.

One of these models, which we will study in this paper, models the chromosomes of a genome with paths and cycles in a graph. For this model, the double cut and join operation (DCJ) was introduced in [YAF05b] and incorporates segment reversals with a number of other operations. Interestingly, a linear time greedy algorithm exists for DCJ sorting two genomes having equal gene content (see [BMS06]).

The incorporation of insertions and deletions of chromosomes and chromosomal intervals (collectively called *indels*) into DCJ distance was discussed in [YF09] and quantified rigorously in [BWS10]. The latter authors provided a linear time algorithm for the associated problem of *DCJ-indel sorting*, which gives a minimum collection of DCJ and indel operations required to transform one genome into another. Yet their argument is case-ridden, and so in this paper, which builds upon[**?**], we wish to provide a much simpler presentation of DCJ-indel sorting that still yields a linear-time solution to the problem.

Main Text

Preliminaries

Say that we are given a perfect matching on $2N$ labeled vertices $\mathcal{V}$, forming a set $\mathcal{G}$ of N edges called *genes*; the vertices of each gene form its *head* and *tail*. We define a *genome* Π as the edge-disjoint union of two matchings. The *genes* of Π, denoted $g(\Pi)$, form a matching on $\mathcal{V}$ such that $g(\Pi) \subseteq \mathcal{G}$; the *adjacencies* of Π, denoted $a(\Pi)$, form a matching on $V(g(\Pi))$. We color the genes of Π black and the adjacencies of Π blue (see Figure 1(a)).

A consequence of these definitions is that Π comprises a disjoint collection of paths and cycles, where each connected component alternates between black genes and blue adjacencies. Each component of Π is called a *chromosome*; paths (cycles) of Π define *linear* (*circular*) chromosomes of Π. The endpoint v of a path in Π is called a *telomere* of Π; v is not incident to an adjacency, and so for clerical purposes, we say that v has the *null adjacency* $\{v, \emptyset\}$. A genome consisting of only circular (linear) chromosomes is called a *circular* (*linear*) *genome*. Note that

Π is circular if and only if the edges of $a(\Pi)$ form a perfect matching on $V(\Pi)$.

Henceforth, we only consider genome pairs $\{\Pi, \Gamma\}$ such that $g(\Pi) \cup g(\Gamma) = \mathcal{G}$. A workhorse data structure encoding the relationship between Π and Γ is the *breakpoint graph* ([BP96b]), denoted by $\mathrm{B}(\Pi, \Gamma)$ and defined as the edge-disjoint union[a] of $a(\Pi)$ and $a(\Gamma)$, where adjacencies of Γ will be colored red (Figure 1 (b)). Observe that $B(\Pi, \Gamma)$ is also a collection of disjoint paths and cycles, which alternate between red and blue edges. The *length* of a connected component of $\mathrm{B}(\Pi, \Gamma)$ is its total number of edges; we consider an isolated vertex in $\mathrm{B}(\Pi, \Gamma)$ to be a path of length 0. The breakpoint graph is also the line graph of the *adjacency graph*, which was first defined in [BMS06] and has also been used in rearrangement studies.

A *double cut and join* operation (DCJ) on Π ([YAF05b]) *uses* one or two adjacencies of Π via one of the following four operations to produce a new genome Π':

1. $\{v, w\}, \{x, y\} \longrightarrow \{v, x\}, \{w, y\}$
2. $\{v, w\}, \{x, \emptyset\} \longrightarrow \{v, x\}, \{w, \emptyset\}$
3. $\{v, \emptyset\}, \{w, \emptyset\} \longrightarrow \{v, w\}$
4. $\{v, w\} \longrightarrow \{v, \emptyset\}, \{w, \emptyset\}$

The DCJ incorporates a wide range of genome rearrangements, as shown in Figure 2.

For the particular case that Π and Γ have the same genes (i.e., $g(\Pi) = g(\Gamma) = \mathcal{G}$), the *DCJ distance* between Π and Γ, written $d_{\mathrm{DCJ}}(\Pi, \Gamma)$, is the minimum number of DCJs required to transform Π into Γ One can easily verify that d_{DCJ} forms a metric on the set of all genomes having gene set $\mathcal{G}$. A closed formula for DCJ distance was derived in [BMS06] and translated into breakpoint graph notation in [TZS09]:

$$d_{\mathrm{DCJ}}(\Pi, \Gamma) = N - c(\Pi, \Gamma) - \frac{p_{\mathrm{even}}(\Pi, \Gamma)}{2} \tag{2.1}$$

Here, $c(\Pi, \Gamma)$ and $p_{\text{even}}(\Pi, \Gamma)$ denote the number of cycles and even-length paths in $B(\Pi, \Gamma)$, respectively.

For the more general case that Π and Γ do not share the same genes, a *deletion* of a chromosomal interval of Π replaces adjacencies $\{v, w\}$ and $\{x, y\}$ (contained in the order (v, w, x, y) along a chromosome of Π) with the adjacency $\{v, y\}$ and removes the path connecting w to x. We also allow deletions of entire chromosomes; however, we must stipulate (following the lead of the authors in [BWS10]) that every vertex removed from Π must belong to $\mathcal{V} - V(\Gamma)$.[b] The *insertion* of a chromosome or chromosomal interval into Π to obtain Π' is defined as the inverse of a corresponding deletion from Π' that yields Π. Note that a consequence of this definition is that we may not insert a gene unless it is contained in $\mathcal{G}$. Insertions and deletions are collectively called *indels*; thus, we define the *DCJ-indel distance* between Π and Γ, written $d_{\text{DCJ}}^{\text{ind}}(\Pi, \Gamma)$, as the minimum number of DCJs and indels required to transform Π into Γ.

Because insertions and deletions are inverse operations, it follows that $d_{\text{DCJ}}^{\text{ind}}(\Pi, \Gamma) = d_{\text{DCJ}}^{\text{ind}}(\Gamma, \Pi)$. However, although $d_{\text{DCJ}}^{\text{ind}}$ is symmetric, unlike d_{DCJ} it does not form a metric, as the triangle inequality does not hold; see [BMRS11] for a more complete discussion.

DCJ-Indel Sorting

Handling Circular Singletons

We begin our discussion of DCJ-indel sorting by defining a *circular singleton* of Π (adapted from[BWS10]) as a chromosome C such that $V(C) \cap V(\Gamma) = \emptyset$. Note that C is defined with respect to Γ as well as Π. Ideally, we could delete (insert) all circular singletons of Π and Γ immediately to simplify the problem of DCJ-indel sorting; fortunately, this is indeed the case, as shown by the following two results.

Proposition: *If Π' is formed by removing a circular singleton C from Π, then $d_{\text{DCJ}}^{\text{ind}}(\Pi', \Gamma) = d_{\text{DCJ}}^{\text{ind}}(\Pi, \Gamma) - 1$. Furthermore, when transforming Π into Γ via a minimum collection of DCJs and indels, no gene belonging to a circular singleton of Π can ever appear in the*

same chromosome as a gene of Γ.

Proof. Any collection of k DCJs and indels transforming Π' into Γ can be supplemented by the deletion of C to yield $k+1$ DCJs and indels transforming Π into Γ; thus, $d_{\text{DCJ}}^{\text{ind}}(\Pi', \Gamma) \geq d_{\text{DCJ}}^{\text{ind}}(\Pi, \Gamma) - 1$.

To obtain the reverse bound, let us view a transformation $\mathbb{T}$ of Π into Γ as a sequence $(\Pi_0, \Pi_1, \ldots, \Pi_n)$ ($n \geq 1$), where $\Pi_0 = \Pi$, $\Pi_n = \Gamma$, and Π_{i+1} is obtained from Π_i as the result of a single DCJ or indel. Consider a sequence $(\Pi'_0, \Pi'_1, \ldots, \Pi'_n)$, where Π'_i is constructed from Π_i by removing the subgraph of Π_i induced by the vertices of C under the stipulation that whenever we remove a path P connecting v to w, we replace adjacencies $\{v, x\}$ and $\{w, y\}$ in Π with $\{x, y\}$ in Π'_i. It is easy to see that $\Pi'_0 = \Pi'$, $\Pi'_n = \Gamma$, and for every i in range, either Π'_{i+1} is the result of a DCJ or indel applied to Π'_i or $\Pi'_{i+1} = \Pi'_i$; thus, $(\Pi'_0, \Pi'_1, \ldots, \Pi'_n)$ encodes a transformation of Π' into Γ using at most n DCJs and indels. Furthermore, one can verify that $\Pi'_{i+1} = \Pi'_i$ only when an adjacency of C is used by a DCJ in $\mathbb{T}$ changing Π_i to Π_{i+1} or when Π_{i+1} is produced from Π_i by a deletion of vertices that all belong to C. At least one such operation must always occur in $\mathbb{T}$; hence, $d_{\text{DCJ}}^{\text{ind}}(\Pi', \Gamma) \leq d_{\text{DCJ}}^{\text{ind}}(\Pi, \Gamma) - 1$.

The proposition's second conclusion follows from the fact that if for some j ($1 \leq j \leq n-1$), a chromosome of Π_j contains a gene g_1 of Π and a gene g_2 of C, then one DCJ was required to combine g_1 and g_2 into the same chromosome, and another will be needed to separate them, yielding two distinct values of i for which $\Pi'_{i+1} = \Pi'_i$. From the first part of the proof, we may conclude that $d_{\text{DCJ}}^{\text{ind}}(\Pi, \Gamma) < n$.

□

Letting $\text{sing}(\Pi, \Gamma)$ denote the total number of circular singletons of Π and Γ, we have an immediate corollary.

Corollary: *The DCJ-indel distance is given by the following:*

$$d_{\text{DCJ}}^{\text{ind}}(\Pi, \Gamma) = \text{sing}(\Pi, \Gamma) + d_{\text{DCJ}}^{\text{ind}}(\Pi^0, \Gamma^0) \tag{2.2}$$

where Π^0 *(*Γ^0*) is formed by removing all circular singletons from* Π *(*Γ*).*

With respect to DCJ-indel sorting, Corollary 1 allows us to assume without loss of generality that Π and Γ do not contain any circular singletons.

We next make an observation taken from[MRR+08b], which is that the deletion of a chromosomal interval of Π connecting w to x may be viewed as a DCJ: $\{v,w\},\{x,y\} \rightarrow \{v,y\},\{w,x\}$; this operation produces a circular chromosome containing w and x that is scheduled for removal, including the case that v or y equals $\emptyset$ (the deletion of an entire linear chromosome is handled by $u = x = \emptyset$); see Figure 3. Because insertions are the inverses of deletions, we would like to conclude that indels may be placed in a one-to-one correspondence with the removal of circular chromosomes. Ironically, the apparent exception to this proposed rule is the deletion of an entire circular chromosome.

Yet if a deleted circular chromosome C is not produced as the result of a DCJ, then C must be a circular singleton of Π in order to be deleted. Otherwise, C has been produced as the result of a DCJ applied to a chromosomal interval; by the method we just described, we can encode the deletion in this DCJ unless it also creates another circular chromosome C' that is scheduled for removal. However, this sequence of operations cannot arise in a minimum collection of DCJs and indels transforming Π into Γ, as we could simply delete the original chromosome(s) from which C and C' were produced by the DCJ in question, thus using at most two operations instead of three.

Toward a New Model of Indels

We will follow the observation made in[MRR+08b] that the actual removal of deleted chromosomes can occur as a final step in the transformation of Π into Γ. As a result, we may view the transformation of Π into Γ as composed of three steps: inserting chromosomes into Π to yield a new genome Π' with $g(\Pi') = \mathcal{G}$; applying a sequence of DCJs to produce a genome Γ' having the same genes as Π'; and finally, deleting chromosomes from Γ' to produce Γ. Note that we can equivalently view the first step as the deletion of chromosomes from Π' to obtain Π. Combining this observation with our correspondence between indels and circular chromosomes above, we may introduce the following framework.

Define a *completion* of Π as a genome Π' having $g(\Pi') = \mathcal{G}$ and for which $a(\Pi')$ is composed of $a(\Pi)$ together with a perfect matching on $V(\Pi') - V(\Pi)$. We call the adjacencies of $a(\Pi') - a(\Pi)$ *new*. Note that the chromosomes of Π embed as chromosomes of Π' and that the components of $\Pi' - \Pi$ form cycles because the new adjacencies of Π' induce a perfect matching on $V(\Pi') - V(\Pi)$; we may now without ambiguity call these circular chromosomes of Π' the *indels* of Π'. A *completion* of a pair of genomes (Π, Γ) is simply a pair (Π', Γ') for which Π' and Γ' are completions of Π and Γ, respectively. The above discussion implies that for any minimum cost transformation of Π into Γ, the indels of Π' correspond bijectively to DCJ operations, so that we will amortize each unit indel cost by that of a DCJ operation. This amortization yields the following equation for DCJ-indel distance:

$$d_{\text{DCJ}}^{\text{ind}}(\Pi, \Gamma) = \min_{(\Pi', \Gamma')} \left\{ d_{\text{DCJ}}(\Pi', \Gamma') \right\} \tag{2.3}$$

where the minimum is taken over all completions of (Π, Γ). A completion (Π^*, Γ^*) is *optimal* if it attains the minimum in (2.3). Applying the closed form equation for the DCJ distance in (2.1) to immediately produces the following result.

Theorem. *The DCJ-indel distance is given by the following equation:*

$$d_{\text{DCJ}}^{\text{ind}}(\Pi, \Gamma) = N - \max_{(\Pi', \Gamma')} \left\{ c(\Pi', \Gamma') + \frac{p_{\text{even}}(\Pi', \Gamma')}{2} \right\} \tag{2.4}$$

where the maximum is taken over all completions of (Π, Γ).

Constructing an Optimal Completion

In light of Theorem 2.1, we have reduced DCJ-indel sorting to the problem of constructing indels intelligently to maximize a weighted sum of breakpoint graph components. Once we have produced an optimal completion (Π^*, Γ^*), we can simply invoke the $O(N)$-time sorting algorithm described in [BMS06] to transform Π^* into Γ^* via a minimum collection of DCJs.

Our goal is to construct (Π^*, Γ^*) by direct analysis of $\mathrm{B}(\Pi, \Gamma)$. Because Π and Γ do not necessarily share the same genes, $\mathrm{B}(\Pi, \Gamma)$ may contain path endpoints that are not telomeres. Accordingly, we define a vertex v to be *ß-open* (*fl-open*) if $v \notin \Pi$ ($v \notin \Gamma$). In other words, v must be matched to some other π-open vertex when constructing the indels of Π^*.[c] The paths of $\mathrm{B}(\Pi, \Gamma)$ are therefore classified according to their endpoints: a *ß-path* (*fl-path*) ends in one π-open (γ-open) vertex and one telomere (of either Π or Γ); a $\{\pi, \gamma\}$*-path* ends in a π-open vertex and a γ-open vertex (such a path must have even length at least 2); a $\{\pi, \pi\}$*-path* ($\{\gamma, \gamma\}$*-path*) ends in two π-open (γ-open) vertices and must therefore have odd length. We should also provide statistics for counting these different components. Define $p^{\pi,\gamma}$ as the number of $\{\pi, \gamma\}$-paths in $\mathrm{B}(\Pi, \Gamma)$; p^{π}_{even} as the number of even-length π-paths in $\mathrm{B}(\Pi, \Gamma)$; and p^{0}_{even} as the number of even-length paths in $\mathrm{B}(\Pi, \Gamma)$ containing no open vertices (i.e., ending in two telomeres). Similar statistics counting odd-length paths can be defined analogously. We have dropped the genomes $\{\Pi, \Gamma\}$ from these statistics for the sake of simplicity; all component statistics will be taken with respect to $\mathrm{B}(\Pi, \Gamma)$ unless otherwise noted.

We first present a proposition regarding the parity of the paths of $\mathrm{B}(\Pi, \Gamma)$.

Proposition: *The component statistics of* $\mathrm{B}(\Pi, \Gamma)$ *satisfy the following condition:*

$$p^{\pi,\gamma} \equiv |p^{\pi}_{\text{odd}} - p^{\pi}_{\text{even}}| \equiv \left|p^{\gamma}_{\text{odd}} - p^{\gamma}_{\text{even}}\right| \mod 2 \tag{2.5}$$

Proof. The total number of π-open vertices is equal to $V(\Pi') - V(\Pi)$ and must therefore be even. Of course, the same is the case for γ-open vertices, and counting π-open and γ-open vertices over the connected components of $\mathrm{B}(\Pi, \Gamma)$ thus produces the following equivalences:

$$p^{\pi}_{\text{odd}} + p^{\pi}_{\text{even}} + p^{\pi,\gamma} \equiv 0 \mod 2 \tag{2.6}$$

$$p^{\gamma}_{\text{odd}} + p^{\gamma}_{\text{even}} + p^{\pi,\gamma} \equiv 0 \mod 2 \tag{2.7}$$

Adding $p^{\pi,\gamma}$ to both sides of (2.6) and (2.7) gives the following:

$$p^{\pi,\gamma} \equiv (p^{\pi}_{\text{odd}} + p^{\pi}_{\text{even}}) \equiv \left(p^{\gamma}_{\text{odd}} + p^{\gamma}_{\text{even}}\right) \mod 2 \tag{2.8}$$

The equivalence of (2.5) and (2.8) is an arithmetical fact. □

We next establish two necessary conditions on optimal completions by culling the set of possible adjacencies of any such completion. Our general strategy is to consider the addition of a new adjacency $\{v, w\}$ to a completion Π' as *linking* the component(s) of $B(\Pi, \Gamma)$ whose endpoints are the (π-open) vertices v and w. Our first result states that we must always link the endpoints of any $\{\pi, \pi\}$-path to each other.

Lemma: *If* (Π^*, Γ^*) *is an optimal completion of* (Π, Γ)*, then every* $\{\pi, \pi\}$*-path* ($\{\gamma, \gamma\}$*-path) of length* $2k - 1$ *in* $B(\Pi, \Gamma)$ ($k \geq 1$) *embeds into a cycle of length* $2k$ *in* $B(\Pi^*, \Gamma^*)$.

Proof. Let P be a path of length $2k - 1$ connecting π-open vertices v and w in $B(\Pi, \Gamma)$. Our claim is that we must link v and w in $B(\Pi^*, \Gamma^*)$. Suppose for the sake of contradiction that we have a completion (Π', Γ') such that P does not embed into a cycle of length $2k$ in $B(\Pi', \Gamma')$; in this case, we must have adjacencies $\{v, x\}$ and $\{w, y\}$ in $a(\Pi')$, where all four vertices are distinct.

Consider the completion Π'' that is identical to Π' except that $\{v, x\}$ and $\{w, y\}$ are replaced by $\{v, w\}$ and $\{x, y\}$. In $B(\Pi'', \Gamma')$, we have closed P into a cycle of length $2k$, and at the same time, we have changed neither the parity nor the linearity/circularity of the component containing x and y. Because we have increased the number of breakpoint graph cycles by 1 without changing the total number of paths, it follows from (2.1) that $d_{\text{DCJ}}(\Pi'', \Gamma') = d_{\text{DCJ}}(\Pi', \Gamma') - 1$, and so (Π', Γ') cannot be optimal. □

Having dealt with $\{\pi, \pi\}$- and $\{\gamma, \gamma\}$-paths of $B(\Pi, \Gamma)$, any remaining component of $B(\Pi^*, \Gamma^*)$ must be either a *j-bracelet*, which is a cycle linking j $\{\pi, \gamma\}$-paths (where $j \geq 2$ and j is even), or a *k-chain*, in which two π-paths or two γ-paths are linked via an intermediate number of $\{\pi, \gamma\}$-paths to form a path containing k components from $B(\Pi, \Gamma)$ ($k \geq 2$). Note that when k is even, a

k-chain C must contain either two π-paths or two γ-paths, and when k is odd, C must contain one π-path and one γ-path.

For the sake of simplicity, we will represent a j-bracelet by $(P_1 : P_2 : \cdots : P_j)$ and a k-chain by $[P_1 : P_2 : \cdots : P_k]$, where every P_i is linked to P_{i+1}, and in the case of a j-bracelet, P_1 is linked to P_j. Because we wish to maximize a weighted sum of breakpoint graph components, we might guess that we should look for many short bracelets and chains. Indeed, the length of a bracelet or chain in $\mathrm{B}(\Pi^*, \Gamma^*)$ is heavily restricted by the following lemma.

Lemma: *If (Π^*, Γ^*) is an optimal completion, then a component C^* of $\mathrm{B}(\Pi^*, \Gamma^*)$ can only contain two or more $\{\pi, \gamma\}$-paths if C^* is a 2-bracelet.*

Proof. Again, say for the sake of contradiction that we have an optimal completion (Π', Γ') for which a component C' of $\mathrm{B}(\Pi', \Gamma')$ contains two or more $\{\pi, \gamma\}$-paths. If C' is not a 2-bracelet, then it must contain $\{\pi, \gamma\}$-paths P_1 and P_2 that are linked by precisely one new adjacency. Say that P_1 joins π-open vertex v to γ-open vertex w and that P_2 joins π-open vertex x to γ-open vertex y. To meet the assumption that P_1 and P_2 are linked by precisely one adjacency, suppose that $\{v, x\} \in a(\Pi')$ but $\{w, y\} \notin a(\Gamma')$, where instead $\{w, w'\}$ and $\{y, y'\}$ are in $a(\Gamma')$. Replacing these two adjacencies with $\{w, y\}$ and $\{w', y'\}$ defines a new completion Γ'' for which $\mathrm{B}(\Pi', \Gamma'')$ contains $(P_1 : P_2)$. Viewed as an operation on $\mathrm{B}(\Pi', \Gamma')$ to yield $\mathrm{B}(\Pi', \Gamma'')$, we have two cases.

First, if C' was a bracelet, then we have formed two new bracelets from C', one of which is $(P_1 : P_2)$. Otherwise, C' was a chain, in which case we have formed a chain (of the same parity) in addition to $(P_1 : P_2)$. In either case, we may check that $d_{\mathrm{DCJ}}(\Pi', \Gamma'') < d_{\mathrm{DCJ}}(\Pi', \Gamma')$, and so (Π', Γ') cannot be optimal. □

Following Lemma 2, we may only have 2-bracelets, 2-chains, and 3-chains in $\mathrm{B}(\Pi^*, \Gamma^*)$. After a simple result about the parity of 2-chain components, we will be ready to state our main result on DCJ-indel sorting.

Proposition: *The breakpoint graph of an optimal completion cannot have one 2-chain joining two odd π-paths and another 2-chain joining two even π-paths. The same holds for γ-paths.*

Proof. Once again, proceed by contradiction and assume that (Π', Γ') is an optimal completion with such 2-chains $[P_1 : P_2]$ and $[P_3 : P_4]$. Replacing these 2-chains with $[P_1 : P_3]$ and $[P_2 : P_4]$ replaces two odd paths in $B(\Pi'', \Gamma'')$ with two even paths; hence, (Π', Γ') cannot be optimal. □

Theorem. *Algorithm* **??**, *given below, defines an $O(N)$ time algorithm for DCJ-indel sorting. For pairs $\{\Pi, \Gamma\}$ having $\operatorname{sing}(\Pi, \Gamma) = 0$, the DCJ-indel distance is given by the following equation:*

$$d_{DCJ}^{ind}(\Pi, \Gamma) = N - \left[\left(c + p^{\pi,\pi} + p^{\gamma,\gamma} + \left\lfloor \frac{p^{\pi,\gamma}}{2} \right\rfloor \right) + \frac{1}{2} \left(p^0_{even} + \min\{p^{\pi}_{odd}, p^{\pi}_{even}\} + \min\{p^{\gamma}_{odd}, p^{\gamma}_{even}\} + \delta \right) \right] \quad (2.9)$$

Here, $\delta = 1$ only if $p^{\pi,\gamma}$ is odd and either $p^{\pi}_{odd} > p^{\pi}_{even}, p^{\gamma}_{odd} > p^{\gamma}_{even}$ or $p^{\pi}_{odd} < p^{\pi}_{even}, p^{\gamma}_{odd} < p^{\gamma}_{even}$; otherwise, $\delta = 0$.

Proof. We aim to construct an optimal completion (Π^*, Γ^*) having

$$c(\Pi^*, \Gamma^*) = c + p^{\pi,\pi} + p^{\gamma,\gamma} + \left\lfloor \frac{p^{\pi,\gamma}}{2} \right\rfloor \quad (2.10)$$

$$p_{even}(\Pi^*, \Gamma^*) = p^0_{even} + \min\{p^{\pi}_{odd}, p^{\pi}_{even}\} + \min\{p^{\gamma}_{odd}, p^{\gamma}_{even}\} + \delta \quad (2.11)$$

First, we count the cycles of $B(\Pi^*, \Gamma^*)$. By Lemma 1, every $\{\pi, \pi\}$-path or $\{\gamma, \gamma\}$-path of $B(\Pi, \Gamma)$ must be closed into a cycle by adding a single new adjacency (Step 1 of Algorithm **??**). We now claim that there exists an optimal completion containing $\left\lfloor \frac{p^{\pi,\gamma}}{2} \right\rfloor$ 2-bracelets. Note that we may always replace 3-chains $[P_1 : P_2 : P_3]$ and $[P_4 : P_5 : P_6]$ (where P_1 and P_4 are π-paths) with $[P_1 : P_4]$, $(P_2 : P_5)$, and $[P_3 : P_6]$, without increasing the DCJ distance of the associated completion because we have obtained a cycle from two paths. This argument implies Step 2 of Algorithm **??** and produces the value of $c(\Pi^*, \Gamma^*)$ stated above.

As for the even paths of $B(\Pi^*, \Gamma^*)$, let us operate under the assumption that $p^{\pi,\gamma}$ is odd. Then after forming a maximal collection of 2-bracelets, we will be left with one additional $\{\pi, \gamma\}$-path P. We claim that (Π^*, Γ^*) will be optimal if we link as many π-paths (γ-paths) of opposite parity as possible. On the one hand, Proposition 3 states that we cannot have 2-chains $[P_1 : P_2]$ and $[P_3 : P_4]$,

where P_1 and P_2 are even π-paths and P_3 and P_4 are odd π-paths. On the other hand, say that we have a 2-chain $[P_1 : P_2]$ and a 3-chain $[P_3 : P : P_4]$, where without loss of generality we assume that P_1 and P_2 are odd π-paths, P_3 is an even π-path, and P_4 is a γ-path. Replacing these chains with the chains $[P_1 : P_3]$ and $[P_2 : P : P_4]$ does not change the number of paths of even length in $\mathrm{B}(\Pi^*, \Gamma^*)$, implying Step 3 of Algorithm **??**.

Thus, all remaining π-paths must have the same parity, as must all the γ-paths; thus, we may choose any π-path and γ-path to link to P (Step 4 of Algorithm **??**) and form a 3-chain. The length of this 3-chain may be even ($\delta = 1$) or odd ($\delta = 0$) depending on whether the length of its π-path and γ-path have equal parity or not. All remaining paths must therefore be 2-chains linking pairs of π-paths or pairs of γ-paths (Step 5 of Algorithm **??**).

If instead $p^{\pi,\gamma}$ is even, then $\delta = 0$, and the argument for constructing an optimal completion proceeds similarly, except that no $\{\pi, \gamma\}$-paths will remain after forming a maximal collection of 2-bracelets, eliminating the need for Step 4. □

The Solution Space of DCJ-Indel Sorting

The problem of DCJ sorting is well understood, its solution space having been described in [BS10]. Thus, by Theorem 2.1, to identify the solution space of DCJ-indel sorting (an open problem), we simply need to enumerate the construction of indels of an optimal completion. We mentioned this enumeration in [**?**], but here we will explore the details of the calculation.

Handling Circular Singletons

By Proposition 1, we may consider the circular singletons of Π and Γ independently of other chromosomes; for that matter, because insertions and deletions are defined symmetrically, we may assume that Π contains k chromosomes and that Γ is the empty genome. Then by Corollary 1 and the trivial fact that any DCJ applied to Π changes the total number of chromosomes of

Given genomes (Π, Γ), the following algorithm constructs an optimal completion (Π^*, Γ^*) in $O(N)$ time.

0. Remove all circular singletons from Π and Γ.
1. Close every $\{\pi, \pi\}$-path ($\{\gamma, \gamma\}$-path) into a cycle by adding a single new adjacency to Π^* (Γ^*).
2. Form a maximum set of 2-bracelets.
3. Form a maximum set of even 2-chains by linking pairs of π-paths (γ-paths) having opposite parity.
4. If $p^{\pi,\gamma}$ is odd, then link the remaining $\{\pi, \gamma\}$-path with any remaining π-path and γ-path to form a 3-chain.
5. Arbitrarily link pairs of remaining π-paths, all of which have the same parity, to form 2-chains. Do the same for remaining γ-paths.

Π by at most 1 (see [YAF05b]), we may obtain Γ from Π in k steps if and only if we perform j successive DCJs ($0 \leq j < k$), each of which fuses two circular chromosomes into one, followed by applying $k - j$ chromosome deletions.

Assuming that k is relatively small, the enumeration of all such transformations of Π into Γ poses a tedious but straightforward task, as a fusion of two circular chromosomes corresponds to a DCJ using two adjacencies from different chromosomes.

Genomes Lacking Circular Singletons

Having handled circular singletons, we may assume that $\text{sing}(\Pi, \Gamma) = 0$. Fortunately, the lemmas presented before Theorem 2.1 have greatly reduced the collection of possible optimal completions, which we now continue to pare down.

Proposition: *Every π-path (γ-path) embedding into a 3-chain of an optimal completion must have the same parity.*

Proof. Say for the sake of contradiction that we have an optimal completion (Π', Γ') such that $B(\Pi', \Gamma')$ contains 3-chains $[P_1 : P_2 : P_3]$ and $[P_4 : P_5 : P_6]$, where P_1 and P_4 are π-paths of opposite parity. Consider the completion (Π'', Γ''), which is defined by rejoining adjacencies of (Π', Γ') to form $[P_1 : P_4]$, $(P_2 : P_5)$, and $[P_3 : P_6]$ in $B(\Pi'', \Gamma'')$. The 2-chain $[P_1 : P_4]$ must have even length, and $(P_2 : P_5)$ is a cycle; thus, $d_{DCJ}(\Pi'', \Gamma'') < d_{DCJ}(\Pi', \Gamma')$, and so (Π', Γ') cannot be optimal. □

Proposition: *If $p^{\pi,\gamma}$ is even, then the breakpoint graph of an optimal completion must contain a maximum set of even-length* 2*-chains.*

Proof. We proceed by contradiction. Say that (Π', Γ') is an optimal completion for which an odd π-path P_1 and an even π-path P_2 are contained in different components of $B(\Pi', \Gamma')$, neither of which is an even 2-chain. By Propositions 3 and 4, we may assume that P_1 and P_2 embed into an odd-length 2-chain $[P_1 : P_5]$ and a 3-chain $[P_2 : P_3 : P_4]$. Because $p^{\pi,\gamma}$ is even, we must have at least one additional 3-chain $[P_6 : P_7 : P_8]$, where (again by Proposition 4) P_6 is an even-length π-path, and the γ-paths P_4 and P_8 have the same parity. With these assumptions in hand, we may rejoin adjacencies to form the four components $[P_1 : P_2]$ (even), $[P_5 : P_6]$ (even), $(P_3 : P_7)$, and $[P_4 : P_8]$ (odd), producing a cycle and two even 2-chains from our original three paths. Hence, by (2.4), (Π', Γ') cannot be optimal. □

We are now ready to fully describe the collection of optimal completions when $p^{\pi,\gamma}$ is even. To construct an optimal completion, after closing each $\{\pi, \pi\}$-path and $\{\gamma, \gamma\}$-path, which can be done uniquely, we must form a maximum collection of even 2-chains by Proposition 5. Recall that our aim is to maximize the statistic $c(\Pi^*, \Gamma^*) + \frac{p_{even}(\Pi^*, \Gamma^*)}{2}$, and consider the following two subcases.

Case 1: $p^{\pi,\gamma}$ is even, $p^{\pi}_{odd} \leq p^{\pi}_{even}$, and $p^{\gamma}_{odd} \geq p^{\gamma}_{even}$. First, a maximal collection of even-length 2-chains will total $p^{\pi}_{odd} + p^{\gamma}_{even}$ components, which requires simply choosing p^{π}_{odd} even-length π-paths, then matching them to odd-length π-paths. This can be achieved in A_1 ways, where

$$A_1 = \binom{p^{\pi}_{even}}{p^{\pi}_{odd}} \cdot (p^{\pi}_{odd})! = P\left(p^{\pi}_{even}, p^{\pi}_{odd}\right) \tag{2.12}$$

Next, we follow the same method for forming even-length 2-chains by linking γ-paths of opposite parity, yielding B_1 total matchings:

$$B_1 = \mathrm{P}\left(p^{\gamma}_{\mathrm{odd}}, p^{\gamma}_{\mathrm{even}}\right) \tag{2.13}$$

Here, we use $\mathrm{P}(n,k)$ to denote the partial permutation statistic: $\mathrm{P}(n,k) = \frac{n!}{(n-k)!}$. We will be left with $p^{\pi}_{\mathrm{even}} - p^{\pi}_{\mathrm{odd}}$ even π-paths and $p^{\gamma}_{\mathrm{odd}} - p^{\gamma}_{\mathrm{even}}$ odd γ-paths. It is impossible to create any more even-length paths in $\mathrm{B}(\Pi^*, \Gamma^*)$, and so we must form a maximum collection of $\frac{p^{\pi,\gamma}}{2}$ 2-bracelets from the $\{\pi,\gamma\}$-paths:

$$C_1 = \left(p^{\pi,\gamma}_{\mathrm{even}} - 1\right)!! = \left(p^{\pi,\gamma}_{\mathrm{even}} - 1\right)\left(p^{\pi,\gamma}_{\mathrm{even}} - 3\right)\cdots(5)(3)(1) \tag{2.14}$$

Note the definition of double factorial. Finally, we link arbitrary remaining π-paths to each other and arbitrary remaining γ-paths to each other:

$$D_1 = \left(p^{\pi}_{\mathrm{even}} - p^{\pi}_{\mathrm{odd}} - 1\right)!! \cdot \left(p^{\gamma}_{\mathrm{odd}} - p^{\gamma}_{\mathrm{even}} - 1\right)!! \tag{2.15}$$

By the independence of these four procedures, the total number of optimal completions is simply given by the product $A_1 \cdot B_1 \cdot C_1 \cdot D_1$.

Case 2: $p^{\pi,\gamma}$ is even, $p^{\pi}_{\mathrm{odd}} > p^{\pi}_{\mathrm{even}}$, and $p^{\gamma}_{\mathrm{odd}} > p^{\gamma}_{\mathrm{even}}$. In this case, we first form a maximum set of 2-chains:

$$A_2 = \mathrm{P}\left(p^{\pi}_{\mathrm{odd}}, p^{\pi}_{\mathrm{even}}\right) \cdot \mathrm{P}\left(p^{\gamma}_{\mathrm{odd}}, p^{\gamma}_{\mathrm{even}}\right) \tag{2.16}$$

We then have $p^{\pi}_{\mathrm{odd}} - p^{\pi}_{\mathrm{even}}$ odd-length π-paths and $p^{\gamma}_{\mathrm{odd}} - p^{\gamma}_{\mathrm{even}}$ odd-length γ-paths remaining. Assume without loss of generality that $p^{\pi}_{\mathrm{odd}} - p^{\pi}_{\mathrm{even}} \geq p^{\gamma}_{\mathrm{odd}} - p^{\gamma}_{\mathrm{even}}$, and set $m = \min\left\{p^{\pi,\gamma}, p^{\gamma}_{\mathrm{odd}} - p^{\gamma}_{\mathrm{even}}\right\}$. We may attain the formula in (2.9) if and only if we form $2j$ even-length 3-chains for some integer j satisfying $0 \leq j \leq \frac{m}{2}$, then create $\frac{p^{\pi,\gamma}}{2} - j$ total 2-bracelets from the remaining $\{\pi,\gamma\}$-paths. Any remaining odd-length π-paths (γ-paths) must then be linked to each other to form (odd-length) 2-chains in $\mathrm{B}(\Pi^*, \Gamma^*)$. The number of such possibilities can be counted by the following statistic B_2:

$$B_2 = \sum_{j=0}^{m/2} \binom{p^{\pi}_{\mathrm{odd}} - p^{\pi}_{\mathrm{even}}}{2j} \binom{p^{\gamma}_{\mathrm{odd}} - p^{\gamma}_{\mathrm{even}}}{2j} \binom{p^{\pi,\gamma}}{2j} [(2j)!]^2 \cdot \left(p^{\pi}_{\mathrm{odd}} - p^{\pi}_{\mathrm{even}} - 2j - 1\right)!! \left(p^{\gamma}_{\mathrm{odd}} - p^{\gamma}_{\mathrm{even}} - 2j - 1\right)!! \left(p^{\pi,\gamma} - 2j - 1\right)!! \tag{2.17}$$

Again, the two statistics can be carried out independently, yielding $A_2 \cdot B_2$ total optimal completions.

In both of the first two cases, reversing the inequalities in the first two cases will lead to analogous arguments. For the next two cases, suppose instead $p^{\pi,\gamma}$ is odd, and select a single $\{\pi, \gamma\}$-path P that must belong to a 3-chain.

Case 3: $p^{\pi,\gamma}$ is odd, $p^{\pi}_{\text{odd}} < p^{\pi}_{\text{even}}$, and $p^{\gamma}_{\text{odd}} > p^{\gamma}_{\text{even}}$. Note that there are $A_3 = p^{\pi,\gamma}$ total ways to select our $\{\pi, \gamma\}$-path P. Of the four possibilities for the parity of the paths to which P may be linked, one may wish to verify that the only way we cannot attain the maximum in (2.9) is if we link P to an odd-length π-path and an even-length γ-path. Thus, we arrive at three mutually exclusive subcases.

In our first subcase, P is linked to an even-length π-path and an odd-length γ-path:

$$B_3 = p^{\pi}_{\text{even}} \cdot p^{\gamma}_{\text{odd}} \tag{2.18}$$

We now have an even number of $\{\pi, \gamma\}$-paths remaining and have reduced our problem to a simpler one that falls under Case 1 above, from which we may obtain some number C_3 of optimal completions.

In the second subcase, we join P to an odd-length π-path and an odd-length γ-path. First, select two such paths:

$$D_3 = p^{\pi}_{\text{odd}} \cdot p^{\gamma}_{\text{odd}} \tag{2.19}$$

Again we have reduced the problem to a subproblem falling under Case 1, from which we may obtain E_3 total optimal completions. In our third and final subcase, we join P to an even π-path and an even γ-path:

$$F_3 = p^{\pi}_{\text{even}} \cdot p^{\gamma}_{\text{even}} \tag{2.20}$$

Say that applying Case 1 to the resulting subcase in which $p^{\pi,\gamma}$ is even yields G_3 total optimal completions. Then by independence, the total number of optimal completions over all three subcases will be given by $A_3 \cdot (B_3 \cdot C_3 + D_3 \cdot E_3 + F_3 \cdot G_3)$.

Case 4: $p^{\pi,\gamma}$ is odd, $p^{\pi}_{\text{odd}} > p^{\pi}_{\text{even}}$, and $p^{\gamma}_{\text{odd}} > p^{\gamma}_{\text{even}}$. Having selected P from the $A_3 = p^{\pi,\gamma}$ total $\{\pi, \gamma\}$-paths, one may verify that the only way

we can achieve the maximum in (2.9) is by linking P to an odd-length π-path and an odd-length γ-path, of which there are $B_4 = p^{\pi}_{\text{odd}} \cdot p^{\gamma}_{\text{odd}}$ total choices. We have therefore reduced our problem of linking components of $\mathrm{B}(\Pi, \Gamma)$ to a smaller problem, falling under Case 2, for which $p^{\pi,\gamma}$ is even. If there are C_4 total solutions to this smaller problem, then the number of optimal completions is given by $A_4 \cdot B_4 \cdot C_4$.

As in the first two cases, reversing the inequalities defining Cases 3 and 4 will result in analogous arguments.

Conclusions

In this paper, we have demonstrated how the problem of DCJ-indel sorting, first solved in[BWS10], can equally be handled via direct inspection of the breakpoint graph. Unfortunately, we still do not see a natural correspondence between the two approaches to DCJ-indel sorting, which appear to be at odds because their definitions of indels are equivalent but fundamentally different.

Furthermore, modeling an indel as a circular chromosome resulting from a DCJ has uncovered the solution space of DCJ-indel sorting, thus resolving an open problem. We wonder if other operations could be adapted to a similar model to yield a straightforward calculation of other genomic distances involving indels. We are also curious whether this model applies to the case of finding a minimum-cost transformation of one genome into another as we vary the parameter associated with the (constant) indel cost.

Competing Interests

The author declares no competing interests.

Authors' Contributions

PC has contributed all intellectual content for this paper.

Acknowledgements

The author would like to acknowledge the support of Pavel Pevzner (UC San Diego Department of Computer Science), who offered guidance during the drafting of the manuscript.

Endnotes

(a) This definition allows $B(\Pi, \Gamma)$ to contain cycles of length 2. (b) In particular, this requirement bars the trivial transformation of Π into Γ in which every chromosome from Π is deleted, and then all the chromosomes of Γ inserted. (c) Note that v cannot be simultaneously π- and γ-open, although it may be a telomere of both Π and Γ or be π-open and a telomere of Γ (in both cases, v is an isolated vertex of $B(\Pi, \Gamma)$, i.e., a path of length 0).

Figures

Figure 1 - Two Genomes and their Breakpoint Graph

(a) Genomes Π and Γ on a collection of 12 genes. We use “h" and “t" to denote the head and tail of a gene. Π is drawn with blue adjacencies, and Γ is drawn with red adjacencies. (b) The breakpoint graph of Π and Γ. We have labeled the endpoint v of a path with π if v is π-open, with γ if v is γ-open, and with $\emptyset$ if v is a telomere of at least one genome.

Figure 2 - The Collection of All Possible DCJ Operations

The DCJ incorporates many operations, depending on the structure of the chromosomes involved and whether the adjacencies used belong to the same chromosome. (a) Operation 1 in the definition of the DCJ incorporates linear internal translocations, reversals, circular fusions/fissions, the excision of a circular chromosome from a linear chromosome, and the integration of a

circular chromosome into a linear chromosome. (b) Operation 2 incorporates telomeric translocations, affix reversals (which involve the telomere of a linear chromosome), and the fission of a linear chromosome into a circular and linear chromosome (together with its inverse). (c) Operations 3 and 4 include linear fusions/fissions as well as the linearization/circularization of a single chromosome.

Figure 3 - Encoding the Deletion of a Chromosomal Interval as a DCJ

The deletion of a chromosomal interval connecting w to x can be encoded by a DCJ that turns the interval connecting w to x into a circular chromosome. The four possible deletions of a chromosomal interval are shown in the above figure; this correspondence holds even when the interval in question is taken to be an entire linear chromosome.

Additional Files

Additional file 1 — Genomes, Breakpoint Graph.pdf

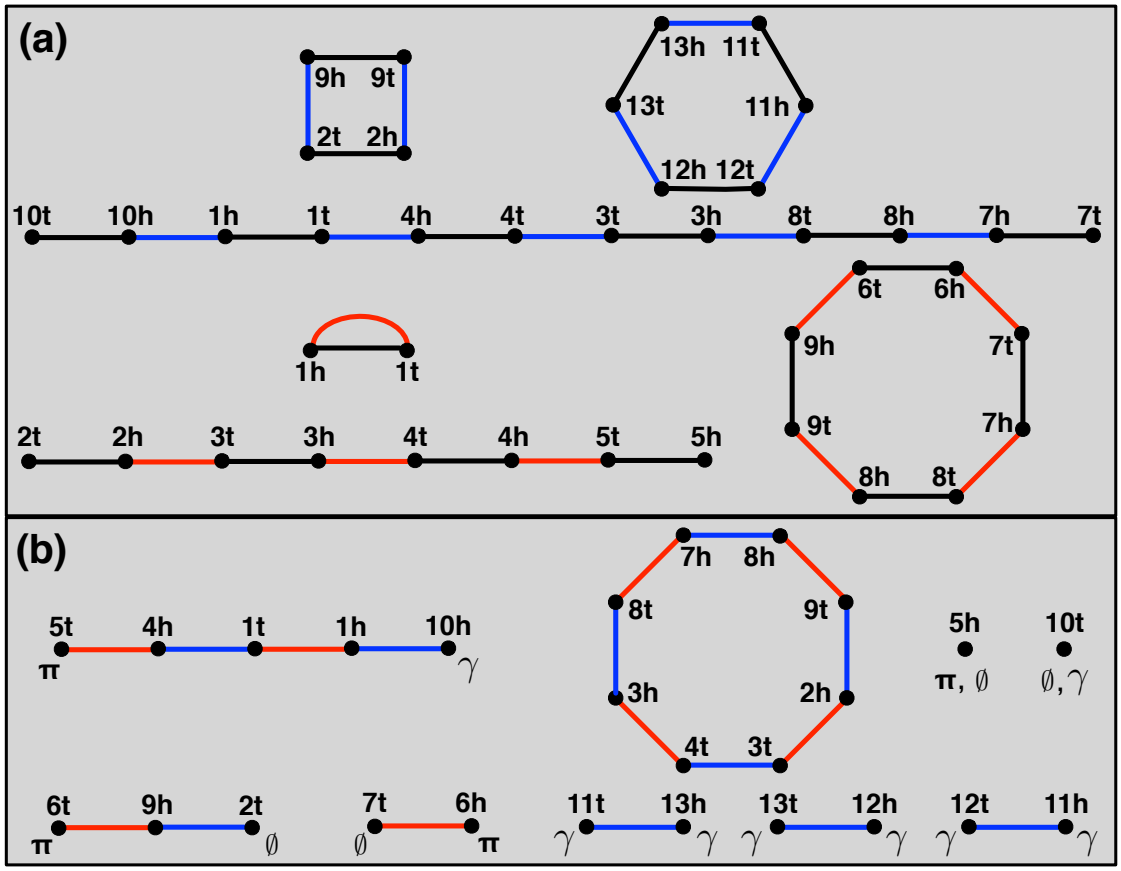
PDF file corresponding to Figure 1.

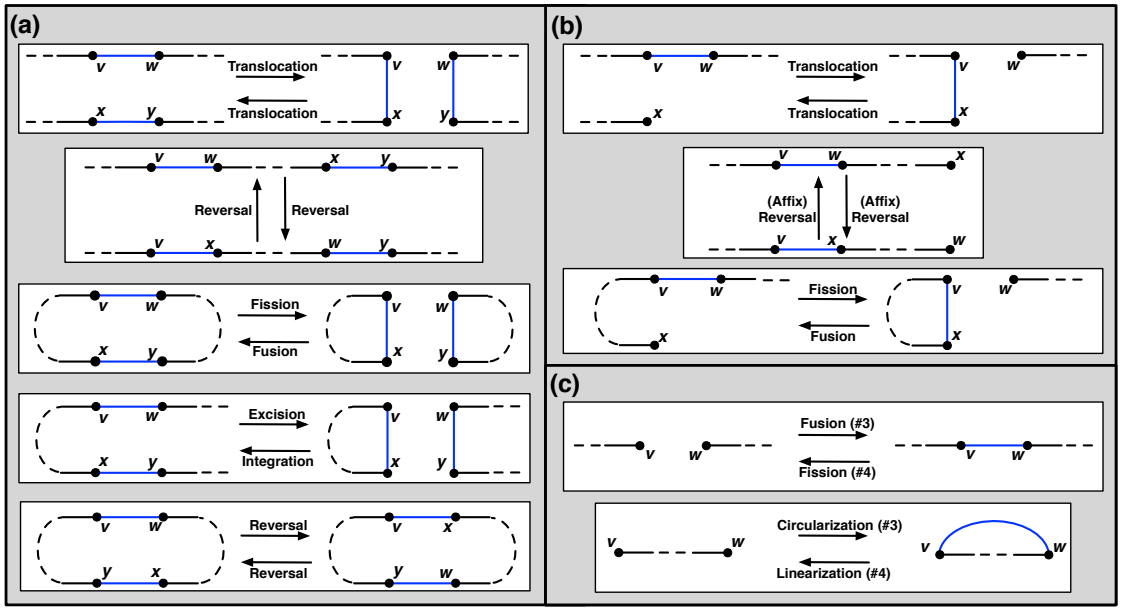
Additional file 2 — DCJ.pdf

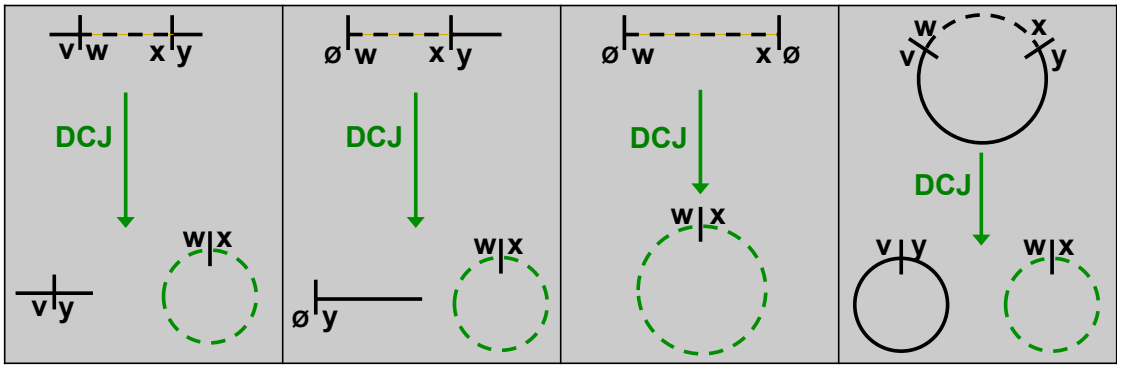
PDF file corresponding to Figure 2.

Additional file 3 — Indel as DCJ.pdf

PDF file corresponding to Figure 3.


(a)
9h 9t
2t 2h
13h 11t
13t 11h
12h 12t
10t 10h 1h 1t 4h 4t 3t 3h 8t 8h 7h 7t
1h 1t
6t 6h
9h 7t
9t 7h
8h 8t
2t 2h 3t 3h 4t 4h 5t 5h
(b)
5t 4h 1t 1h 10h
π
γ
7h 8h
8t 9t
3h 2h
4t 3t
5h 10t
π, ∅ ∅, γ
6t 9h 2t
π ∅
7t 6h
∅ π
11t 13h 13t 12h 12t 11h
γ γ γ γ γ γ


(a)
Translocation
Translocation
Reversal
Reversal
Fission
Fusion
Excision
Integration
Reversal
Reversal
(b)
Translocation
Translocation
(Affix) Reversal
(Affix) Reversal
Fission
Fusion
(c)
Fusion (#3)
Fission (#4)
Circularization (#3)
Linearization (#4)


v w x y
DCJ
w x
v y
ø w x y
DCJ
w x
ø y
ø w x ø
DCJ
w x
w x
v y
DCJ
v y
w x

Chapter 3

A Generalized Cost Model for DCJ-Indel Sorting

3.1 Preliminaries

A **genome** Π is a graph containing an even number of labeled nodes and comprising the edge-disjoint union of two perfect matchings: the **genes**[1] of Π, denoted $g(\Pi)$; and the **adjacencies** of Π, denoted $a(\Pi)$. Consequently, each node of Π has degree 2, and the connected components of Π form cycles that alternate between genes and adjacencies; these cycles are called **chromosomes**. This genomic model, in which chromosomes are circular, offers a reasonable and commonly used approximation of genomes having linear chromosomes.

A **double cut and join operation (DCJ)** on Π, introduced in [YAF05b], forms a new genome by replacing two adjacencies of Π with two new adjacencies on the same four nodes. Despite being simply defined, the DCJ incorporates the **reversal** of a chromosomal segment, the **fusion** of two chromosomes into one chromosome, and the **fission** of one chromosome into two chromosomes (Fig. **??** (top)).[2] For genomes Π and Γ with the same genes, the **DCJ distance**, denoted

[1] In practice, gene edges typically represent synteny blocks containing a large number of contiguous genes

[2] When the DCJ is applied to circularized linear chromosomes, it encompasses a larger variety of operations. See [BWS10] for details.

$d(\Pi, \Gamma)$, is the minimum number of DCJs needed to transform Π into Γ.

The **breakpoint graph** of Π and Γ, denoted $B(\Pi, \Gamma)$ (introduced in [BP96b]), is the edge-disjoint union of $a(\Pi)$ and $a(\Gamma)$ (Fig. **??** (bottom)). The line graph of the breakpoint graph is the *adjacency graph*, which was introduced in [BMS06] and is also commonly used in genome rearrangement studies. Note that the connected components of $B(\Pi, \Gamma)$ form cycles (of length at least 2) that alternate between adjacencies of Π and Γ. Letting $c(\Pi, \Gamma)$ denote the number of cycles in $B(\Pi, \Gamma)$, the authors in [YAF05b] showed that the DCJ distance is given by

$$d(\Pi, \Gamma) = |g(\Pi)| - c(\Pi, \Gamma) \ . \tag{3.1}$$

The DCJ distance offers a useful metric for measuring the evolutionary distance between two genomes having the same genes, but we strive toward a genomic model that incorporates insertions and deletions as well. A **deletion** in Π is defined as the removal of either an entire chromosome or chromosomal interval of Π, i.e., if adjacencies $\{v, w\}$ and $\{x, y\}$ are contained in the order (v, w, x, y) on some chromosome of Π, then a deletion replaces the path connecting v to y with the single adjacency $\{v, y\}$. An **insertion** is simply the inverse operation of a deletion. The term **indels** refers collectively to insertions and deletions.

To consider genomes with unequal gene content, we will henceforth assume that any pair of genomes Π and Γ satisfy $g(\Pi) \cup g(\Gamma) = \mathcal{G}$, where $\mathcal{G}$ is a perfect matching on a collection of nodes $\mathcal{V}$. A **transformation** of Π into Γ is a sequence of DCJs and indels such that any deleted node must belong to $\mathcal{V} - V(\Gamma)$ and any inserted node must belong to $\mathcal{V} - V(\Pi)$.[3]

The **cost** of a transformation $\mathbb{T}$ is equal to the weighted sum of the number of DCJs in $\mathbb{T}$ plus ω times the number of indels in $\mathbb{T}$, where ω is some nonnegative constant determined in advance. The **DCJ-indel distance** between Π and Γ, denoted $d_\omega(\Pi, \Gamma)$, is the minimum cost of any transformation of Π into Γ. Note that since a transformation of Π into Γ can be inverted to yield a transformation of Γ into Π, the DCJ-indel distance is symmetric by definition. Yet unlike the DCJ

[3]This assumption follows the lead of the authors in [BWS10]. It means that we can view any transformation of Π into Γ prevents, among other things, a trivial transformation of one genome into another genome of similar gene content in which we simply delete all the chromosomes of Π and replace them with the chromosomes of Γ.

distance, the DCJ-indel distance does not form a metric, as the triangle inequality does not hold; see [BMRS11] for a discussion in the case that $\omega = 1$.

Although we would like to compute DCJ-indel distance, here we are interested in the more difficult problem of **DCJ-indel sorting**, or producing a minimum cost transformation of Π into Γ. The case $\omega = 1$ was resolved by the authors in [BWS10]; this result was extended to cover all values $0 \leq \omega \leq 1$ in [SBMD12] and [dSMDB13]. This work aims to use the simplifying ideas in [Com12] and [Com13] as a stepping stone for a generalized presentation that will solve the problem of DCJ-indel sorting for all $\omega \geq 0$, thus resolving the open case that $\omega > 1$.

3.2 Encoding Indels as DCJs

A chromosome of Π (Γ) sharing no genes with Γ (Π) is called a **singleton**. We use the notation $\text{sing}_{\Gamma}(\Pi)$ to denote the number of singletons of Π with respect to Γ and the notation $\text{sing}(\Pi, \Gamma)$ to denote the sum $\text{sing}_{\Gamma}(\Pi) + \text{sing}_{\Pi}(\Gamma)$. We will deal with singletons later; for now, we will show that in the absence of singletons, the insertion or deletion of a chromosomal interval is the only type of indel that we need to consider for the problem of DCJ-indel sorting.

Theorem. *If* $\text{sing}(\Pi, \Gamma) = 0$*, then any minimum-cost transformation of* Π *into* Γ *cannot include the insertion or deletion of entire chromosomes.*

Proof. We proceed by contradiction. Say that we have a minimum-cost transformation of Π into Γ in which (without loss of generality) we are deleting an entire chromosome C. Because Π has no singletons, C must have been produced as the result of the deletion of a chromosomal interval or as the result of a DCJ. C cannot have been produced from the deletion of an interval, since we could have simply deleted the chromosome that C came from. Thus, assuming C was produced as the result of a DCJ, there are now three possibilities:

1. The DCJ could be a reversal. In this case, we could have simply deleted the chromosome to which the reversal was appiled, yielding a transformation of strictly smaller cost.

2. The DCJ could be a fission of a chromosome C' that produced C along with another chromosome. In this case, the genes of C appeared as a contiguous interval of C', which we could have simply deleted at lesser total cost.

3. The DCJ could be the fusion of two chromosomes, C_1 and C_2. This case is somewhat more difficult to deal with and is handled by Lemma 3.

Lemma: *If* $\text{sing}_\Gamma(\Pi) = 0$*, then any minimum-cost transformation of* Π *into* Γ *cannot include the deletion of a chromosome that was produced by a fusion.*

Proof. Suppose for the sake of contradiction that a minimum-cost transformation $\mathbb{T}$ of Π into Γ involves k fusions of $k+1$ chromosomes $C_1, C_2, \ldots, C_{k+1}$ to form a chromosome C, which is then deleted. Without loss of generality, we may assume that this collection of fusions is "maximal", i.e., none of the C_i is produced as the result of a fusion.

Because Π has no singletons, each C_i must have been produced as a result of a DCJ. Similar reasoning to that used in the main proof of Theorem 3.2 shows that this DCJ cannot be a reversal, and by the assumption of maximality, it cannot be the result of a fusion. Thus, each C_i is produced by a fission applied to some chromosome C_i' to produce C_i in addition to some other chromosome C_i^*.

Now, let Π' be the genome in $\mathbb{T}$ occurring immediately before these $2k+2$ operations. Assume that the $k+1$ fissions applied to the C_i' replace adjacencies $\{v_i, w_i\}$ and $\{x_i, y_i\}$ with $\{v_i, y_i\}$ and $\{w_i, x_i\}$.[4] Furthermore, assume that the ensuing k fusions are as follows:

$$\begin{aligned}
\{v_1, y_1\}, \{v_2, y_2\} &\rightarrow \{y_1, v_2\}, \{v_1, y_2\} \\
\{y_1, v_2\}, \{v_3, y_3\} &\rightarrow \{y_1, v_3\}, \{v_2, y_3\} \\
\{y_1, v_3\}, \{v_4, y_4\} &\rightarrow \{y_1, v_4\}, \{v_3, y_4\} \\
&\vdots \\
\{y_1, v_k\}, \{v_{k+1}, y_{k+1}\} &\rightarrow \{y_1, v_{k+1}\}, \{v_k, y_{k+1}\}
\end{aligned}$$

The genome resulting from these $2k+1$ operations, which we call $\Pi''_{\mathbb{T}}$, is identical to Π' except that for each i ($1 \leq i \leq k+1$), it has replaced the

[4] It can be verified that these $4k+4$ nodes must be distinct by the assumption that $\mathbb{T}$ has minimum cost.

adjacencies $\{v_i, w_i\}$ and $\{x_i, y_i\}$ in C'_i with the adjacencies $\{w_i, x_i\} \in C^*_i$ and $\{v_i, y_{i+1 \mod (k+1)}\} \in C$. In $\mathbb{T}$, we then delete C from $\Pi''_{\mathbb{T}}$.

Now consider the transformation $\mathbb{U}$ that is identical to $\mathbb{T}$ except that when we reach Π', $\mathbb{U}$ first applies the following k DCJs:

$$\begin{aligned}
\{v_1, w_1\}, \{x_2, y_2\} &\rightarrow \{v_1, y_2\}, \{w_1, x_2\} \\
\{v_2, w_2\}, \{x_3, y_3\} &\rightarrow \{v_2, y_3\}, \{w_2, x_3\} \\
\{v_3, w_3\}, \{x_4, y_4\} &\rightarrow \{v_3, y_4\}, \{w_3, x_4\} \\
&\vdots \\
\{v_k, w_k\}, \{x_{k+1}, y_{k+1}\} &\rightarrow \{v_k, y_{k+1}\}, \{w_k, x_{k+1}\}
\end{aligned}$$

$\mathbb{U}$ then applies k subsequent DCJs as follows:

$$\begin{aligned}
\{x_1, y_1\}, \{w_1, x_2\} &\rightarrow \{y_1, x_2\}, \{w_1, x_1\} \\
\{y_1, x_2\}, \{w_2, x_3\} &\rightarrow \{y_1, x_3\}, \{w_2, x_2\} \\
\{y_1, x_3\}, \{w_3, x_4\} &\rightarrow \{y_1, x_4\}, \{w_3, x_3\} \\
&\vdots \\
\{y_1, x_k\}, \{w_k, x_{k+1}\} &\rightarrow \{y_1, x_{k+1}\}, \{w_k, x_k\}
\end{aligned}$$

The resulting genome, which we call $\Pi''_{\mathbb{U}}$, has the exact same adjacencies as $\Pi''_{\mathbb{T}}$ except that it contains the adjacencies $\{y_1, x_{k+1}\}$ and $\{v_{k+1}, w_{k+1}\}$ instead of $\{v_{k+1}, y_1\}$ and $\{w_{k+1}, x_{k+1}\}$. Because two genomes on the same genes are equivalent if and only if they share the same adjacencies, a single DCJ on $\{y_1, x_{k+1}\}$ and $\{v_{k+1}, w_{k+1}\}$ would change $\Pi''_{\mathbb{U}}$ into $\Pi''_{\mathbb{T}}$. Furthermore, in $\Pi''_{\mathbb{T}}$, $\{v_{k+1}, y_1\}$ belongs to C and $\{w_{k+1}, x_{k+1}\}$ belongs to C^*_{k+1}, so that this DCJ in question must be a fission producing C and C^*_{k+1}. In $\mathbb{U}$, rather than applying this fission, we simply delete the chromosomal interval containing the genes of C. As a result, $\mathbb{U}$ is identical to $\mathbb{T}$ except that it replaces $2k + 1$ DCJs and a deletion by $2k$ DCJs and a deletion. Hence, $\mathbb{U}$ has strictly smaller cost than $\mathbb{T}$, which provides the desired contradiction. □

□

Following Theorem 3.2, we recall the observation in [AT11] that we can view the deletion of a chromosomal interval replacing adjacencies $\{v, w\}$ and $\{x, y\}$ with

the single adjacency as a fission replacing $\{v, w\}$ and $\{x, y\}$ by the two adjacencies $\{w, x\}$ and $\{v, y\}$, thus forming a circular chromosome containing $\{v, y\}$ that is scheduled for later removal. By viewing this operation as a DCJ, we establish a bijective correspondence between the deletions of a minimum cost transformation of Π into Γ (having no singletons) and a collection of chromosomes sharing no genes with Π. (Insertions are handled symmetrically.)

Therefore, define a **completion** of genomes Π and Γ as a pair of genomes (Π', Γ') such that Π is a subgraph of Π', Γ is a subgraph of Γ', and $g(\Pi') = g(\Gamma') = \mathcal{G}$. Each of $\Pi' - \Pi$ and $\Gamma' - \Gamma$ is formed of alternating cycles called **new chromosomes**; by our bijective correspondence, we use $\text{ind}(\Pi', \Gamma')$ to denote the total number of new chromosomes of Π' and Γ'. We will amortize the cost of a deletion by charging unit cost for the DCJ that produces a new chromosome, followed by $1 - \omega$ for the removal of this chromosome, yielding our next result. For simplicity, we henceforth set $N = |\mathcal{V}|/2$, the number of genes of Π and Γ.

Theorem. *If* $\text{sing}(\Pi, \Gamma) = 0$, *then*

$$d_\omega(\Pi, \Gamma) = \min_{(\Pi', \Gamma')} \{d(\Pi', \Gamma') + (\omega - 1) \cdot \text{ind}(\Pi', \Gamma')\} \tag{3.2}$$

$$= N - \max_{(\Pi', \Gamma')} \{c(\Pi', \Gamma') + (1 - \omega) \cdot \text{ind}(\Pi', \Gamma')\} \tag{3.3}$$

where the optimization is taken over all completions of Π *and* Γ.

A completion (Π^*, Γ^*) is called **optimal** if it achieves the maximum in (3.3). We plan to use Theorem 3.2 to construct an optimal completion for genomes lacking singletons. Once we have formed an optimal completion (Π^*, Γ^*), we can simply invoke the $O(N)$-time sorting algorithm described in [BMS06] to transform Π^* into Γ^* via a minimum collection of DCJs.

3.3 DCJ-Indel Sorting Genomes without Singletons

Define a node $v \in \mathcal{V}$ to be **Π-open** (**Γ-open**) if $v \notin \Pi$ ($v \notin \Gamma$). When forming adjacencies of Π^* (Γ^*), we connect pairs of Π-open (Γ-open) nodes. Given genomes Π and Γ with unequal gene content, we can still define the

breakpoint graph $B(\Pi, \Gamma)$ as the edge-disjoint union of $a(\Pi)$ and $a(\Gamma)$; however, because the adjacencies of Π and Γ are not necessarily perfect matchings on $\mathcal{V}$, $B(\Pi, \Gamma)$ may contain paths (of positive length) in addition to cycles.

We can view the problem of constructing an optimal completion (Π^*, Γ^*) as adding edges to $B(\Pi, \Gamma)$ to form $B(\Pi^*, \Gamma^*)$. Our hope is to construct these edges via direct analysis of $B(\Pi, \Gamma)$. First, note that cycles of $B(\Pi, \Gamma)$ must embed as cycles of $B(\Pi^*, \Gamma^*)$, whereas odd-length paths of $B(\Pi, \Gamma)$ end in either two Π-open nodes or two Γ-open nodes, and even-length paths of $B(\Pi, \Gamma)$ end in a Π-open node and a Γ-open node. The paths of $B(\Pi, \Gamma)$ must be **linked** in some way by edges in $a(\Pi^*) - a(\Pi)$ or $a(\Gamma^*) - a(\Gamma)$ to form cycles alternating between edges of $a(\Pi^*)$ and $a(\Gamma^*)$. Our basic intuition is to do so in such a way as to create as many cycles as possible, at least when ω is small; this intuition is confirmed by the following two results.

Proposition: *If* $0 < \omega < 2$ *and* $\text{sing}(\Pi, \Gamma) = 0$, *then for any optimal completion* (Π^*, Γ^*) *of* Π *and* Γ, *every path of length* $2k - 1$ *in* $B(\Pi, \Gamma)$ $(k \geq 1)$ *embeds into a cycle of length* $2k$ *in* $B(\Pi^*, \Gamma^*)$.

Proof. Let P be a path of length $2k - 1$ in $B(\Pi, \Gamma)$. Without loss of generality, assume that P has Π-open nodes v and w as endpoints. Suppose that for some completion (Π', Γ'), P does not embed into a cycle of length $2k$ in $B(\Pi', \Gamma')$ (i.e., $\{v, w\}$ is not an adjacency of Π'); in this case, we must have distinct adjacencies $\{v, x\}$ and $\{w, y\}$ in Π' belonging to the same cycle of $B(\Pi, \Gamma)$.

Consider the completion Π'' that is formed from Π' by replacing $\{v, x\}$ and $\{w, y\}$ with $\{v, w\}$ and $\{x, y\}$. It is clear that $c(\Pi'', \Gamma') = c(\Pi', \Gamma') + 1$ and $|\text{ind}(\Pi'', \Gamma') - \text{ind}(\Pi', \Gamma')| < 1$. Thus, it follows from (3.3) that (Π', Γ') cannot be optimal. □

As a result of Proposition 6, when $0 < \omega < 2$, any cycle of $B(\Pi^*, \Gamma^*)$ that is not induced from a cycle or odd-length path of $B(\Pi, \Gamma)$ must be a ***k*-bracelet**, which contains k even-length paths of $B(\Pi, \Gamma)$, where k is even. We use the term **bracelet links** to refer to adjacencies of a bracelet belonging to new chromosomes; each k-bracelet in $B(\Pi^*, \Gamma^*)$ contains $k/2$ bracelet links from $\Pi^* - \Pi$ and $k/2$ bracelet

links from $\Gamma^* - \Gamma$. According to (3.3), we need to make $c(\Pi^*, \Gamma^*)$ large, which means that when indels are inexpensive, we should have bracelets containing as few bracelet links as possible.

Proposition: *If* $0 < \omega < 2$ *and* $\text{sing}(\Pi, \Gamma) = 0$, *then for any optimal completion* (Π^*, Γ^*) *of* Π *and* Γ, *all of the even-length paths of* $B(\Pi, \Gamma)$ *embed into 2-bracelets of* $B(\Pi^*, \Gamma^*)$.

Proof. Suppose that a completion (Π', Γ') of Π and Γ contains a k-bracelet for $k \geq 4$. This bracelet must contain two bracelet adjacencies $\{v, w\}$ and $\{x, y\}$ belonging to $a(\Pi')$, where these four nodes are contained in the order (v, w, x, y) in the bracelet. Consider the genome Π'' that is obtained from Π' by replacing $\{v, w\}$ and $\{x, y\}$ with $\{v, y\}$ and $\{w, x\}$. As in the proof of Proposition 6, $c(\Pi'', \Gamma') = c(\Pi', \Gamma') + 1$ and $|\text{ind}(\Pi'', \Gamma') - \text{ind}(\Pi', \Gamma')| < 1$, so that (Π', Γ') cannot be optimal. □

The conditions provided by the previous two propositions are very strong. To resolve the case that $0 < \omega < 2$, note that if we must link the endpoints of any odd-length path in $B(\Pi, \Gamma)$ to construct an optimal completion, then we may first create some new chromosomes before dealing with the case of even-length paths. Let $k_\Gamma(\Pi)$ be the number of new chromosomes formed by linking the endpoints of odd-length paths of $B(\Pi, \Gamma)$ that end with Π-open nodes, and set $k(\Pi, \Gamma) = k_\Gamma(\Pi) + k_\Pi(\Gamma)$. After linking the endpoints of odd-length paths, if we can link pairs of even-length paths (assuming any exist) into 2-bracelets so that one *additional* new chromosome is created in each of Π^* and Γ^*, then we will have constructed an optimal completion. This construction is guaranteed by the following proposition.

Proposition: *If* $0 < \omega < 2$ *and* $\text{sing}(\Pi, \Gamma) = 0$, *then any optimal completion* (Π^*, Γ^*) *of* Π *and* Γ *has the property that one new chromosome of* Π^* (Γ^*) *contains all of the bracelet adjacencies of* Π^* (Γ^*).

Proof. By Proposition 6, we may assume that we have started forming an optimal completion (Π^*, Γ^*) by linking the endpoints of any odd-length paths in $B(\Pi, \Gamma)$

to each other. Given any even-length path P in $B(\Pi, \Gamma)$, there is exactly one other even-length path P_1 that would form a new chromosome in Γ^* if linked with P, and exactly one other even-length path P_2 in $B(\Pi, \Gamma)$ that would form a new chromosome in Π^* if linked with P (P_1 and P_2 may be the same). As long as there are more than two other even-length paths to choose from, we can simply link P to any path other than P_1 or P_2. We then iterate this process until two even-length paths remain, which we link to complete the construction of Π^* and Γ^*; each of these genomes has one new chromosome containing all of that genome's bracelet adjacencies. □

It is easy to see that the conditions in the preceding three propositions are sufficient (but not necessary) when constructing an optimal completion for the boundary cases $\omega = 0$ and $\omega = 2$. We are now ready to state our first major result with respect to DCJ-indel sorting.

When $0 \leq \omega \leq 2$ and $\text{sing}(\Pi, \Gamma) = 0$, the following algorithm solves the problem of DCJ-indel sorting Π into Γ in $O(N)$ time.

1. Link the endpoints of any odd-length path in $B(\Pi, \Gamma)$, which may create some new chromosomes in Π^* and Γ^*.

2. Arbitrarily select an even-length path P of $B(\Pi, \Gamma)$ (if one exists).

   (a) If there is more than one additional even-length path in $B(\Pi, \Gamma)$, link P to an even-length path that produces no new chromosomes in Π^* or Γ^*.

   (b) Otherwise, link the two remaining even-length paths in $B(\Pi, \Gamma)$ to form a new chromosome in each of Π^* and Γ^*.

3. Iterate Step 2 until no even-length paths of $B(\Pi, \Gamma)$ remain. The resulting completion is (Π^*, Γ^*).

4. Apply the $O(N)$-time algorithm for DCJ sorting from [YAF05b] to transform Π^* into Γ^*.

Let $p_{\text{odd}}(\Pi, \Gamma)$ and $p_{\text{even}}(\Pi, \Gamma)$ equal the number of odd- and even-length paths in $B(\Pi, \Gamma)$, respectively. The optimal completion (Π^*, Γ^*) constructed by Algorithm **??** has the following properties:

$$c(\Pi^*, \Gamma^*) = c(\Pi, \Gamma) + p_{\text{odd}}(\Pi, \Gamma) + \frac{p_{\text{even}}(\Pi, \Gamma)}{2} \tag{3.4}$$

$$\text{ind}(\Pi^*, \Gamma^*) = k(\Pi, \Gamma) + \min\{2, p_{\text{even}}(\Pi, \Gamma)\} \tag{3.5}$$

These formulas, when combined with Theorem 3.2, yield a formula for the DCJ-indel distance as a function of Π, Γ, and ω alone.

Corollary: *If* $0 \leq \omega \leq 2$ *and* $\text{sing}(\Pi, \Gamma) = 0$*, the DCJ-indel distance between* Π *and* Γ *is given by the following equation:*

$$\begin{aligned} d_\omega(\Pi, \Gamma) = N - \Bigg[\Big(c(\Pi, \Gamma) + p_{\text{odd}}(\Pi, \Gamma) + \frac{p_{\text{even}}(\Pi, \Gamma)}{2} \Big) + \Big(1 - \omega\Big) \cdot \\ \Big(k(\Pi, \Gamma) + \min\big\{2, p_{\text{even}}(\Pi, \Gamma)\big\} \Big) \Bigg] \end{aligned} \tag{3.6}$$

We now turn our attention to the case $\omega > 2$. Intuitively, as ω grows, we should witness fewer indels. Let $\delta_\Gamma(\Pi)$ be equal to 1 if $g(\Pi) - g(\Gamma)$ is nonempty and 0 otherwise; then, set $\delta(\Pi, \Gamma) = \delta_\Gamma(\Pi) + \delta_\Pi(\Gamma)$. Note that $\delta(\Pi, \Gamma)$ is a lower bound on the number of indels in any transformation of Π into Γ. The following result shows that in the absence of singletons, this bound is achieved by every minimum-cost transformation when $\omega > 2$.

Theorem. *If* $\omega > 2$ *and* $\text{sing}(\Pi, \Gamma) = 0$*, then any minimum-cost transformation of* Π *into* Γ *has at most insertion and at most one deletion. As a result,*

$$d_\omega(\Pi, \Gamma) = N - \max_{\text{ind}(\Pi', \Gamma') = \delta(\Pi, \Gamma)} \{c(\Pi', \Gamma') + (1 - \omega) \cdot \delta(\Pi, \Gamma)\} \ . \tag{3.7}$$

Proof. Suppose for the sake of contradiction that $\mathbb{T}$ is a minimum-cost transformation of Π into Γ and that (without loss of generality) $\mathbb{T}$ contains two deletions of chromosomal intervals P_1 and P_2, costing 2ω. Say that one of these deletions replaces adjacencies $\{v, w\}$ and $\{x, y\}$ with $\{v, y\}$ (deleting the interval connecting w to x) and the other deletion replaces adjacencies $\{a, b\}$ and $\{c, d\}$ with $\{a, d\}$ (deleting the interval connecting b to c).

Consider a second transformation $\mathbb{U}$ that is otherwise identical to $\mathbb{T}$, except that it replaces the deletions of P_1 and P_2 with three operations. First, a DCJ replaces $\{v, w\}$ and $\{a, b\}$ with $\{v, a\}$ and $\{w, b\}$; the new adjacency $\{w, b\}$ joins P_1 and P_2 into a single chromosomal interval P. Second, a deletion removes P and replaces adjacencies $\{c, d\}$ and $\{x, y\}$ with the single adjacency $\{d, y\}$. Third, another DCJ replaces $\{v, a\}$ and $\{d, y\}$ with the adjacencies $\{v, y\}$ and $\{a, d\}$, yielding the same genome as the first scenario at a cost of $2 + \omega$. Because $\mathbb{U}$ is otherwise the same as $\mathbb{T}$, $\mathbb{U}$ will have strictly lower cost precisely when $\omega > 2$, in which case $\mathbb{T}$ cannot have minimum cost. □

One can verify that the condition in Theorem 3.3 is sufficient but not necessary to guarantee a minimum-cost transformation when $\omega = 2$. Furthermore, a consequence of Theorem 3.3 is that the optimal completion is independent of the value of ω. In other words, if a completion achieves the maximum in (3.7), then this completion is automatically optimal for *all* values of $\omega \geq 2$.

Fortunately, Algorithm **??** already describes the construction of a completion (Π', Γ') that is optimal when $\omega = 2$. Of course, we cannot guarantee that this completion has the desired property that $\text{ind}(\Pi', \Gamma') = \delta(\Pi, \Gamma)$. However, if $\text{ind}(\Pi', \Gamma') > \delta(\Pi, \Gamma)$, then we can apply $\text{ind}(\Pi', \Gamma') - \delta(\Pi, \Gamma)$ total fusions to Π' and Γ' in order to obtain a different completion (Π^*, Γ^*). Each of these fusions reduces the number of new chromosomes by 1 and (by (3.3)) must also decrease the number of cycles in the breakpoint graph by 1, since (Π', Γ') is optimal for $\omega = 2$. As a result, $c(\Pi^*, \Gamma^*) - \text{ind}(\Pi^*, \Gamma^*) = c(\Pi', \Gamma') - \text{ind}(\Pi', \Gamma')$. Thus, (Π^*, Γ^*) is optimal for $\omega = 2$, and since $\text{ind}(\Pi^*, \Gamma^*) = \delta(\Pi, \Gamma)$, we know that (Π^*, Γ^*) must be optimal for any $\omega > 2$ as already noted. This discussion immediately implies the following algorithm.

The optimal completion (Π^*, Γ^*) returned by Algorithm **??** has the property that

$$c(\Pi^*, \Gamma^*) = c(\Pi, \Gamma) + p_{\text{odd}}(\Pi, \Gamma) + \frac{p_{\text{even}}(\Pi, \Gamma)}{2} - \Big[\text{ind}(\Pi', \Gamma') - \delta(\Pi, \Gamma)\Big] \ , \tag{3.8}$$

where (Π', Γ') is the optimal completion for $\omega = 2$ returned by Algorithm **??**. Combining this equation with (3.5) and (3.7) yields a closed formula for the DCJ-indel distance when $\omega > 2$ in the absence of singletons.

If $\omega \geq 2$ and $\text{sing}(\Pi, \Gamma) = 0$, then the following algorithm solves the problem of DCJ-indel sorting Π into Γ in $O(N)$ time.

1. Follow the first three steps of Algorithm **??** to construct a completion (Π', Γ') that is optimal for $\omega = 2$.

2. Apply a total of $\text{ind}(\Pi', \Gamma') - \delta(\Pi, \Gamma)$ fusions to Π' and Γ' in order to produce a completion (Π^*, Γ^*) having $\text{ind}(\Pi^*, \Gamma^*) = \delta(\Pi, \Gamma)$.

3. Apply the $O(N)$-time algorithm for DCJ sorting from [YAF05b] to transform Π^* into Γ^*. Any DCJ involving a new chromosome can be viewed as an indel.

Corollary: *If $\omega \geq 2$ and $\text{sing}(\Pi, \Gamma) = 0$, then the DCJ-indel distance between Π and Γ is given by the following equation:*

$$d_\omega(\Pi, \Gamma) = N - \Bigg[\Big(c(\Pi, \Gamma) + p_{\text{odd}}(\Pi, \Gamma) + \frac{p_{\text{even}}(\Pi, \Gamma)}{2} - k(\Pi, \Gamma) - \min\big\{2, p_{\text{even}}(\Pi, \Gamma)\big\}\Big) + (2 - \omega) \cdot \delta(\Pi, \Gamma)\Bigg] \tag{3.9}$$

3.4 Incorporating Singletons into DCJ-Indel Sorting

We have thus far avoided genome pairs with singletons because Theorem 3.2, which underlies the main results in the preceding section, only applied in the absence of singletons. Yet fortunately, genomes with singletons will be relatively easy to incorporate into a single DCJ-indel sorting algorithm. As we might guess, different values of ω produce different results.

Theorem. *If $\Pi^\emptyset$ and $\Gamma^\emptyset$ are produced from genomes Π and Γ by removing all singletons, then*

$$\begin{aligned} d_\omega(\Pi, \Gamma) = & d_\omega(\Pi^\emptyset, \Gamma^\emptyset) + \min\{1, \omega\} \cdot \text{sing}(\Pi, \Gamma) + \max\{0, \omega - 1\} \cdot \\ & \Big[(1 - \delta_{\Gamma^\emptyset}(\Pi^\emptyset)) \cdot \min\{1, \text{sing}_\Gamma(\Pi)\} + \\ & (1 - \delta_{\Pi^\emptyset}(\Gamma^\emptyset)) \cdot \min\{1, \text{sing}_\Pi(\Gamma)\}\Big] \end{aligned} \tag{3.10}$$

Proof. Any transformation of $\Pi^{\varnothing}$ into $\Gamma^{\varnothing}$ can be supplemented by the deletion of each singleton of Π and the insertion of each singleton of Γ to yield a collection of DCJs and indels transforming Π into Γ. As a result, for any value of ω,

$$d_{\omega}(\Pi, \Gamma) \leq d_{\omega}(\Pi^{\varnothing}, \Gamma^{\varnothing}) + \omega \cdot \text{sing}(\Pi, \Gamma) \ . \tag{3.11}$$

Next, we will view an arbitrary transformation $\mathbb{T}$ of Π into Γ as a sequence $(\Pi_0, \Pi_1, \ldots, \Pi_n)$ $(n \geq 1)$, where $\Pi_0 = \Pi$, $\Pi_n = \Gamma$, and Π_{i+1} is obtained from Π_i as the result of a single DCJ or indel. Consider a sequence $(\Pi_0^{\varnothing}, \Pi_1^{\varnothing}, \ldots, \Pi_n^{\varnothing})$, where $\Pi_i^{\varnothing}$ is constructed from Π_i by removing the subgraph of Π_i induced by the nodes of the singletons of Π and Γ under the stipulation that whenever we remove a path P connecting v to w, we replace adjacencies $\{v, x\}$ and $\{w, y\}$ in Π_i with $\{x, y\}$ in $\Pi_i^{\varnothing}$. Certainly, $\Pi_0^{\varnothing} = \Pi^{\varnothing}$ and $\Pi_n^{\varnothing} = \Gamma^{\varnothing}$. Furthermore, for every i in range, if $\Pi_{i+1}^{\varnothing}$ is not the result of a DCJ or indel applied to $\Pi_i^{\varnothing}$, then $\Pi_{i+1}^{\varnothing} = \Pi_i^{\varnothing}$. Thus, $(\Pi_0^{\varnothing}, \Pi_1^{\varnothing}, \ldots, \Pi_n^{\varnothing})$ can be viewed as encoding a transformation of $\Pi^{\varnothing}$ into Γ using *at most* n DCJs and indels. One can verify that $\Pi_{i+1}^{\varnothing} = \Pi_i^{\varnothing}$ precisely when Π_{i+1} is produced from Π_i either by a DCJ that involves an adjacency belonging to a singleton or by an indel containing genes that all belong to singletons. At least $\text{sing}(\Pi, \Gamma)$ such operations must always occur in $\mathbb{T}$; hence,

$$d_{\omega}(\Pi, \Gamma) \geq d_{\omega}(\Pi^{\varnothing}, \Gamma^{\varnothing}) + \min\{1, \omega\} \cdot \text{sing}(\Pi, \Gamma) \ . \tag{3.12}$$

In the case that $\omega \leq 1$, the bounds in (3.11) and (3.12) immediately yield (3.10).

Assume, then, that $\omega > 1$. If $\delta_{\Gamma^{\varnothing}}(\Pi^{\varnothing}) = 0$, then $g(\Pi^{\varnothing}) \subseteq g(\Gamma^{\varnothing})$, meaning that every deleted gene of Π must belong to a singleton of Π. In this case, the total cost of removing any singletons of Π is trivially minimized by $\text{sing}_{\Gamma}(\Pi) - 1$ fusions consolidating the singletons of Π into a single chromosome, followed by the deletion of this chromosome. Symmetric reasoning applies to the singletons of Γ if $\delta_{\Pi^{\varnothing}}(\Gamma^{\varnothing}) = 0$.

On the other hand, assume that $\omega > 1$ and that $\delta_{\Gamma^{\varnothing}}(\Pi^{\varnothing}) = 1$, so that $g(\Pi^{\varnothing}) - g(\Gamma^{\varnothing})$ is nonempty. In this case, if Π has any singletons, then we can create a minimum-cost transformation by applying $\text{sing}_{\Gamma}(\Pi) - 1$ fusions consolidating the singletons of Π into a single chromosome, followed by another fusion that consolidates these chromosomes into a chromosomal interval of Π

that is about to be deleted. Symmetric reasoning applies to the singletons of Γ if $\delta_{\Pi^\emptyset}(\Gamma^\emptyset) = 1$.

Regardless of the particular values of $\delta_{\Gamma^\emptyset}(\Pi^\emptyset)$ and $\delta_{\Pi^\emptyset}(\Gamma^\emptyset)$, we will obtain the formula in (3.10). □

This proof immediately provides us with an algorithm incorporating the case of genomes with singletons into the existing DCJ-indel sorting framework.

3.5 Conclusion

With the problem of DCJ-indel sorting genomes with circular chromosomes unified under a general model, we see three obvious future applications of this work.

First, an extension of these results for genomes with linear chromosomes would prevent us from having to first circularize linear chromosomes when comparing eukaryotic genomes. This work promises to be extremely tedious (if it is indeed possible) without offering dramatic new insights.

Second, we would like to implement the linear-time method for DCJ-indel sorting described in Algorithm **??** and publish the code publicly. Evolutionary study analysis on real data will hopefully determine appropriate choices of ω.

Third, we are currently attempting to extend these results to fully characterize the space of all solutions to DCJ-indel sorting, which would generalize the result in [Com13] to arbitrary values of ω.

The following algorithm solves the general problem of DCJ-indel sorting genomes Π and Γ for any indel cost $\omega \geq 0$ in $O(N)$ time.

1. Case 1: $\omega \leq 1$.
   (a) Delete any singletons of Π, then insert any singletons of Γ.
   (b) Apply Algorithm **??** to transform the resulting genome into Γ.
2. Case 2: $\omega > 1$.
   (a) If Π has any singletons, apply $\text{sing}_\Gamma(\Pi) - 1$ fusions to consolidate the singletons of Π into a single chromosome C_Π.
      i. If $g(\Pi^\emptyset) \subseteq g(\Gamma^\emptyset)$, delete C_Π.
      ii. Otherwise, save C_Π for later.
   (b) If Γ has any singletons, apply $\text{sing}_\Pi(\Gamma) - 1$ fusions to consolidate the singletons of Γ into a single chromosome C_Γ.
      i. If $g(\Gamma^\emptyset) \subseteq g(\Pi^\emptyset)$, delete C_Γ.
      ii. Otherwise, save C_Γ for later.
   (c) Apply a sorting algorithm as needed to construct an optimal completion (Π^*, Γ^*) for $\Pi^\emptyset$ and $\Gamma^\emptyset$.
      i. If $1 < \omega \leq 2$, apply the first three steps of Algorithm **??**.
      ii. If $\omega > 2$, apply the first two steps of Algorithm **??**.
   (d) If $g(\Pi^\emptyset) - g(\Gamma^\emptyset)$ is nonempty, apply a fusion incorporating C_Π into a new chromosome of Π^*. If $g(\Gamma^\emptyset) - g(\Pi^\emptyset)$ is nonempty, apply a fusion incorporating C_Γ into a new chromosome of Γ^*.
   (e) Apply the final step of Algorithm **??** or Algorithm **??**, depending on the value of ω.

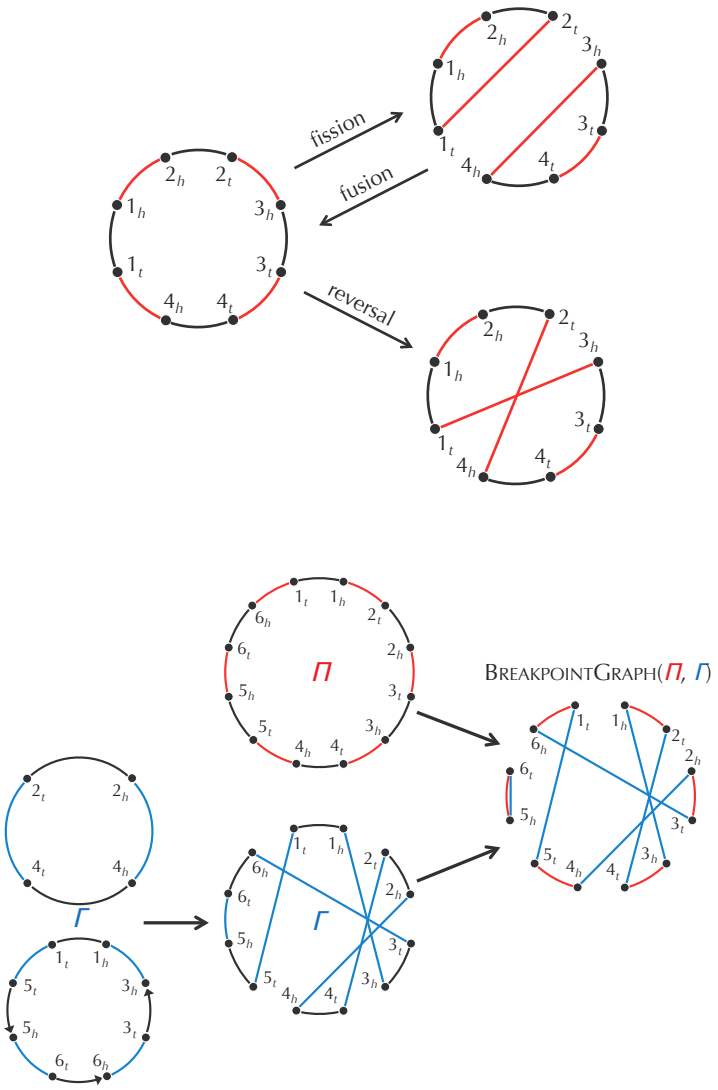


FIGURE 3.1: (Top) DCJs replace two adjacencies of a genome and incorporate three operations on circular chromosomes: reversals, fissions, and fusions. Genes are shown in black, and adjacencies are shown in red. (Bottom) The construction of the breakpoint graph of genomes Π and Γ having the same genes. First, the nodes of Γ are rearranged so that they have the same position in Π. Then, the adjacency graph is formed as the disjoint union of adjacencies of Π (red) and Γ (blue).

Appendix A

Final notes

Remove me in case of abdominal pain.

Bibliography

[AP09] M. A. Alekseyev and P. A. Pevzner. Breakpoint graphs and ancestral genome reconstructions. *Genome Research*, 19:943–957, May 2009.

[AT11] W. Arndt and Jijun Tang. Emulating insertion and deletion events in genome rearrangement analysis. In *Bioinformatics and Biomedicine (BIBM), 2011 IEEE International Conference on*, pages 105–108, Nov 2011.

[BFRnt] Laurent Bulteau, Guillaume Fertin, and Irena Rusu. Pancake flipping is hard. *CoRR*, abs/1111.0434, preprint.

[BMRS11] Marilia Braga, Raphael Machado, Leonardo Ribeiro, and Jens Stoye. On the weight of indels in genomic distances. *BMC Bioinformatics*, 12(Suppl 9):S13, 2011.

[BMS06] Anne Bergeron, Julia Mixtacki, and Jens Stoye. A unifying view of genome rearrangements. *WABI 2006. LNCS (LNBI)*, pages 163–173, 2006.

[BP96a] Vineet Bafna and Pavel A. Pevzner. Genome rearrangements and sorting by reversals. *SIAM Journal on Computing*, 25:272–289, February 1996.

[BP96b] Vineet Bafna and Pavel A. Pevzner. Genome rearrangements and sorting by reversals. *SIAM J. Comput.*, 25(2):272–289, 1996.

[BS10] Marília D.V. Braga and Jens Stoye. The solution space of sorting by DCJ. *Journal of Computational Biology*, 17(9):1145–1165, September 2010.

[BWS10] Marília D. V. Braga, Eyla Willing, and Jens Stoye. Genomic distance with DCJ and indels. *Proceedings of the 10th international conference on Algorithms in bioinformatics*, pages 90–101, 2010.

[CB95] David S. Cohen and Manuel Blum. On the problem of sorting burnt pancakes. *Discrete Applied Mathematics*, 61:105–120, July 1995.

[CFM+09] B. Chitturi, W. Fahle, Z. Meng, L. Morales, C.O. Shields, I.H. Sudborough, and W. Voit. An upper bound for sorting by prefix reversals. *Theoretical Computer Science*, 410(36):3372 – 3390, 2009. Graphs, Games and Computation: Dedicated to Professor Burkhard Monien on the Occasion of his 65th Birthday.

[Com12] Phillip E. C. Compeau. A simplified view of dcj-indel distance. In Benjamin J. Raphael and Jijun Tang, editors, *WABI*, volume 7534 of *Lecture Notes in Computer Science*, pages 365–377. Springer, 2012.

[Com13] Phillip Compeau. Dcj-indel sorting revisited. *Algorithms for Molecular Biology*, 8(1):6, 2013.

[DS38] Theodosius Dobzhansky and Alfred H. Sturtevant. Inversions in the chromosomes of drosophila pseudoobscura. *Genetics*, 23(1):28–64, January 1938.

[dSMDB13] Poly da Silva, Raphael Machado, Simone Dantas, and Marilia Braga. Dcj-indel and dcj-substitution distances with distinct operation costs. *Algorithms for Molecular Biology*, 8(1):21, 2013.

[FLR+09] Guillaume Fertin, Anthony Labarre, Irena Rusu, É Tannier, and Stéphane Vialette. *Combinatorics of Genome Rearrangements*. MIT Press, 2009.

[GP79a] William H. Gates and Christos H. Papadimitriou. Bounds for sorting by prefix reversal. *Discrete Mathematics*, 27:47–57, 1979.

[GP79b] William H. Gates and Christos H. Papadimitriou. Bounds for sorting by prefix reversal. *Discrete Mathematics*, 27(1):47 – 57, 1979.

[Har75] Harry Dweighter (pseudonym of Goodman, J.). Problem E2569. *American Mathematical Monthly*, 82:1010, 1975.

[HP99] Sridhar Hannenhalli and Pavel A. Pevzner. Transforming cabbage into turnip: Polynomial algorithm for sorting signed permutations by reversals. *Journal of the ACM*, 46:1–27, January 1999.

[HS97] Mohammad H. Heydari and I.Hal Sudborough. On the diameter of the pancake network. *Journal of Algorithms*, 25(1):67 – 94, 1997.

[MRR+08a] J. Ma, A. Ratan, B. J. Raney, B. B. Suh, W. Miller, and D. Haussler. The infinite sites model of genome evolution. *Proceedings of the National Academy of Sciences of the United States of America*, 105:14254–14261, Sep 2008.

[MRR+08b] Jian Ma, Aakrosh Ratan, Brian J. Raney, Bernard B. Suh, Webb Miller, and David Haussler. The infinite sites model of genome evolution. *Proceedings of the National Academy of Sciences of the United States of America*, 105(38):14254–14261, September 2008.

[NT84] J. H. Nadeau and B. A. Taylor. Lengths of chromosomal segments conserved since divergence of man and mouse. *Proceedings of the National Academy of Sciences of the United States of America*, 81:814–818, Feb 1984.

[Ohn73] S. Ohno. Ancient linkage groups and frozen accidents. *Nature*, 244:259–262, Aug 1973.

[PT03a] P. Pevzner and G. Tesler. Genome rearrangements in mammalian evolution: lessons from human and mouse genomes. *Genome Research*, 13:37–45, Jan 2003.

[PT03b] P. Pevzner and G. Tesler. Human and mouse genomic sequences reveal extensive breakpoint reuse in mammalian evolution. *Proceedings of the National Academy of Sciences of the United States of America*, 100:7672–7677, Jun 2003.

[SBMD12] PolyH. Silva, MarÃ■liaD.V. Braga, Raphael Machado, and Simone Dantas. Dcj-indel distance with distinct operation costs. In Ben Raphael and Jijun Tang, editors, *Algorithms in Bioinformatics*, volume 7534 of *Lecture Notes in Computer Science*, pages 378–390. Springer Berlin Heidelberg, 2012.

[SD36] A. H. Sturtevant and T. Dobzhansky. Inversions in the Third Chromosome of Wild Races of Drosophila Pseudoobscura, and Their Use in the Study of the History of the Species. *Proceedings of the National Academy of Sciences of the United States of America*, 22:448–450, Jul 1936.

[Stu21] A. H. Sturtevant. A Case of Rearrangement of Genes in Drosophila. *Proceedings of the National Academy of Sciences of the United States of America*, 7:235–237, Aug 1921.

[TZS09] Eric Tannier, Chunfang Zheng, and David Sankoff. Multichromosomal median and halving problems under different genomic distances. *BMC Bioinformatics*, 10(1):120, 2009.

[YAF05a] S. Yancopoulos, O. Attie, and R. Friedberg. Efficient sorting of genomic permutations by translocation, inversion and block interchange. *Bioinformatics*, 21:3340–3346, Aug 2005.

[YAF05b] Sophia Yancopoulos, Oliver Attie, and Richard Friedberg. Efficient sorting of genomic permutations by translocation, inversion and block interchange. *Bioinformatics*, 21(16):3340–3346, 2005.

[YF09] Sophia Yancopoulos and Richard Friedberg. DCJ path formulation for genome transformations which include insertions, deletions, and duplications. *Journal of Computational Biology*, 16(10):1311–1338, October 2009.

[ZB09] H. Zhao and G. Bourque. Recovering genome rearrangements in the mammalian phylogeny. *Genome Research*, 19:934–942, May 2009.