Descrambling Order Analysis in Ciliates

Nazifa Azam Khan and Ian McQuillan* **

Department of Computer Science,
University of Saskatchewan, Saskatoon, SK, Canada
nak310@mail.usask.ca, mcquillan@cs.usask.ca

Abstract. Certain genera of ciliates undergo a large genomic transformation, where many segments get rearranged and removed. A topic of interest is to predict a (partial) order on the rearrangement of segments to descramble. Similar to phylogenetic analysis, this prediction can be based on the principle of parsimony, whereby the smallest sequence of operations is likely close to the actual number. The *Oxytricha trifallax* genome is analyzed, providing evidence that multiple parallel recombination operations occur during descrambling, with alignment of interleaving segments in a manner that can be captured with the shuffle operation. Two similar systems involving shuffle are created, an optimal algorithm for each is created, and executed on the genomic data. One system can descramble 96.63% of the scrambled micronuclear chromosome fragments by 1 or 2 applications of shuffle, and every sequence can be descrambled with at most seven operations.

Keywords: ciliates, macronucleus, micronucleus, scrambled genes, shuffle, parsimony.

1 Introduction

Ciliated protozoa are a group of unicellular organisms, where each cell has two types of nuclei; the *micronucleus* (MIC) and the *macronucleus* (MAC). When two cells mate, they exchange haploid micronuclei, destroy their own macronuclei, and then develop a new MAC from the genetic material in the new MIC. In the MIC of stichotrichs (a group of ciliates), less than 5% of the DNA actually encodes genes, with a large amount of non-coding DNA both between genes, and also within genes. In contrast, the MAC largely consists of single gene chromosomes, and the intragenic spacer is not present. Indeed, certain segments get removed when converting to MAC chromosomes, called *internal eliminated segments* (IESs), while certain segments remain, called *macronuclear destined segments* (MDSs); see Figure 1. Even stranger, many genes have the MDSs in a different order between the MIC and MAC version of a gene, and these MDSs become rearranged, or descrambled, during the conversion of the MIC to the MAC in a process known as the gene assembly process [17].

* Supported, in part, by a grant from the Natural Sciences and Engineering Council of Canada (Ian McQuillan).

** Published in Proceedings of UCNC 2017. The final authenticated version is available online at https://doi.org/10.1007/978-3-319-58187-3_16.

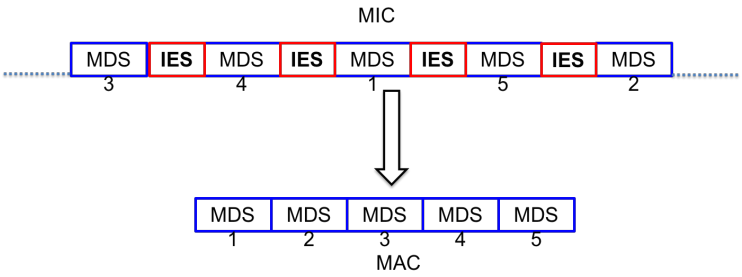

Fig. 1: Simplified conversion of a MAC chromosome fragment from a MIC chromosome fragment.

The process of gene assembly is a fascinating example of computing taking place in nature [17]. An extensive amount of parallel computation occurs during the gene assembly process. In fact, descrambling MDSs is a computationally hard problem, as even the problem of aligning a micronuclear gene to a macronuclear gene to partition it into segments is an NP-complete problem [8], meaning that very likely no optimal polynomial time algorithm exists to solve it. Knowledge about how nature is solving these computationally complex problems may assist computer scientists to construct new algorithms and techniques, and conversely, computational results could be used to infer biological conclusions.

There are a variety of biological and computational models and hypotheses that have been created to explain the gene assembly process in ciliates. Originally, a model known as the intermolecular model viewed this gene descrambling as a computational process, consisting of one intramolecular and two intermolecular operations of DNA recombination on pointers [10]. Another theoretical model for gene assembly, known as the intramolecular model was introduced by Prescott et al. [16] and Ehrenfeucht et al. [5]. It consists of three unary molecular operations based on pointers: loop excision, hairpin excision, and double loop deletion, that explains IES excision and MDS rearrangements during gene assembly. In 2009, the notion of assembly graphs was introduced to model the DNA structure during the recombination process [1]. They introduced another model in 2012 that describes rearrangement pathways of DNA recombination events with three rewriting rules: insertion, deletion, and inversion [2].

In 1980, Meyer et al. studied *Stylonychia mytilus* by means of electron microscopy and observed that at the very beginning of the gene assembly process, IESs are eliminated in the form of chromatin rings (loops) [12]. Then, micronuclear chromatin becomes organised into coiled, lampbrush patterns, or loop-like structures (Figure 2) that might be a necessary prerequisite for later IES elimination and MDS rearrangement [13,14]. Chromatin consists of DNA that is tightly coiled around proteins called histones that condenses to form chromosomes. In 2008, Matthias et al. concluded that multiple descrambling pathways may produce functional macronuclear molecules [15], and that there are occurrences of multiple parallel inversion and transposition events through each pathway during

assembly [15]. *Inversion* takes a particular segment of MDSs in a MIC gene and puts it back in the opposite direction, whereas *transposition* excises a segment of DNA and puts it back in a different position.

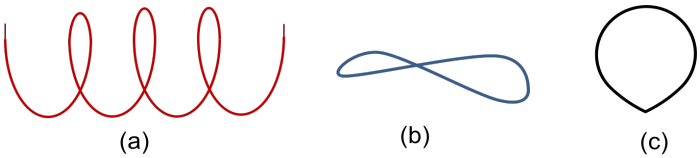

Fig. 2: Coiled structure (a) lampbrush pattern (b) loop, or ring shape (c).

The *Oxytricha trifallax* MIC genome has been recently sequenced, allowing for deeper analysis that what was previously possible. Very recently, Burns et al. investigated the scrambled gene architectures in the *Oxytricha trifallax* genome [3]. For scrambled genes, they identified the precursor scrambled patterns with so-called sequence rearrangement maps, and assembly graph representations. From their analysis of the MIC and MAC genes, they deduced that 87.2% of the MIC loci is non-scrambled, and among the scrambled MIC contigs, 81.7% follow a pattern involving either a sequence of consecutive odd numbered MDSs, followed by a sequence of consecutive even numbered MDSs, or vice versa [3]. These statistics are very similar to those we independently calculated [9].

The order of MDSs in the MIC genome provides evidence that multiple parallel transpositions occur, where the structure allows for interleaving between two sections that can be captured with a string operation called *shuffle*. The shuffle operation on two strings results in new strings by weaving together the first two, preserving the order within each string. For example, if $x = 2\ 4\ 5\ 6\ 7$ and $y = 1\ 3\ 8\ 9$ are two strings of numbers, then the shuffle of x and y is any permutation r of $1\ 2\ 3 \cdots 9$ where the order of the members of x and y is followed in r as well (for example $r = 2\ 4\ 1\ 3\ 5\ 6\ 7\ 8\ 9$). The sequences in Figure 3a can be rearranged computationally by shuffle between two segments, 1 3 5 7 10 12 and 2 4 6 8 9 11 13 14 15 16 17, as $1\ 2\ 3 \cdots 17$ is one of the results of the shuffle of the two segments. Figure 3b is even more complex, but the result can be obtained by splitting the whole sequence into two segments and applying shuffle once.

Shuffle is nondeterministic, and therefore multiple strings can be in the shuffle of two strings, however it is thought that structural components allow the developing MAC to align in a shuffle-like fashion, similar to the coiled and lampbrush patterns in Figure 2. Furthermore, the sheer number of genes that can be rearranged with very few applications — as seen in Sections 3 and 4 — yields evidence that this type of behaviour is occurring.

Predicting the order to descramble a gene or chromosomal segment, can be based on the principle of parsimony, whereby the smallest sequence of operations is likely close to the actual number of operations that occurred [15,7]. The genome rearrangement problem similarly uses the principle of parsimony

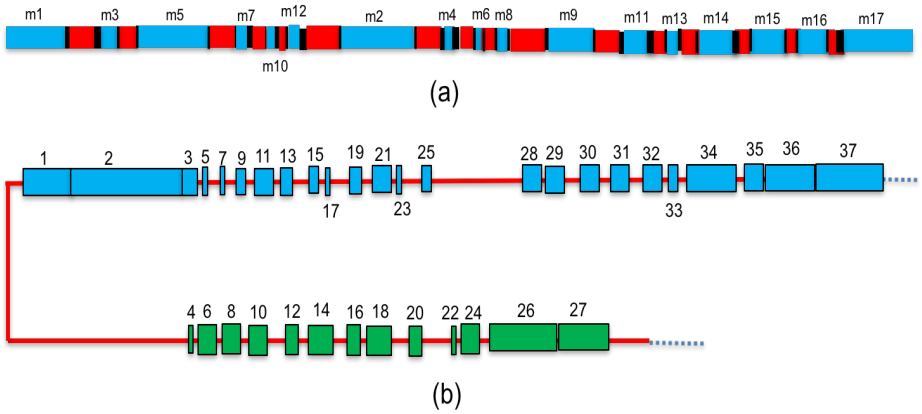

Fig. 3: (a) MDS organization found in the scrambled alpha-telomere-binding protein genes of *Oxytricha trifallax* [15]. (b) A schematic alignment of the micronuclear genes encoding the large catalytic subunit of DNA-polymerase α in *Oxytricha nova* [11] (inverted MDSs indicated by green MDSs).

for predicting genomic rearrangement operations [7]. This is now a well-studied problem, and indeed is quite similar to gene assembly [6]. Similarly, maximum parsimony is also an established method for phylogeny reconstruction.

This study aims to determine the order of parallel rearrangements by examining the number of applications of shuffle needed to assemble MIC genes.

2 Preliminaries

First, some notation that is used will be described.

An *alphabet* is a finite, non-empty set of symbols. Given an alphabet A, A^* is the set of all words over A, and A^+ is the set of all non-empty words over A. Let $\mathbb{N}$ be the natural numbers. Let $n \in \mathbb{N}$. Then $\mathbb{Z}_+(n) = \{1, 2, \ldots, n\}$ and $\mathbb{Z}_-(n) = \{-n, \ldots, -1\}$, and $\mathbb{Z}_{+-}(n) = \mathbb{Z}_+(n) \cup \mathbb{Z}_-(n)$ (0 is not in this set). For $i \in \mathbb{Z}_{+-}(n)$, let $sgn(i)$ be $+1$ if $i > 0$ and -1 otherwise. It is also common to examine sequences of numbers represented in the form of words with numbers for the alphabet. Therefore, $\mathbb{Z}_{+-}(n)^+$ is the set of all non-empty strings over the alphabet $\mathbb{Z}_{+-}(n)$. A string $\pi = \pi_1 \cdots \pi_n$, $\pi_1, \ldots, \pi_n \in \mathbb{Z}_{+-}(n)$ is *positive* if $\pi \in \mathbb{Z}_+(n)^+$, and *negative* if $\pi \in \mathbb{Z}_-(n)^+$. Also, π is *increasing* if $\pi_1 < \pi_2 < \cdots < \pi_n$, and is *decreasing* if $\pi_1 > \pi_2 > \cdots > \pi_n$ (here, $<$ and $>$ are the usual orderings of the integers). A *subword* of π is any word $\pi_i \pi_{i+1} \cdots \pi_j$, $1 \leq i \leq j \leq n$. The *inversion* of π, π^I, is the string obtained by reversing π and switching the sign of each number (this is the reverse complement). For $i \in \mathbb{Z}_{+-}(n)$, let $\Psi(\pi, i)$ be the number of i's in π. If $\pi \in \mathbb{Z}_+(n)^+ \cup \mathbb{Z}_-(n)^+$, then let $\overline{\pi}$ be equal to π if π is positive, and the inversion of π if π is negative (by reversing the numbers and making them all positive). Then $\overline{\pi}$ is always positive

for every such π. For $n \in \mathbb{N}$, let $id_n = 1\ 2 \cdots n$, which we call the *identity permutation*.

A sequence $\pi \in \mathbb{Z}_{+-}(n)^+$ is called a *permutation* if, for each $i \in \mathbb{Z}_+(n)$, $\Psi(\pi, i) + \Psi(\pi, -i) = 1$. π is a partial permutation if, for each $i \in \mathbb{Z}_+(n)$, $\Psi(\pi, i) + \Psi(\pi, -i) \leq 1$.

Let $u, v \in \mathbb{Z}_{+-}(n)^+$ be two sequences of integers. Then the *shuffle* of u and v, $u \sqcup\!\sqcup v$, is the set

$$\{x_1 y_1 x_2 y_2 \cdots x_r y_r \mid u = x_1 x_2 \cdots x_r, v = y_1 y_2 \cdots y_r,\ x_i, y_i \in \mathbb{Z}_{+-}(n)^*, 1 \leq i \leq r\}.$$

Given two words $u, v \in \mathbb{Z}_{+-}(n)^*$, u is a *subsequence* of v if $v = v_0 u_1 v_1 u_2 v_2 \cdots u_n v_n$, and $u = u_1 u_2 \cdots u_n$, where $u_i, v_i, v_0 \in \mathbb{Z}_{+-}(n)^*,\ 1 \leq i \leq n$. Notice that in the definitions of shuffle and subsequence, the variables $x_i,\ y_i,\ u_i$ refer to words of any length.

3 Data Preprocessing

The main purpose of this section is to preprocess the *Oxytricha trifallax* genome in order to obtain a sufficiently large data set for the analysis of parsimony. Although this does involve calculating some basic statistical properties of the genome, we refer to [3] for a more thorough investigation.

The data used was raw genome data from *Oxytricha trifallax* retrieved from NCBI on May 20, 2015 in the form of 22,363 MAC contigs and 25,720 MIC contigs (a contiguous sequence of DNA created by repeatedly assembling overlapping sequenced fragments of a chromosome). The procedure for determining the order of MDSs on the micronuclear chromosomes was chosen to be the same as Chen et al. [4] for the same purpose. The MAC contigs were aligned against the MIC contigs by using Nucleotide BLAST (parameters of [-ungapped -word_size 20 -outfmt 10]). For each MAC contig (almost all containing a single gene [4]), the MIC contig that matched with the lowest E-value (Expect value) was chosen and defined to be a *MAC/MIC sequence pair* (a MIC contig could then be matched with many MAC contigs). Of the 22,363 MAC contigs, only 9 of these sequences did not match with a MIC contig. Other MIC contigs that matched with lower scoring values were ignored as only the best matches were needed for the parsimony analysis.

Then, for each MAC contig, if n subwords matched a MIC contig, then a permutation of $1\ 2 \cdots n$ was determined giving the order of the matching segments on the matching MIC segment. The *MIC MDS sequence* of a MAC contig is the order of MDSs in the contig as determined by this procedure. Among the 22,354 matching sequences, the MIC MDS sequence of $18,315$ MAC contigs were unscrambled of the form $1\ 2 \cdots n$ for some n, or equivalently, $-n\ -(n-1) \cdots -1$ (here the '$-$' sign represents that the MDS is oriented in the opposite direction). These are called the *unscrambled sequences*. There are 4039 other MAC contigs called the *scrambled sequences*.

Every scrambled sequence was divided into two categories: one with MDSs only in one direction, called the *unidirectional sequences* (2443 total) and those

with MDSs in both direction, the *bidirectional sequences* (1596 total). Of the bidirectional sequences, each is divided into its two subsequences, one containing the non-inverted MDSs, and one containing the inverted subsequences. These were treated separately for the next part of the analysis and will be managed later. The set of unidirectional sequences produced is called the *extracted unidirectional sequences*. Then, the total number of scrambled sub-sequences for further analysis is 5635 (2443 scrambled unidirectional sequences, and 3192 scrambled extracted unidirectional sequences).

Next, the 5635 scrambled sequences were processed by collapsing all consecutive numbers, to one number (removing unused numbers). Henceforth, only these sequences will be used. This was done to investigate more about the scrambled patterns of the data. There is also some evidence that IESs between consecutive MDSs are removed first [15]. Of the 2443 unidirectional sequences after removing consecutive numbers from the unidirectional sequences, these are called *renumbered unidirectional sequences*, and of the 3192 extracted unidirectional sequences, after removing consecutive numbers, these are called *renumbered extracted unidirectional sequences*. Consecutive even-odd and odd-even patterns were common. Of the 2443 renumbered unidirectional sequences, there were 986 consecutive odd-even sequences, and, 985 consecutive even-odd sequences, and 472 others, called *complex scrambled sequences*. This is consistent with the analysis from [3]. From the 3192 renumbered extracted unidirectional sequences, there were 2443 unscrambled sequences, 280 consecutive odd-even sequences, 302 consecutive even-odd sequences, and 167 other complex scrambled sequences.

4 Parallel Descrambling Order Analysis

As discussed in Section 1, the patterns of the order of MDSs in micronuclear genes (Figure 3) shows evidence of some parallel operations that can be computationally described with shuffle. The order of MDSs of a MIC chromosome corresponding to a MAC chromosome is represented by a permutation $\pi = \pi_1\pi_2\pi_3\cdots\pi_n$ (as defined in Section 2). Informally, the descrambling order analysis problem is to determine the minimum number of "parallel steps" required to transform an input permutation π into the identity permutation. This attempts to predict the descrambling order at a higher level of abstraction suggested by the patterns occurring in ciliate MIC data, instead of changes at the molecular level. Thus, the operations do not represent any single molecular level biological operation (inversions, or loop deletion operations); instead, it represents a parallel operation. Because we do not know the exact mechanism by which descrambling takes place, we will study two similar systems involving shuffle to see how the minimum number of operations differs. If one system of moves gives significantly lower numbers of required moves, then there are advantages to this system in terms of parsimony.

As seen in Section 3, there are a number of sequences with consecutive odd-even (even-odd) patterns. When the odd and even numbers are in consecutive order, it only requires a single application of shuffle to transform the permutation

into the identity permutation. For example, the permutation of alpha-telomere-binding protein genes of *Sterkiella histriomuscorum* is 1 3 5 7 2 4 6 (Figure 3), which can be descrambled in one step by taking the shuffle of two subwords 2 4 6 with 1 3 5 7. In that case, there is a possibility that recombinations take place in parallel (or without significantly changing the structure between individual recombination) to descramble the MIC chromosome, and therefore a structural component is partially enforcing an alignment of appropriate MDSs so that the operation is applied correctly. Note that the this operation applies shuffle to segments of the same input string rather than on two separate strings. The two systems will be described next.

- **Contiguous Increasing System (CIS):** Given an input permutation $\pi = \pi_1 \cdots \pi_n$, $\pi_i \in \mathbb{Z}_{+-}(n)$, $1 \leq i \leq n$, a CIS partition of π is a set of subwords $\{u_1, \ldots, u_m\}$ of π, with each $u_i \in \mathbb{Z}_+(n)^+ \cup \mathbb{Z}_-(n)^+$, with u_i increasing for each i, $1 \leq i \leq m$, such that $\pi = u_1 u_2 \cdots u_m$.
- **Non-Contiguous Increasing System (NIS):** For a given input permutation $\pi = \pi_1 \cdots \pi_n$, $\pi_i \in \mathbb{Z}_{+-}(n)$, $1 \leq i \leq n$, a NIS partition of π is a set of subsequences $\{u_1, \ldots, u_m\}$, with each $u_i \in \mathbb{Z}_+(n)^+ \cup \mathbb{Z}_-(n)^+$, and u_i is increasing for each i, $1 \leq i \leq m$, such that $\pi \in u_1 \sqcup u_2 \sqcup \cdots \sqcup u_m$.

Notice that for every CIS partition $\{u_1, \ldots, u_m\}$, the identity permutation is in $\overline{u_1} \sqcup \cdots \sqcup \overline{u_m}$. This system allows the shuffle on increasing subwords (each either positive or negative). The smallest number of increasing subwords of the input permutation such that the input permutation is the concatenation of the subwords is desired. In such a case, the identity is in the shuffle of the subwords after taking the inversion of any negative subwords. For example, the permutation 5 6 2 −8 −7 −4 −3 −1 can be split into three increasing subwords, $u = 5\ 6$, $v = 2$, $w = -8\ -7\ -4\ -3\ -1$, the input is indeed uvw, and the identity is in $\overline{u} \sqcup \overline{v} \sqcup \overline{w}$. For every NIS partition, the input permutation is in the shuffle of the segments, and the identity permutation is in $\overline{u_1} \sqcup \cdots \sqcup \overline{u_m}$. In this case, both the input permutation, and the identity permutation are strings of numbers derived from shuffle. For example, the permutation 5 6 2 1 3 4 7 8 can be split into three increasing subsequences, $u = 5\ 6\ 7\ 8$, $v = 2$, and $w = 1\ 3\ 4$, with the input permutation in $u \sqcup v \sqcup w$, and the identity in $\overline{u} \sqcup \overline{v} \sqcup \overline{w}$. For both of the systems, we are interested in calculating the number of segments, which corresponds to the number of shuffle applications plus one. A CIS partition of π can be thought of as a parallel recombination of different subwords, where the inversion of a subword can occur before a parallel recombination. The NIS system is intended as an investigation as to whether the number of operations can be reduced by adding an addition layer of the shuffle operation.

Proposition 1. *Given an input permutation π of n elements. Let m be the smallest such that there exists $u_1, \ldots, u_m$ with $\pi = u_1 \cdots u_m$ and $id_n \in \overline{u_1} \sqcup \cdots \sqcup \overline{u_m}$. Then $m - 1$ is the size of a smallest CIS partition.*

Proof. Given any two positive increasing words u, v which use disjoint numbers, then there is a positive increasing word in $u \sqcup v$. Similarly, if both are negative

and increasing then there is a positive increasing word in $\overline{u} \sqcup\!\sqcup \overline{v}$. If u is positive and v is negative, then there is a positive increasing sequence in $\overline{u}$ shuffled with $\overline{v}$. And in general, given $\pi = \pi_1 \cdots \pi_n$ and a CIS partition $u_1, \ldots, u_m$, then the identity must be in $\overline{u_1} \sqcup\!\sqcup \cdots \sqcup\!\sqcup \overline{u_m}$. That is, if there is a partition of size m, then the identity can be obtained with $m-1$ shuffle operations.

Conversely, if $\pi = u_1 \cdots u_m$, and the identity is in $\overline{u_1} \sqcup\!\sqcup \cdots \sqcup\!\sqcup \overline{u_m}$, the each u_i, $1 \leq i \leq m$ is either positive increasing or negative increasing. Hence, counting the number of segments in a CIS partition is always exactly one more than counting the number of applications of shuffle. □

4.1 Contiguous Increasing System

Next, an algorithm to determine the minimum sized CIS partition will be given. In an input permutation $\pi = \pi_1 \pi_2 \cdots \pi_n$, a pair of adjacent elements π_i and π_{i+1}, $1 \leq i \leq n-1$, are called *neighbours* if $\pi_i < \pi_{i+1}$ and either π, π_{i+1} are both positive or both negative; otherwise the pair is called a *cut-off point*. Then $c(\pi)$ is the number of cut-off points in π. If π has an increasing positive or negative subword $\pi_i \cdots \pi_j$, ie. $\pi_i < \cdots < \pi_j$ all the same sign, then each adjacent pair in $\pi_i, \pi_{i+1}, \ldots, \pi_j$ are neighbours. Thus, in an increasing positive or negative permutation $\pi = \pi_1 < \pi_2 < \pi_3 < \cdots < \pi_n$, $c(\pi) = 0$. In contrast, if π is a positive or negative decreasing permutation, then $c(\pi) = n-1$, because $\pi = \pi_1 > \pi_2 > \pi_3 > \cdots > \pi_n$. The size of the smallest CIS partition depends on the number of cut-off points.

Proposition 2. *Let π be an input permutation. The size of the smallest CIS partition is $c(\pi) + 1$.*

Proof. Let $\pi = \pi_1 \pi_2 \pi_3 \cdots \pi_n$ have $c(\pi)$ cut-off points after positions (in order) $c_1, \ldots, c_m$ of π. Then, it is impossible to have an increasing segment that includes a number from both before and after a cut-off point, as the identity must be in the shuffle of the potentially inverted segments. Therefore, any CIS partition has at most $m+1$ elements in it. Furthermore, there exists a CIS partition with $m+1$ elements in it, because there is one increasing positive or negative subword that has the elements in between π_1 and the element at position c_1, and an increasing positive or negative subword between positions $c_i + 1$ and c_{i+1} for each i, $1 \leq i < m$, and one last increasing positive or negative subword that has the elements starting from $c_m + 1$ to π_n. As $m = c(\pi)$, the smallest number of increasing subwords in π will always be $c(\pi) - 1 + 2 = c(\pi) + 1$. □

Therefore, we constructed an optimal algorithm called *IncreasingSubwords* (Algorithm 1) that determines the minimum size of a CIS partition of an input permutation π by simply counting the number of cut-off points $c(\pi)$ for each input permutation π which can be done in linear time (the algorithm is optimal in the sense that it finds the smallest segments). For example, 1 3 5 7 9 2 4 6 8 10 has one cut-off point between 9 and 2, and two increasing subwords: 1 3 5 7 9, and 2 4 6 8 10. The sequences having consecutive odd-even patterns (or even-odd patterns) will always have two increasing subwords — one with the consecutive

odd numbers, and the other with the even numbers, as these sequences will always have a single cut-off point.

Algorithm 1 Minimum number of segments in CIS.

```
1: procedure IncreasingSubwords(π = π_1 ··· π_n)
2:     segments ← 0;
3:     cut_off_points ← 0;
4:     for i ← 1 to n − 1 do
5:         if π_i > π_{i+1} or sgn(π_i) ≠ sgn(π_{i+1}) then
6:             cut_off_points ← cut_off_points + 1;
7:         end if
8:     end for
9:     segments ← cut_off_points + 1;
10:    return segments;
11: end procedure
```

The graph in Figure 4 shows the number of segments versus the number of sequences achieving that minimal number of segments from the preprocessed *Oxytricha* data. The sequences that have only 1 increasing subword are already unscrambled, and only the sequences having the consecutive odds and evens will have 2 increasing subwords. In the graph, the rest of the sequences have at least 3 increasing subwords. The following sequence 1 4 6 8 10 12 2 5 7 11 13 3 9 requires at least 3 increasing subwords, as the identity is in 1 4 6 8 10 12⧢2 5 7 11 13⧢3 9.

Table 1 shows the average number of increasing subwords determined by Algorithm 1, along with the maximum number of increasing subwords, the number of sequences in each sequence pattern, the average length of increasing subwords, and the maximum length of increasing subwords.

4.2 Non-Contiguous Increasing System

This system allows shuffle on increasing subsequences (i.e. non-contiguous) instead of increasing subwords only. As discussed above, if $\pi = \pi_1 \cdots \pi_n$ is a given input permutation, then a NIS partition of π is a set of increasing subsequences $\{s_1, \ldots, s_m\}$, such that $\pi \in s_1 ⧢ s_2 ⧢ \cdots ⧢ s_m$, and the identity is in $\overline{s_1} ⧢ \cdots ⧢ \overline{s_m}$. For example, consider π = 6 1 7 2 8 3 4 9 10 11 5. Then both the input permutation and the identity permutation is in 6 8 9 10 11 ⧢ 1 2 3 4 5. Next, an optimal algorithm is described for determining the segments within this system. At first, it adds the first element of π to an increasing subsequence s. Then it finds the next larger element to the right, adds it to increasing subsequence s, and continues doing this until reaching the end of π. Then s becomes the first increasing subsequence. Next, the algorithm deletes the elements of s from π, and repeats until π becomes empty. The final number of increasing subsequences is the size of the smallest NIS partition.

We will prove that Algorithm 2 calculates an optimal NIS partition, but first, an intermediate lemma is needed.

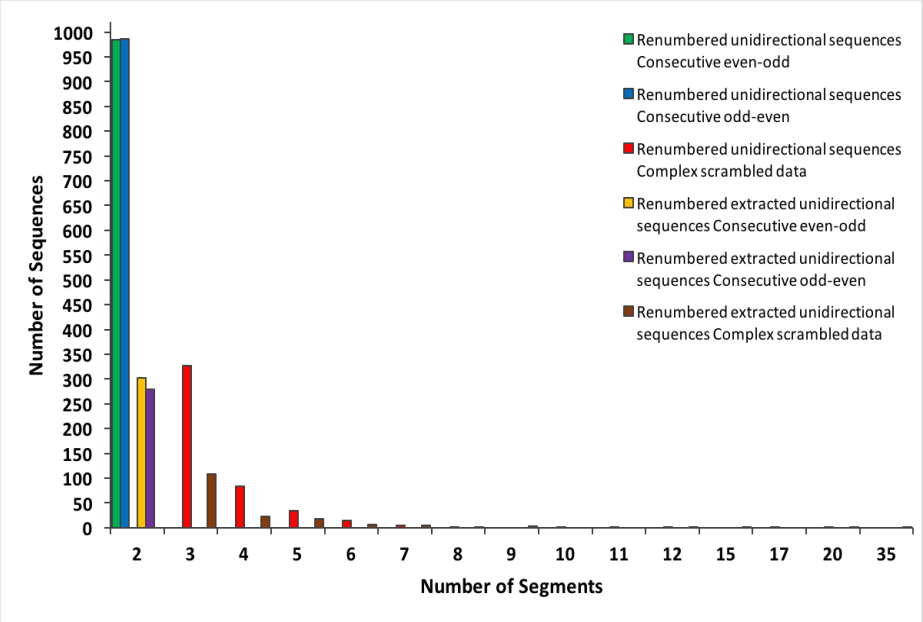

Fig. 4: Relationship between the number of segments (increasing subwords) and the number of sequences in the dataset achieving that for renumbered unidirectional sequences and renumbered extracted unidirectional sequences.

Lemma 1. *Let π be a positive or negative input permutation. If π has a decreasing subsequence of length m, then every NIS partition of π has at least m elements.*

Proof. Let $\pi = \pi_1 \cdots \pi_n$ be an input permutation, and assume that there exists $i_1, \ldots, i_m$ such that $1 \leq i_1 < \cdots < i_m \leq n$, but $\pi_{i_1} > \pi_{i_2} > \cdots > \pi_{i_m}$. Assume, by contradiction that there exists some NIS partition X with $k < m$ elements. Then there has to be two of π_{i_α} and π_{i_β}, $\alpha < \beta$ such that π_{i_α} and π_{i_β} are in the same sequence s of X. But, then the identity cannot be in the shuffle of $\overline{s}$ with other elements. □

Proposition 3. *Let π be a positive or negative input permutation. Then Algorithm 2 calculates the minimum size of an NIS partition.*

Proof. Let $\pi = \pi_1 \cdots \pi_n$ be an input permutation, and assume first that π is positive (similarly if π is negative). Let $X = \{s_1, \ldots, s_m\}$ be the output from Algorithm 2, such that $s_1, \ldots, s_m$ is the order as determined by the algorithm.

Let k be the size of the smallest NIS partition. It is clear then that $k \leq m$. Let i satisfy $1 \leq i \leq n$. Let $f(i)$ be the number at position i of π, and also for $1 \leq j \leq m$, let $g_i(j)$ be the largest position x of π such that $x < i$ and $\pi(x)$ is in s_j. By the algorithm, $s_2(1)$, at position i_2 say of π, must be smaller than the number at

Table 1: Increasing subword statistics with Contiguous Increasing System for renumbered unidirectional sequences (and renumbered extracted unidirectional sequences in parentheses).

Sequence patterns	**No. of Sequences**	**Avg. number of increasing subwords**	**Max. number of increasing subwords**	**Avg. length of increasing subwords**	**Max. length of increasing subwords**
Consecutive odd-even patterns	986 (280)	2 (2)	2 (2)	2.300 (2.323)	36 (20)
Consecutive even-odd patterns	985 (302)	2 (2)	2 (2)	2.305 (2.488)	37 (25)
Complex scrambled patterns	472 (167)	3.619 (4.126)	20 (25)	2.741 (3.452)	43 (86)

position $g_{i_1}(1)$ of π (a letter of s_1). More generally, let i_m be the position of $s_m(1)$ in π. Then notice that $f(g_{i_m}(1)) > f(g_{i_m}(2)) > \cdots > f(g_{i_m}(m-1)) > s_m(1)$, otherwise, the smallest α such that $f(g_{i_m}(\alpha)) < f(g_{i_m}(\alpha{+}1))$ would have caused Algorithm 2 to include $f(g_{i_m}(\alpha{+}1))$ in s_α. Thus, π has a decreasing subsequence of length m, and therefore by the lemma above, m is the smallest size of an NIS partition, and $k = m$. Hence, Algorithm 2 is optimal. □

Also notice that even though Algorithm 2 is not linear time, there is a linear time variation whereby, whenever it is determined that the current sequence s_1 should not contain the next element, in line 7, it starts a new sequence s_2, and then preferably adds new elements to s_1, and if not s_2, and if not start s_3, etc. Therefore, an optimal linear time algorithm exists to solve this problem.

Table 2 shows the average number of increasing subsequences determined by Algorithm 2, along with other properties. The graph in Figure 5 shows the number of segments (minimum size of NIS partitioning) versus the number of sequences achieving that number of segments. Most of the complex scrambled sequences were partitioned into only 3 increasing subsequences by this algorithm, and the maximum number of increasing subsequences is only 7.

5 Result Analysis and Discussion

NIS often gives a much smaller number of applications of shuffle versus CIS. The largest of the minimum CIS partition sizes for CIS is 20 and 35, for renumbered unidirectional and extracted unidirectional sequences respectively, whereas for NIS, this number is 6 and 7, respectively (Tables 1 and 2).

There are 986 consecutive odd-even and 985 consecutive even-odd sequences in the renumbered unidirectional dataset, and 280 consecutive odd-even and 302 consecutive even-odd sequences in the renumbered extracted unidirectional

Algorithm 2

```
1: procedure NextIncreasingElement(π = π_1 ··· π_n)
2:     segments ← 0;
3:     while π is not empty do
4:         element ← π_1;
5:         append element in subsequence s;
6:         for i ← 2 to n do
7:             if element < π_i then
8:                 element ← π_i;
9:                 append element in subsequence s;
10:            end if
11:        end for
12:        if s is not empty then
13:            segments ← segments + 1;
14:            delete the elements of s from π;
15:        end if
16:        Clear s;
17:    end while
18:    return segments;
19: end procedure
```

dataset. Thus, there are a total of 1266 consecutive odd-even, and 1287 consecutive even-odd sequences among the 3192 scrambled input sequences. Algorithm 2 partitioned all of these 2553 consecutive odd/even sequences into 2 increasing segments, which is optimal. There are 472 categorized as complex scrambled sequences in the renumbered unidirectional dataset, and 167 complex scrambled sequences in the renumbered extracted unidirectional dataset. Thus, there are a total of 639 complex scrambled sequences among the 3192 scrambled input sequences. Algorithm 2 partitioned 136 sequences of these, into size 2, and 405 sequences into 3 segments. There are 98 sequences that have an NIS partition size between 4 and 7, and there is none higher than 7.

Each parallel step, represented by an application of shuffle, can descramble a section of MDSs that might or might not reside beside each other. As the chromosomes fold mostly in a coiled and lampbrush structures, such an alignment of the non-contiguous MDSs subsequently via structural component, might be practical. Hence, the NIS has potential advantages in terms of parsimony. However, the feasibility of any such hypothesis needs experimental validation.

Recall that bidirectional sequences were separated into two sub-subsequences: one holding non-inverted MDSs, and one holding inverted MDSs. The NIS system determines the minimum number of applications of shuffle to descramble its two subsequences, separately. To combine the two, it requires exactly one extra application of shuffle, as the identity permutation of the bidirectional sequences is in the shuffle of its two descrambled subsequences. However, it could be more for CIS. For example, $1\ -6\ 2\ -5\ 3\ -4$ has 5 cut-off points, because an individual segment cannot contain both positive and negative numbers, and the minimum

Table 2: Increasing subsequence statistics with Non-Contiguous Increasing System for renumbered unidirectional sequences (and renumbered extracted unidirectional sequences in parentheses).

Sequence patterns	**No. of Sequences**	**Avg. number of increasing subsequences**	**Max. number of increasing subsequences**	**Avg. length of increasing subsequences**	**Max. length of increasing subsequences**
Consecutive odd-even patterns	986 (280)	2 (2)	2 (2)	2.301 (2.323)	36 (20)
Consecutive even-odd patterns	985 (302)	2 (2)	2 (2)	2.306 (2.488)	38 (26)
Complex scrambled patterns	472 (167)	2.915 (3.221)	6 (7)	3.403 (4.421)	43 (87)

CIS partition is 6. But if positive and negative parts are separated, then it is 1 2 3 and $-6\ -5\ -4$, each having no cut-off points.

The analysis shows that 96.63% of the scrambled MIC chromosome fragments of *Oxytricha trifallax* can be partitioned into 2 to 3 segments, and therefore can be descrambled by only 1 or 2 applications of shuffle, where each application of shuffle corresponds to a parallel recombination. This small number lends theoretical evidence that some structural component is enforcing the shuffle-like behaviour, by properly aligning segments in an interleaving fashion, and then parallel recombination is taking place for MDS rearrangement. The sheer number of MIC chromosome fragments that can be rearranged with very few applications of shuffle yields evidence that this type of behaviour could be occurring, using parsimony. Indeed, the number of applications of shuffle is far lower than the number of MDSs, and therefore the principle of parsimony dictates that there is significant computational advantages to such a system, as the arrangement of the large number of MDSs do not need a large number of parallel steps to descramble.

References

1. Angeleska, A., Jonoska, N., Saito, M.: DNA recombination through assembly graphs. Discrete Applied Mathematics 157(14), 3020–3037 (2009)
2. Angeleska, A., Jonoska, N., Saito, M.: Rewriting rule chains modeling DNA rearrangement pathways. Theoretical Computer Science 454, 5–22 (2012)
3. Burns, J., Kukushkin, D., Chen, X., Landweber, L.F., Saito, M., Jonoska, N.: Recurring patterns among scrambled genes in the encrypted genome of the ciliate *Oxytricha trifallax*. Journal of Theoretical Biology 410, 171–180 (2016)

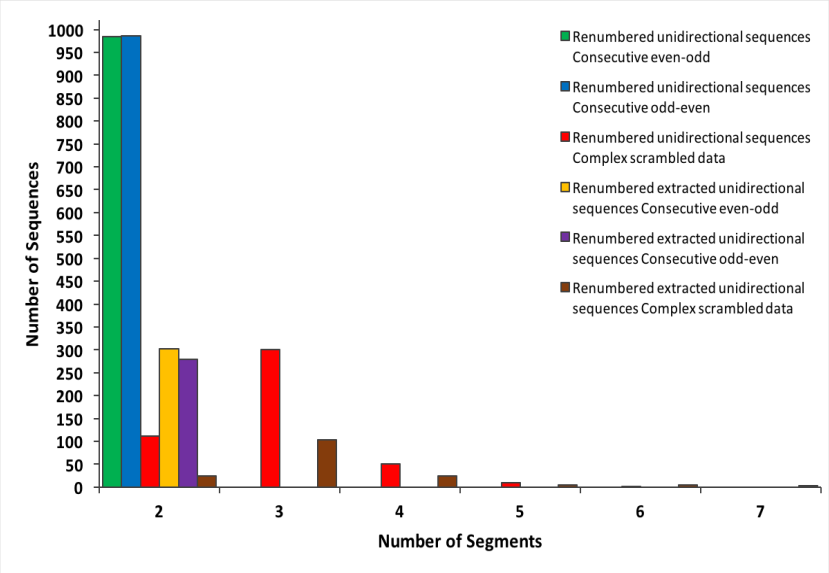

Fig. 5: Relationship between the number of segments (increasing subsequences) and the number of sequences in the dataset achieving that for renumbered unidirectional sequences and renumbered extracted unidirectional sequences.

4. Chen, X., Bracht, J.R., Goldman, A.D., Dolzhenko, E., Clay, D.M., Swart, E.C., Perlman, D.H., Doak, T.G., Stuart, A., Amemiya, C.T., Sebra, R.P., Landweber, L.F.: The architecture of a scrambled genome reveals massive levels of genomic rearrangement during development. Cell 158(5), 1187–1198 (2014)
5. Ehrenfeucht, A., Prescott, D.M., Rozenberg, G.: Computational aspects of gene (un)scrambling in ciliates. In: Evolution as Computation, pp. 216–256. Springer (2002)
6. Herlin, J.L., Nelson, A., Scheepers, M.: Using ciliate operations to construct chromosome phylogenies. Involve, a Journal of Mathematics 9(1), 1–26 (2015)
7. Jones, N.C., Pevzner, P.: An Introduction to Bioinformatics Algorithms. MIT press, Cambridge, MA (2004)
8. Keil, J.M., Liu, J., McQuillan, I.: Algorithmic properties of ciliate sequence alignment. Theoretical Computer Science 411(6), 919–925 (2010)
9. Khan, N.A.: Chromosome Descrambling Order Analysis in Ciliates. Master's thesis, University of Saskatchewan, Saskatoon (2016)
10. Landweber, L.F., Kari, L.: The evolution of cellular computing: Nature's solution to a computational problem. Biosystems 52(1), 3–13 (1999)
11. Landweber, L.F., Kuo, T.C., Curtis, E.A.: Evolution and assembly of an extremely scrambled gene. PNAS 97(7), 3298–3303 (2000)
12. Meyer, G., Lipps, H.: Chromatin Elimination in the Hypotrichous Ciliate *Stylonychia mytilus*. Chromosoma 77(3), 285–297 (1980)
13. Meyer, G., Lipps, H.: The Formation of Polytene Chromosomes during Macronuclear Development of the Hypotrichous Ciliate *Stylonychia mytilus*. Chromosoma 82(2), 309–314 (1981)

14. Meyer, G., Lipps, H.: Electron microscopy of surface spread polytene chromosomes of *Drosophila* and *Stylonychia*. Chromosoma 89(2), 107–110 (1984)
15. Möllenbeck, M., Zhou, Y., Cavalcanti, A.R., Jönsson, F., Higgins, B.P., Chang, W.J., Juranek, S., Doak, T.G., Rozenberg, G., Lipps, H.J., et al.: The pathway to detangle a scrambled gene. PLoS One 3(6), e2330 (2008)
16. Prescott, D.M., Ehrenfeucht, A., Rozenberg, G.: Molecular operations for DNA processing in hypotrichous ciliates. European Journal of Protistology 37(3), 241–260 (2001)
17. Prescott, D.M., Rozenberg, G.: How ciliates manipulate their own DNA—a splendid example of natural computing. Natural Computing 1(2-3), 165–183 (2002)