Greedy Algorithms And Genome Rearrangements

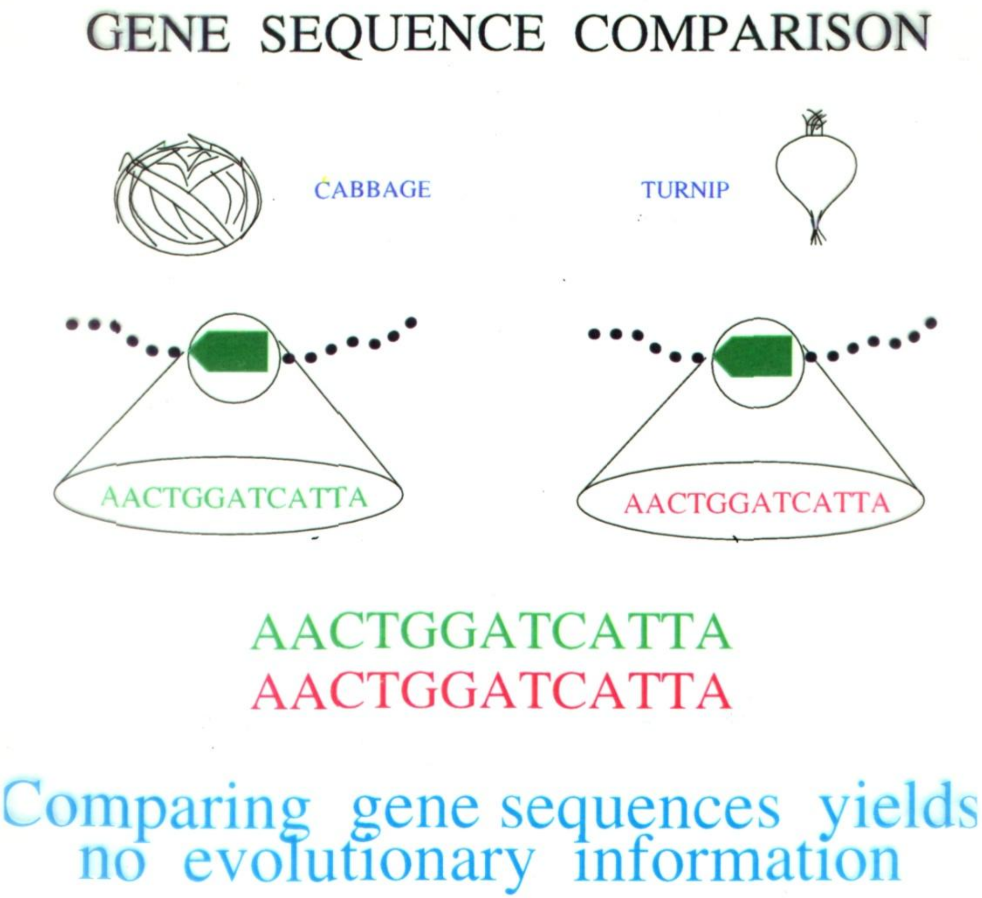
Outline

- Transforming Cabbage into Turnip
- Genome Rearrangements
- Sorting By Reversals
- Pancake Flipping Problem
- Greedy Algorithm for Sorting by Reversals
- Approximation Algorithms
- Breakpoints: a Different Face of Greed
- Breakpoint Graphs

Turnip vs Cabbage: Look and Taste Different

- Although cabbages and turnips share a recent common ancestor, they look and taste different

Turnip vs Cabbage: Comparing Gene Sequences Yields No Evolutionary Information



GENE SEQUENCE COMPARISON

AACTGGATCATTA
AACTGGATCATTA

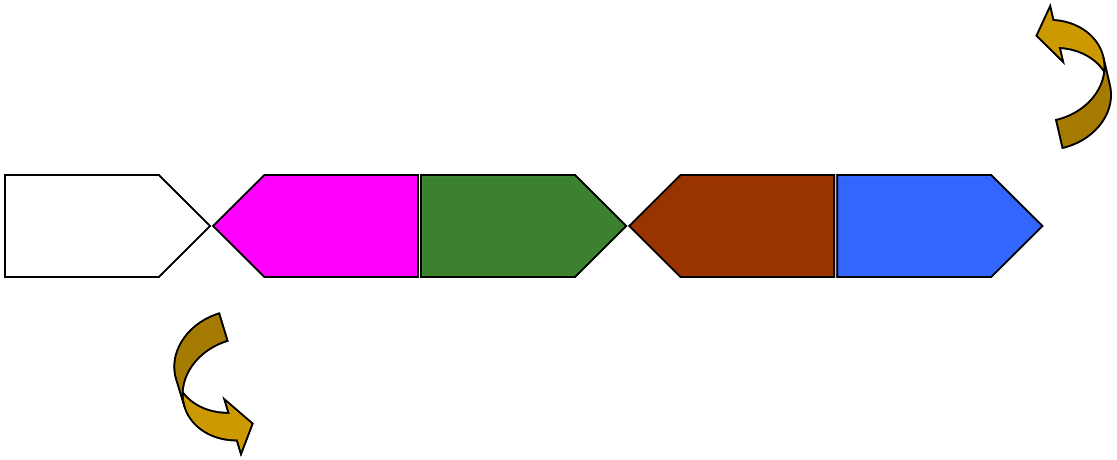
Comparing gene sequences yields
no evolutionary information

Turnip vs Cabbage: Almost Identical mtDNA gene sequences

- In 1980s Jeffrey Palmer studied evolution of plant organelles by comparing mitochondrial genomes of the cabbage and turnip
- 99% similarity between genes
- These surprisingly identical gene sequences differed in gene order
- This study helped pave the way to analyzing genome rearrangements in molecular evolution

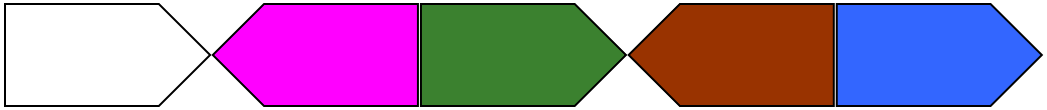
Turnip vs Cabbage: Different mtDNA Gene Order

- Gene order comparison:



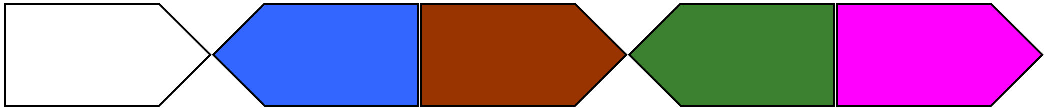
Turnip vs Cabbage: Different mtDNA Gene Order

- Gene order comparison:



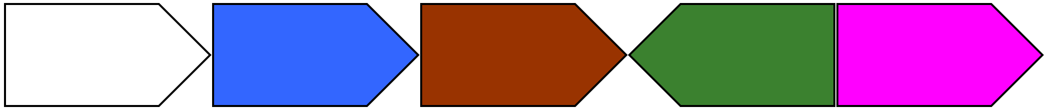
Turnip vs Cabbage: Different mtDNA Gene Order

- Gene order comparison:



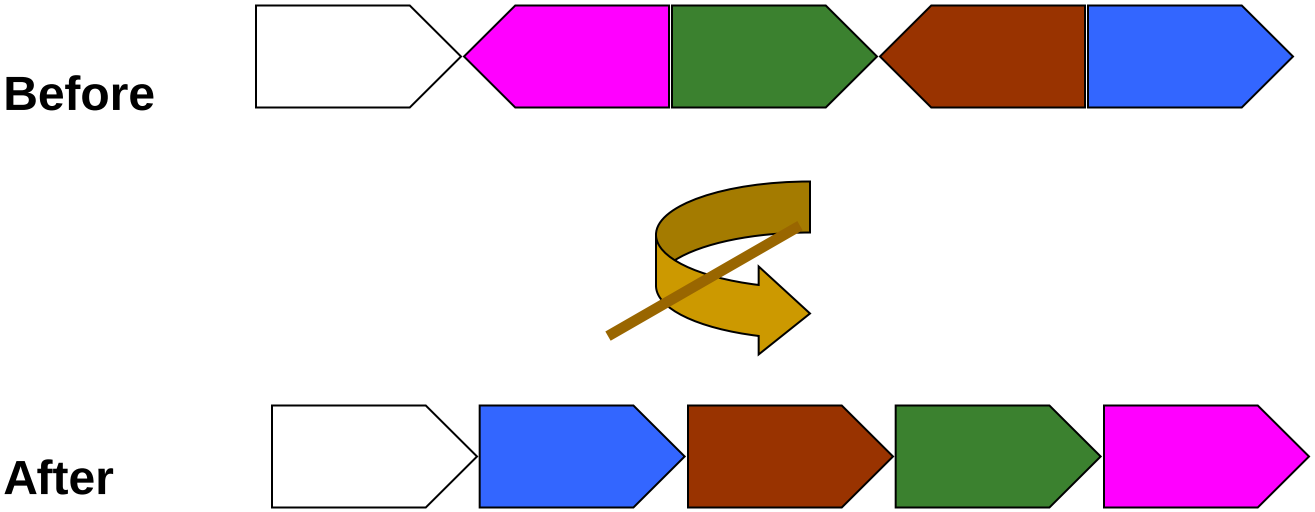
Turnip vs Cabbage: Different mtDNA Gene Order

- Gene order comparison:



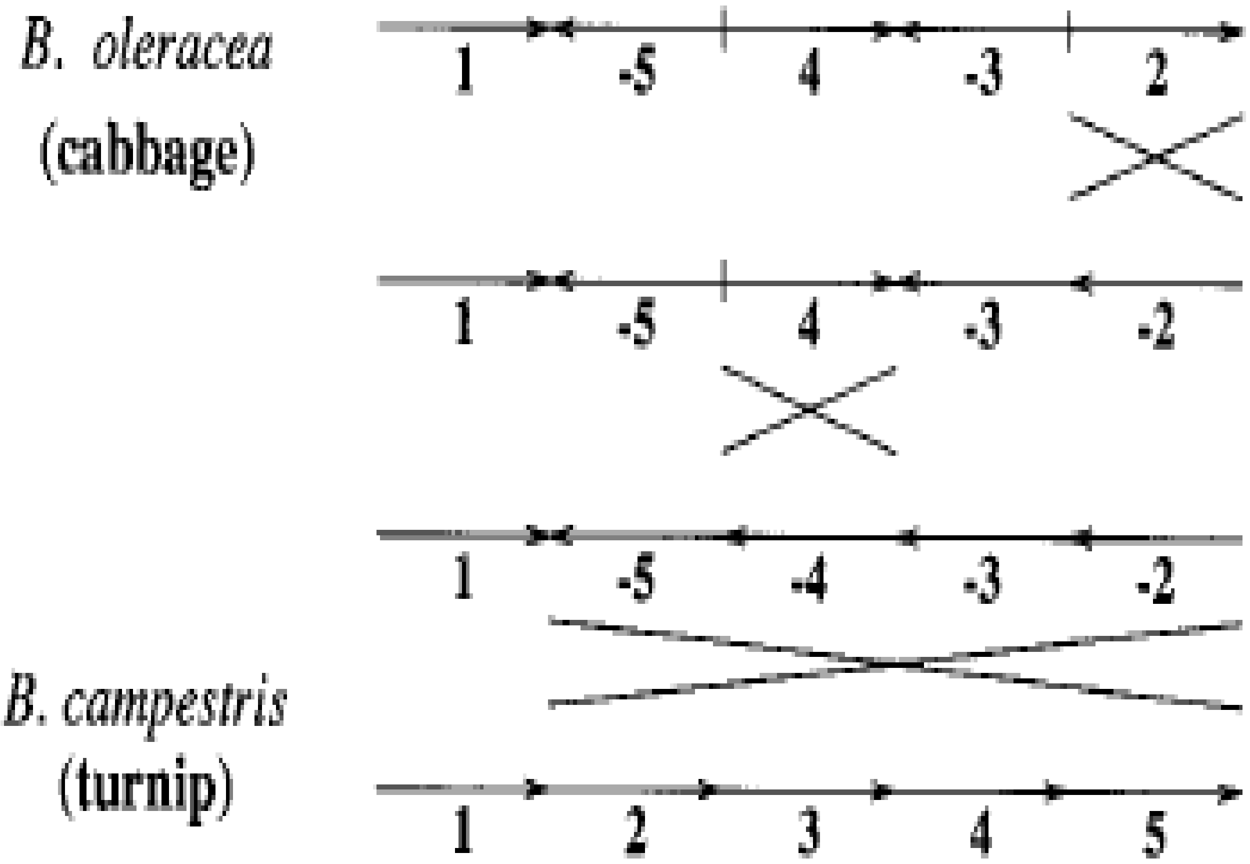
Turnip vs Cabbage: Different mtDNA Gene Order

- Gene order comparison:

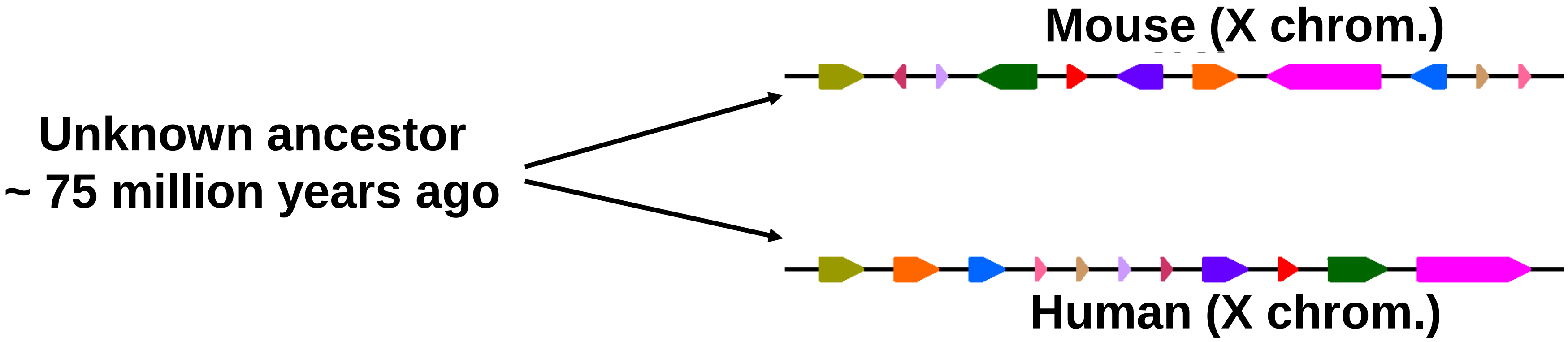


Evolution is manifested as the divergence in gene order

Transforming Cabbage into Turnip



Genome rearrangements



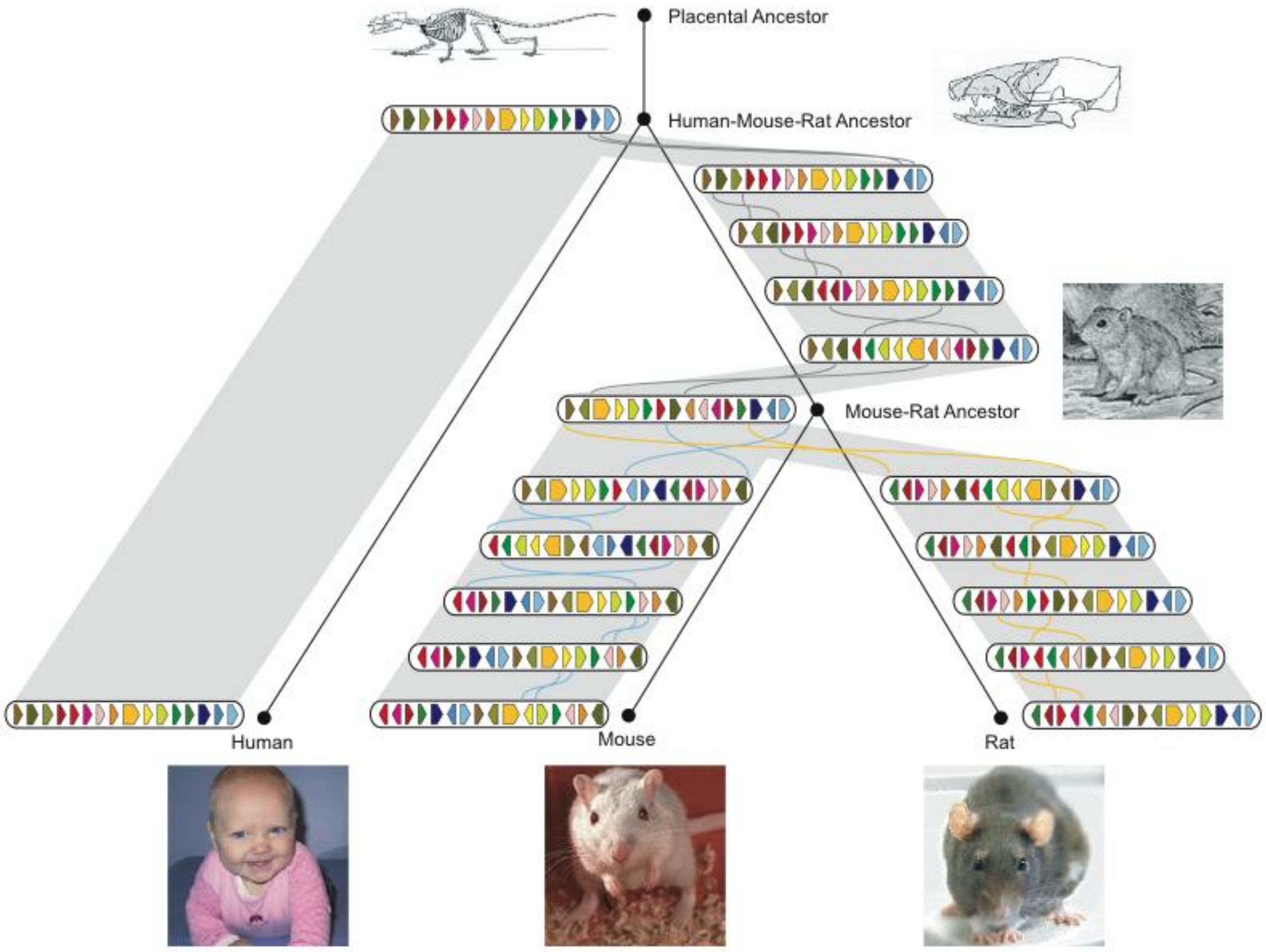
Mouse (X chrom.)

Unknown ancestor
~ 75 million years ago

Human (X chrom.)

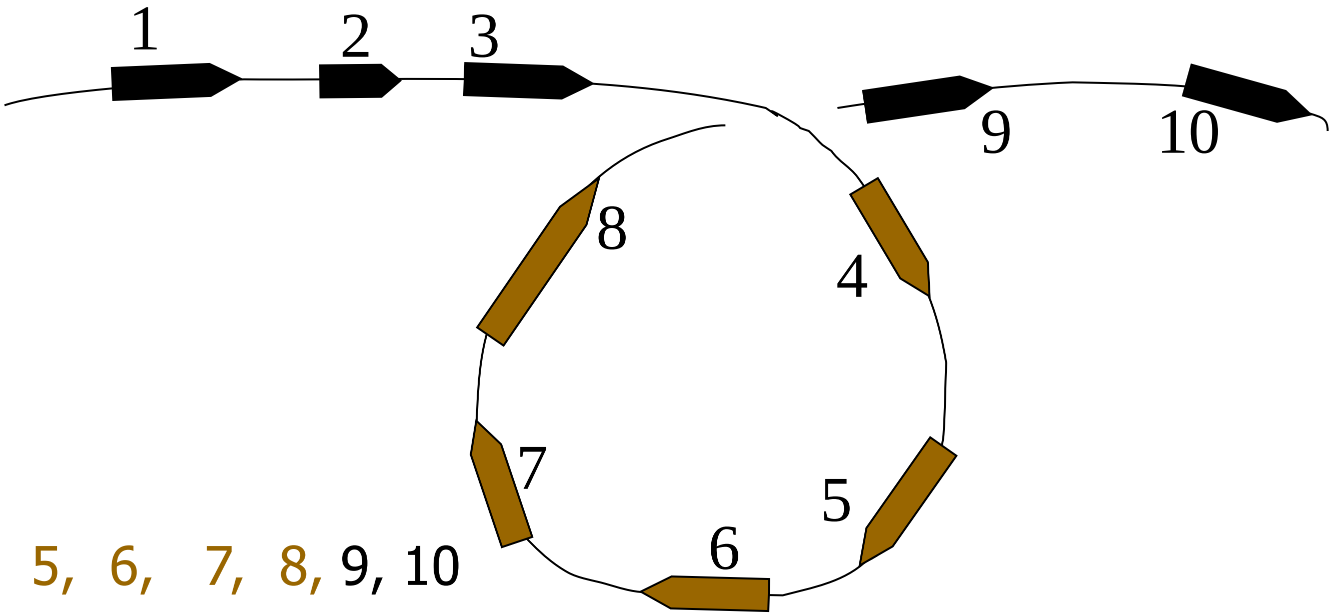
- What are the similarity (synteny) blocks and how to find them?
- What is the architecture of the ancestral genome?
- What is the evolutionary scenario for transforming one genome into the other?

History of Chromosome X



Rat Consortium, *Nature*, 2004

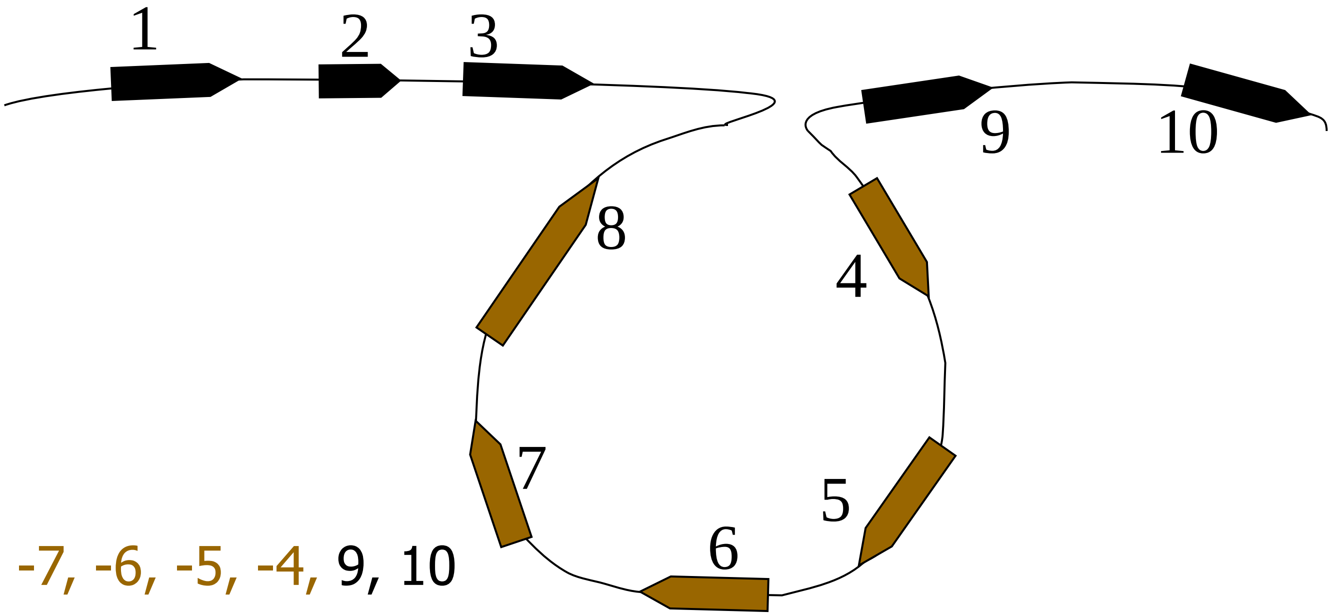
Reversals



1, 2, 3, 4, 5, 6, 7, 8, 9, 10

- Blocks represent conserved genes.

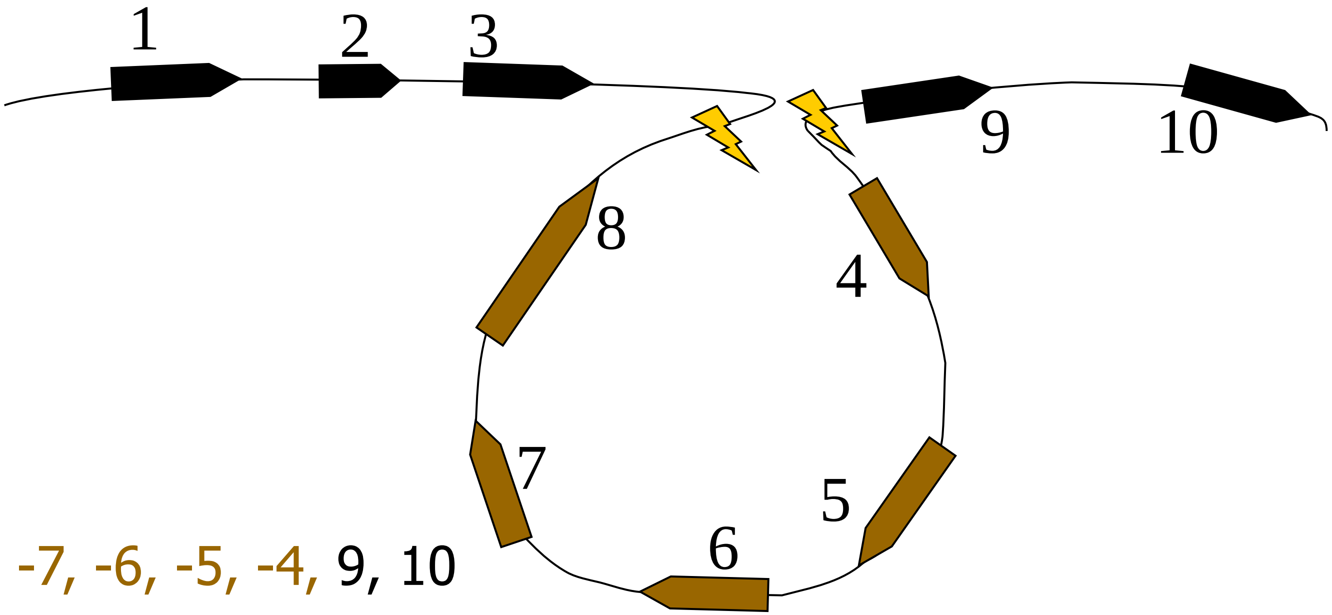
Reversals



1, 2, 3, -8, -7, -6, -5, -4, 9, 10

- Blocks represent conserved genes.
- In the course of evolution or in a clinical context, blocks 1,…,10 could be misread as 1, 2, 3, -8, -7, -6, -5, -4, 9, 10.

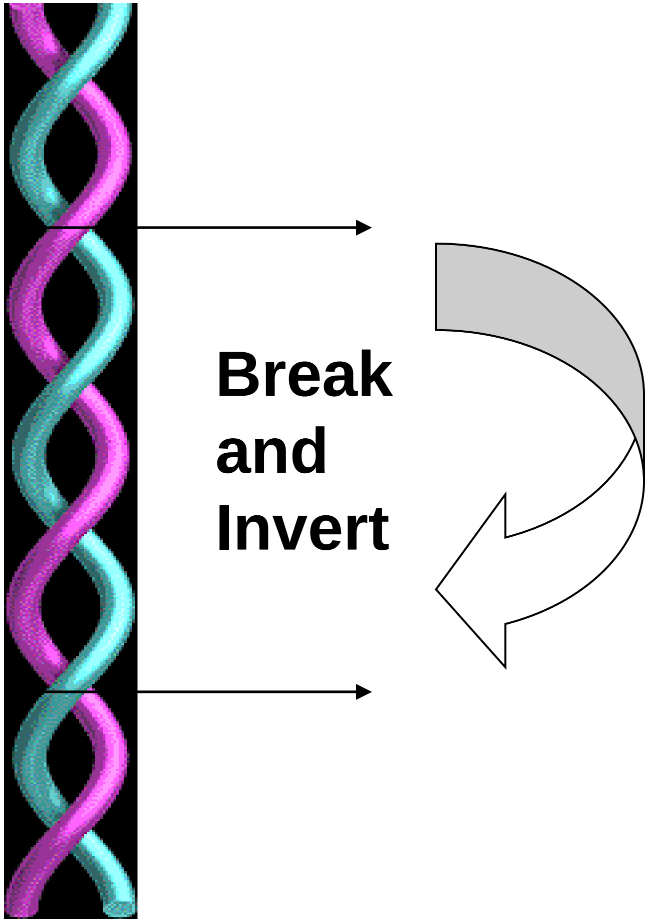
Reversals and Breakpoints



1, 2, 3, -8, -7, -6, -5, -4, 9, 10

The reversion introduced two *breakpoints* ⚡ (disruptions in order).

Reversals: Example



Break and Invert

5’ ATGCCTGTACTA 3’

3’ TACGGACATGAT 5’

5’ ATGTACAGGCTA 3’

3’ TACATGTCCGAT 5’

Types of Rearrangements

Reversal

1 2 3 4 5 6 → 1 2 -5 -4 -3 6

Translocation

1 2 3 → 1 2 6

4 5 6 → 4 5 3

Fusion

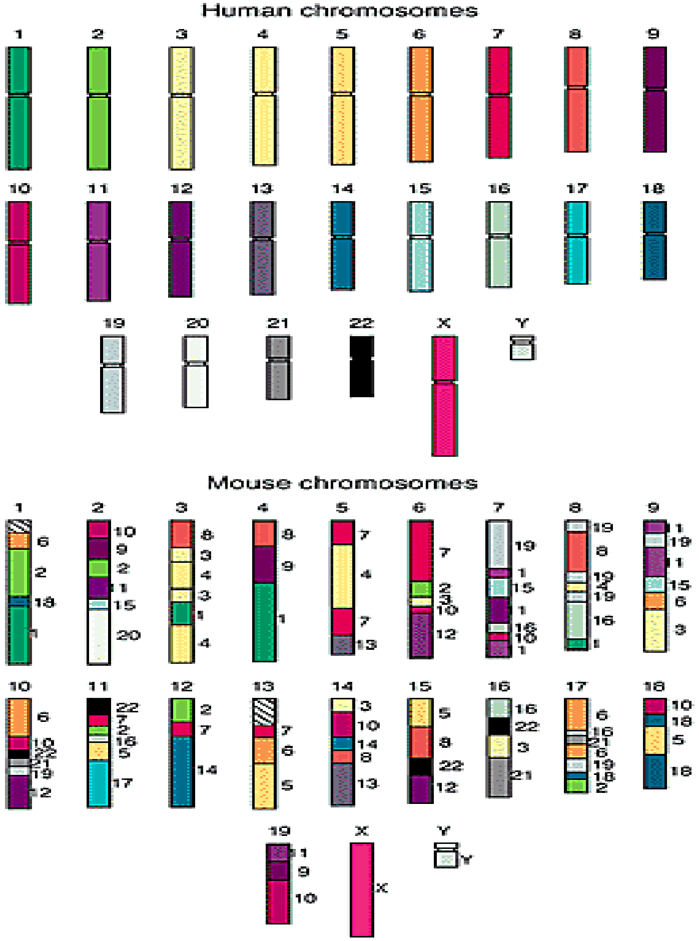
1 2 3 4

5 6 ⇄ 1 2 3 4 5 6

Fission

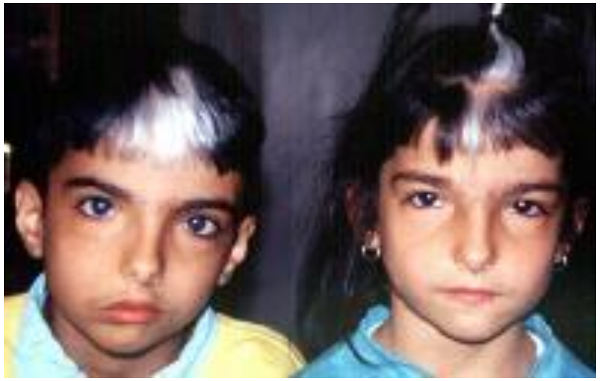
Comparative Genomic Architectures: Mouse vs Human Genome

- Humans and mice have similar genomes, but their genes are ordered differently
- ~245 rearrangements
  - Reversals
  - Fusions
  - Fissions
  - Translocation



Waardenburg’s Syndrome: Mouse Provides Insight into Human Genetic Disorder

- Waardenburg’s syndrome is characterized by pigmentary dysphasia (such as two differently colored eyes coupled with hearing problem).
- Gene implicated in the disease was linked to human chromosome 2 but it was not clear where exactly it is located on chromosome 2

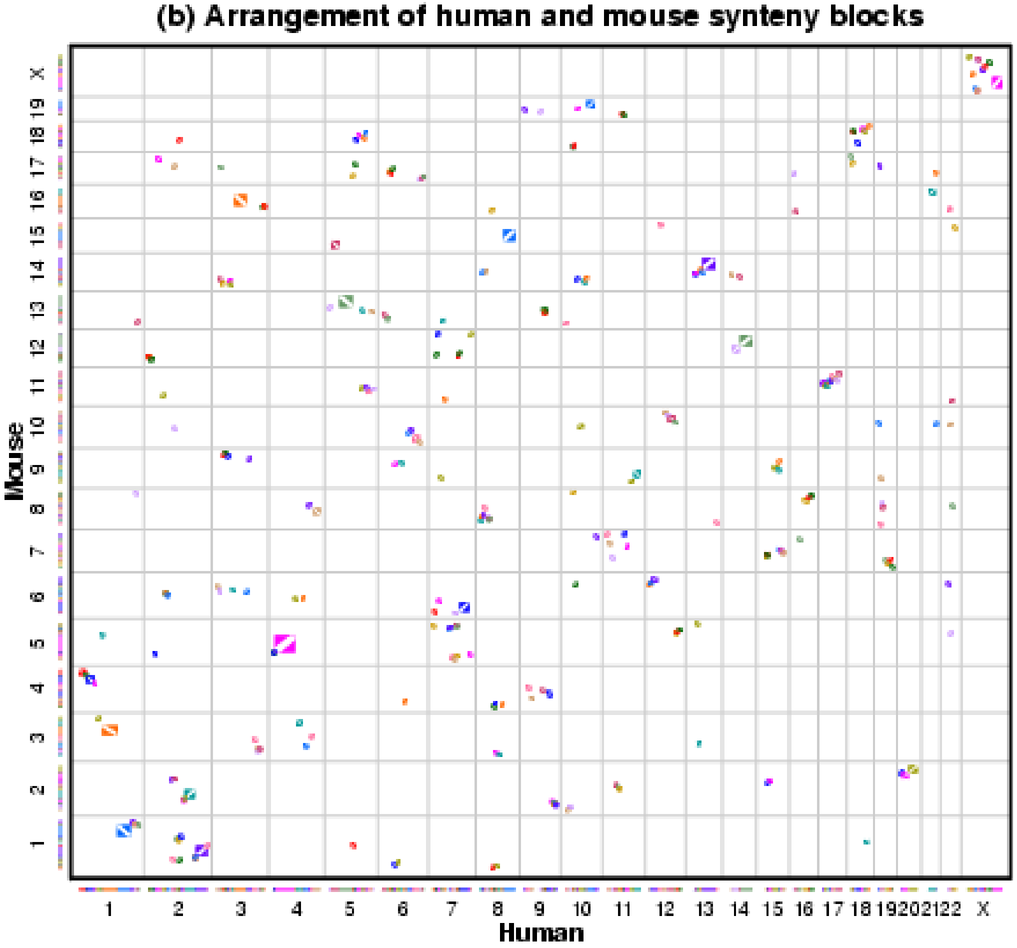


Waardenburg’s syndrome and splotch mice

- A breed of mice (with splotch gene) had similar symptoms caused by the same type of gene as in humans
- Scientists succeeded in identifying location of gene responsible for disorder in mice
- Finding the gene in mice gives clues to where the same gene is located in humans

Comparative Genomic Architecture of Human and Mouse Genomes

To locate where corresponding gene is in humans, we have to analyze the relative architecture of human and mouse genomes



Gene Order

There are groups of genes in a mice that appear in the same order as they do in humans. These genes are likely to be present in some common ancient mammalian genome. The human genome is just a mouse genome cut into about 300 *synteney blocks* that have been pasted together in a different order.

For example, the chromosome 2 in humans is built from fragments that are similar to mouse genomes residing in chromosome 1,2,3,5,6,7,10,11,12,14 and 17.

This phenomenon is called *genome rearrangement.* About 250 genome rearrangements have occurred since the divergence of human and mice 80 million years ago.

Genome Rearrangement

The combinatorial problem that we are interested is to find a minimum (most parsimonious) number of reversals or inversions that transform one genome into another. While there is no guarantee that this scenario represents an actual evolutionary sequence (excluding translocation, fusion or fission), this gives a lower bound on the number of rearrangement events that occurred and indicates similarity between two species.

The exhaustive gene finding algorithms that we discussed earlier do not work well for rearrangement studies because the number of possible variants is very large.

Reversals and Gene Orders

- The order of genes in a genome can be represented by a permutation (For simplicity we are ignoring directionality which needs signed permutation) ***π*:**

$$\pi = \pi_1 \text{ ------ } \pi_{i-1} \underbrace{\pi_i \, \pi_{i+1} \text{ ------ } \pi_{j-1} \, \pi_j}_{\rho(i,j)} \pi_{j+1} \text{ ----- } \pi_n$$

$$\Big\downarrow$$

$$\pi_1 \text{ ------ } \pi_{i-1} \, \pi_j \, \pi_{j-1} \text{ ------ } \pi_{i+1} \, \pi_i \, \pi_{j+1} \text{ ----- } \pi_n$$

- Reversal ρ (*i, j*) reverses (flips) the elements from *i* to *j* in π

Reversals: Example

$$\pi = 1\ 2\ \underline{3\ 4\ 5}\ 6\ 7\ 8$$

$$\rho(3,5) \qquad \Big\downarrow$$

$$1\ 2\ 5\ 4\ 3\ 6\ 7\ 8$$

Reversals: Example

$$\begin{tikzcd}
\pi = 1\ 2\ \underline{3\ 4\ 5}\ 6\ 7\ 8 \arrow[d, "\rho(3,5)"'] \\
1\ 2\ 5\ 4\ \underline{3\ 6}\ 7\ 8 \arrow[d, "\rho(5,6)"'] \\
1\ 2\ 5\ 4\ 6\ 3\ 7\ 8
\end{tikzcd}$$

Reversal Distance Problem

- Goal: Given two permutations, find the shortest series of reversals that transforms one into another

- Input: Permutations $\boldsymbol{\pi}$ and $\boldsymbol{\sigma}$

- Output: A series of reversals $\rho_1,\ldots\rho_t$ transforming $\boldsymbol{\pi}$ into $\boldsymbol{\sigma}$, such that *t* is minimum
- ***t*** - reversal distance between π and σ
- ***d***(π, σ) - smallest possible value of *t*, given π and σ.

If we set $\sigma = (1,2,3,\quad, n)$, it becomes Sorting by Reversal problem and the input is simply $\boldsymbol{\pi}$

Sorting By Reversals Problem

- Goal: Given a permutation, find a shortest series of reversals that transforms it into the identity permutation ($1\ 2\ \ldots\ n$)
- Input: Permutation π
- Output: A series of reversals $\rho_1, \ldots\ \rho_{\mathbf{t}}$ transforming π into the identity permutation such that t is minimum

Sorting By Reversals: Example

- $t = d(\pi)$ - reversal distance of π
- Example :

$$
\begin{array}{ll}
\pi = & \underline{3\ \ 4}\ \ 2\ \ 1\ \ 5\ \ 6\ \ 7\ \ 10\ \ 9\ \ 8 \\
 & 4\ \ 3\ \ 2\ \ 1\ \ 5\ \ 6\ \ 7\ \ \underline{10\ \ 9\ \ 8} \\
 & \underline{4\ \ 3\ \ 2\ \ 1}\ \ 5\ \ 6\ \ 7\ \ 8\ \ 9\ \ 10 \\
 & 1\ \ 2\ \ 3\ \ 4\ \ 5\ \ 6\ \ 7\ \ 8\ \ 9\ \ 10
\end{array}
$$

So $d(\pi) = 3$

Sorting by reversals: 5 steps

Step 0: π $2\ \underline{-4\ -3}\ 5\ -8\ -7\ -6\ 1$

Step 1: $2\ 3\ 4\ 5\ \underline{-8\ -7\ -6}\ 1$

Step 2: $2\ 3\ 4\ 5\ 6\ 7\ 8\ \underline{1}$

Step 3: $\underline{2\ 3\ 4\ 5\ 6\ 7\ 8}\ -1$

Step 4: $\underline{-8\ -7\ -6\ -5\ -4\ -3\ -2\ -1}$

Step 5: γ $1\ 2\ 3\ 4\ 5\ 6\ 7\ 8$

Sorting by reversals: 4 steps

Step 0: π $2\ \ \underline{-4\ \ -3}\ \ 5\ \ -8\ \ -7\ \ -6\ \ 1$

Step 1: $\underline{2\ \ 3\ \ 4\ \ 5}\ \ -8\ \ -7\ \ -6\ \ 1$

Step 2: $-5\ \ -4\ \ -3\ \ -2\ \ \underline{-8\ \ -7\ \ -6\ \ 1}$

Step 3: $\underline{-5\ \ -4\ \ -3\ \ -2\ \ -1}\ \ 6\ \ 7\ \ 8$

Step 4: γ $1\ \ 2\ \ 3\ \ 4\ \ 5\ \ 6\ \ 7\ \ 8$

Sorting by reversals: 4 steps

Step 0: $\pi \quad 2 \ \underline{-4 \ -3} \ 5 \ -8 \ -7 \ -6 \ 1$

Step 1: $\underline{2 \ 3 \ 4 \ 5} \ -8 \ -7 \ -6 \ 1$

Step 2: $-5 \ -4 \ -3 \ -2 \ \underline{-8 \ -7 \ -6 \ 1}$

Step 3: $\underline{-5 \ -4 \ -3 \ -2 \ -1} \ 6 \ 7 \ 8$

Step 4: $\gamma \quad 1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \ 8$

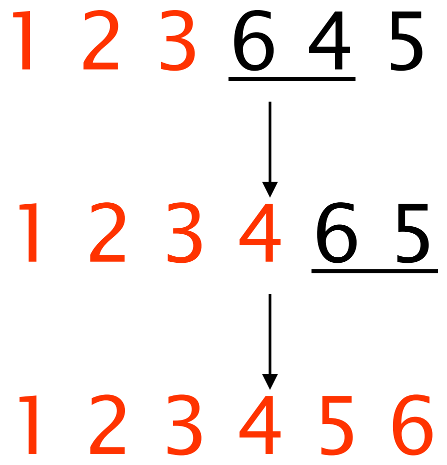
What is the reversal distance for this permutation? Can it be sorted in 3 steps?

Sorting By Reversals: A Greedy Algorithm

- If sorting permutation π = 1 2 3 6 4 5, the first three elements are already in order so it does not make any sense to break them.
- The length of the already sorted prefix of π is denoted *prefix*(π)
  - *prefix*(π) = 3
- This results in an idea for a greedy algorithm: increase *prefix*(π) at every step

Greedy Algorithm: An Example

- Doing so, π can be sorted



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

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

$1\ 2\ 3\ 4\ 5\ 6$

- Number of steps to sort permutation of length *n* is at most *(n – 1)*

Greedy Algorithm: Pseudocode

```
SimpleReversalSort($\pi$)
for $i \leftarrow 1$ to $n - 1$
    $j \leftarrow$ position of element $i$ in $\pi$ (i.e., $\pi_j = i$)
    if $j \neq i$
        $\pi \leftarrow \pi \cdot \rho(i, j)$
        output $\pi$
    if $\pi$ is the identity permutation
        return
```

Pancake Flipping Problem

- The chef is sloppy; he prepares an unordered stack of pancakes of different sizes
- The waiter wants to rearrange them (so that the smallest winds up on top, and so on, down to the largest at the bottom)
- He does it by flipping over several from the top, repeating this as many times as necessary

Christos Papadimitrou and Bill Gates flip pancakes

Pancake Flipping Problem: Formulation

- Goal: Given a stack of *n* pancakes, what is the minimum number of flips to rearrange them into perfect stack?
- Input: Permutation π
- Output: A series of prefix reversals $\rho_1, \ldots \rho_{\mathbf{t}}$ transforming π into the identity permutation such that *t* is minimum

Pancake Flipping Problem: Greedy Algorithm

- Greedy approach: 2 prefix reversals at most to place a pancake in its right position, *2n – 2* steps total

  at most

- William Gates and Christos Papadimitriou showed in the mid-1970s that this problem can be solved by at most *5/3 (n + 1) prefix reversals*

Analyzing SimpleReversalSort

- SimpleReversalSort chooses the “best” reversal at every step – simply increase *prefix(π)* - *but* does not guarantee the smallest number of reversals. It takes five steps on π = 6 1 2 3 4 5 :
  - Step 1: 1 6 2 3 4 5
  - Step 2: 1 2 6 3 4 5
  - Step 3: 1 2 3 6 4 5
  - Step 4: 1 2 3 4 6 5
  - Step 5: 1 2 3 4 5 6
- But it can be sorted in two steps:

  π = 6 1 2 3 4 5
  - Step 1: 5 4 3 2 1 6
  - Step 2: 1 2 3 4 5 6

So, SimpleReversalSort(π) is not optimal

Approximation Algorithms

- These algorithms find approximate solutions rather than optimal solutions
- The approximation ratio of an algorithm A on input π is:

$$A(\pi) \,/\, OPT(\pi)$$

where

$A(\pi)$ -solution produced by algorithm A

$OPT(\pi)$ - optimal solution of the problem

Approximation Ratio/Performance Guarantee

- Approximation ratio (performance guarantee) of algorithm A: max approximation ratio of all inputs of size *n, that is, as*
- $\max_{|\pi| = n} [A(\pi) / OPT(\pi)]$

We assume that A is a minimization algorithm which minimizes the objective function, that is, it minimizes the maximum number of reversals and the ratio gives the worst case scenario.

Adjacencies and Breakpoints

$$\pi = \pi_1\pi_2\pi_3\ldots\pi_{n-1}\pi_n$$

- A pair of elements π_{i} and π_{i+1} are adjacent if

$$\pi_{i+1} = \pi_i \pm 1$$

- For example:

$$\pi = 1\ \ 9\ \ \underline{3\ \ 4}\ \ \underline{7\ \ 8}\ \ 2\ \ \underline{6\ \ 5}$$

- (3, 4) or (7, 8) and (6,5) are adjacent pairs

Breakpoints: An Example

There is a breakpoint between any adjacent element that are non-consecutive:

$$\pi = 1 \mid 9 \mid 3 \;\; 4 \mid 7 \;\; 8 \mid 2 \mid 6 \;\; 5$$

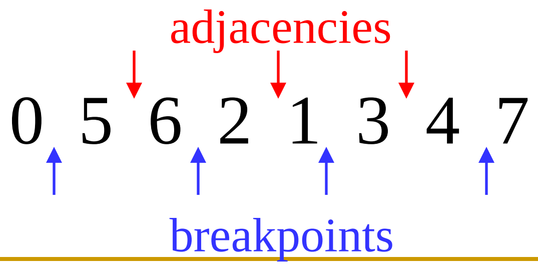
- Pairs (1,9), (9,3), (4,7), (8,2) and (2,5) form breakpoints of permutation π
- $b(\pi)$ - # breakpoints in permutation π

Adjacency & Breakpoints

- An adjacency - a pair of adjacent elements that are consecutive

- A breakpoint - a pair of adjacent elements that are not consecutive

$\pi = 5\ \ 6\ \ 2\ \ 1\ \ 3\ \ 4 \longrightarrow$ Extend π with $\pi_0 = 0$ and $\pi_7 = 7$



adjacencies

0 5 6 2 1 3 4 7

breakpoints

Extending Permutations

- We put two elements $\pi_0 = 0$ and $\pi_{n+1} = n+1$ at the ends of π

  Example:

$$\pi = 1 \mid 9 \mid 3 \;\; 4 \mid 7 \;\; 8 \mid 2 \mid 6 \;\; 5$$

$\downarrow$ Extending with *0* and *10*

$$\pi = 0 \; 1 \mid 9 \mid 3 \;\; 4 \mid 7 \;\; 8 \mid 2 \mid 6 \;\; 5 \mid 10$$

Note: A new breakpoint was created after extending

Reversal Distance and Breakpoints

- Each reversal eliminates at most 2 breakpoints.

$\pi = 2\ \ 3\ \ 1\ \ 4\ \ 6\ \ 5$

$0 \mid \underline{2\ \ 3 \mid 1}\ \mid 4 \mid 6\ \ 5 \mid 7$ $\qquad b(\pi) = 5$

$0\ \ 1 \mid \underline{3\ \ 2}\ \mid 4 \mid 6\ \ 5 \mid 7$ $\qquad b(\pi) = 4$

$0\ \ 1\ \ 2\ \ 3\ \ 4 \mid \underline{6\ \ 5}\ \mid 7$ $\qquad b(\pi) = 2$

$0\ \ 1\ \ 2\ \ 3\ \ 4\ \ 5\ \ 6\ \ 7$ $\qquad b(\pi) = 0$

Reversal Distance and Breakpoints

- Each reversal eliminates at most 2 breakpoints.
- This implies:

reversal distance ≥ #breakpoints / 2

π = 2 3 1 4 6 5

$\mathit{0} \mid \underline{2\ \ 3 \mid 1} \mid 4 \mid 6\ \ 5 \mid \mathit{7}$ $\qquad b(\pi) = 5$

$\mathit{0}\ \ 1 \mid \underline{3\ \ 2} \mid 4 \mid 6\ \ 5 \mid \mathit{7}$ $\qquad b(\pi) = 4$

$\mathit{0}\ \ 1\ \ 2\ \ 3\ \ 4 \mid \underline{6\ \ 5} \mid \mathit{7}$ $\qquad b(\pi) = 2$

$\mathit{0}\ \ 1\ \ 2\ \ 3\ \ 4\ \ 5\ \ 6\ \ \mathit{7}$ $\qquad b(\pi) = 0$

Sorting By Reversals: A Better Greedy Algorithm

```
BreakPointReversalSort($\pi$)
while $b(\pi) > 0$
    Among all possible reversals, choose reversal $\rho$ minimizing $b(\pi \bullet \rho)$
    $\pi \leftarrow \pi \bullet \rho(i, j)$
    output $\pi$
return
```

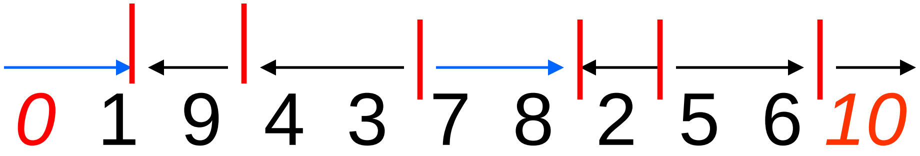
Sorting By Reversals: A Better Greedy Algorithm

```
BreakPointReversalSort($\pi$)
while $b(\pi) > 0$
    Among all possible reversals,
    choose reversal $\rho$ minimizing $b(\pi \cdot \rho)$
    $\pi \leftarrow \pi \cdot \rho(i, j)$
    output $\pi$
return
```

Problem: this algorithm may work forever. How can we make sure that removing some breakpoints does not introduce others, leading to an endless cycle.

Strips

- Strip: an interval between two consecutive breakpoints in a permutation, that is, any maximal segments without breakpoints. Strips are further divided into:
  - Decreasing strip: *strip* of elements in decreasing order (e.g. 6 5 and 3 2).
  - Increasing strip: *strip* of elements in increasing order (e.g. 7 8)



0 1 9 4 3 7 8 2 5 6 *10*

A single-element strip can be declared either increasing or decreasing. We will choose to declare them as decreasing with exception of the strips with *0* and *n+1*

Reducing the Number of Breakpoints

Theorem 1:

If permutation π contains at least one decreasing strip, then there exists a reversal ρ which decreases the number of breakpoints (i.e. $b(\pi \bullet \rho) < b(\pi)$)

Things To Consider

- For π = 1 4 6 5 7 8 3 2

  $\mathit{0}\ \ 1\,|\,4\,|\ 6\ \ 5\,|\,7\ \ 8\,|\,3\ \ 2\,|\,\mathit{9}$ $\qquad b(\pi) = 5$

  - Choose decreasing strip with the smallest element *k* in π (*k* = 2 in this case)

Things To Consider (cont'd)

- For π = 1 4 6 5 7 8 3 2

  $0\ \ 1 \mid 4 \mid 6\ \ 5 \mid 7\ \ 8 \mid 3\ \ 2 \mid 9$ $\qquad b(\pi) = 5$

  - Choose decreasing strip with the smallest element k in π (k = 2 in this case)

Things To Consider (cont'd)

- For π = 1 4 6 5 7 8 3 2

  $0\ \ 1 \mid 4 \mid 6\ \ 5 \mid 7\ \ 8 \mid 3\ \ 2 \mid 9$ $b(\pi) = 5$

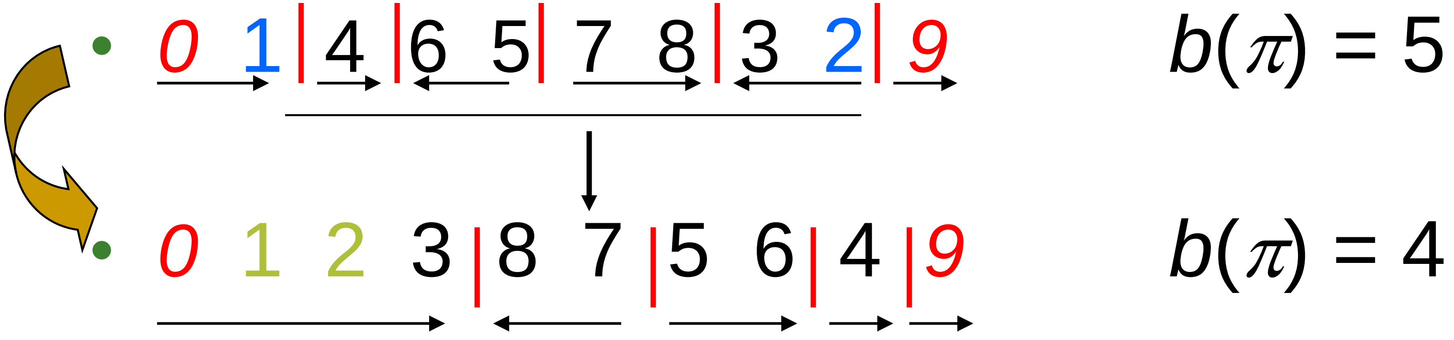
  - Choose decreasing strip with the smallest element *k* in π (*k* = 2 in this case)
  - Find *k – 1* in the permutation

Things To Consider (cont'd)

- For π = 1 4 6 5 7 8 3 2

  0 1 | 4 | 6 5 | 7 8 | 3 2 | *9* $b(\pi) = 5$

  - Choose decreasing strip with the smallest element *k* in π (*k* = 2 in this case)
  - Find *k – 1* in the permutation
  - Reverse the segment between *k* and *k-1*:



  - *0* 1 | 4 | 6 5 | 7 8 | 3 2 | *9* $b(\pi) = 5$
  - *0* 1 2 3 | 8 7 | 5 6 | 4 | *9* $b(\pi) = 4$

Reducing the Number of Breakpoints Again

- If there is no decreasing strip, there may be no reversal ρ that reduces the number of breakpoints (i.e. $b(\pi \bullet \rho) \geq b(\pi)$ for any reversal ρ).
- By reversing an increasing strip (# of breakpoints stay unchanged), we will create a decreasing strip at the next step. Then the number of breakpoints will be reduced in the next step (theorem 1).

Things To Consider (cont'd)

- There are no decreasing strips in π, for:

$$\pi = \underrightarrow{0\ \ 1\ \ 2}\,|\,\underrightarrow{5\ \ 6\ \ 7}\,|\,\underrightarrow{3\ \ 4}\,|\,\underrightarrow{8} \qquad b(\pi) = 3$$

$$\pi \bullet \rho(6,7) = \underrightarrow{0\ \ 1\ \ 2}\,|\,\underrightarrow{5\ \ 6\ \ 7}\,|\,\underleftarrow{4\ \ 3}\,|\,\underrightarrow{8} \qquad b(\pi) = 3$$

✓ $\rho(6,7)$ does not change the # of breakpoints

✓ $\rho(6,7)$ creates a decreasing strip thus guaranteeing that the next step will decrease the # of breakpoints.

ImprovedBreakpointReversalSort

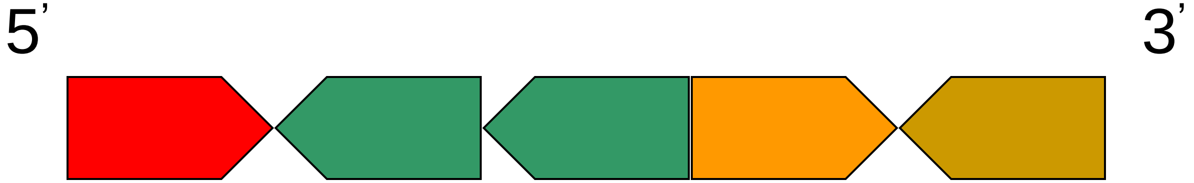
```
ImprovedBreakpointReversalSort($\pi$)
while $b(\pi) > 0$
    if $\pi$ has a decreasing strip
        Among all possible reversals, choose reversal $\rho$
                that minimizes $b(\pi \bullet \rho)$
    else
        Choose a reversal $\rho$ that flips an increasing strip in $\pi$
    $\pi \leftarrow \pi \bullet \rho$
    output $\pi$
return
```

ImprovedBreakpointReversalSort: Performance Guarantee

- *ImprovedBreakPointReversalSort* is an approximation algorithm with a performance guarantee of at most 4. Implicitly it means that the "no progress" situation is rare.
    - It eliminates at least one breakpoint in every two steps; at most $2b(\pi)$ steps
    - Approximation ratio: $2b(\pi) / d(\pi)$
    - Optimal algorithm eliminates at most 2 breakpoints in every step: $d(\pi) \geq b(\pi) / 2$
    - Performance guarantee:
        - $(2b(\pi) / d(\pi)) \geq [2b(\pi) / (b(\pi) / 2)] = 4$

Signed Permutations

- Up to this point, all permutations to sort were unsigned
- But genes have directions… so we should consider signed permutations

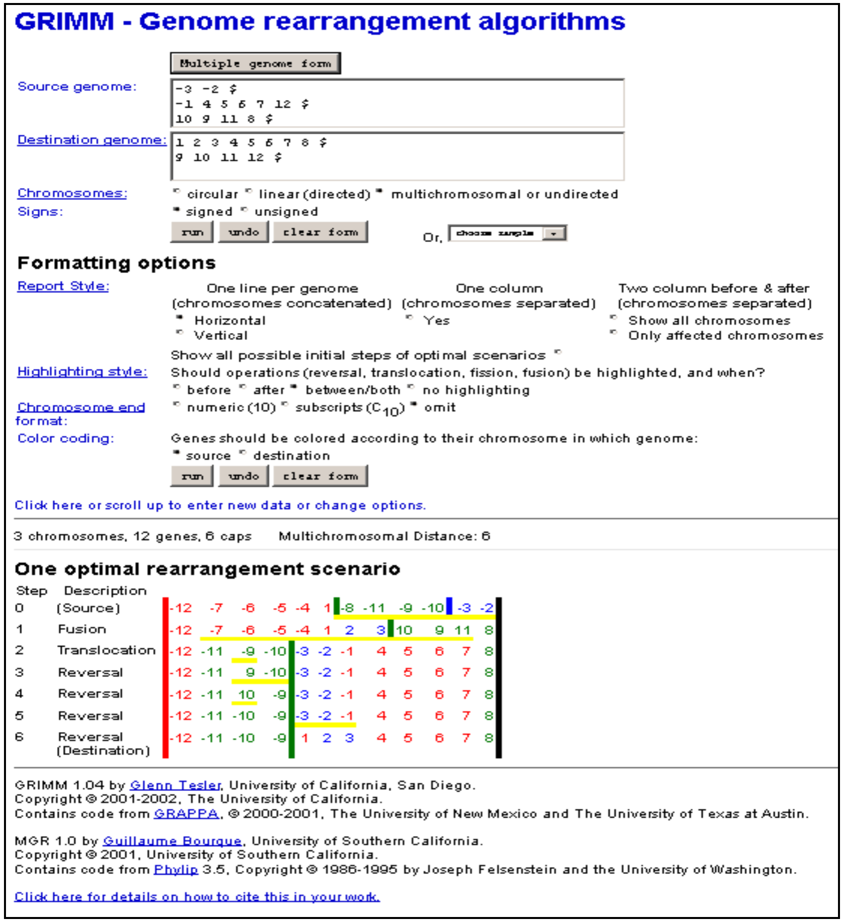


π = 1 -2 -3 4 -5

GRIMM Web Server

- Real genome architectures are represented by signed permutations
- Efficient algorithms to sort signed permutations have been developed
- GRIMM web server computes the reversal distances between signed permutations:

GRIMM Web Server



GRIMM - Genome rearrangement algorithms

Source genome:

```
-3 -2 $
-1 4 5 6 7 12 $
10 9 11 8 $
```

Destination genome:

```
1 2 3 4 5 6 7 8 $
9 10 11 12 $
```

Chromosomes: ○ circular ○ linear (directed) ● multichromosomal or undirected

Signs: ● signed ○ unsigned

Formatting options

Report Style:

One line per genome (chromosomes concatenated)	One column (chromosomes separated)	Two column before & after (chromosomes separated)
● Horizontal ○ Vertical	○ Yes	○ Show all chromosomes ○ Only affected chromosomes

Show all possible initial steps of optimal scenarios ○

Highlighting style: Should operations (reversal, translocation, fission, fusion) be highlighted, and when?
○ before ○ after ● between/both ○ no highlighting

Chromosome end format: ○ numeric (10) ○ subscripts (C_{10}) ● omit

Color coding: Genes should be colored according to their chromosome in which genome:
● source ○ destination

Click here or scroll up to enter new data or change options.

3 chromosomes, 12 genes, 6 caps   Multichromosomal Distance: 6

One optimal rearrangement scenario

```
Step  Description
0     (Source)       |-12  -7  -6  -5  -4   1|-8  -11  -9  -10|-3  -2|
1     Fusion         |-12  -7  -6  -5  -4   1   2    3|10    9   11   8|
2     Translocation  |-12 -11  -9 -10|-3  -2  -1   4   5   6   7   8|
3     Reversal       |-12 -11   9 -10|-3  -2  -1   4   5   6   7   8|
4     Reversal       |-12 -11  10  -9|-3  -2  -1   4   5   6   7   8|
5     Reversal       |-12 -11 -10  -9|-3  -2  -1   4   5   6   7   8|
6     Reversal       |-12 -11 -10  -9| 1   2   3   4   5   6   7   8|
      (Destination)
```

GRIMM 1.04 by Glenn Tesler, University of California, San Diego.
Copyright © 2001-2002, The University of California.
Contains code from GRAPPA, © 2000-2001, The University of New Mexico and The University of Texas at Austin.

MGR 1.0 by Guillaume Bourque, University of Southern California.
Copyright © 2001, University of Southern California.
Contains code from Phylip 3.5, Copyright © 1986-1995 by Joseph Felsenstein and the University of Washington.

Click here for details on how to cite this in your work.

http://www-cse.ucsd.edu/groups/bioinformatics/GRIMM

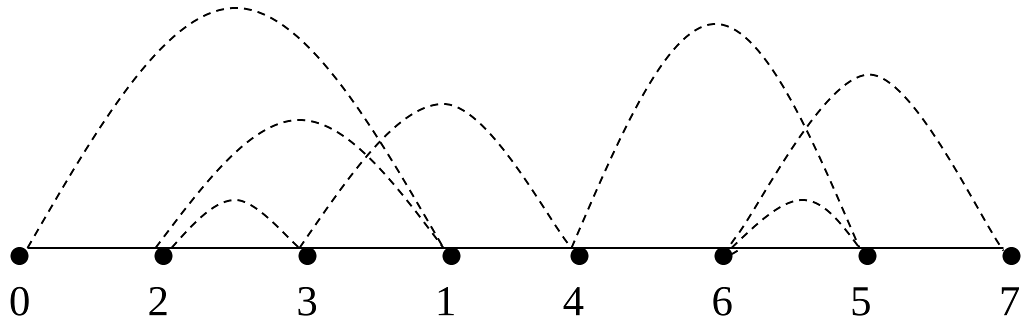
Breakpoint Graph

1) Represent the elements of the permutation π = 2 3 1 4 6 5 as vertices in a graph (ordered along a line)

2) Connect vertices in order given by π with black edges (black path)

3) Connect vertices in order given by 1 2 3 4 5 6 with grey edges (grey path)

4) Superimpose black and grey paths

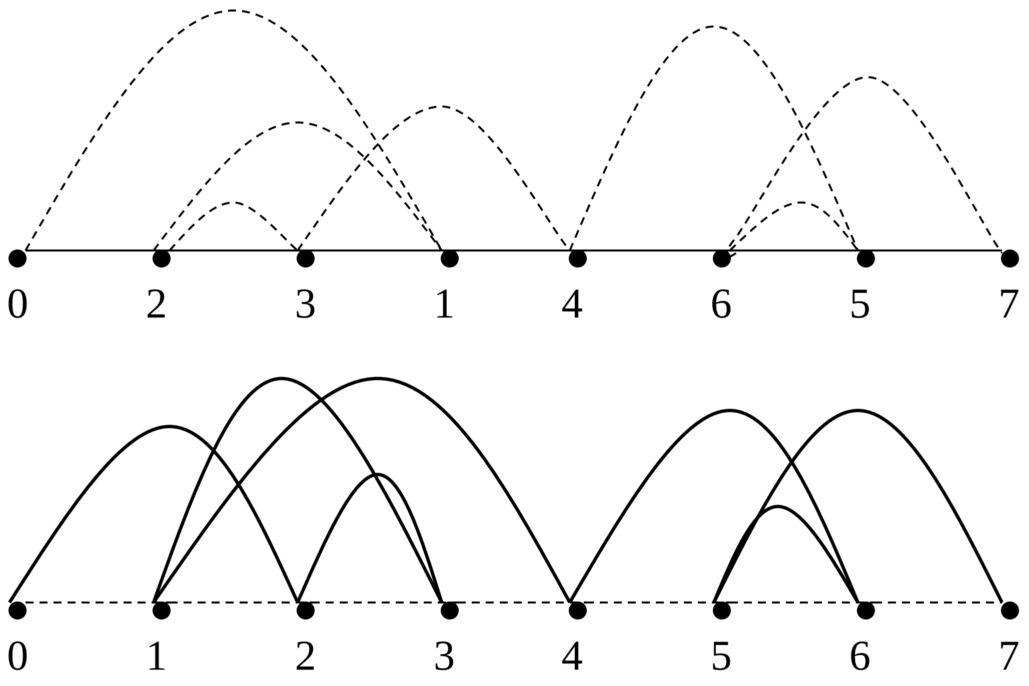


Two Equivalent Representations of the Breakpoint Graph

- Consider the following Breakpoint Graph

- If we line up the gray path (instead of black path) on a horizontal line, then we would get the following graph

- Although they may look different, these two graphs are the same

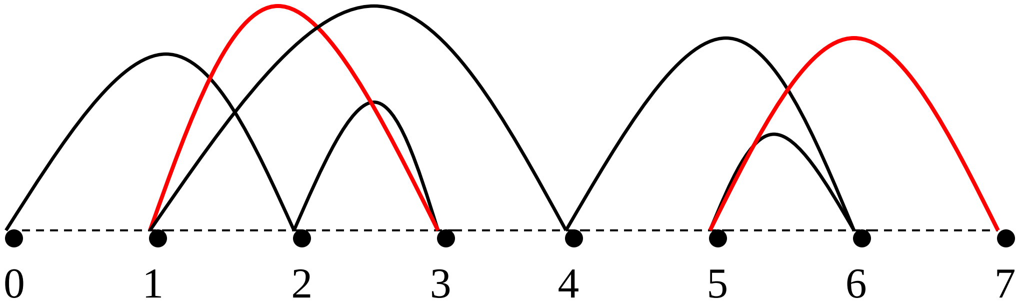


What is the Effect of the Reversal ?

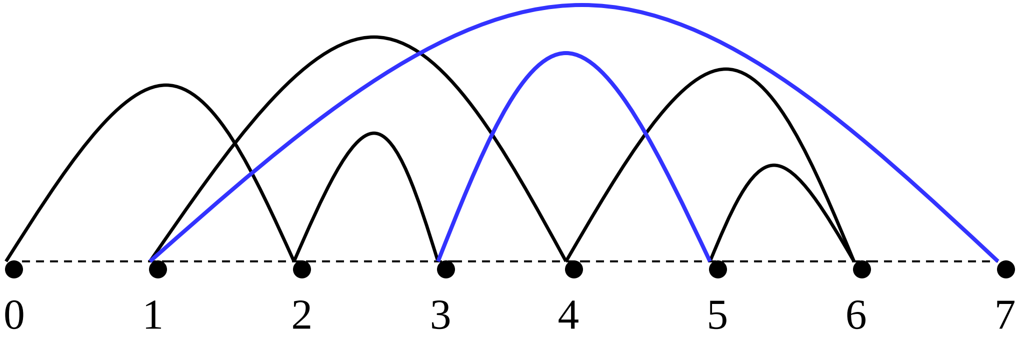
How does a reversal change the breakpoint graph?

- The gray paths stayed the same for both graphs
- There is a change in the graph at this point
- There is another change at this point
- The black edges are unaffected by the reversal so they remain the same for both graphs

Before: 0 2 3 1 4 6 5 7

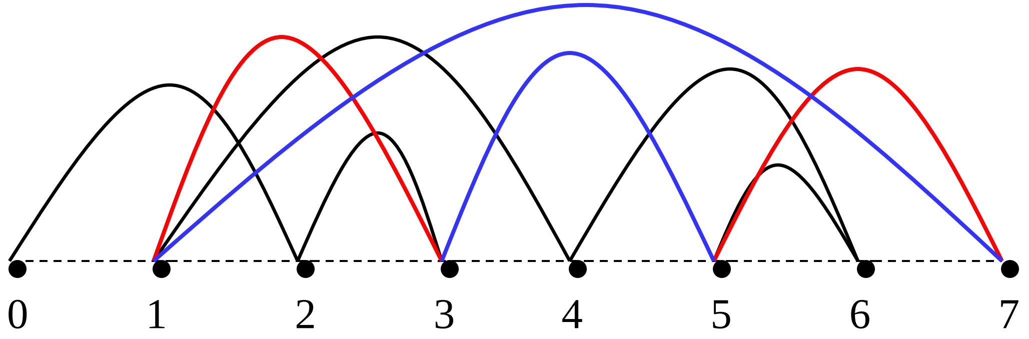


After: 0 2 3 5 6 4 1 7



A reversal affects 4 edges in the breakpoint graph

- A reversal removes 2 edges (red) and replaces them with 2 new edges (blue)



Effects of Reversals

Case 1:

Both edges belong to the same cycle

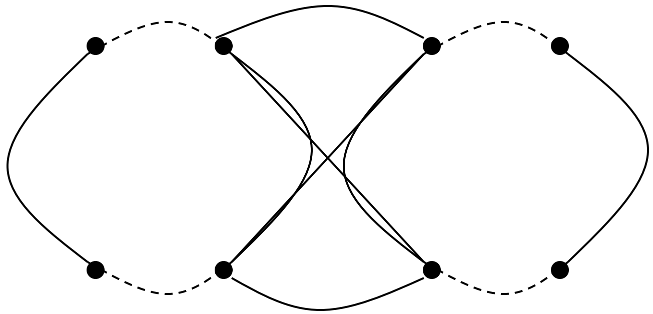
- Remove the center black edges and replace them with new black edges (there are two ways to replace them)
- (a) After this replacement, there now exists 2 cycles instead of 1 cycle
- (b) Or after this replacement, there still exists 1 cycle

Therefore, after the reversal

$c(\pi\rho) - c(\pi) = 0$ or 1

$$c(\pi\rho) - c(\pi) = 1$$
$$c(\pi\rho) - c(\pi) = 0$$

This is called a proper reversal since there's a cycle increase after the reversal.



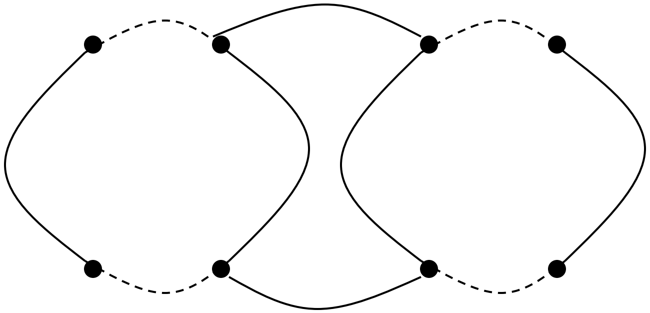
Effects of Reversals (Continued)

Case 2:

Both edges belong to different cycles

- Remove the center black edges and replace them with new black edges
- After the replacement, there now exists 1 cycle instead of 2 cycles

➡ $c(\pi\rho) - c(\pi) = -1$



Therefore, for every permutation π and reversal ρ, $c(\pi\rho) - c(\pi) \leq 1$

Reversal Distance and Maximum Cycle Decomposition

- Since the identity permutation of size n contains the maximum cycle decomposition of $n+1$, $c(identity) = n+1$

- $c(identity) – c(\pi)$ equals the number of cycles that need to be “added” to $c(\pi)$ while transforming π into the identity

- Based on the previous theorem, at best after each reversal, the cycle decomposition could increased by one, then:

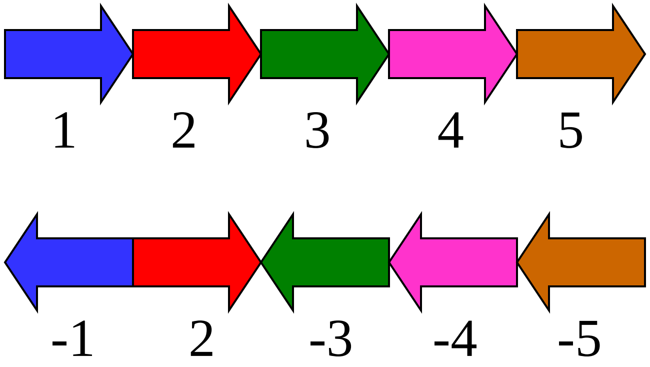
$$d(\pi) = c(identity) – c(\pi) = n+1 – c(\pi)$$

- Yet, not every reversal can increase the cycle decomposition

➡ *Therefore,* $d(\pi) \geq n+1 – c(\pi)$

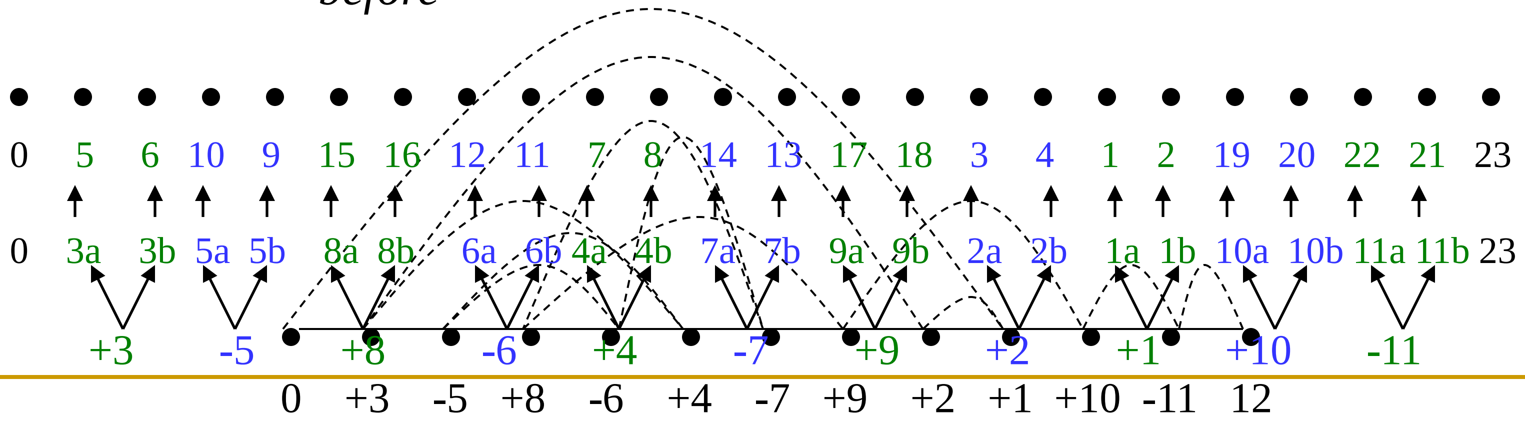
Signed Permutation

- Genes are *directed* fragments of DNA and we represent a genome by a signed permutation
- If genes are in the same position but there orientations are different, they do not have the equivalent gene order
- For example, these two permutations have the same order, but each gene's orientation is the reverse; therefore, they are not equivalent gene sequences



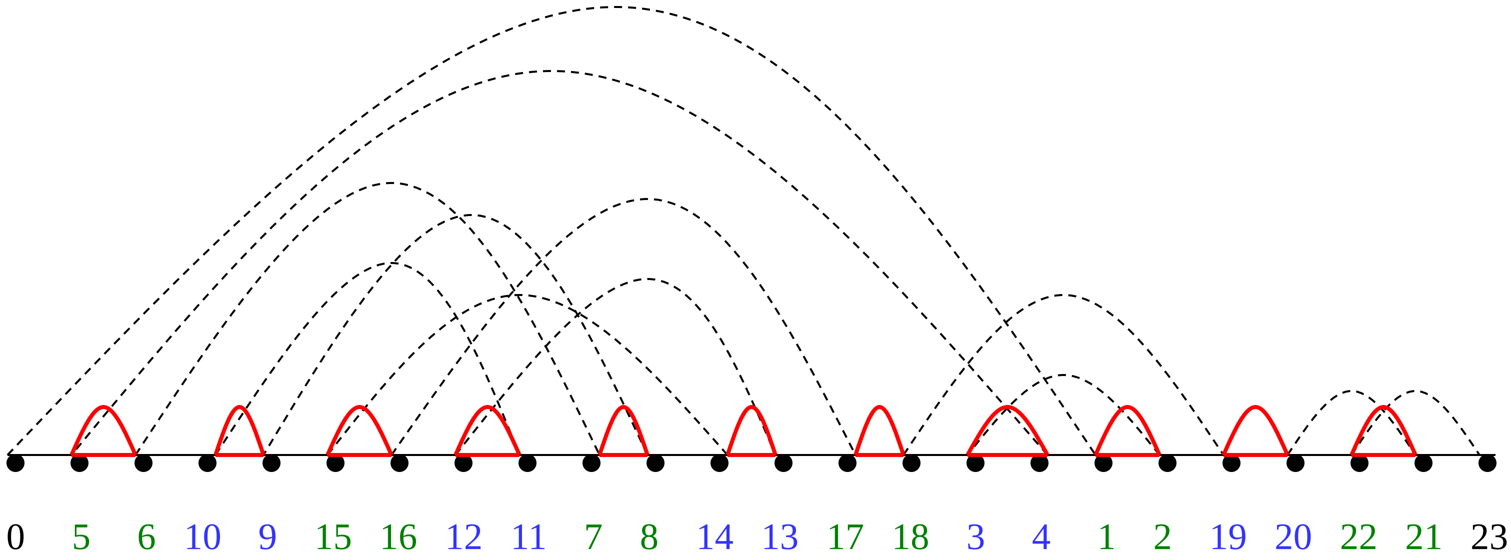
From Signed to Unsigned Permutation

- Begin by constructing a normal signed breakpoint graph
- Redefine each vertex x with the following rules:
  - *If vertex x is positive, replace vertex x with vertex 2x-1 and vertex 2x in that order*
  - *If vertex x is negative, replace vertex x with vertex 2x and vertex 2x-1 in that order*
  - *The extension vertices x = 0 and x = n+1 are kept as it was before*



From Signed to Unsigned Permutation (Continued)

- Construct the breakpoint graph as usual
- Notice the alternating cycles in the graph between every other vertex pair
- Since these cycles came from the same signed vertex, we will not be performing any reversal on both pairs at the same time; therefore, these cycles can be removed from the graph

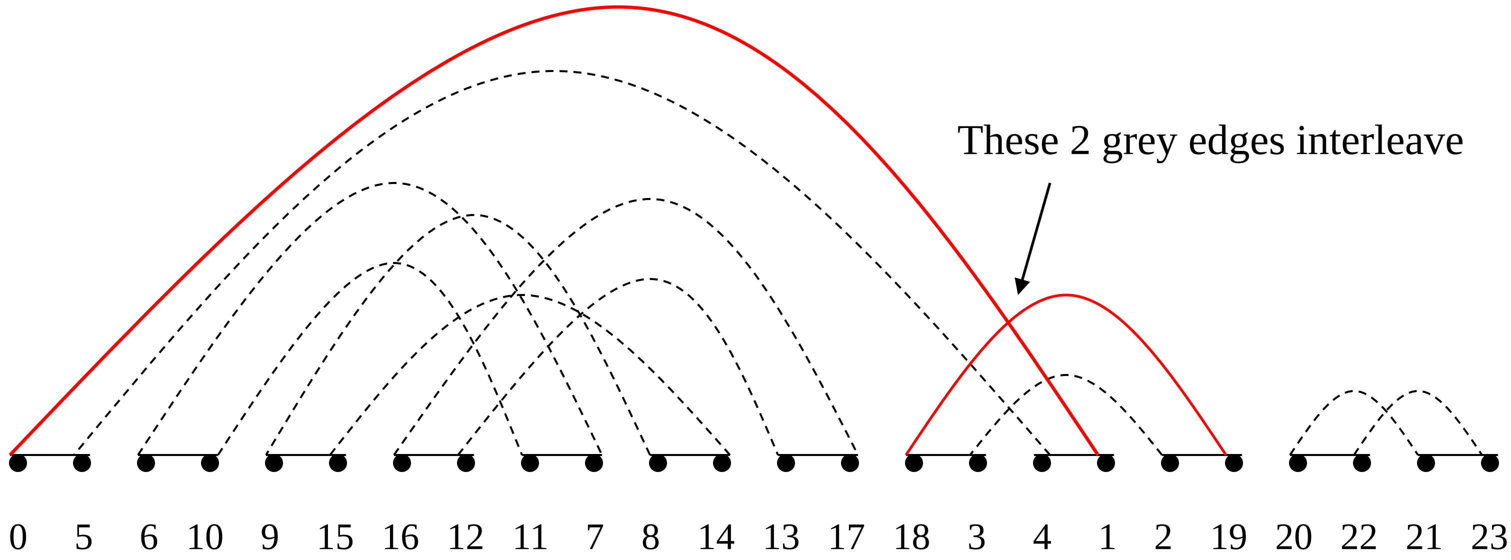


Interleaving Edges

- Interleaving edges are grey edges that cross each other

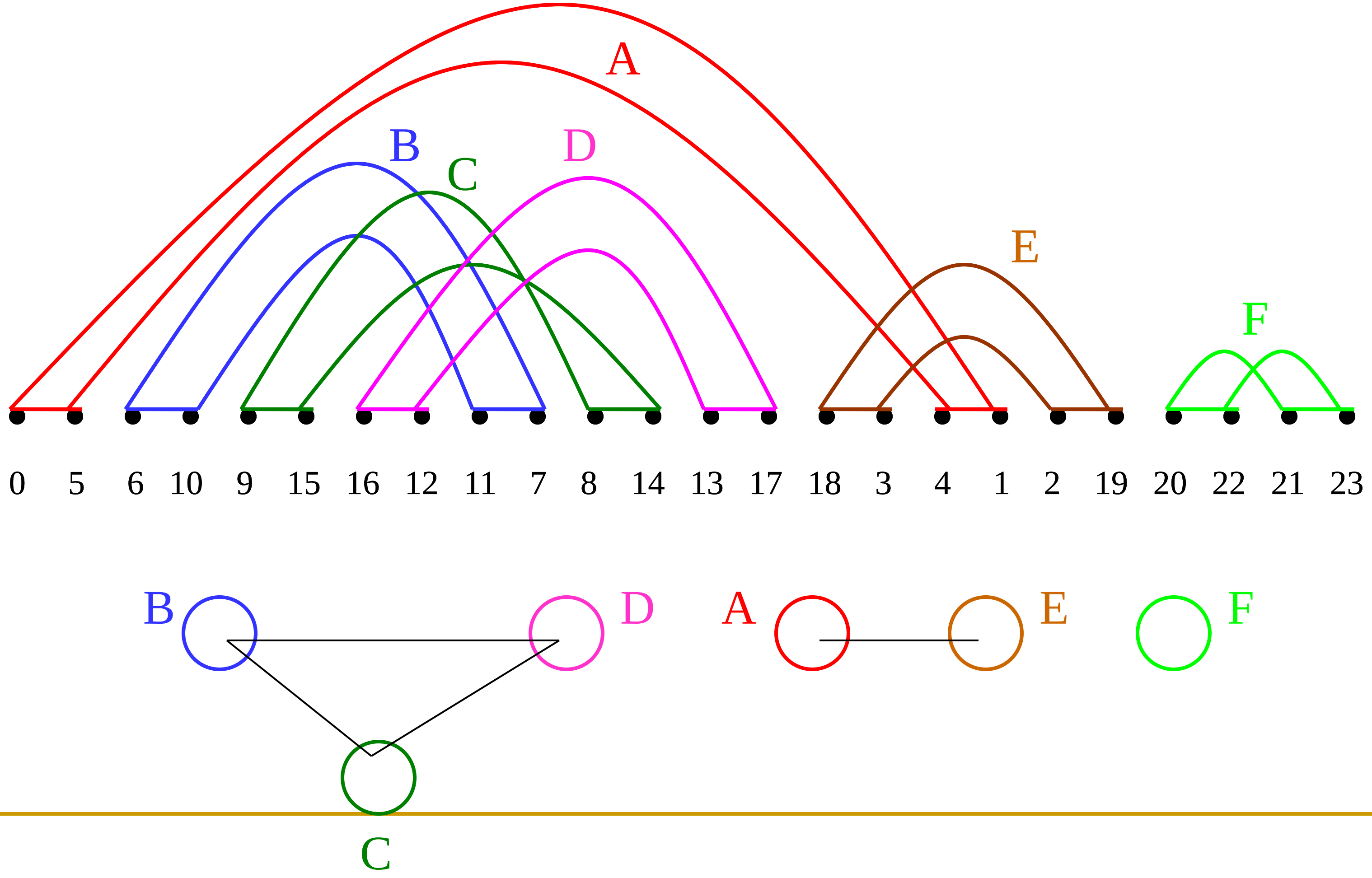
  Example: Edges (0,1) and (18, 19) are interleaving

- Cycles are interleaving if they have an interleaving edge



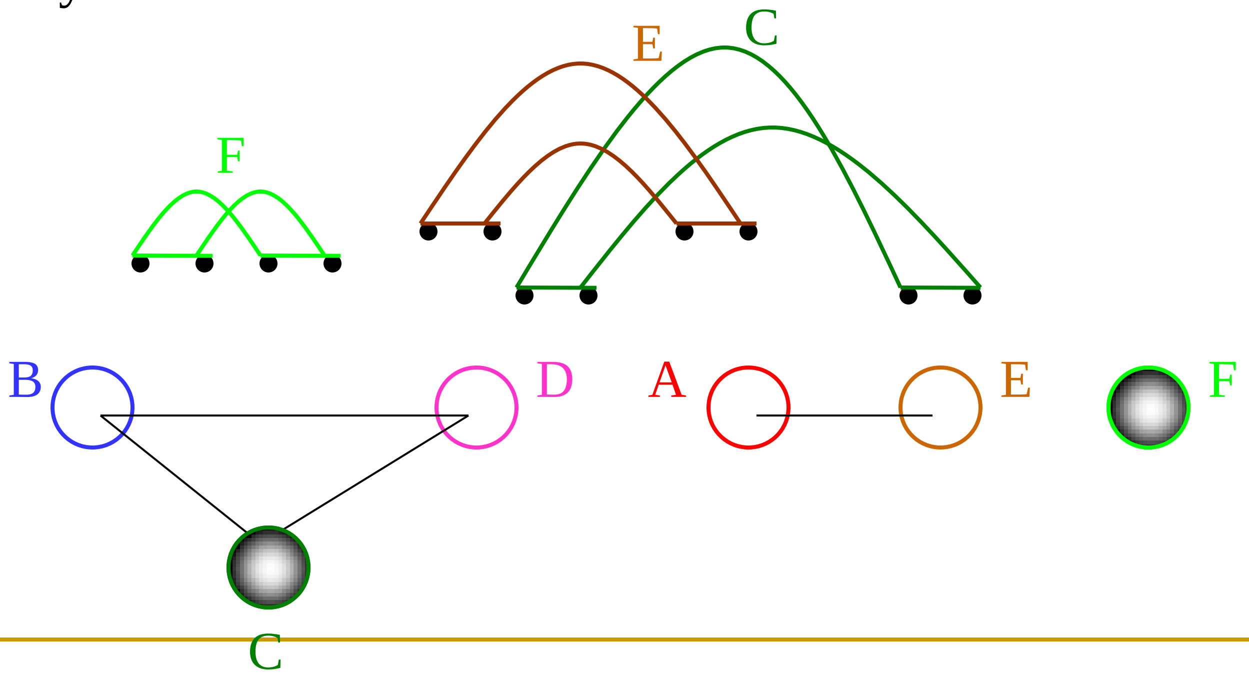
Interleaving Graphs

- An Interleaving Graph is defined on the set of cycles in the Breakpoint graph and are connected by edges where cycles are interleaved



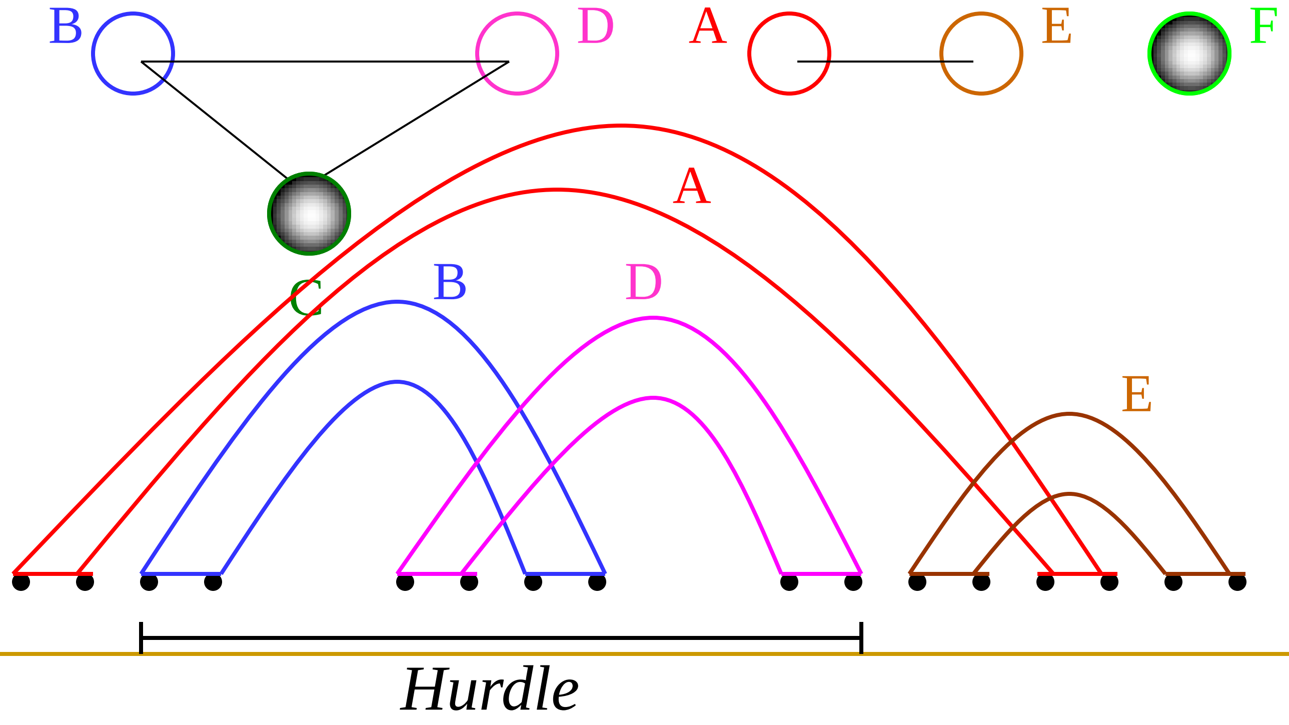
Interleaving Graphs (Continued)

- Oriented cycles are cycles that have the following form
- Mark them on the interleave graph
- Unoriented cycles are cycles that have the following form
- In our example, A, B, D, E are unoriented cycles while C, F are oriented cycles



Hurdles

- Remove the oriented components from the interleaving graph
- The following is the breakpoint graph with these oriented components removed
- Hurdles are connected components that do not contain any other connected components within it



Reversal Distance with Hurdles

- Hurdles are obstacles in the genome rearrangement problem
- They cause a higher number of required reversals for a permutation to transform into the identity permutation
- Let h(π) be the number of hurdles in permutation π
- Taking into account of hurdles, the following formula gives a tighter bound on reversal distance:

$$\Rightarrow \quad d(\pi) \geq n+1 - c(\pi) + h(\pi)$$