

# Lecture 19: Genome Rearrangement

## BCB 5300 Algorithms in Computational Biology

Fall 2019

**Tae-Hyuk (Ted) Ahn**

Department of Computer Science  
Program of Bioinformatics and Computational Biology  
Saint Louis University



**SAINT LOUIS  
UNIVERSITY™**

— EST. 1818 —

# Outline

- **Learn about Genome Rearrangement**

# 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      $\pi \leftarrow \pi \circ \rho(i, j)$ 
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  $\pi = \underline{6} \underline{1} 2 3 4 5$  :
  - 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

# Analyzing SimpleReversalSort

- But it can be sorted in two steps:

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

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

- So, SimpleReversalSort( $\pi$ ) is not optimal
- Optimal algorithms are unknown for many problems; approximation algorithms are used

# Adjacencies

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

- A pair of elements  $\pi_i$  and  $\pi_{i+1}$  are **adjacent** if

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

- For example:

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

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

# Breakpoints

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

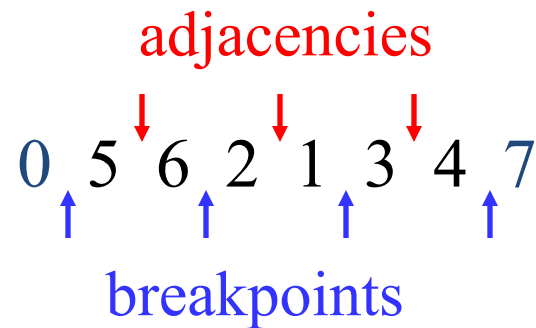
$$\pi = 1 \mid 9 \mid 3 \mid 4 \mid 7 \mid 8 \mid 2 \mid 6 \mid 5$$

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

# Adjacencies & 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  $\pi$  with  $\pi_0 = 0$  and  $\pi_7 = 7$





# Extending Permutations

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

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



Extending with 0 and 10

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

A new breakpoint was created after extending

A permutation of  $n$  may have up to  $(n+1)$  breakpoints

# Reversal Distance and Breakpoints

- Breakpoints are the *bottlenecks* for sorting by reversals.
- Each reversal eliminates at most 2 breakpoints.

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

$$0 | \underline{2 \ 3} | 1 | 4 | 6 \ 5 | 7$$

$$b(\pi) = 5$$

$$0 \ 1 | \underline{3 \ 2} | 4 | 6 \ 5 | 7$$

$$b(\pi) = 4$$

$$0 \ 1 \ 2 \ 3 \ 4 | \underline{6 \ 5} | 7$$

$$b(\pi) = 2$$

$$0 \ 1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7$$

$$b(\pi) = 0$$

$$d(\pi) \geq \frac{b(\pi)}{2}$$

# Sorting By Reversals: A Better 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      $\pi \leftarrow \pi \cdot \rho(i, j)$ 
5   output  $\pi$ 
6 if  $\pi$  is the identity permutation
7   return
```

## BreakPointReversalSort( $\pi$ )

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

Does it always terminate?

How can we be sure that removing some breakpoints does not introduce others?

# Strips

- Strip: an interval between two consecutive breakpoints in a permutation
  - 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)




- 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

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

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



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

# Things to Consider


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

$0\ 1\ 4\ 6\ 5\ 7\ 8\ 3\ 2\ 9$

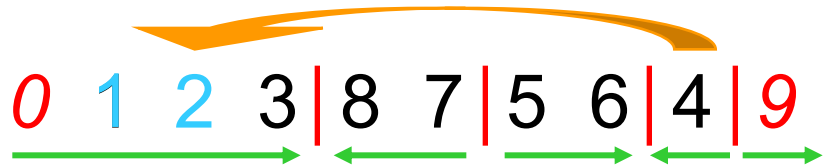


$$b(\pi) = 5$$

reduced by 1!



$0\ 1\ 2\ 3\ 8\ 7\ 5\ 6\ 4\ 9$

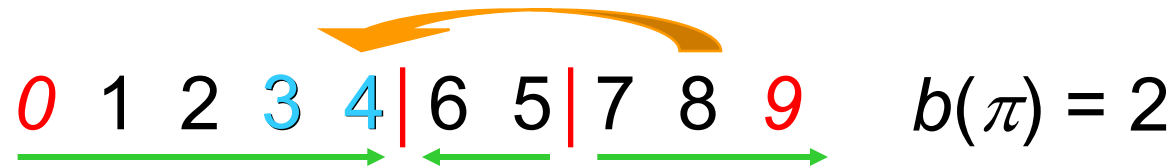
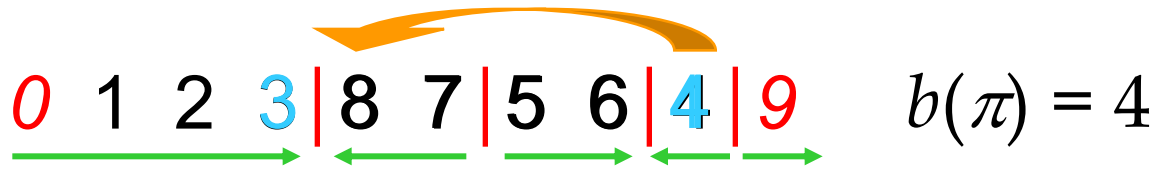


$$b(\pi) = 4$$

- Choose the decreasing strip with the smallest element  $k$  in  $\pi$
- Find  $k - 1$  in the permutation
- Reverse the segment between  $k$  and  $k-1$
- Repeat until there is no decreasing strip

# Things to Consider

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



- Choose the decreasing strip with the smallest element  $k$  in  $\pi$
- Find  $k - 1$  in the permutation
- Reverse the segment between  $k$  and  $k-1$
- Repeat until there is no decreasing strip

# Things to Consider

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



- Choose the decreasing strip with the smallest element  $k$  in  $\pi$
- Find  $k - 1$  in the permutation
- Reverse the segment between  $k$  and  $k-1$
- Repeat until there is no decreasing strip



# Things to Consider

Consider  $p = 1\ 4\ 6\ 5\ 7\ 8\ 3\ 2$

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

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

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

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

$$d(\pi) = 3$$

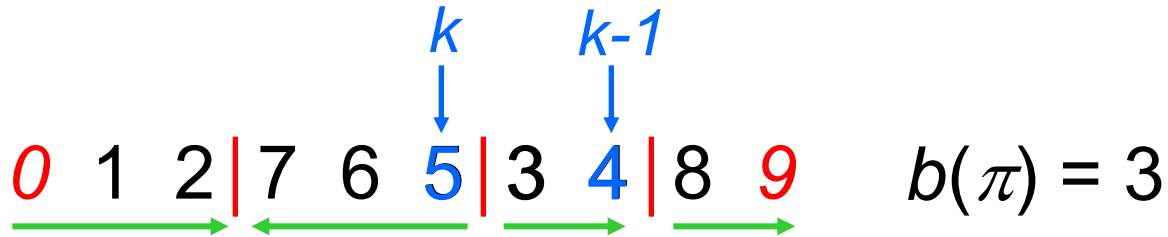
Q: Does it work for arbitrary permutation?

# What if there is no Decreasing Strip?

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

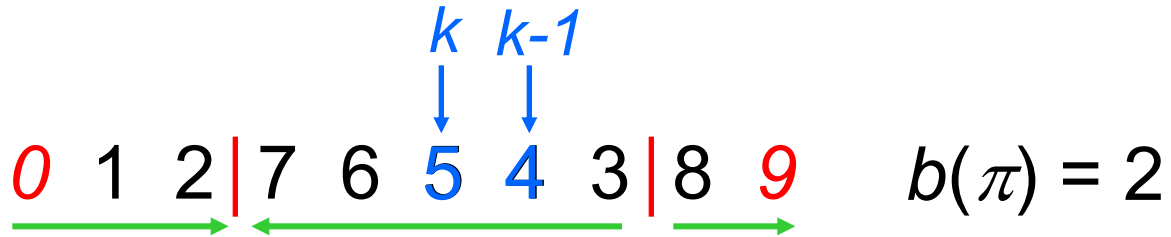
- If there is no decreasing strip, there may be **no reversal  $r$  that reduces the number of breakpoints** (i.e.  $b(p \cdot r) \geq b(p)$  for any reversal  $r$ ).
- By reversing an increasing strip ( # of breakpoints remains unchanged ), we will create a decreasing strip. Then the number of breakpoints will be reduced in the following step.

# What if there is no Decreasing Strip?



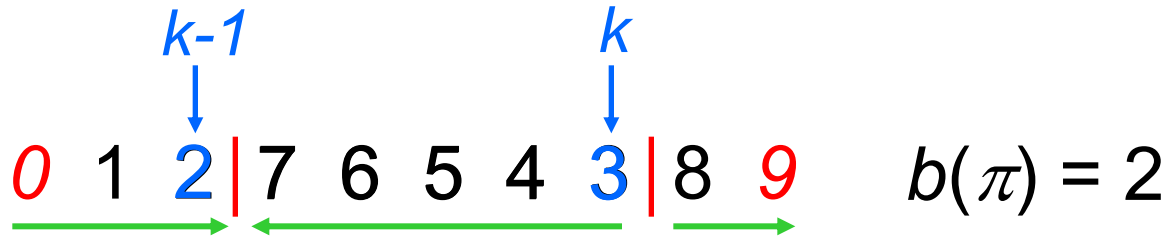
- If there is no decreasing strip, there may be **no reversal  $r$  that reduces the number of breakpoints** (i.e.  $b(p \cdot r) \geq b(p)$  for any reversal  $r$ ).
- By reversing an increasing strip ( # of breakpoints remains unchanged ), we will create a decreasing strip. Then the number of breakpoints will be reduced in the following step.

# What if there is no Decreasing Strip?



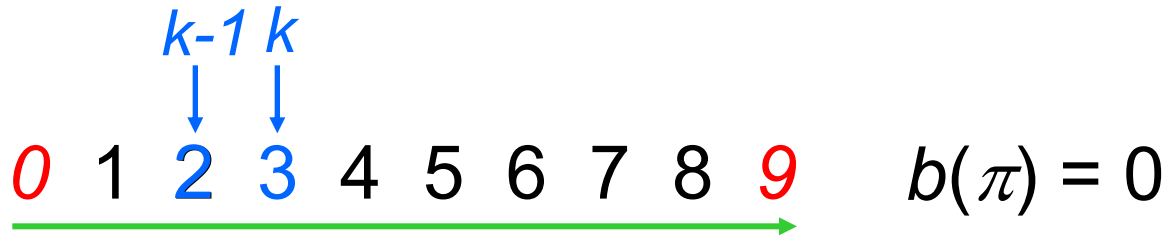
- If there is no decreasing strip, there may be **no reversal  $r$  that reduces the number of breakpoints** (i.e.  $b(p \cdot r) \geq b(p)$  for any reversal  $r$ ).
- By reversing an increasing strip ( # of breakpoints remains unchanged ), we will create a decreasing strip. Then the number of breakpoints will be reduced in the following step.

# What if there is no Decreasing Strip?



- If there is no decreasing strip, there may be **no reversal  $r$  that reduces the number of breakpoints** (i.e.  $b(p \cdot r) \geq b(p)$  for any reversal  $r$ ).
- By reversing an increasing strip ( # of breakpoints remains unchanged ), we will create a decreasing strip. Then the number of breakpoints will be reduced in the following step.

# What if there is no Decreasing Strip?



- If there is no decreasing strip, there may be **no reversal  $r$  that reduces the number of breakpoints** (i.e.  $b(p \cdot r) \geq b(p)$  for any reversal  $r$ ).
- By reversing an increasing strip ( # of breakpoints remains unchanged ), we will create a decreasing strip. Then the number of breakpoints will be reduced in the following step.

# ImprovedBreakpointReversalSort

## ImprovedBreakpointReversalSort( $\pi$ )

```
1 while  $b(\pi) > 0$ 
2   if  $\pi$  has a decreasing strip
3     Among all possible reversals, choose reversal  $\rho$ 
        that minimizes  $b(\pi \cdot \rho)$ 
4   else
5     Choose a reversal  $\rho$  that flips an increasing strip in  $\pi$ 
6    $\pi \leftarrow \pi \cdot \rho$ 
7   output  $\pi$ 
8 return
```

## BreakPointReversalSort( $\pi$ )

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