# CS342: Bioinformatics
# Lecture 7

## Greedy Algorithms

- **Def:** Algorithms that make locally optimal choices using a metric with the hope of finding a globally optimal solution.

- **Example**: Making change with US coins.

- **Optimization Problem**: Given an input, compute a solution, subject to various constraints, that either minimizes cost or maximizes profit.

## Coin-Changing: Greedy Algorithm

Cashier's algorithm. At each iteration, add coin of the largest value that does not take us past the amount to be paid.

```
Sort coins denominations by value: $c_1 < c_2 < … < c_n$.

    coins selected
S ← $\phi$
while ($x \neq 0$) {
    let k be largest integer such that $c_k \leq x$
    if (k = 0)
        return "no solution found"
    $x \leftarrow x - c_k$
    S ← S ∪ {k}
}
return S
```

# Greedy Motif Search

```
GreedyMotifSearch(DNA, k, t)
        BestMotifs ← empty motif list
        BestScore   ← t * k
        for each k-mer Motif in the first string from DNA
            Motif1 ← Motif
            for i = 2 to t
                form Profile from motifs Motif1, …, Motifi – 1
                Motifi ← Profile-most probable k-mer in the i-th string in DNA
            Motifs ← (Motif1, …, Motift)
            if Score(Motifs) < BestScore:
                BestMotifs ← Motifs
                BestScore ← Score(Motifs)
        return BestMotifs
```

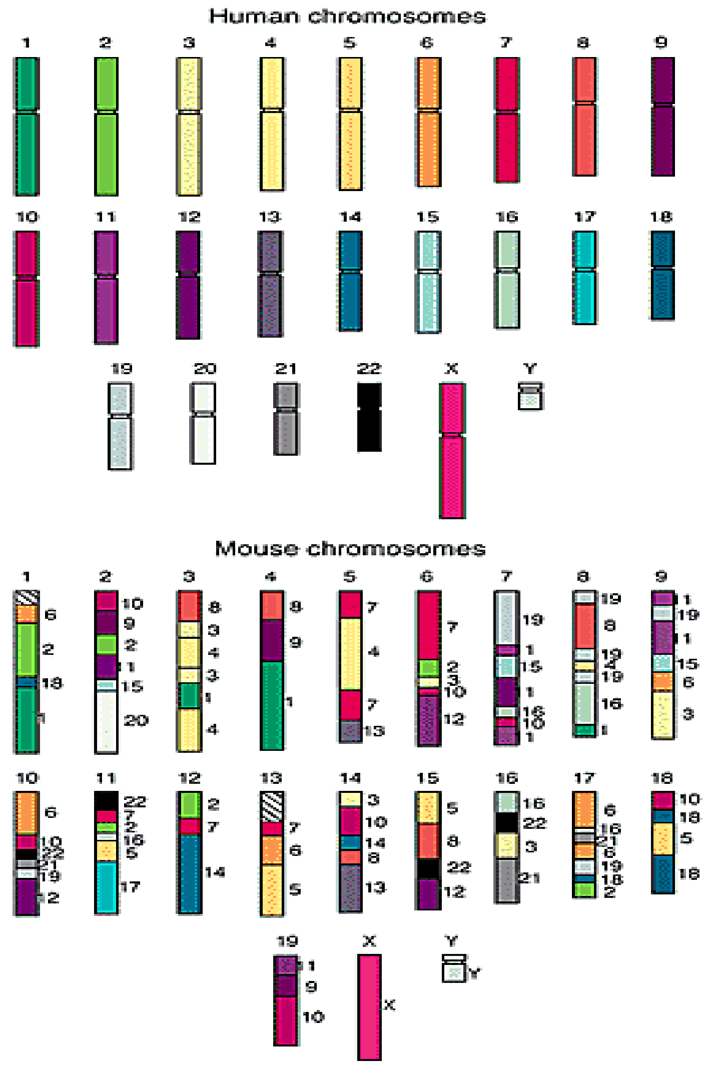
## A Serious Scientific Problem …

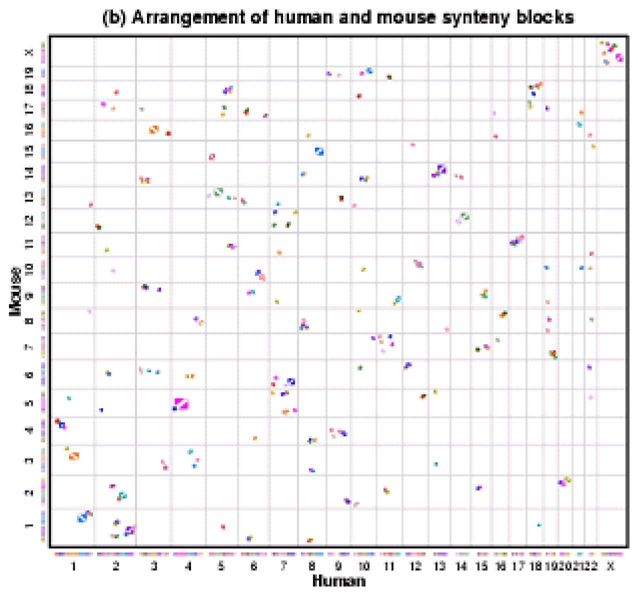
Differences between species?

- 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!

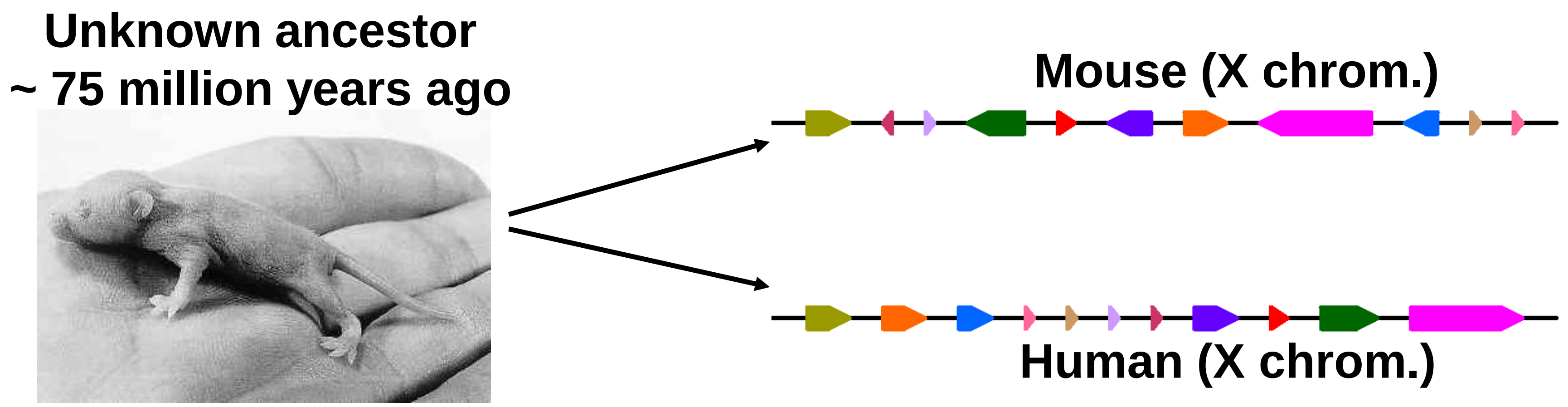
# Genome Rearrangements

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





# Genome Rearrangements



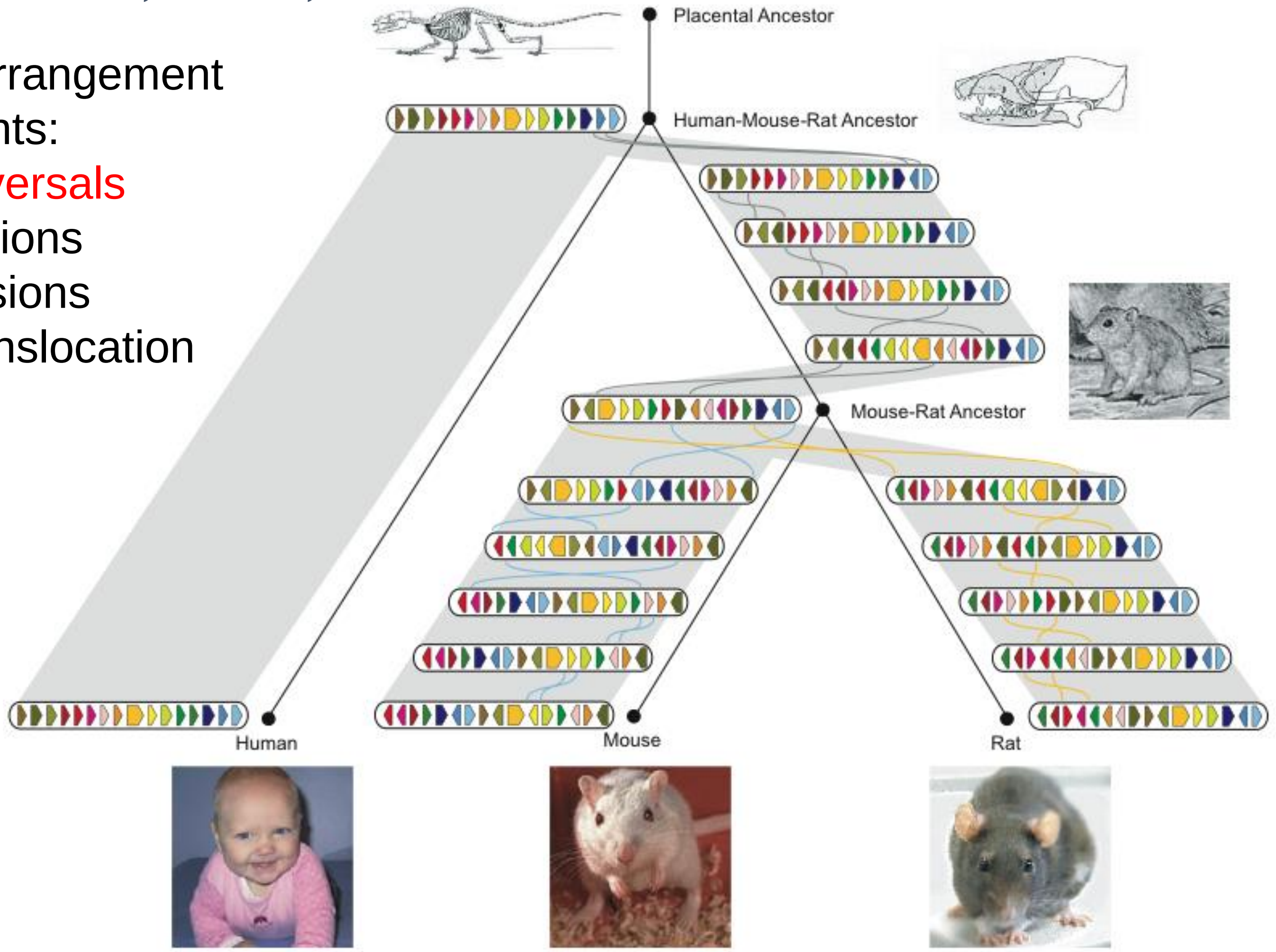
- What are the similarity blocks and how to find them?
- What is the architecture of the ancestral genome?
- What is the evolutionary scenario for transforming one genome into the other?

# History of Chromosome X

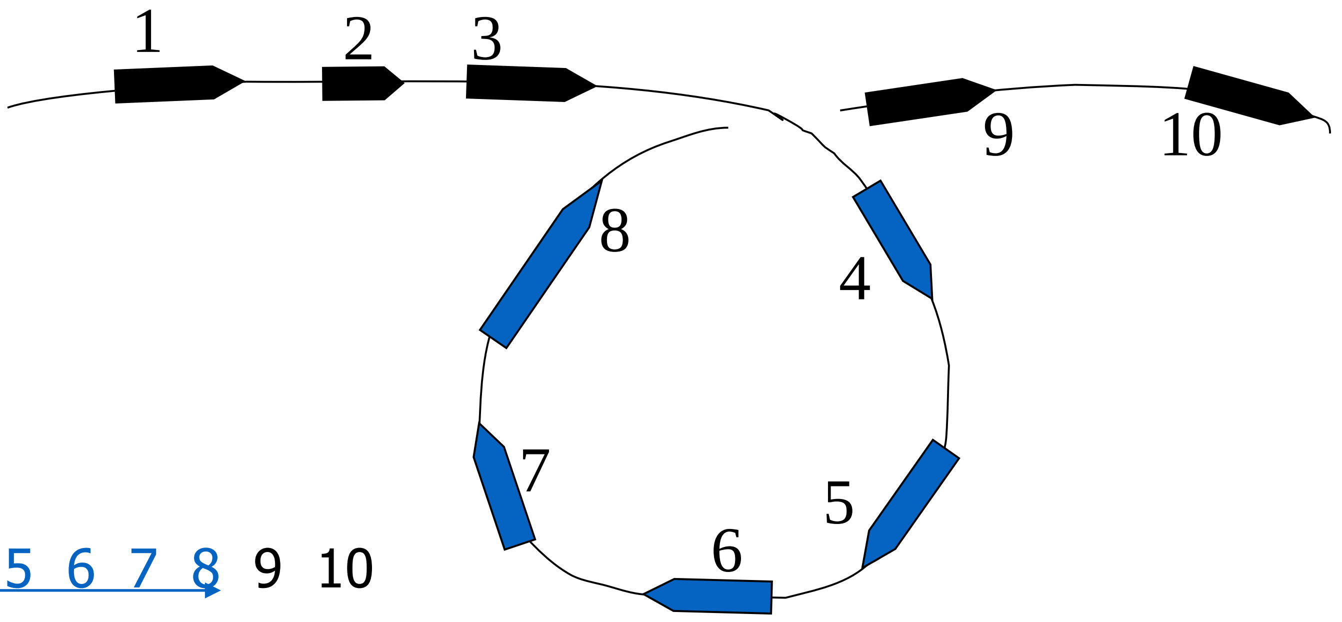
Rat Consortium, *Nature*, 2004

Rearrangement Events:

- Reversals
- Fusions
- Fissions
- Translocation



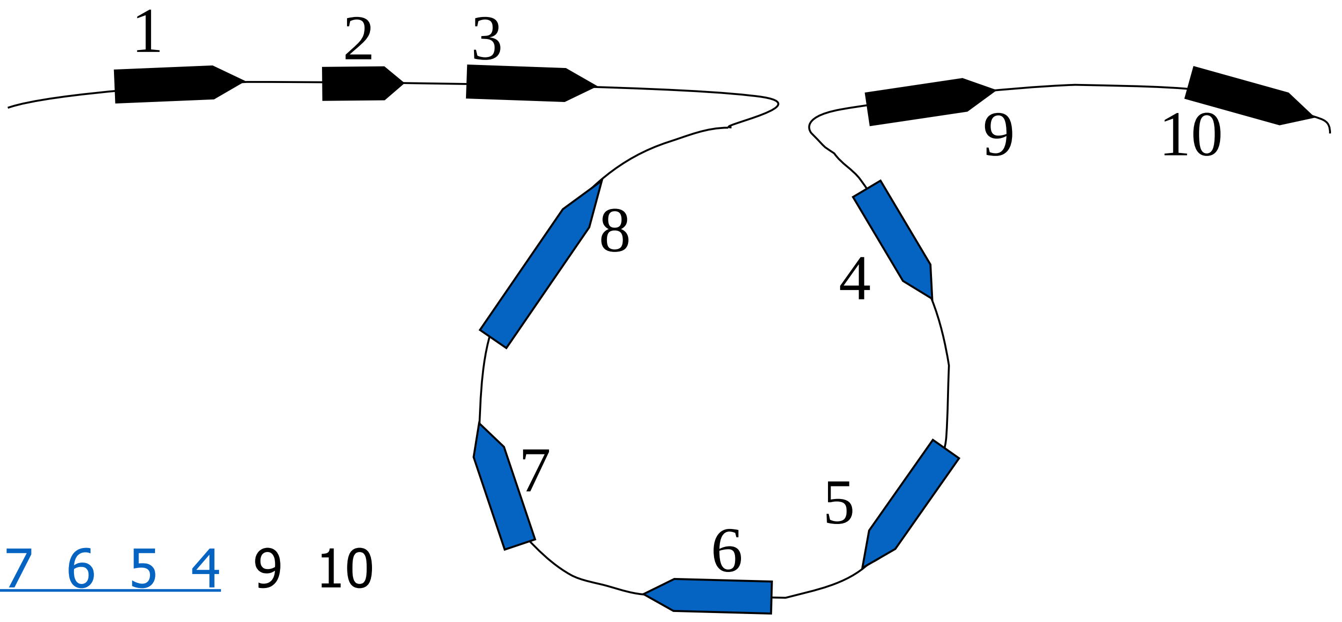
# Reversals



1 2 3 $\underrightarrow{4\ 5\ 6\ 7\ 8}$ 9 10

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

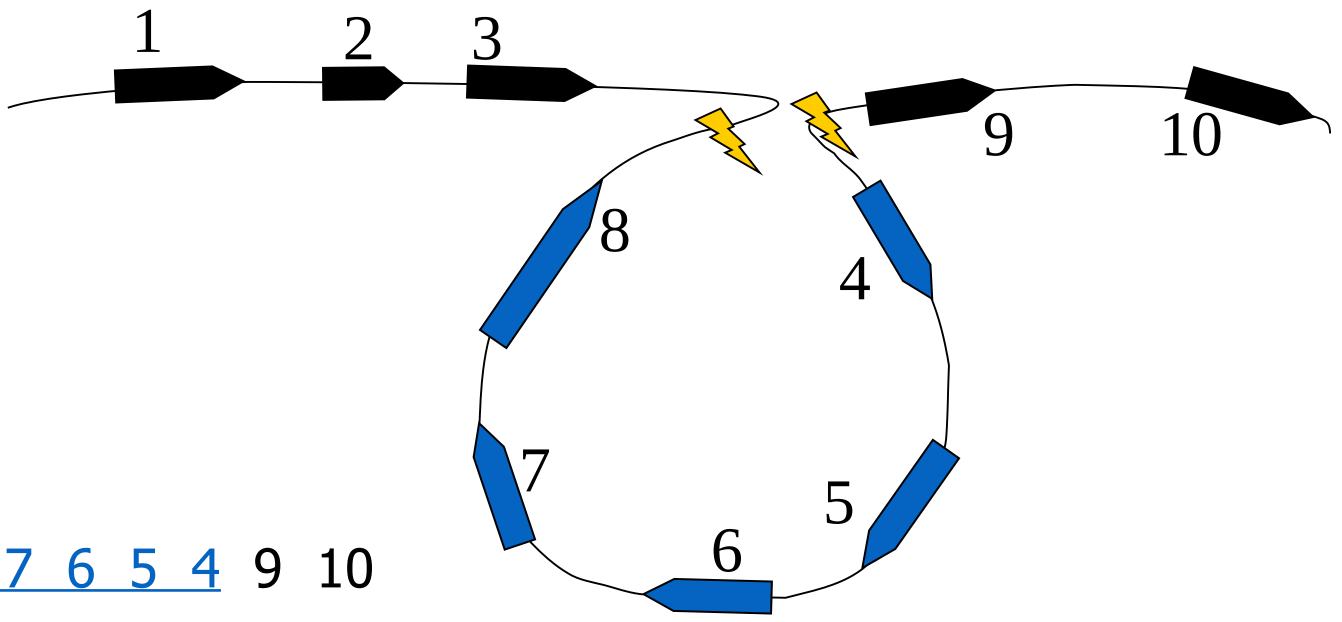
## Reversals



1 2 3 $\underleftarrow{8\ 7\ 6\ 5\ 4}$ 9 10

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

# Reversals and Breakpoints



1 2 3 $\underleftarrow{8\ \ 7\ \ 6\ \ 5\ \ 4}$ 9 10

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

# 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 $\pi$

# 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 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

- ***t*** - reversal distance between $\pi$ and $\sigma$ *(# of reversals)*
- ***d***$(\pi, \sigma)$ - smallest possible value of *t*, given $\pi$ and $\sigma$

## “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 $\rho_1, \ldots \rho_{\boldsymbol{t}}$ transforming $\boldsymbol{\pi}$ into the identity permutation such that *t* is minimum
- ***t* =*d*($\pi$ )** - reversal distance of $\pi$

# Sorting By Reversals: Example

$$\pi = \underline{3\ \ 4}\ \ 2\ \ 1\ \ 5\ \ 6\ \ 7\ \ 10\ \ 9\ \ 8$$

$$4\ \ 3\ \ 2\ \ 1\ \ 5\ \ 6\ \ 7\ \ \underline{10\ \ 9\ \ 8}$$

$$\underline{4\ \ 3\ \ 2\ \ 1}\ \ 5\ \ 6\ \ 7\ \ 8\ \ 9\ \ 10$$

$$1\ \ 2\ \ 3\ \ 4\ \ 5\ \ 6\ \ 7\ \ 8\ \ 9\ \ 10$$

$d(\pi) = 3$

## Sorting by Reversals: 4 flips

| | |
|---|---|
| Step 0: | $2\ \ \underline{4\ \ 3}\ \ 5\ \ 8\ \ 7\ \ 6\ \ 1$ |
| Step 1: | $2\ \ 3\ \ 4\ \ 5\ \ \underline{8\ \ 7\ \ 6}\ \ 1$ |
| Step 2: | $\underline{2\ \ 3\ \ 4\ \ 5\ \ 6\ \ 7\ \ 8}\ \ 1$ |
| Step 3: | $\underline{8\ \ 7\ \ 6\ \ 5\ \ 4\ \ 3\ \ 2\ \ 1}$ |
| 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?
How can we know?

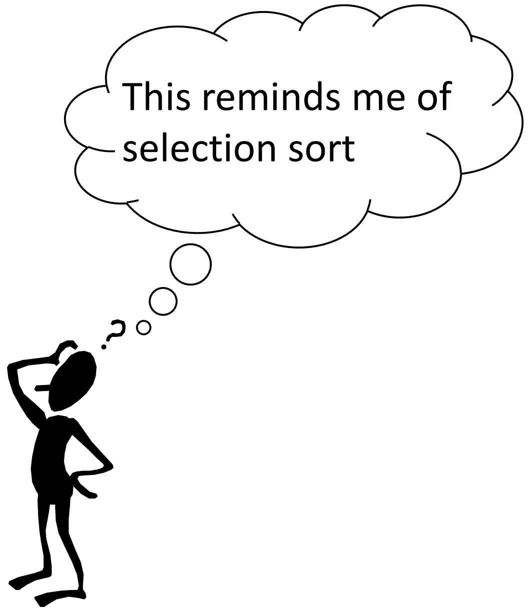
## Sorting By Reversals: A Greedy Algorithm

- If sorting permutation $\pi$ = 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 $\pi$ is denoted *prefix*($\pi$)
  - *prefix*($\pi$) = 3
- This results in an idea for a greedy algorithm: increase *prefix*($\pi$) at every step

# Sort by Reversals: An Example

- Doing so, $\pi$ can be sorted

$$\begin{tikzcd} 1\ 2\ 3\ \underline{6\ 4}\ 5 \arrow[d] \\ 1\ 2\ 3\ 4\ \underline{6\ 5} \arrow[d] \\ 1\ 2\ 3\ 4\ 5\ 6 \end{tikzcd}$$



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

## Greedy Algorithm

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

# 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:

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

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

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

- So, SimpleReversalSort($\pi$) is not optimal

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

# Approximation Algorithms

- Find *approximate* solutions rather than *optimal* solutions
- The approximation ratio of an algorithm $\mathcal{A}$ on input $\pi$ is:

$$\mathcal{A}(\pi) \,/\, \mathrm{OPT}(\pi)$$

where

$\mathcal{A}(\pi)$ - solution produced by algorithm $\mathcal{A}$

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

## Approximation Ratio/Performance Guarantee

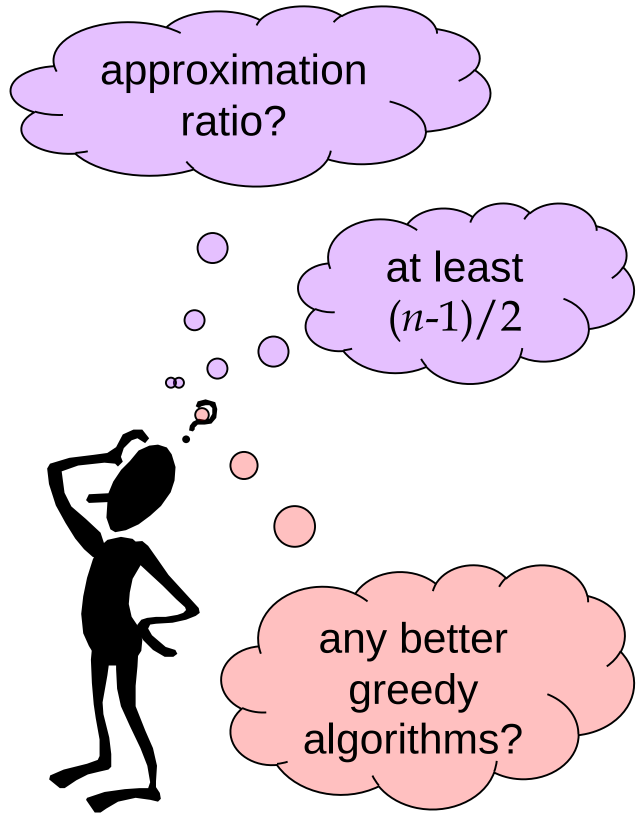
- Approximation ratio (performance guarantee) of algorithm $\mathcal{A}$: max approximation ratio over all inputs of size *n*
  - For a minimizing algorithm $\mathcal{A}$ (like ours):
    - Approx Ratio $= \max_{|\pi| = n} \mathcal{A}(\pi) \ / \ \mathrm{OPT}(\pi) \geq 1.0$
  - For maximization algorithms:
    - Approx Ratio $= \min_{|\pi| = n} \mathcal{A}(\pi) \ / \ \mathrm{OPT}(\pi) \leq 1.0$

# Approximation Ratio

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

Step 0: $\underline{6\ 1}\ 2\ 3\ 4\ 5$
Step 1: $1\ \underline{6\ 2}\ 3\ 4\ 5$
Step 2: $1\ 2\ \underline{6\ 3}\ 4\ 5$
Step 3: $1\ 2\ 3\ \underline{6\ 4}\ 5$
Step 4: $1\ 2\ 3\ 4\ \underline{6\ 5}$
Step 5: $1\ 2\ 3\ 4\ 5\ 6$

Step 0: $\underline{6\ 1\ 2\ 3\ 4\ 5}$
Step 1: $\underline{5\ 4\ 3\ 2\ 1}\ 6$
Step 2: $1\ 2\ 3\ 4\ 5\ 6$


approximation ratio?
at least $(n-1)/2$
any better greedy algorithms?