Bounds for Pancake Problem

Mehrdad Aksari Mahabadi
Supervisor: Dr. Fatemeh Zare
Jan. 2024

Contents

1 Introduction

DNA usually evolves by tiny mutations. However, sometimes something more radical happens, and the order of a contiguous segment of genome is rearranged. Understanding these rearrangement sheds light on the process of evolution. Every genome rearrangement results in a change of gene ordering, and a series of these rearrangements can alter the genomic architecture of a species. Biologists are interested in the most parsimonious evolutionary scenario, that is, the scenario involving the smallest number of reversals. While there is no guarantee that this scenario represents an actual evolutionary sequence, it gives us a lower bound on the number of rearrangements that have occurred and indicates the similarity between two species.

1.1 Human and Mice

In some ways, the human genome is just the mouse genome cut into about 300 large genomic fragments, called synteny blocks, that have been pasted together in a different order. Both sequences are just two different shufflings of the ancient mammalian genome.

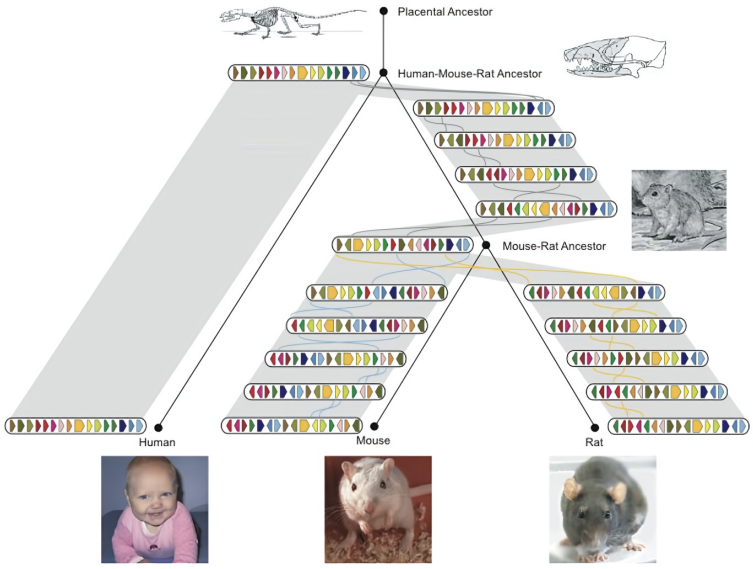

Figure 1: Evolutionary tree of mouse, rat, and human showing genome rearrangements[1]

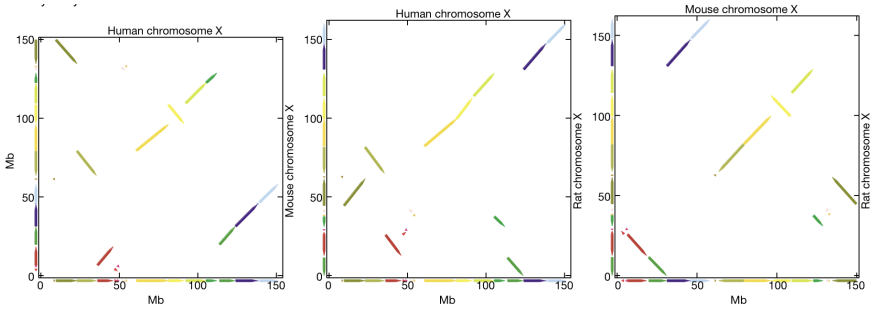

Figure 2: sentry block alignments of each pair of species

[1]Extreme conservation of genes on X chromosomes across mammalian species provides an opportunity to study the evolutionary history of X chromosomes independently of the rest of the genomes, since the gene content of X chromosomes has barely changed throughout mammalian evolution. However, the order of genes on X chromosomes has been disrupted several times. In other words, genes that reside on the X chromosome stay on the X chromosome (but their order may change). All other chromosomes may exchange genes, that is, a gene can move from one chromosome to another.

1.2 Combinatorial Problems

Every study of genome rearrangements involves solving the combinatorial puzzle of finding a series of rearrangements that transform one genome into another. Due to the parsimonious nature of evolution, we are interested in the minimum number of steps in which one genome sequence can be transformed into the other using only reversals, the most common evolutionary event.

1.2.1 Sorting by Reversals

In the simplest form, the order of genes can be represented by a permutation $\pi = \pi_1\pi_2 \dots \pi_n$. We define a reversal $\rho(i, j)$ transforms permutation in the following way

$$\pi = \pi_1\pi_2 \dots \pi_i\pi_{i+1} \dots \pi_j\pi_{j+1} \dots \pi_n$$
$$\pi.\rho(i, j) = \pi_1\pi_2 \dots \pi_j\pi_{j-1} \dots \pi_{i-1}\pi_i \dots \pi_n.$$

In the sorting by reversals problem, we are interested in applying a number of reversals $\rho_1\rho_2 \dots \rho_n$ to transform permutation π into the identity permutation.

1.2.2 Pancake Problem

In 1975, Jacob E. Goodman (under the pseudonym Harry Dweighter) posed the following problem:

> The chef in our place is sloppy, and when he prepares a stack of pancakes they come out all different sizes. Therefore, when I deliver them to a customer, on the way to the table I rearrange them (so that the smallest winds up on top, and so on, down to the largest at the bottom) by grabbing several from the top and flipping them over, repeating this (varying the number I flip) as many times as necessary. If there are n pancakes, what is the maximum number of flips (in terms of n) that I will ever have to use to rearrange them?

The Pancake Problem, is also know as Sorting by Prefix Reversals. In this setting we can only apply reversals that start from the beginning of the sequence, $\rho(1, j)$ or simply $\rho(j)$ in this case.

1.2.3 Bounds for Pancake Flipping Problem

On obvious algorithm for sorting the permutation is as follows: Given permutation π assume that the elements $\pi_i\pi_{i+1} \dots \pi_n$ are already in the correct order for $1 \leq i \leq n$. If $i = 1$ the permutation is already sorted, hence we assume that $i > 1$. We denote the position of $i - 1$ by π_j, and perform the reversal operations $\rho(\pi_j).\rho(\pi_{i-1})$. Thus, we can put each element in its correct position by 2 operations. Since we only need to put $n - 1$ elements in their correct position, this algorithms sorts any permutation in at most $2(n - 1)$ operations. In 1979, Gates and Papadimitriou [4] improved bounds for the pancake flipping problem. In this report, we'll examine their presented algorithm, that sorts any permutation in at most $\frac{5}{3}(n + 1)$ steps.

1.2.4 Related Work

The MIN-SBR problem has been shown to be NP-Complete by Caprara [1]. However, Hannenhalli and Pevzner [2] gave an exact polynomial time algorithm for a slightly restricted version of the problem. In 2011, Laurent Bulteau, Guillaume Fertin, and Irena Rusu [3] showed that MIN-SBPR is also NP-Complete, thereby answering a question that had been open for over three decades.

2 An Algorithm

Let $f(n)$ denote the maximum number of steps required to sort a permutation of size n. The table below shows the value for f for $1 \leq n \leq 9$.

n	1	2	3	4	5	6	7	8	9
$f(n)$	0	1	3	4	5	7	8	9	10

2.1 Definitions

As before, π_j represents the jth position in permutation π. We call π_j and π_{j+1} an **adjacency** if $|\pi_j - \pi_{j+1}| = 1$. If $\pi = xby$, and (π_j, π_{j+1}) are adjacencies for $j = |x| + 1, \ldots, |b|$, and $(\pi_{|x|}, \pi_{|x|+1})$, $(\pi_{|b|}, \pi_{|b|+1})$ are not adjacencies, we call b a **block**. If π_j is not in a block, it is said to be **free**. The algorithm presented here will create $n-1$ adjacencies of the forms shown in Fig. 2 (a) and Fig. 2 (b)[2], and then transform the permutation into the identity permutation by at most 4 operations.

$k \ldots n$	$k-1 \ldots 1$
$n \ldots k$	$k-1 \ldots 1$
$1 \ldots k-1$	$k \ldots n$

(a)

$k-1 \ldots 1$	$k \ldots n$
$k \ldots n$	$k-1 \ldots 1$

proceed like (a)

(b)

Fig. 2

2.2 The Algorithm

algorithm ϕ
input a permutation π
output a permutation σ with $n-1$ adjacencies
set $\sigma = \pi$ and repeat the following steps until the algorithm halts
let t be the first element of σ, at least one of the following cases hold

1. if t is free, and $t + o$[3] is free, apply the flipping shown in Fig. 3 (a)
2. if t is free, and $t + o$ is the first element of a block, flip according to Fig. 3 (b)
3. if t is free, and both $t+1$ and $t-1$ are the last element of a block, use Fig. 3 (c) flips
4. if t is in a block, and $t + o$ is free, perform the sequence of flips shown in Fig 3. (d)
5. if t is in a block, and $t + o$ is the first element of a block, perform the sequence of flips shown in Fig 3. (e)
6. if t is in a block with the last element $t+k.o, k > 0$. $t-o$ is the last element of another block. If $t+(k+1).o$ is free, perform according to Fig. 3 (f) or Fig. 3 (g) based on the relative position of $t+(k+1)o$ and $t-o$
7. if t is in a block with the last element $t+k.o, k > 0$, and $t+(k+1).o$ is in another block, perform according to Fig. 3 (h) or Fig. 3 (k), depending on whether $t + (k+1).o$ is in the beginning or the end of its block
8. none of the above, σ has $n-1$ adjacencies, return σ

[2] addition is understood modulo n here and therefore 1 and n are considered adjacent
[3] $o \in \{1, -1\}$

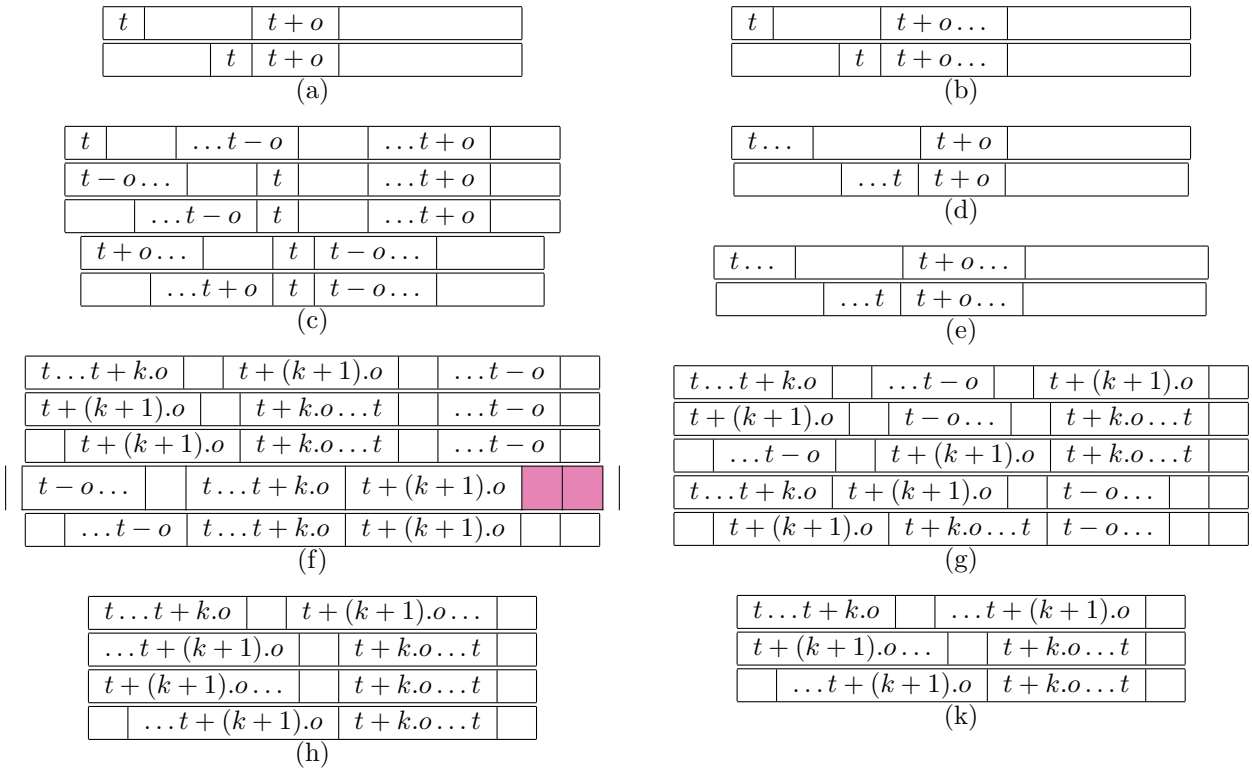

Fig. 3

The first three conditions correspond to the case where t is free. If t is free, $t+o$ could be either free or in a block. The situation where $t+o$ is not in a block is covered in the first case. If $t+o$ is in a block, it's either the first or the last element of a block. If it's the first element of a block, then second case is applied. If it's not the first element of a block, we can be sure that both $t+1$ and $t-1$ are the last elements of their blocks, which corresponds to case 3. When t is in a block cases 4-7 are applied.

Claim Algorithm ϕ creates a permutation with $n-1$ adjacencies in at most $\frac{5n-7}{3}$ steps
Proof To see why the algorithms halts and returns a permutation with $n-1$ adjacencies, note that at each step at least 1 adjacency is added without destroying the existing ones. Moreover, at each step one of the cases 1-7 applies. Thus, it remain to show that the algorithm runs in at most $\frac{5n-7}{3}(n+1)$ steps.
Call the action of case 1 action 1, case 2 action 2, case 3 and 6 action 3, case 4 action 4, case 5 and case 7, action 5 and action 7 respectively. We denote x_i as the number of times action i was performed in the execution of the algorithm. The total number of moves in the algorithm is given by

$$z = x_1 + x_2 + 4x_3 + x_4 + x_5 + 2x_7 \tag{1}$$

Where x_i is multiplied by the number of flips involved in the execution of action i. Action 3 can be divided into four special subcases, according to what happens in the flipping of Fig. 3(c) (or 3(f), 3(g)) in the step that comes before the last step[4]:

1. be non-adjacent
2. form a new block
3. merge a block with a singleton
4. merge to blocks

We distinguish between these cases by writing $x_3 = x_{31}+x_{32}+x_{33}+x_{34}$. We start with a adjacencies and increase the number of adjacencies according to Table. 2, and eventually end up with $n-1$ adjacencies. Additionally, we begin with b blocks and end up with only 1 after the permutation is sorted. Therefore, we can write:

$$n-1 = a + x_1 + x_2 + 2x_{31} + 3x_{32} + 3x_{33} + 3x_{34} + x_4 + x_5 + x_7 \tag{2}$$

$$1 = b + x_1 - x_{31} - x_{33} - 2x_{34} - x_5 - x_7 \tag{3}$$

[4]we have colored the two squares in Fig. 3 for which the following cases can happen

action	1	2	31	32	33	34	4	5	7
increase in adjacencies	1	1	2	3	3	3	1	1	1
decrease in blocks	-1	0	1	0	1	2	0	1	1

Table. 2

Since each block consists of at least one adjacent pair, we can say that $b \leq a$. Therefore, equation (2) becomes:

$$b + x_1 + x_2 + 2x_{31} + 3x_{32} + 3x_{33} + 3x_{34} + x_4 + x_5 + x_7 \leq n - 1 \tag{4}$$

Hence, if we apply the algorithm in any way[5], we would, at worst, maximize equation (1) subject to constraints (3), (4). This is an integer linear programming problem. By the weak duality theorem, we know that every solution to the dual problem is an upper bound on the original problem. Therefore, we consider the dual problem instead. Hence, we will need to minimize $w = y_2 + (n-1)y_3$, subject to the following equations:

$$y_2 + y3 \geq 1, \tag{5}$$
$$y_3 \geq 1, \tag{6}$$
$$-y_2 + 2y_3 \geq 4, \tag{7}$$
$$3y_3 \geq 4, \tag{8}$$
$$-y_2 + 3y_3 \geq 4, \tag{9}$$
$$-2y_2 + 3y_3 \geq 4, \tag{10}$$
$$-y_2 + y_3 \geq 1, \tag{11}$$
$$-y_2 + y_3 \geq 2, \tag{12}$$
$$y_2 + y_3 \geq 0 \tag{13}$$

Therefore, in order to prove our claim, we only need to find a pair (y_2, y_3) such that $y_2 + (n-1)y_3 = \frac{5n-7}{3}$ and (y_2, y_3) is a feasible solution to the stated minimization problem. Putting $(y_2, y_3) = (\frac{-2}{3}, \frac{5}{3})$ satisfies all the conditions. Since there will be at most 4 steps after we run our algorithm, the bound $f(n) \leq \frac{5n-7}{3} + 4 = \frac{5n+5}{3}$ follows immediately.

3 A Lower Bound

To prove the lower bound, $f(n) \geq \frac{17n}{16}$, we will only have to find a permutation, τ, that can't be sorted in less than $\frac{17n}{16}$ steps. Let $\tau = 17536428$. For a positive integer k, τ_k denotes τ with each of the integers increased by $8.(k-1)$. In other words, τ_k is the sequence $1_k 7_k 5_k 3_k 6_k 4_k 2_k 8_k$, where $m_k = m + (k-1).8$. For example, τ_2 =9 15 13 11 14 12 10 16. Consider the permutation $\chi = \tau_1 \tau_2^R \tau_3 \tau_4^R \dots \tau_{m-1} \tau_m^R$, where m is an even integer, and $n = |\chi| = 8.m$.

Claim $\frac{17n}{16} \leq f(\chi) \leq \frac{19n}{16}$.

Proof We will first prove the upper bound. In order to show that $f(\chi) \leq \frac{19n}{16}$, we need to find a sequence of reversals that transforms χ into the identity permutation in at most $\frac{19n}{16}$ flips. We first do the following sequence of moves:

$$\chi \longrightarrow \tau_2 \tau_1^R \tau_3 \tau_4^R \dots \tau_{m-1} \tau_m^R \longrightarrow \tau_2^R \tau_1^R \tau_3 \tau_4^R \dots \tau_{m-1} \tau_m^R \longrightarrow \tau_1 \tau_2 \tau_3 \tau_4^R \dots \tau_{m-1} \tau_m^R$$

Following the same procedure, we can convert χ to $\chi\prime = \tau_1 \tau_1 \tau_3 \tau_4 \dots \tau_{m-1} \tau_m$ in $\frac{3m}{2} = \frac{3n}{16}$ reversals. After that for each τ_i in $\chi\prime$ we perform the following reversals, for simplicity we removed the subscript k.

$$\chi\prime = x17536428y \longrightarrow 571x^R 36428y \longrightarrow 63x175428y \longrightarrow 1x^R 3675428y \longrightarrow 45763x128y$$
$$\longrightarrow 67543x128y \longrightarrow 76543x128y \longrightarrow 21x^R 345678y \longrightarrow x1234567y$$

Hence, each τ_i is sorted in 8 reversals, and permutation χ is sorted in $8.m + \frac{3n}{16} = \frac{19n}{16}$ reversals.

To prove the lower bound $\chi = \chi_0 \longrightarrow \chi_1 \longrightarrow \chi_2 \longrightarrow \dots \longrightarrow \chi_{f(\chi)} = e$ be an optimal sequence of reversals for χ. Each of χ_j, $j = 1, \dots, f(\chi)$ is called a **move**. Let us call a move **k-stable**, if it contains a substring of the form $1_k 7_k \sigma 2_k 8_k$ (or its reverse), where σ can be any permutation of the string 3456. A move χ_j is called an **event**, if χ_j is k-stable, for some k, but $\chi_{j+1}, \chi_{j+2}, \dots, \chi_{f(\chi)}$ are not.

[5]by any ordering we mean any way of applying the cases 1-7

Claim There are exactly m events
Proof χ_0 is k-stable for every k, $1 \leq k \leq m$. Moreover, the identity permutation is not k-stable. The claim follows immediately from the fact that no permutation can cease to be k_1-stable and k_2-stable, where $k_1 \neq k_2$, in only one reversal operation.

Let us call χ_j a waste if it has no more adjacencies than χ_{j-1}. Let w denote the total number of wastes among $\chi_j, \chi_{j+1}, \ldots, \chi_{f(\chi)}$.

Claim $f(\chi) \geq n + w$
Proof χ has no adjacencies, and after applying the reversal operations we end up with $n-1$ adjacencies. Since any move that is not a waste creates one adjacency, it must be that $f(\chi) \geq n + w$.

Claim For all j, $1 \leq j \leq m-1$, there exists a waste χ_l with $i_j \leq l \leq i_{j+1}$.
Proof Since by previous claims there are exactly m events, we can rewrite the optimal sequence in the following way, where each χ_{i_j} corresponds to an event

$$\chi_{i_1} \rightsquigarrow \chi_{i_2} \rightsquigarrow \cdots \rightsquigarrow \chi_{i_m}$$

Suppose that the claim is not correct. That is for some j, all the moves $\chi_j, i_j \leq j \leq i_{j-1}$, construct a new adjacency without destroying the existing ones. Choose k such that χ_{i_j-1} is the last k-sortable permutation in the sequence. Then $\chi_{i_j-1} = x1_k7_k\sigma 2_k8_k y$. WLOG, assume that $\tau = x17536428y$, again we have omitted subscript k for the sake of simplicity. We distinguish among two cases. If $x = \epsilon$, the next move must be $\chi_{i_j} = 46357128y$. This is due to the fact that any other move would have been a waste or preserved k-sortability. Now, according to our hypothesis, no move is a waste until after the next event. However, if any flip with more than 4 elements is a waste. If $x \neq \epsilon$, that is $\tau = x17536428y$, since χ_{i_j} is neither a waste nor k-sortable, it must be that $x = 9z$. Therefore, $\chi_{i_j} = 2463571z^R98y$. Since 2 is at the beginning of χ_{i_j}, any reversal other than the local rearrangements of elements $\{1, 2, 3, 4, 5, 6, 7\}$ would be a waste. χ_{i_j} will eventually turn into $7654321z^R98y$, since we must increase adjacencies at every step according to our hypothesis. After that, any move will be a waste, which is a contradiction to our hypothesis.

The lower bound now follows directly from the claims.

$$f(\chi) \geq n + w \geq n + \frac{m}{2} = \frac{17}{16}n$$

4 Conclusion

In this report, we introduced genome rearrangement and two relevant combinatorial problems, MIN-SBPR and MIN-SBR. We then examined bounds for MIN-SBPR. It is worth noting that in 2008, the upper bound was improved, thirty years later, to $\frac{18}{11}$ by a team of researchers at the University of Texas at Dallas, led by Founders Professor Hal Sudborough[7].

5 Acknowledgement

This report is mostly based on Gates and Papadimitriou[4]. Most of the text in the introduction is inspired by the chapter on greedy algorithms of Pevzner and Jones [5]. The synteny block alignments in section 1.1 are taken from [6].

References

[1] A. Caprara: Sorting Permutations by Reversals and Eulerian Cycle Decompositions. SIAM J. on Discrete Mathematics 12(1): 91–110, 1999.

[2] S. Hannenhalli, P. A. Pevzner. Transforming Cabbage into Turnip: Polynomial Algorithm for Sorting Signed

[3] Bulteau, Laurent, Guillaume Fertin, and Irena Rusu. "Pancake flipping is hard." Journal of Computer and System Sciences 81.8 (2015): 1556-1574.

[4] William H. Gates, Christos H. Papadimitriou, Bounds for sorting by prefix reversal, Discrete Mathematics, Volume 27, Issue 1, 1979, Pages 47-57

[5] Neil C. Jones Pavel A. Pevzner, An Introduction to Bioinformatics Algorithms

[6] Genome sequence of the Brown Norway rat yields insights into mammalian evolution

[7] Chitturi, B.; Fahle, W.; Meng, Z.; Morales, L.; Shields, C. O.; Sudborough, I. H.; Voit, W. (August 31, 2009). "An (18/11)n upper bound for sorting by prefix reversals".