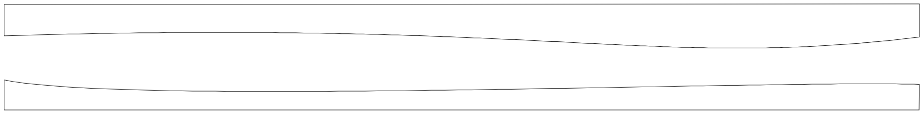
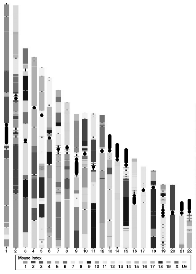
A4G

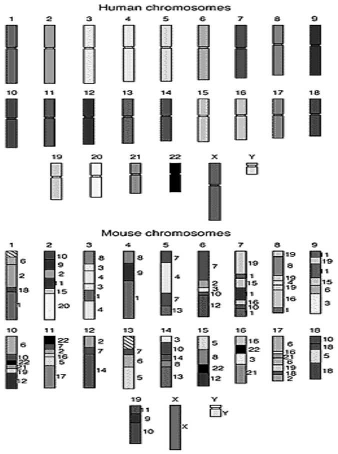


Genome rearrangements

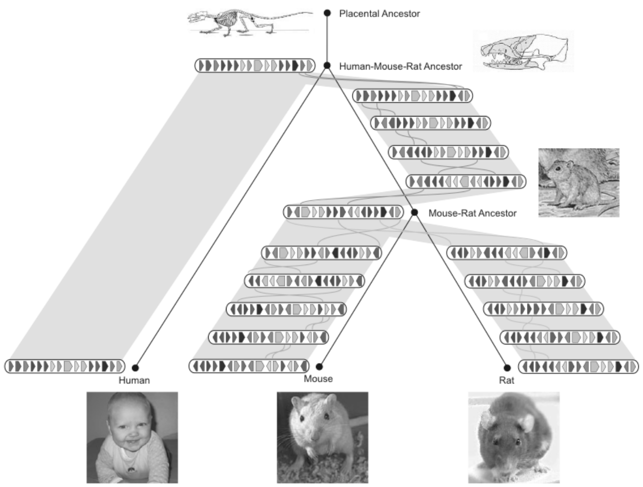


Comparative Genomic Architectures: Mouse vs Human Genome

- Humans and mice have similar genomes, but their genes are ordered differently
- ~300 rearrangements
  - inversions
  - Fusions
  - Fissions
  - Translocation



History of Chromosome X



Rat Consortium, *Nature*, 2004

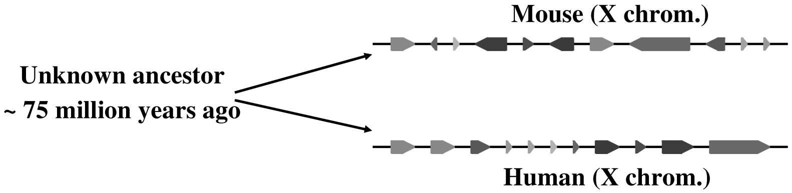
Rearrangements

- Genome-level mutations
- Duplication+divergence: dominating evolutionary force
- Both short- and long- evolutionary distances
- Within-population differences
  - 3.5% of human genome (Patchwork people, E. Check,Nature, Vol 437, 20 Oct 2005)

Conserved synteny block

- A conserved synteny block
  - *Greek*: “same thread”
  - locally conserved gene order
  - long for closely related organism
  - disturbed by rearrangements

Genome rearrangements



- Architecture of the ancestral genome
- Evolutionary scenario for transforming one genome into the other

---

Types of Rearrangements

Inversion

1 2 3 4 5 6 ⇒ 1 2 -5 -4 -3 6

Translocation

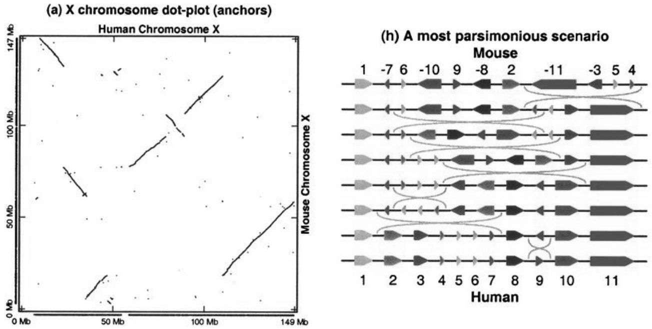
1 2 3
4 5 6 ⇒ 1 2 6
4 5 3

Fusion

1 2 3 4
5 6 ⇒ 1 2 3 4 5 6 ⇐

Fission

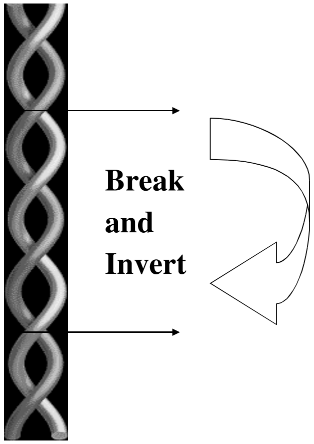
Transforming one genome into the other



Most parsimonious scenarios give a lower bound on actual # of inversions

Pavel Pevzner and Glenn Tesler, *Genome Rearrangements in Mammalian Evolution: Lessons From Human and Mouse Genomes*, Genome Res., Jan 2003; 13: 37 - 45 ; doi:10.1101/gr.757503

Inversions: Example



Break and Invert

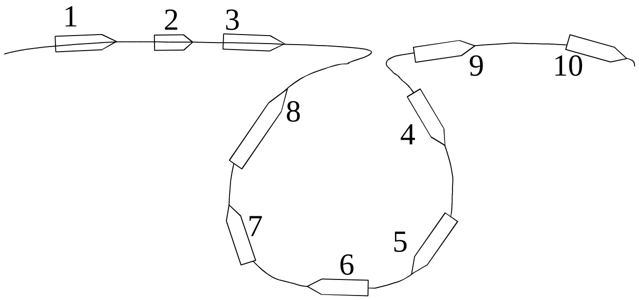
5’ ATGCCTGTACTA 3’

3’ TACGGACATGAT 5’

5’ ATGTACAGGCTA 3’

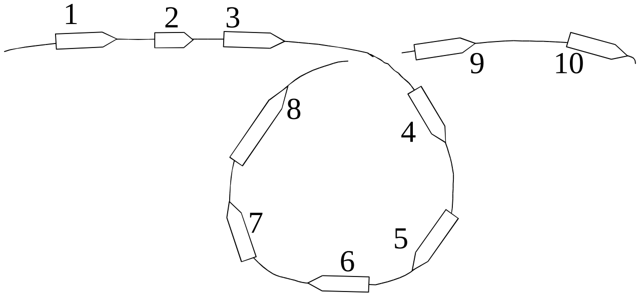
3’ TACATGTCCGAT 5’

Inversion



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

Inversion



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

Inversions and Gene Orders

- Gene order is represented by a permutation π:

$$\pi = \pi_1 \text{------} \pi_{i-1} \underline{\pi_i \pi_{i+1} \text{------} \pi_{j-1} \pi_j} \pi_{j+1} \text{-----} \pi_n$$

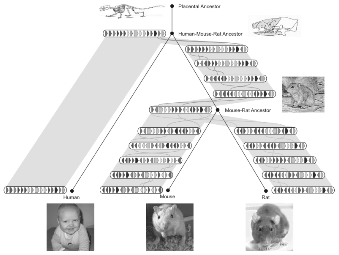
$$\rho(i, j) \Big\downarrow$$

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

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

Inversion distance

- π_1, π_2– permutations, no sign
- min number of inversions transforming permutations
  - less than evolutionary events



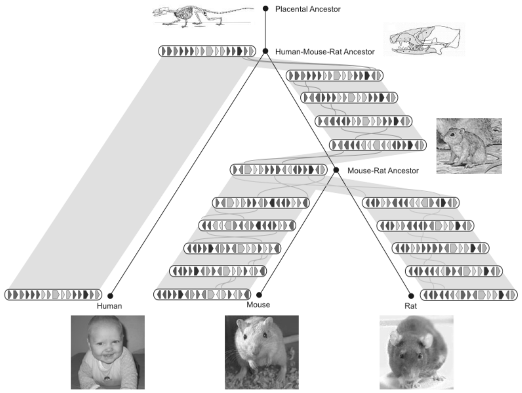
Sorting By Inversions Problem

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

---

Inversion distance

- Even simpler: π_2 = identity
- from π to id in d(π)
- NP-hard :-(



Sorting By inversions: Example

- *t* =*d*(π) - inversion 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*(π) = 3

Sorting By inversions: An Inefficient Algorithm

- The length of the already sorted prefix of π is denoted *prefix*(π)
  - *prefix*(1 2 3 6 4 5) = 3
- This results in an idea for a greedy algorithm: increase *prefix*(π) at every step

---

Approximation Ratio

- Approximation ratio of algorithm A: max approximation ratio of all inputs of size *n*
- A(π) -solution produced by algorithm A
- OPT(π) - optimal solution of the problem
- For algorithm A that minimizes objective function (minimization algorithm):

$$\max_{|\pi| = n} A(\pi) / OPT(\pi)$$

An Inefficient Algorithm, example

- Doing so, π can be sorted

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

$$\downarrow$$

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

$$\downarrow$$

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

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

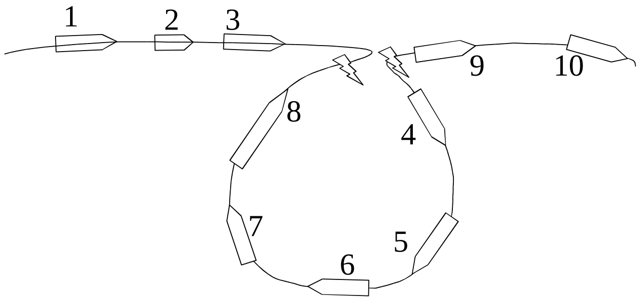
Breakpoints: An Example

- $\pi = \pi_1\pi_2\pi_3\ldots\pi_{n-1}\pi_n$
- There is a breakpoint between neighbors in permutation if are non-consecutive:

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

- *b*(π) - # breakpoints in permutation π

Inversions and Breakpoints



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

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

Inversion Distance and Breakpoints

- Each inversion eliminates at most 2 breakpoints.
- $d(\pi) \geq b(\pi)/2$

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

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

Extending Permutations

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

$$0\underset{\uparrow}{\ }5\ \ 6\underset{\uparrow}{\ }2\ \ 1\underset{\uparrow}{\ }3\ \ 4\underset{\uparrow}{\ }7$$

breakpoints

Strips

- Strip: an interval between two consecutive breakpoints in a permutation
  - Decreasing strip: *strip* of elements in decreasing order
  - Increasing strip: *strip* of elements in increasing order

$$\underset{\longrightarrow}{0\ \ 1}\ \ \underset{\longleftarrow}{9}\ \ \underset{\longleftarrow}{4\ \ 3}\ \ \underset{\longrightarrow}{7\ \ 8}\ \ \underset{\longleftarrow}{2}\ \ \underset{\longrightarrow}{5\ \ 6}\ \ \underset{\longrightarrow}{10}$$

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

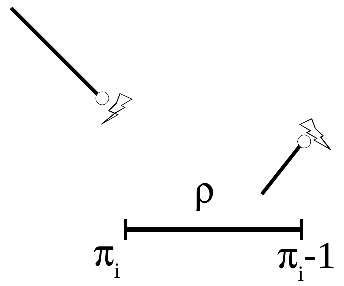
Reducing the Number of Breakpoints

Theorem 1:

If π contains at least one decreasing strip, then there exists an inversion ρ for which $b(\pi \bullet \rho) < b(\pi)$

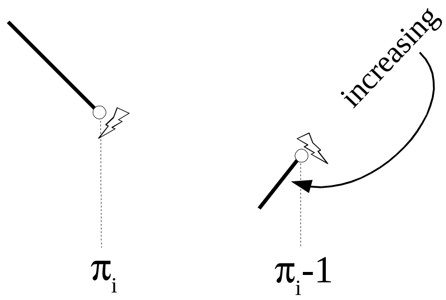
Choosing an inversion

- Choose the inversion ρ



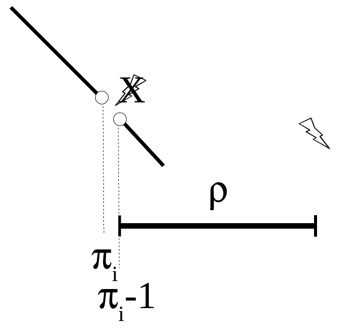
Choosing a ↓ strip

- Choose a ↓ strip with the smallest element π_i in π



Choosing an inversion

- $\pi \bullet \rho$: at least one breakpoint less



ImprovedBreakpointReversalSort

BreakpointInversionSort

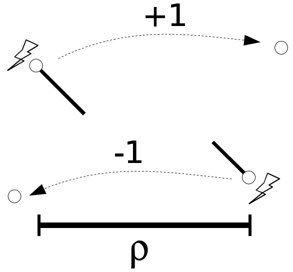
```
BreakpointInversionSort($\pi$)
while $b(\pi) > 0$
    if $\pi$ has a decreasing strip
        choose an inversion $\rho$ that removes a breakpoint
    else
        choose an inversion $\rho$ that flips an increasing strip in $\pi$
    $\pi = \pi \bullet \rho$
    output $\pi$
return
```

Improving the Approximation Ratio

Theorem 2:

✿ {Suppose there is no inversion which removes a ⚡ and leaves a ↓ strip.

Then there exists an inversion ρ which removes 2⚡ .



ImprovedBreakpointReversalSort: Performance Guarantee

BreakpointinversionSort: Approximation Ratio

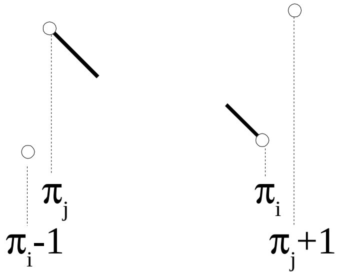
- at most $2b(\pi)$ steps
  - one for breakpoint,
  - one to get ↓ strip
- we know that $d(\pi) \geq b(\pi) / 2$
- therefore the approximation ratio

$$\max_{|\pi| = n} A(\pi) / OPT(\pi) = 4$$

Improving the Approximation Ratio

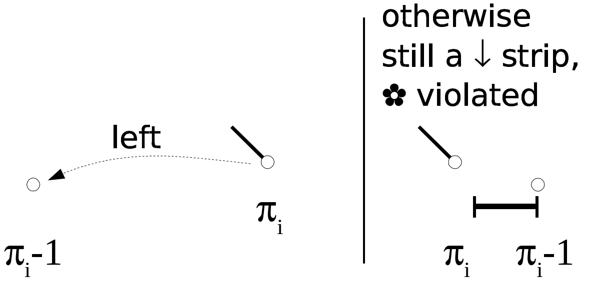
π_i: the smallest number in any ↓ strip

π_j: the largest number in any ↓ strip



Proof of Theorem 2

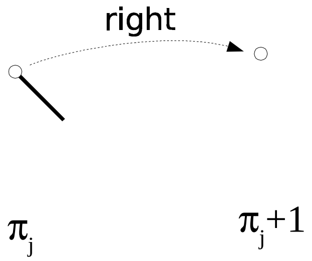
π_i-1 is to the left of π_i



otherwise
still a ↓ strip,
✿ violated

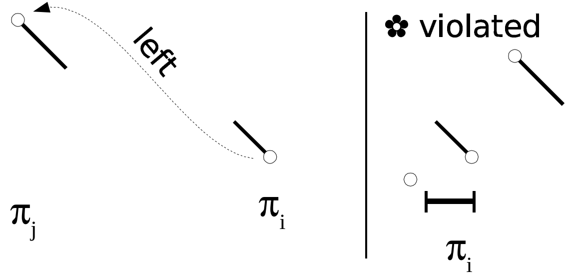
Proof of Theorem 2

π_j+1 is to the right of π_j



Proof of Theorem 2

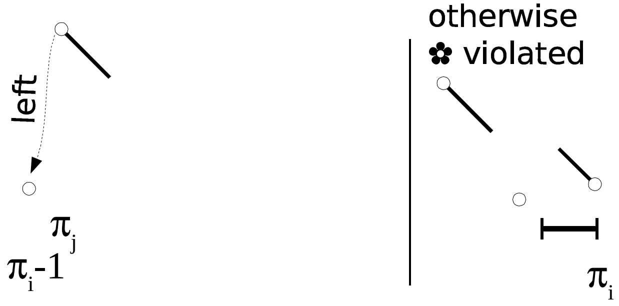
π_j is to the left of π_i



otherwise
✿ violated

Proof of Theorem 2

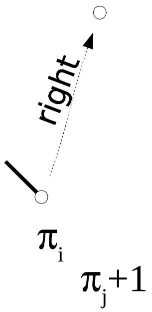
π_{i-1} is to the left of π_j



otherwise
✿ violated

Proof of Theorem 2

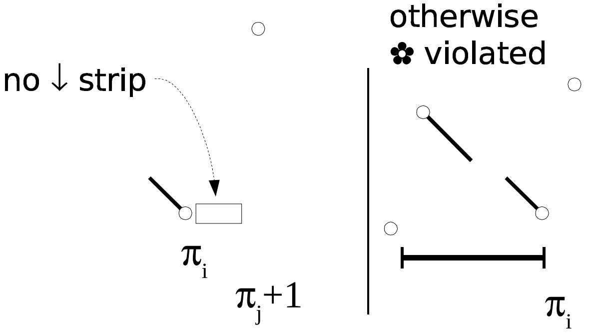
π_j+1 is to the right of π_i



Proof of Theorem 2

π_i is next to π_j+1

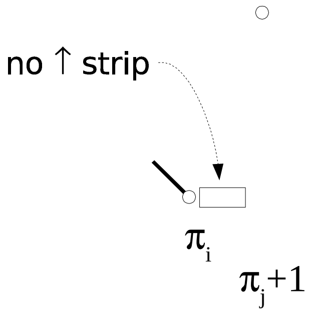
case a)



Proof of Theorem 2

π_i is next to π_j+1

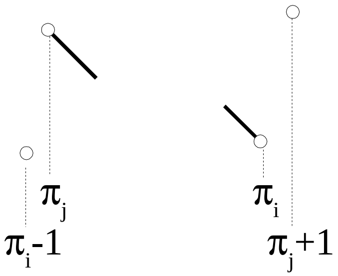
case b)



Improving the Approximation Ratio

$\pi_{i+1} = \pi_j+1$, similarly,

$\pi_{j-1} = \pi_i-1$

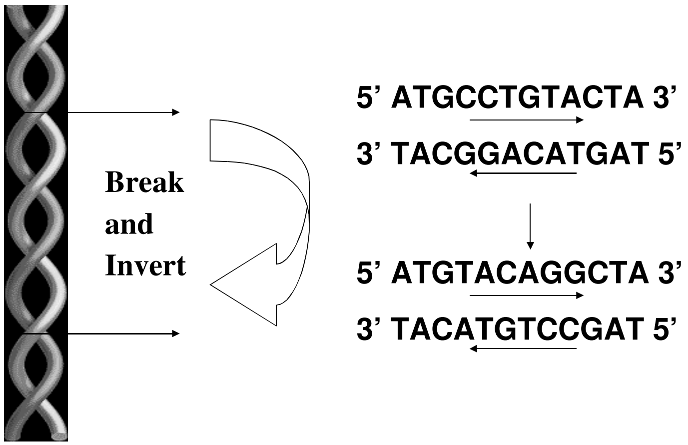


Much Better Algorithm

```
MuchBetterBreakpointInversionSort(π)
while b(π) > 0
    if there's an inversion ρ that removes a breakpoint and
            leaves a ↓ strip
            π = π • ρ; output π
    else
            choose an inversion ρ that removes 2 breakpoints π
            π = π • ρ; output π
            if b(π) > 0
                    choose ρ that flips an increasing strip in π
                    π = π • ρ; output π
return
```

- at most $b(\pi)$ steps, approx ratio = 2
- approx 1.5 or 1.375 when looking two steps ahead

Inversion and Signed Permutations



5’ ATGCCTGTACTA 3’

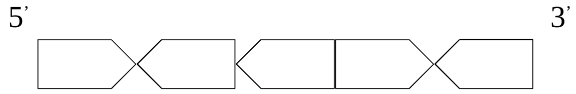
3’ TACGGACATGAT 5’

5’ ATGTACAGGCTA 3’

3’ TACATGTCCGAT 5’

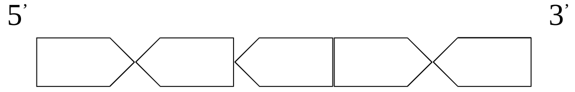
Signed Permutations

- 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

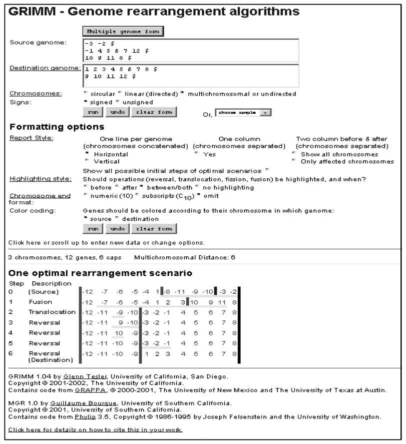
Signed Permutations



π = 1 -2 -3 4 -5

- Easy problem!
- We can find $d(\pi)$ in $O(n^2)$

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): ● Horizontal ○ Vertical

One column (chromosomes separated): ○ Yes

Two column before & after (chromosomes separated): ○ 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 (Destination)	-12	-11	-10	-9	1	2	3	4	5	6	7	8

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.

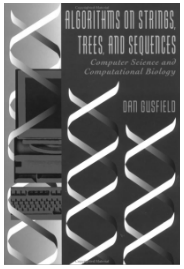
MGR 1.0 by Guillaume Bourque, University of Southern California.
Copyright © 2001, University of Southern California.
Contains code from Phylip 3.5, Copyright © 1986-1995 by Joseph Felsenstein and the University of Washington.

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

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

Where to find the proof to Theorem 2

- print.google.com
- search for the book by Dan Gusfield
- search for "improving the guarantee" (p. 496)



References

- www.bioalgorithms.info
- Book “Algorithms on strings…” D. Gusfield