Genome rearrangements and approximation algorithms

Somewhere, something went wrong…

Turnip vs Cabbage: Look and Taste Different

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

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

- Almost identical mtDNA sequences:
  - In 1980s Jeffrey Palmer studied evolution of plant organelles by comparing mitochondrial genomes of the cabbage and turnip
  - 99% similarity between genes

Turnip vs Cabbage: Different mtDNA Gene Order

- Gene order comparison:
  - These surprisingly identical gene sequences differed in gene order

Transforming Cabbage into Turnip

- This study helped pave the way to analyzing genome rearrangements in molecular evolution

Transforming Mouse into Human

- Humans and mice have similar genomes, but their genes are ordered differently
  - ~245 rearrangements
    - Reversals
    - Fusions
    - Fissions
    - Translocation
Human/mouse gene order conservation: Implications

- Waardenburg’s syndrome is characterized by pigmentary dysphasia
- Gene implicated in the disease was linked to human chromosome 2, but it was not clear where exactly it is located on chromosome 2.
- 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.

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

Types of Rearrangements

- Reversal
  
  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

  Simple representation: inversions in concatenated chromosomes

Reversals

- 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

The reversal introduced two breakpoints (disruptions in order).
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

Reversal Distance Problem

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

• Input: Permutations $\pi$ and $\sigma$

• Output: A series of reversals $\rho_1, \ldots, \rho_t$ transforming $\pi$ into $\sigma$ such that $t$ is minimum

• $t$: reversal distance between $\pi$ and $\sigma$

• $d(\pi, \sigma)$: smallest possible value of $t$, given $\pi$ and $\sigma$

Gene order rearrangement: overlapping inversions

Inference of inversion history: “sorting by reversals”

Reversals and Gene Orders

• Gene order is represented by a permutation $\pi$:
  \[ \pi = \pi_1 \pi_2 \pi_3 \pi_4 \pi_5 \pi_6 \pi_7 \pi_8 \pi_9 \pi_{10} \]
  \[ \rho(i, j) \]

• Reversal $\rho(i, j)$ reverses (flips) the elements from $i$ to $j$ in $\pi$.

Sorting By Reversals Problem

• Goal: Given one permutation, find a shortest series of reversals that transforms it into the identity permutation ($1 \ 2 \ \ldots \ n$)

• Input: Permutation $\sigma$

• Output: A series of reversals $\rho_1, \ldots, \rho_t$ transforming $\pi$ into the identity permutation such that $t$ is minimum
Sorting by reversals: how many steps?

<table>
<thead>
<tr>
<th>2</th>
<th>4</th>
<th>3</th>
<th>5</th>
<th>8</th>
<th>7</th>
<th>6</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
</tbody>
</table>

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

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

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 \( \pi \)
- **Output**: A series of prefix reversals \( \rho_1, \ldots, \rho_t \) transforming \( \pi \) 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
### 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.
- The length of the already sorted prefix of \( \pi \) is denoted \( \text{prefix}(\pi) \)
  - \( \text{prefix}(\pi) = 3 \)
- This results in an idea for a greedy algorithm: increase \( \text{prefix}(\pi) \) at every step

### Greedy Algorithm: An Example

- Doing so, \( \pi \) can be sorted
  
  \[
  1 \, 2 \, 3 \, 6 \, 4 \, 5 \\
  \downarrow \\
  1 \, 2 \, 3 \, 4 \, 6 \, 5 \\
  \downarrow \\
  1 \, 2 \, 3 \, 4 \, 5 \, 6
  \]

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

### Greedy Algorithm: Pseudocode

```plaintext
SimpleReversalSort(\pi)
1 for i \in 1 to n - 1
2 j \in 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

- Does SimpleReversalSort does guarantee the smallest number of reversals ?
  - NO
- E.g., takes five steps on \( \pi = 6 \, 1 \, 2 \, 3 \, 4 \, 5 \):
  - Step 1: \( 1 \, 6 \, 2 \, 3 \, 4 \, 5 \)
  - Step 2: \( 1 \, 2 \, 3 \, 4 \, 5 \, 6 \)
  - 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 (cont’d)

- 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

### 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
Approximation Algorithms

- These algorithms find approximate solutions rather than optimal solutions.
- The approximation ratio of an algorithm \( A \) on input \( x \) is:
  \[
  \frac{A(x)}{OPT(x)}
  \]
  where
  \( A(x) \) - solution produced by algorithm \( A \)
  \( OPT(x) \) - optimal solution of the problem

Approximation Ratio/Performance Guarantee

- Approximation ratio (performance guarantee) of algorithm \( A \): \( \max_{|x| = n} \frac{A(x)}{OPT(x)} \)
  - For algorithm \( A \) that minimizes objective function (minimization algorithm):
    \[
    \max_{|x| = n} \frac{A(x)}{OPT(x)}
    \]

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

Adjacencies and Breakpoints

Consider \( \pi = \pi_1\pi_2\pi_3\ldots\pi_{n-1}\pi_n \)

- A pair of elements \( \pi_i \) and \( \pi_{i+1} \) are adjacent if
  \[\pi_{i+1} - \pi_i = \pm 1\]
- For example:
  \[\pi = 1 \ 9 \ 3 \ 4 \ 7 \ 8 \ 2 \ 6 \ 5\]
  - (3, 4) or (7, 8) and (6, 5) are adjacent pairs

Breakpoints: An Example

There is a breakpoint between any pair of non-adjacent elements:

\[\pi = 1 \ | \ 9 \ | \ 3 \ 4 \ | \ 7 \ 8 \ | \ 2 \ | \ 6 \ 5\]

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

- We put two elements \( \pi_0 = 0 \) and \( \pi_{n+1} = n+1 \) at the ends of \( \pi \).
  - Example:

\[
\pi = 1 9 3 4 7 8 2 6 5
\]

Extending with 0 and 10

\[
\pi = 0 1 9 3 4 7 8 2 6 5 10
\]

Note: A new breakpoint was created after extending.

Reversal Distance and Breakpoints

- Each reversal eliminates at most 2 breakpoints.
- This implies:
  \[
  \text{reversal distance} \geq \frac{\#\text{breakpoints}}{2}
  \]

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).

\[
0 1 9 4 3 7 8 2 5 6 10
\]

- 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 \( n+1 \).

Strip

- For \( \pi = 1 4 6 5 7 8 3 2 \)

\[
0 1 | 4 6 5 7 8 3 2 | 9
\]

\( b(\pi) = 5 \)

- Choose decreasing strip with the smallest element \( k \) in \( \pi \) (\( k = 2 \) in this case)
- Find \( k-1 \) in the permutation
- Reverse the segment between \( k \) and \( k-1 \)

\[
0 1 | 4 6 5 7 8 3 2 | 9
\]

\( b(\pi) = 5 \)

\[
0 1 2 3 8 7 5 6 4 9
\]

\( b(\pi) = 4 \)

When no decreasing strip is available

- Reverse an increasing strip
  (\( \# \) of breakpoints stay unchanged)
- The number of breakpoints will be reduced in the next step (theorem 1).
ImprovedBreakpointReversalSort

ImprovedBreakpointReversalSort(π)
1 while $b(π) > 0$
2     if π has a decreasing strip
3         Among all possible reversals, choose reversal $ρ$
4             that minimizes $b(π • ρ)$
5     else
6         Choose a reversal $ρ$ that flips an increasing strip in $π$
7     output $π$
8 return

ImprovedBreakpointReversalSort Approximation Guarantee

- $ImprovedBreakpointReversalSort$ is an approximation algorithm with a performance guarantee of at most 4
  - It eliminates at least one breakpoint in every two steps; at most $2b(π)$ steps
  - Approximation ratio: $2b(π) / d(π)$
  - Optimal algorithm eliminates at most 2 breakpoints in every step: $d(π) ≥ b(π) / 2$
  - Performance guarantee:
    $\left\lfloor \frac{2b(π)}{b(π) / 2} \right\rfloor = 4$

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

Calculate min # of breakpoints exactly: Breakpoint graph

$G(π, γ)$

reversal distance = number of elements + 1 - number of cycles + h

where h is rather complicated, but can be computed from breakpoint graph in polynomial time.

- Here, reversal distance $= 8 + 1 - 5 + 0 + 0 = 4$

Constructing Breakpoint Graph: Dot Plot

- Duality Theorem for Sorting by Reversals - simple and imprecise version.

reversal distance = number of elements + 1 - number of cycles

- Dot Plot
Constructing Breakpoint Graph: Black Path

Constructing Breakpoint Graph: Gray Path

Constructing Breakpoint Graph: Superimposing Two Paths

Constructing Breakpoint Graph: Removing Dot-Plot

Comparing complete Human and Mouse genomes

Human-mouse breakpoint graph
Summary

• 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