Greedy Algorithms

Study Chapters 5.1-5.2

Greedy Algorithms

- An iterative algorithm where at each step
  - Take what seems to be the best option
- Cons:
  - It may return incorrect results
  - It may require more steps than necessary
- Pros:
  - it often takes very little time to make a greedy choice
  - we consider choices independently
- Did we see any greedy algorithm in previous lectures?

Coin change problem

Pancake Flipping Problem

The chef at "Breadman's" is sloppy.
He makes pancakes of nonuniform sizes.,
and throws them on the plate.

Before the waitress delivers them to your table, she rearranges them so that the smaller pancakes are stacked on larger ones.

Since she has only one hand to perform this culinary rearrangement, she does it with spatula with which she flips the pancakes.
I was wondering, how many such flips are needed for this rearrangement?

Pancake Flipping Problem: Formulation

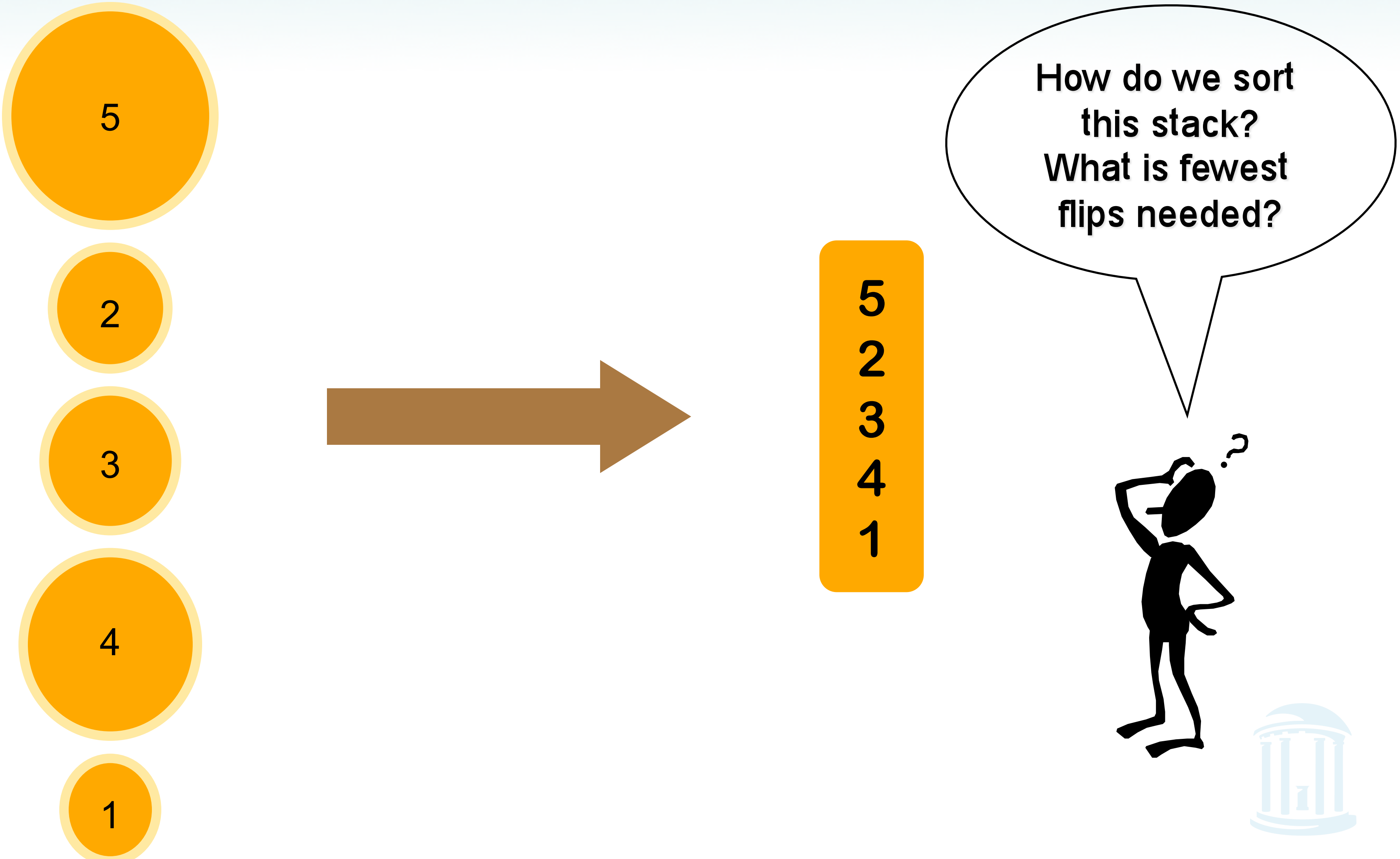
- Goal: Given a stack of n pancakes, what is the minimum number of flips to rearrange them into a perfect (small-to-large ordered) stack?
- Input: Permutation π
- Output: A series of prefix reversals $\rho_1, \ldots \rho_t$ transforming π into the identity permutation such that t is minimum

$$\pi = \underrightarrow{\pi_1 \ldots \pi_{i-1}\, \pi_i}\, \pi_{i+1} \ldots \pi_n$$

$$\big\downarrow \rho$$

$$\pi = \underleftarrow{\pi_i\, \pi_{i-1} \ldots \pi_1}\, \pi_{i+1} \ldots \pi_n$$

Turning Pancakes into Numbers



"Bring to Top" Method

Flip the biggest to top.

Flip the whole stack (*n*), to place it on bottom.

Flip the next largest to top.

Flip the *n-1* pancakes, thus placing the second largest second from bottom.

And so on…

Bring-to-Top Method for n Pancakes

- If ($n = 1$), the smallest is on top - we are done.
- otherwise: flip pancake n to top and then
  flip it to position n.

- Now use:

Bring-to-Top Method
for n-1 Pancakes

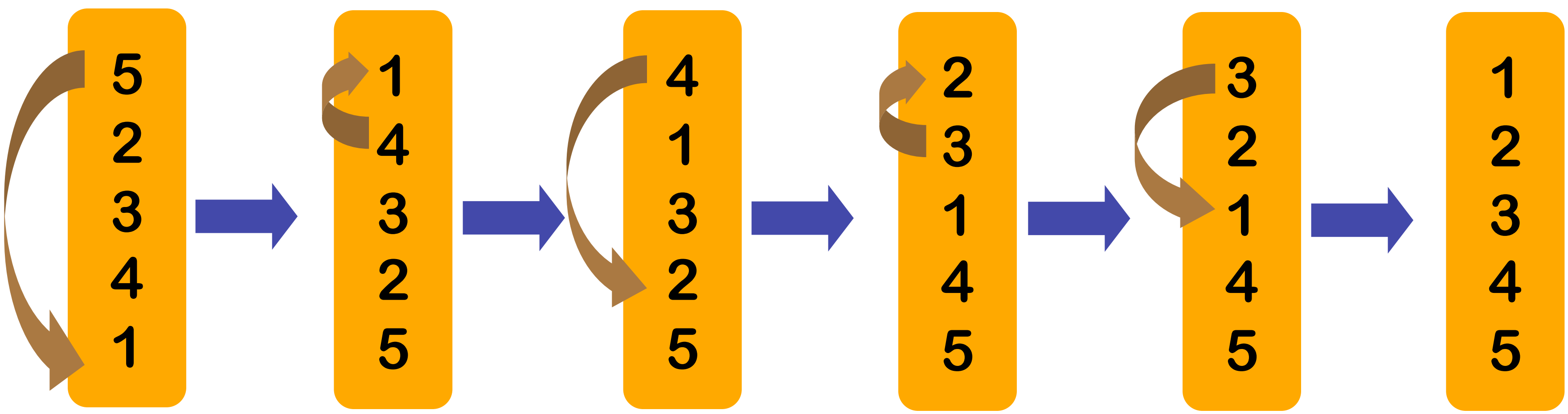
Greedy algorithm: 2 flips to put a pancake in its right position.

Total Cost: at most $2(n-1) = 2n - 2$ flips.

Good Enough?

- Our algorithm is correct, but is it the best we could do?
- Consider the following:

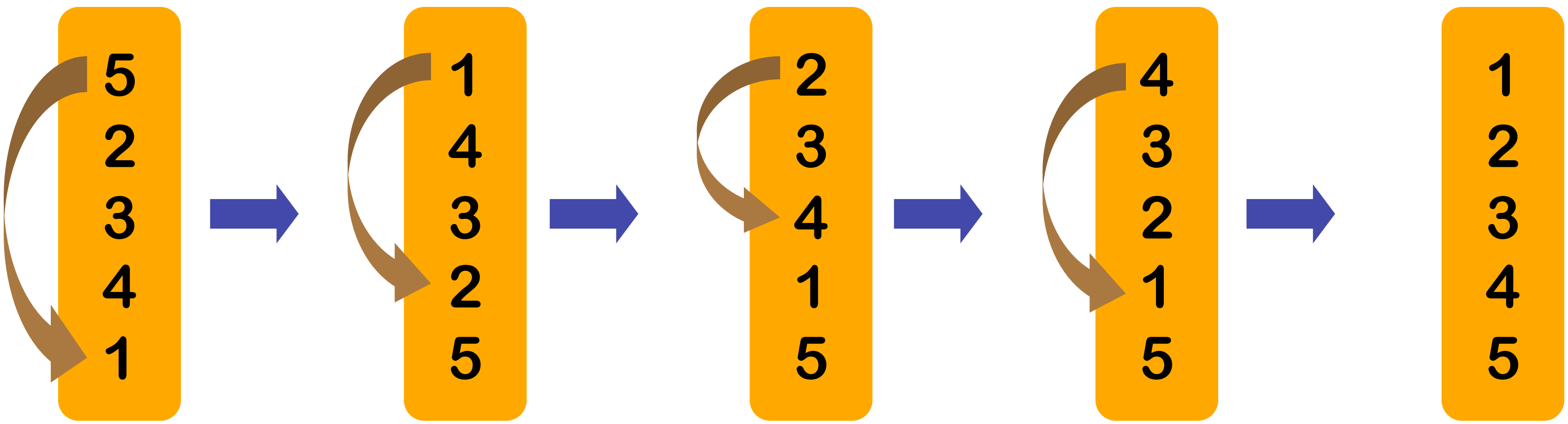
  Our algorithm predicts 2(5-1) = 8 flips, but…



The “Biggest-to-top” algorithm did it in 5 flips! The predicted “8” flips is an upper-bound for ***any*** input.

Does there exist another algorithm do in fewer flips?

4 Flips Are Sufficient

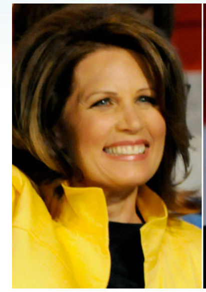


William Gates (yeah, that Microsoft guy) and Christos Papadimitriou showed in the mid-1970s that this problem can be solved by at least 17/16 n and at most 5/3 $(n + 1)$ *prefix reversals* (*flips*) for n pancakes.

A Serious Scientific Problem …

Differences between species?

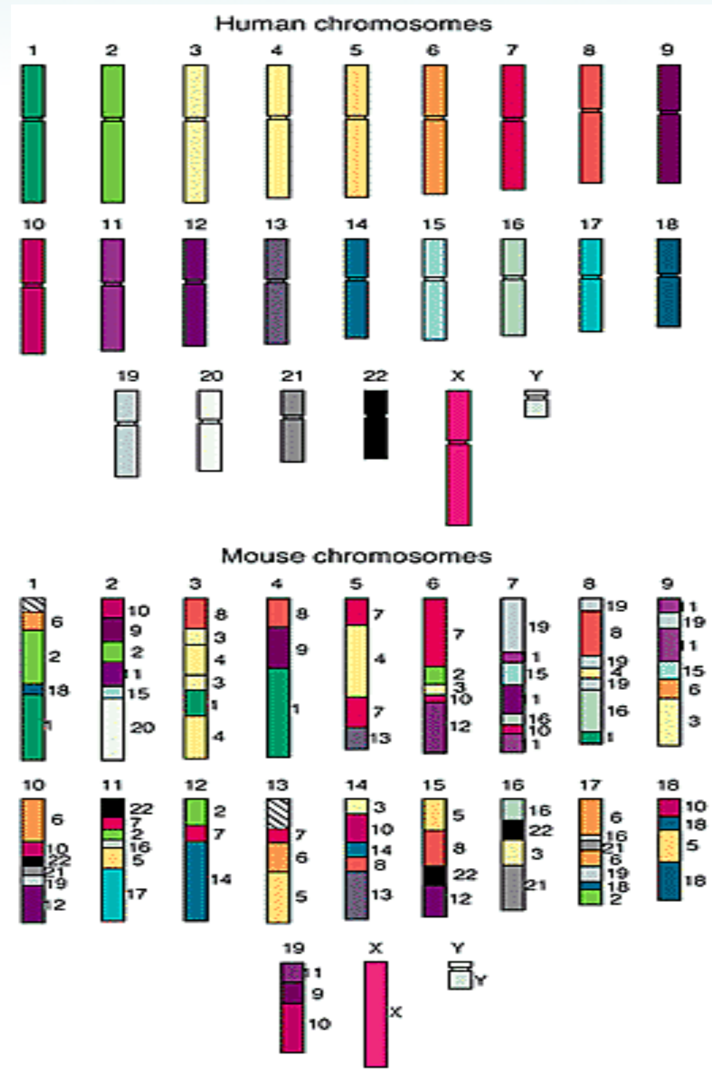
- Some are obviously similar…
- Some are obviously different…
- Some are close calls…
- The differences that matter are in the genes!
- And the gene order is important!

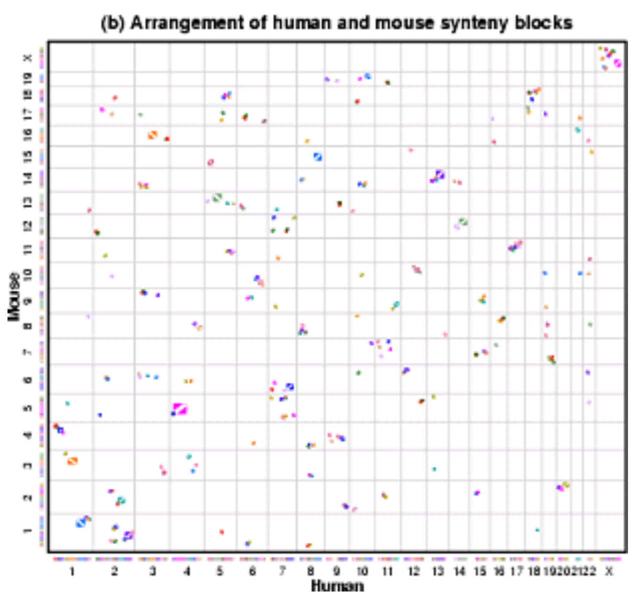




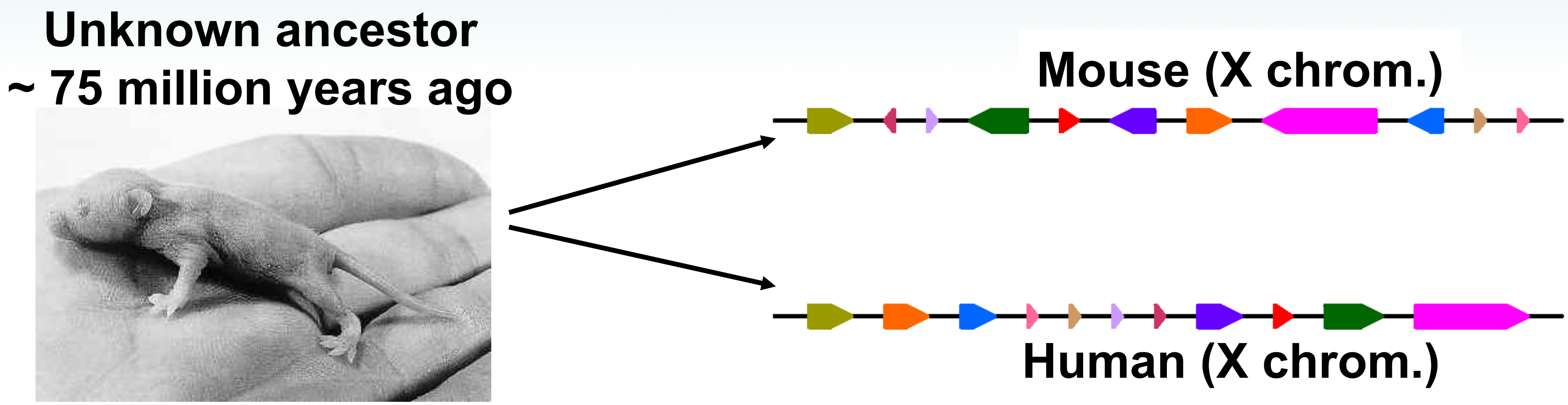
Genome Rearrangements

- Humans and mice have similar genomes, but their genes are ordered differently
- ~245 rearrangements
- ~ 300 large *synteny blocks*





Genome Rearrangements



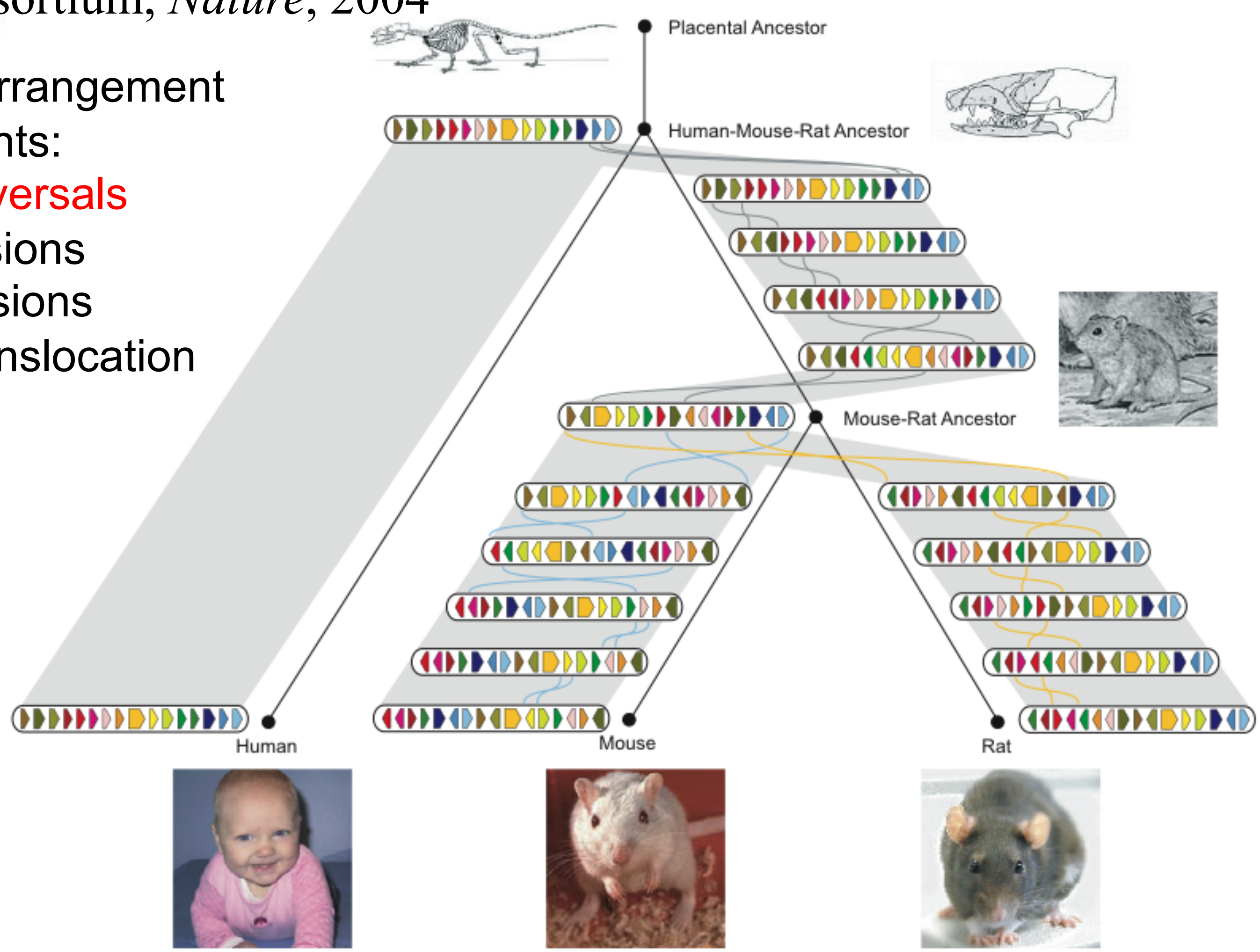
- What are the similarity blocks and how to find them?
- What is the architecture of the ancestral genome?
- What is the evolutionary scenario for transforming one genome into the other?

History of Chromosome X

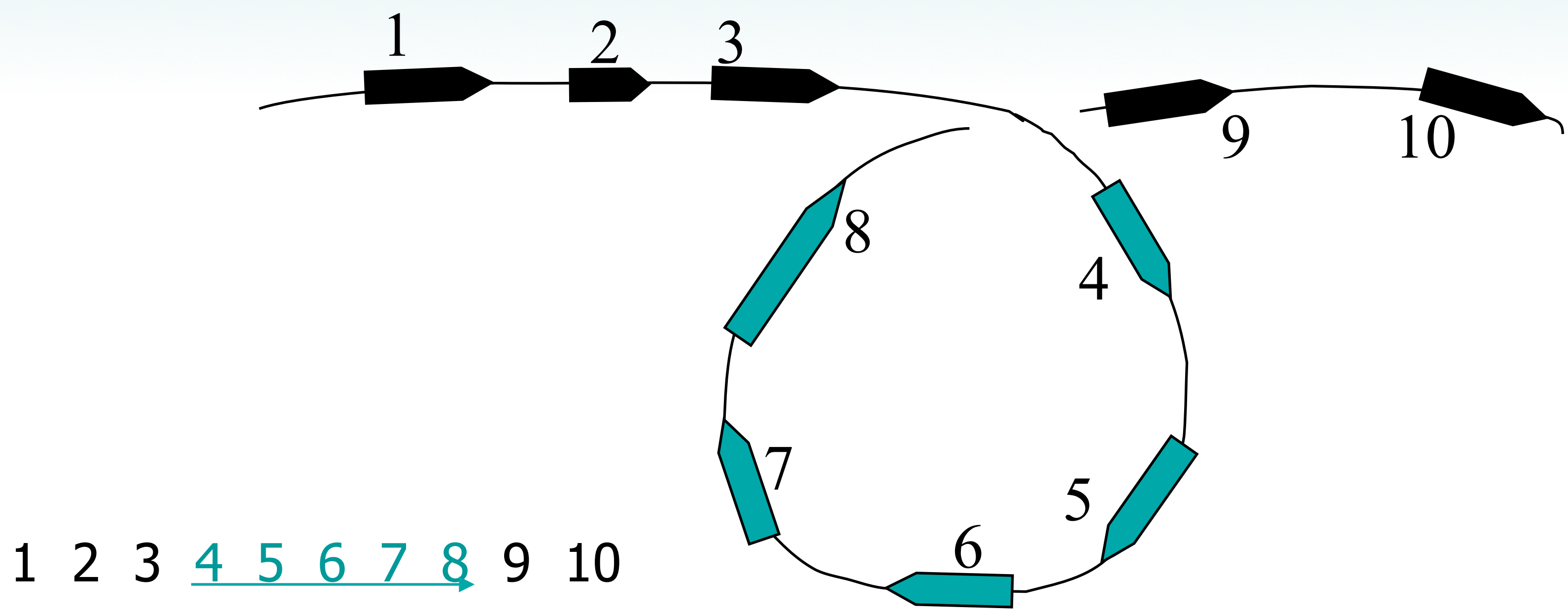
Rat Consortium, *Nature*, 2004

Rearrangement Events:

- Reversals
- Fusions
- Fissions
- Translocation

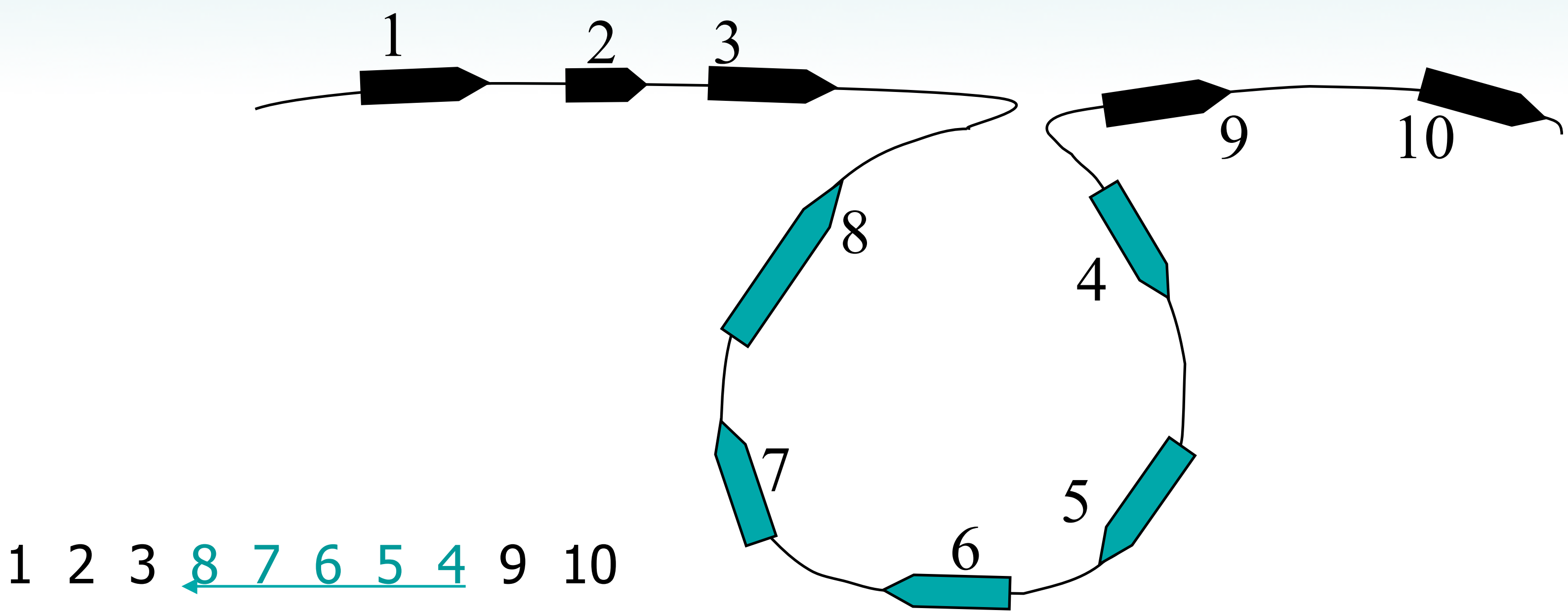


Reversals



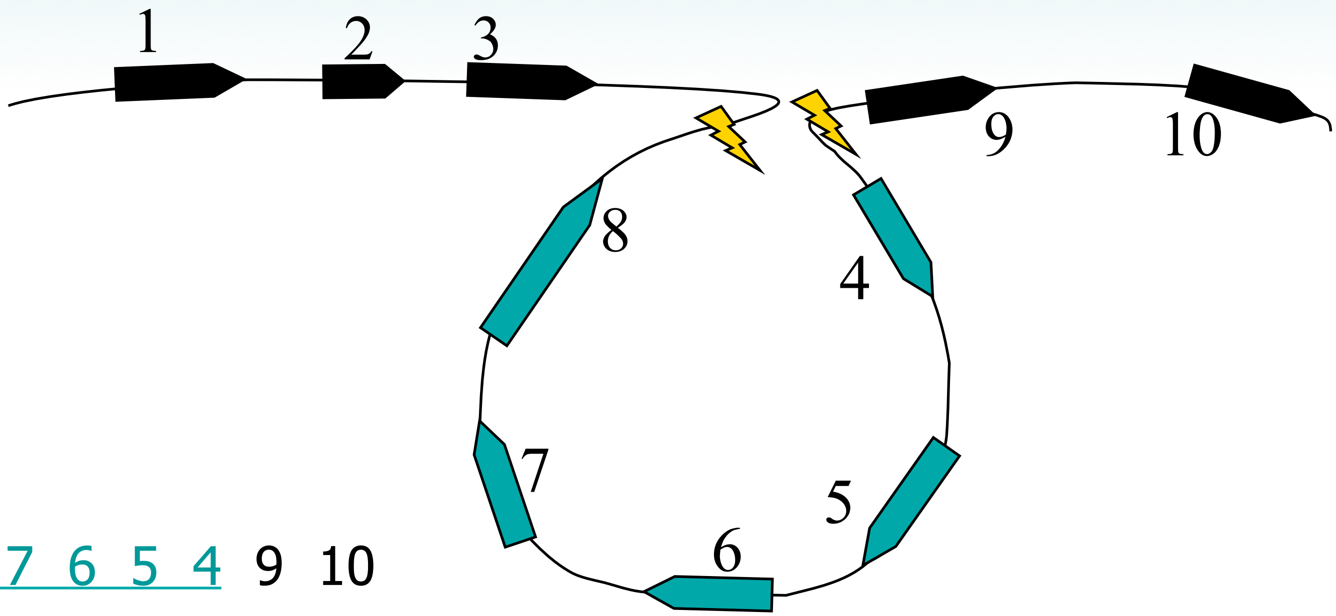
- Blocks represent conserved genes.
- Reversals, or *inversions*, are particularly relevant to speciation. Recombinations cannot occur between reversed and normally ordered segments.

Reversals



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

Reversals and Breakpoints



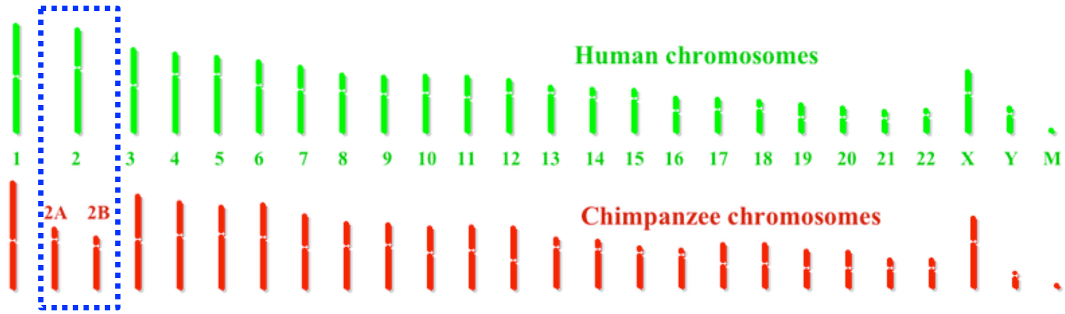
1 2 3 8 7 6 5 4 9 10

The inversion introduced two *breakpoints* (disruptions in order).

Other Types of Rearrangements

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

Reversals and Gene Orders

- Gene order can be represented by a permutation ***π*:**

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

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

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

- Reversal $\rho(i, j)$ reverses (flips) the elements from i to j in π

Reversals: Example

π = 1 2 $\underline{3\ 4\ 5}$ 6 7 8

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

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

$\rho(5,6)$ $\downarrow$

1 2 5 4 6 3 7 8

"Reversal Distance" Problem

- Goal: Given two permutations over n elements, 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

- $\boldsymbol{t}$ - reversal distance between π and σ
- $\boldsymbol{d}(\pi, \sigma)$ - smallest possible value of t, given π and σ

"Sorting By Reversals" Problem

A simplified restatement of the same problem….

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

Sorting By Reversals: Example

π = 3 4 2 1 5 6 7 10 9 8

4 3 2 1 5 6 7 10 9 8

4 3 2 1 5 6 7 8 9 10

1 2 3 4 5 6 7 8 9 10

$d(\boldsymbol{\pi}) = 3$

Sorting by Reversals: 4 flips

Step 0: π 2 4 3 5 8 7 6 1

Step 1: 2 3 4 5 8 7 6 1

Step 2: 2 3 4 5 6 7 8 1

Step 3: 8 7 6 5 4 3 2 1

Step 4: 1 2 3 4 5 6 7 8

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

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 apart.
- 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

Sort by Reversals: An Example

- Doing so, π can be sorted

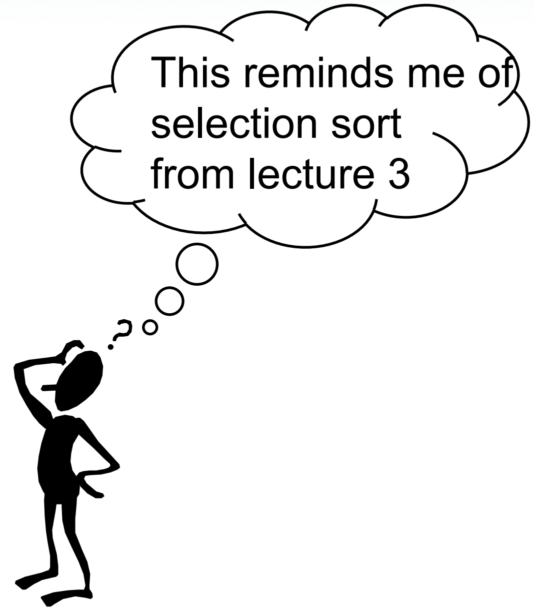
$$1\ 2\ 3\ \underline{6\ 4}\ 5$$

$$\downarrow$$

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

$$\downarrow$$

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



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

Greedy Algorithm

SimpleReversalSort(π)

for $i \leftarrow 1$ to $n-1$

  $j \leftarrow$ position of element i in π (i.e., $\pi_j = i$)

  if $j \neq i$

    $\pi \leftarrow \pi \cdot \rho(i, j)$

    output π

  if π is the identity permutation

    return

In Python

```
def SimpleReversalSort(pi):
    for i in xrange(len(pi)):
        j = pi.index(min(pi[i:]))
        if (j != i):
            pi = pi[:i] + [v for v in reversed(pi[i:j+1])] + pi[j+1:]
            print i, j, pi
    return pi
```

Analyzing SimpleReversalSort

- SimpleReversalSort does not guarantee the smallest number of reversals and takes five steps on $\pi = \underline{6\ 1}\ 2\ 3\ 4\ 5$:

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

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

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

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

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

Analyzing SimpleReversalSort

- But it can be sorted in two flips:

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

$$\text{Flip 1:} \quad \underline{5\ 4\ 3\ 2\ 1}\ 6$$

$$\text{Flip 2:} \quad 1\ 2\ 3\ 4\ 5\ 6$$

- So, SimpleReversalSort(π) is not optimal

- Optimal algorithms are unknown for many problems; approximation algorithms are used

Next Time

- Approximation ratios
  - How close are non-optimal algorithms to optimal solutions?
- Genome rearrangement and breakpoints