Simple gene assembly is deterministic

Miika Langille
Turku Centre for Computer Science
Department of Computer Science, Åbo Akademi University
Turku 20520 Finland
miika.langille@abo.fi

Ion Petre
Academy of Finland and
Turku Centre for Computer Science
Department of Computer Science, Åbo Akademi University
Turku 20520 Finland
ion.petre@abo.fi

Abstract
We investigate in this paper a simple intramolecular model for gene assembly in ciliates. Unlike the general intramolecular model, the folds that a micronuclear chromosome may form to assemble the genes is very restricted (minimal) here: between any two pointers there may be at most one coding block. It has been known that the general model is universal, being able to assemble any gene pattern (to sort any signed permutation). The simple model on the other hand is not universal: there exist signed permutations that cannot be sorted in this model. Remarkably though, all known micronuclear gene patterns in ciliates can indeed be assembled in the simple model. We prove in this paper that while the general model is non-deterministic, the simple model is “weakly deterministic”: any gene pattern either has only successful or only unsuccessful sorting strategies. Moreover, although different strategies may lead to different permutations for the same input, these final results have the same structure.

1 Introduction
Ciliates form a rich group of unicellular eukaryotes, estimated to be some 2.5 billion years old. A unique feature of the ciliates is that they have two functionally different types of nuclei, *macronucleus* and *micronucleus*, each present in multiple (up to several hundred in some species) copies in the same cell.

The macronucleus is the somatic nucleus where all RNA transcripts are produced. Macronucleus genes are short contiguous sequences of nucleotides, each gene being placed on its own DNA molecule (with only a couple of known exceptions). Incidentally, these are the shortest DNA molecules known in Nature, even shorter than those of viruses, explaining in part the remarkable ability of ciliates to adapt quickly to hostile environments.
The micronucleus is the germline nucleus that is dormant (no RNA transcripts are produced here) throughout most of the life cycle, becoming active only at some stage in sexual reproduction. When mating, two ciliates exchange a haploid micronucleus that fuses with a local one to form a new diploid micronucleus. After multiplying it a few times, ciliates transform one such micronucleus into a new macronucleus, while destroying all old micronuclei and macronuclei. The significance of this transformation, especially from the point of view of computer science, lies in the drastically different structure of micronuclear genes with respect to their macronuclear forms. Thus, micronuclear genes are placed on long chromosomes, separated by long stretches of non-coding sequences. Moreover, the micronuclear genes are split into blocks called MDSs (macronuclear-destined sequences) separated by non-coding blocks called IESs (internally-eliminated sequences). Ciliates have to assemble all MDSs and excise the IESs to produce the transcriptionally active macronuclear gene, in a process called gene assembly. The intricacy of this process is best seen in a species of ciliates called Stichotrichs that we consider in this paper. In Stichotrichs the MDSs are often shuffled and some of them are even inverted! With up to 30,000 MDSs to place in the orthodox form and an equal number of IESs to excise precisely, gene assembly is often called the most involved process of DNA manipulation in living cells that we know of today.

A hint on how the ciliates accomplish gene assembly is given by the structure of MDSs. Thus, each MDS \( M \) ends with a specific sequence of nucleotides that is repeated in the beginning of the MDS that should follow \( M \) in the macronuclear gene. These sequences are called pointers and make the structure of micronuclear genes an astonishing implementation of the linked list data structure in computer science. The first MDS of the macronuclear gene begins with a specific sequence of nucleotides that we call the beginning marker, while the last MDS ends with a specific sequence that we call the ending marker.

We consider in this paper the intramolecular model for gene assembly introduced in [6], [21], see also [3] for a recent monograph. A different intermolecular model was introduced earlier in [12], [13], see also [9] for a comparison of the two models. Three molecular operations were conjectured in the intramolecular model: ld, hi and dlad. In all of them, the chromosome is conjectured to make a specific fold on itself so that one (two in the case of dlad) pair of pointers align and through recombination, two or more MDSs get spliced together into a bigger coding block, see Figure 1.

The intramolecular model was formalised on three levels of abstraction: signed permutations (denoting the sequence and orientation of the MDSs), signed strings (denoting the sequence and orientation of the pointers) and signed graphs (denoting the overlap of pointers and their orientation), see [3]. It is convenient for the purpose of this paper to work within the framework of signed permutations.

It was proved in [4], [2] that the intramolecular model is universal and non-deterministic: any gene pattern (any sequence of MDSs) may be assembled (sorted) according to this model and in general, several assembly strategies exist for a given pattern. However, in its most general form, the model allows for two pointers to be aligned and participate in an operation, regardless of the number of nucleotides separating them on the chromosome. The following question is then natural: what is the power of a simplified intramolecular model, where only at most one MDS separates two pointers participating in an
operation? The simple model for gene assembly was formalised in [10] in terms of signed permutations and signed strings. In this model, the pointers participating in an operation are indeed separated by at most one MDS, be that a micronuclear MDS, or a composite MDS obtained by the previous assembly of several micronuclear MDSs. Investigations of this model were done in [7] but only in its most elementary form, where the MDS separating two pointers of any operation is always micronuclear (no composite MDS may be involved in an operation). It turns out that this elementary model is still non-deterministic in a very strong way: not only that a given gene pattern may have several sorting strategies, but it may also have strategies that lead the pattern to a form that cannot be sorted anymore. Since gene assembly is an irreversible process, that means that ciliates, should they use this model, would have to be careful which operations to select at any stage of the process, out of a range of possibilities. However, it turns out that ciliates do not use this model: e.g. the actin I gene in S. nova, see Figure 2, cannot be assembled in the elementary model.

In this paper we show that the situation is drastically different in the unrestricted simple model for gene assembly, when the MDS separating two pointers may also be composite. We prove that the model is in fact deterministic in the following “weak” sense: no gene pattern may have both successful and unsuccessful strategies. Moreover, even though a pattern may indeed have several strategies (either all successful, or all unsuccessful), all of them lead to molecules having essentially the same “structure”, where structure is defined in details in Section 3.1. Regarding the assembly power of the simple model, it was shown already in [10] and [4] that the model is not universal: certain sequences of MDSs cannot be assembled in this model. However, as reported in [10] all known micronuclear gene patterns, see [1] for a database of known patterns, may be assembled according to the simple model for gene assembly.

2 Notations

For a finite alphabet $A = \{a_1, \ldots, a_n\}$, we denote by $A^*$ the free monoid generated by $A$ and call any element of $A^*$ a word. Let $\overline{A} = \{\overline{a_1}, \ldots, \overline{a_n}\}$, where $A \cap \overline{A} = \emptyset$. For $p, q \in A \cup \overline{A}$, we say that $p, q$ have the same signature if either $p, q \in A$, or $p, q \in \overline{A}$ and we say that they have different signatures otherwise.

We denote $A^\bullet = (A \cup \overline{A})^\ast$. For any $u \in A^\bullet$, $u = x_1 \ldots x_k$, with $x_i \in A \cup \overline{A}$, for all $1 \leq i \leq k$, we denote $\|u\| = \|x_1\| \ldots \|x_k\|$, where $\|a\| = \|\overline{a}\| = a$, for all $a \in A$. We also denote $\overline{\pi} = \overline{x}_k \ldots \overline{x}_1$, where $\overline{x} = a$, for all $a \in A$. For two alphabets $A, B$, a mapping $f: A^\bullet \to B^\bullet$ is called a morphism if $f(uv) = f(u)f(v)$ and $f(\overline{u}) = f(u)$.

A permutation $\pi$ over $A$ is a bijection $\pi: A \to A$. Fixing the order relation $(a_1, a_2, \ldots, a_m)$ over $A$, we often denote $\pi$ as the word $\pi(a_1) \ldots \pi(a_m) \in A^\ast$. A signed permutation over $A$ is a string $\psi \in A^\bullet$, where $\|\psi\|$ is a permutation over $A$.

3 The intramolecular model

We describe in this section how we represent genes as signed permutations. We then introduce the intramolecular model for gene assembly in its general and
simple forms.

3.1 Gene structure

For the purpose of this paper, it is enough to represent a gene through the sequence of its MDSs. We will use the alphabet $M_n = \{M_1, M_2, \ldots, M_n\}$ to denote the MDSs, where the numbering is given by the order in which MDSs are assembled in the macronuclear gene, and $n$ is an arbitrary but fixed positive integer. Thus, a micronuclear gene will be a signed permutation over $M_n$ and a macronuclear gene will be a sorted signed permutation over $M_n$.

**Definition 1.** We say that a signed permutation $\mu$ is *sorted* if either:

i. $\mu = M_i M_{i+1} \ldots M_n M_1 \ldots M_{i-1}$, or

ii. $\mu = \overline{M_{i-1}} \ldots \overline{M_1} M_n \ldots \overline{M_{i+1}} \overline{M_i}$,

for some $1 \leq i \leq n$. If $i = 1$ we say that $\mu$ is a *linear* permutation. We call $\mu$ *circular* otherwise. In case i. we say that $\mu$ is sorted in the *orthodox order*, while in case ii. we say that $\mu$ is sorted in the *inverted order*.

The term circular in the above definition refers to a gene that gets assembled, say in the form $M_i \ldots M_n M_1 \ldots M_{i-1}$, and then gets excised from the chromosome by an $ld$ operation applied on the pointer in the beginning of $M_i$ and its identical copy at the end of $M_{i-1}$.

3.2 Molecular operations

The intramolecular model consists of three operations: $ld$, $hi$ and $dlad$. In each of them, the chromosome is folded in such a way that specific folds are formed and recombination takes place as shown in Figure 1. In all of these folds, the alignment and subsequent recombination is done on pointers. For a detailed presentation of these operations we refer to [3] and [8].

![Image of molecular operations](image_url)

Figure 1: Illustration of the $ld$, $hi$, $dlad$ molecular operation showing in each case: (i) the folding, (ii) the recombination, and (iii) the result. Courtesy of T.Harju.
3.3 Simple operations

The ld operation excises a circular molecule, see Figure 1. Since our model is intramolecular, that excised circular molecule cannot recombine later with the remaining linear molecule. For that reason, the ld operation must always be simple (even in the general model): the fold may only contain IESs. Equivalently, between the two pointers aligning the fold cannot be any MDSs. The only exception is when ld is used as a post-assembly operation to excise from the chromosome the gene assembled in the order \( (M_i \ldots M_n) \) or \( (M_{i-1} \ldots M_1 M_{n} \ldots M_i) \).

![Figure 2: Structure of the micronuclear gene encoding actin protein in Sterkiella nova.](image)

In the case of hi and dlad, the folds and thus, the rearranged sequences, may be arbitrarily large. E.g., in actin I, see Figure 2, the pointer in the beginning of MDS \( M_3 \) has a second (inverted) occurrence at the end of the MDS \( M_2 \) and thus, may be used to align a hairpin consisting of MDSs \( M_3, M_4, M_6, M_5, M_7 \) and \( M_9 \) to facilitate a hi operation. Similarly, in case of the MDS sequence \( M_2 M_8 M_6 M_5 M_1 M_7 M_3 M_{10} M_9 M_4 \), a double loop could be folded as in Figure 1, aligned by the pointer ending \( M_2 \) and beginning \( M_3 \) and by the pointer beginning \( M_5 \) and ending \( M_4 \), to facilitate a dlad operation. In this case, the rearranged sequences would be \( M_8 M_6 \) and \( M_3 M_{10} M_9 M_4 \). We define the simple hi and simple dlad in the following.

An application of the hi operation on a pointer \( p \) is called simple if the sequence separating the two occurrences of \( p \) consists of exactly one IES and one MDS. The two possible patterns are shown in Figure 3, see [10] for more details.

An application of the dlad operation on pointers \( p \) and \( q \) is called simple if the sequence separating the first occurrence of \( p \) and \( q \) and the sequence separating their second occurrences consist of either one MDS, or of one IES. The two possible patterns are shown in Figure 4, see [10] for more details.

**Example 1.** Consider the MDS sequence \( M_3 M_4 M_6 M_5 M_7 M_9 M_8 \) describing the actin I gene in S. nova, see Figure 2.

i. If \( p_3 \) is the pointer beginning \( M_3 \) and ending \( M_2 \), then the hi operation directed by \( p_3 \) is not simple. Indeed, the two occurrences of \( p_3 \) are separated by 6 MDSs: \( M_3, M_4, M_6, M_5, M_7, M_9 \).

ii. Denote by \( p_6 \) the pointer beginning \( M_6 \) ending \( M_5 \) and by \( p_7 \) the pointer beginning \( M_7 \) and ending \( M_6 \). Then the dlad operation directed by \( p_6 \) and \( p_7 \) is simple: the first occurrences of \( p_6 \) and \( p_7 \) are separated by \( M_6 \), while their second occurrences are separated by the IES in-between \( M_5 \) and \( M_7 \).
Figure 3: The MDS/IES structures where the simple hi-rule is applicable. An MDS is shown here as a black rectangle, while a straight line denotes one IES. A zig-zag line denotes an arbitrarily long sequence of MDSs and IESs. Courtesy of T.Harju.

Figure 4: The MDS/IES structures where simple dlad-rules is applicable. An MDS is shown here as a black rectangle, while a straight line denotes one IES. A zig-zag line denotes an arbitrarily long sequence of MDSs and IESs. Courtesy of T.Harju.

4 Gene assembly as a sorting of permutations

We formalize in this section the simple operations as operations on signed permutations over the alphabet $M_n$. Representing the genes by their sequence of MDSs only and denoting their orientations we obtain signed permutations over $M_n$. Since the assembled gene is a sorted permutation in the sense of Definition 1, the process of gene assembly will thus be seen as a sorting of signed permutations. We will define the simple operations as transformation rules for signed permutations in such a way that a simple operation is applicable on a gene pattern if and only if the corresponding rule is applicable on the associated signed permutation.

It is important to note that when formalizing gene assembly as a sorting of permutations we will effectively ignore the ld operation. Indeed, ld only combines two MDSs $M_i$ and $M_{i+1}$ already placed next to each other into a bigger composite MDS, by excising the IES separating them. However, in our representation we ignore all IES and thus, once $M_i$ and $M_{i+1}$ are placed next to each other at some point, we will consider that as far as the two of them are concerned, the assembly is already accomplished: an ld operation effectively “gluing” them together may be applied at any later point in the process. By doing so, we avoid renaming our alphabet of MDSs after each application of an operation. For the rest of this paper we will replace the alphabet $M_n$ with $\Sigma_n = \{1, 2, \ldots, n\}$.

The molecular model of simple hi and simple dlad in Figures 3 and 4 can be formalized as follows.

i. For each $p \geq 1$, $sh_p$ is defined as follows:
\[\text{sh}_i(xp \ldots (p+i)(p+i+1) \ldots (p+k)y) = xp \ldots (p+i)(p+i+1) \ldots (p+k)y,\]
\[\text{sh}_i(x(p+i) \ldots (p+k)y) = xp \ldots (p+i)(p+i+1) \ldots (p+k)y,\]
\[\text{sh}_i(x(p+i+1) \ldots (p+k) \ldots y) = x(p+k) \ldots (p+i+1)(p+i) \ldots y,\]
\[\text{sh}_i(x(p+k) \ldots (p+i+1)p \ldots (p+i) = x(p+k) \ldots (p+i+1)(p+i) \ldots y,\]
\[
\text{where } k > i \geq 0 \text{ and } x, y, z \text{ are signed strings over } \Sigma_n. \text{ We denote } \text{SH} = \{\text{sh}_i | 1 \leq i \leq n\}.
\]

ii. For each \( p, 2 \leq p \leq n - 1, \text{sd}_p \) is defined as follows:
\[\text{sd}_p(xp \ldots (p+i)y(p-1)(p+i+1)z) = xy(p-1)p \ldots (p+i)(p+i+1)z,\]
\[\text{sd}_p(x(p-1)(p+i)y \ldots (p+i)z) = x(p-1)p \ldots (p+i)(p+i+1)yz,\]
\[
\text{where } i \geq 0 \text{ and } x, y, z \text{ are signed strings over } \Sigma_n. \text{ We also define } \text{sd}_p \text{ as follows:}
\]
\[\text{sd}_p(x(p+i+1)(p+i+1) \ldots yz) = x(p+i+1)(p+i+1) \ldots yz,\]
\[\text{sd}_p(x(p+i) \ldots yz(p+i+1)(p-1)z) = xy(p-i+1)(p+i) \ldots yz,\]
\[
\text{where } i \geq 0 \text{ and } x, y, z \text{ are signed strings over } \Sigma_n. \text{ We denote } \text{SD} = \{\text{sd}_i, \text{sd}_j | 1 \leq i \leq n\}.
\]

We say that a signed permutation \( \pi \) over the set of integers \( \{i, i+1, \ldots, i+l\} \) is \textit{sortable} if there are operations \( \phi_1, \ldots, \phi_k \in \text{SH} \cup \text{SD} \) such that \( (\phi_1 \circ \ldots \circ \phi_k)(\pi) \) is a sorted permutation. We say that \( \pi \) is \textit{blocked} if neither an sh operation, nor a sd one is applicable to \( \pi \) and \( \pi \) is not sorted.

Let \( \phi = \phi_1 \circ \ldots \circ \phi_k \), \( \phi_i \in \text{SH} \cup \text{SD} \), for all \( 1 \leq i \leq k \). We say that \( \phi \) is a \textit{strategy} for \( \pi \) if \( \phi(\pi) \) is either sorted, or blocked. In the former case we say that \( \phi \) is a \textit{sorting strategy}, while in the latter case we say that \( \phi \) is a \textit{unsuccessful strategy} for \( \pi \).

If \( \phi = \phi_1 \circ \ldots \circ \phi_k \) is a sorting strategy for \( \pi \), we say that \( \pi \) is \textit{SH-sortable} if \( \phi_1, \ldots, \phi_k \in \text{SH} \) and we say that \( \pi \) is \textit{SD-sortable} if \( \phi_1, \ldots, \phi_k \in \text{SD} \).

**Example 2.**

i. The permutation \( \pi_1 = 21435 \) is sortable. It has the following sorting strategies: \( \text{sh}_{27} \circ \text{sd}_4 \circ \text{sh}_1(\pi_1) = 12345 \) and \( \text{sh}_{27} \circ \text{sh}_1 \circ \text{sd}_4(\pi_1) = 12345 \).

ii. The permutation \( \pi_2 = 25314 \) is blocked as no operations are applicable and it is not sorted.

iii. The permutation \( \pi_3 = 35124 \) has two assembly strategies which lead to different results: \( \text{sd}_3(\pi_3) = 51234 \) and \( \text{sd}_4(\pi_3) = 34512 \).

## 5 Simple operations are deterministic

We prove in this section our main results establishing that the simple model for gene assembly is deterministic: no signed permutation has both successful and unsuccessful strategies. We also show that even if different strategies may eventually lead to different (either sorted, or blocked) permutations, the “structure” of those permutations is the same.
5.1 The structure of a permutation

For any \( i \in \Sigma_n \) let \( \xi_i : \Sigma_n^* \to \Sigma_n^* \) be the morphism defined by \( \xi_i(i) = \lambda \), \( \xi_i(k) = k \), for \( 1 \leq k < i \), and \( \xi_i(k) = k - 1 \), for \( i < k \leq n \).

Consider also the mapping \( \sigma_i : \Sigma_n^* \to \Sigma_n^* \) where \( \sigma_i(u) \in \Sigma_n^* \), is defined as follows:

i. If \( i(i + 1) \not\leq u \) and \( (i + 1)i \not\leq u \), then \( \sigma_i(u) = u \) and

ii. Otherwise, \( \sigma_i(u) = \xi_i(u) \).

**Example 3.** If \( u = 231\overline{54} \in \Sigma_5^* \), then \( \sigma_1(u) = u \), \( \sigma_2(u) = 214\overline{3} \) and \( \sigma_4(u) = 231\overline{7} \).

**Definition 2.** We say that a word \( u \in \Sigma_n^* \) is simple if \( \sigma_i(u) = u \), for any \( i \in \Sigma_n \). Equivalently, \( i(i + 1) \not\leq u \) and \( (i + 1)i \not\leq u \), for any \( i \in \Sigma_n \). We say that \( u \) is a simple word associated to \( v \in \Sigma_n^* \) if \( u = \sigma_{i_1}\sigma_{i_2} \ldots \sigma_{i_k}(v) \), for some \( i_1, i_2, \ldots, i_k \in \Sigma_n \).

**Theorem 1.** For any word \( v \in \Sigma_n^* \), there exists a unique simple word \( \sigma(v) \) associated to \( v \). Moreover, if \( v \) is a signed permutation over \( \Sigma_n \), then \( \sigma(v) \) is a signed permutation over \( \Sigma_m \), for some \( m \leq n \).

**Proof.** Assume that there is a word \( v \in \Sigma_n^* \), for some \( v \geq 1 \), with two simple words \( w_1, w_2 \) associated to \( v \). Without loss of generality, assume that \( v \) is a minimal such word with respect to length. It follows then that there are \( i_1 \neq i_2 \) such that \( w_k \) is the simple word associated to \( \sigma_{i_k}(v) \), \( k = 1, 2 \).

If \( |i_2 - i_1| > 1 \), then \( \sigma_{i_1}(w_1(v)) = \sigma_{i_2}(w(v)) \). Due to the minimality of \( v \), \( \sigma_{i_k}(v) \), \( k = 1, 2 \), has a unique simple word associated to it and so, \( w_1 = w_2 \), a contradiction.

If \( |i_2 - i_1| = 1 \), say \( i_2 = i_1 + 1 \), then \( i_1 \) \( (i_1 + 1) ((i_1 + 2) (i_1 + 1) i_1 \leq v \) and so, \( \sigma_{i_1}(v) = \sigma_{i_2}(v) \), again contradicting the minimality of \( v \). The second part of the claim is obvious from the definition.

For any signed permutation \( v \) over \( \Sigma_n \) we call \( \sigma(v) \) the structure of \( v \). Without risk of confusion, for any \( p \in \Sigma_n \), we denote \( \sigma(\sh_p) = \sh_{\sigma(p)} \). Also, for \( p \in \Sigma_n \), \( \sum n \), we denote \( \sigma(\sd_p) = \sd_{\sigma(p)} \).

**Example 4.**

i. The simple permutation associated to \( \pi_1 = 18567432 \) is \( \sigma(\pi_1) = 1432 \).

ii. The simple permutation associated to \( \pi_2 = 1234 \) is \( \sigma(\pi_2) = 1 \).

iii. The simple permutation associated to \( \pi_3 = 3412 \) is \( \sigma(\pi_3) = 21 \).

iv. Operation \( \sh_3 \) is applicable to \( \pi_4 = 1234 \) and \( \sh_3(\pi_4) = 1234 \). In this case, \( \sigma(\pi_4) = 1324 \), \( \sigma(3) = 2 \) and \( \sigma(\sh_3) = \sh_2 \). Note that \( \sigma(\sh_4) \) is applicable to \( \sigma(\pi_4) \); indeed, \( \sigma(\sh_3)(\sigma(\pi_4)) = \sigma(\sh_3(\pi_4)) \).

Note now that for a simple permutation \( \pi \), if \( \sh_p \) is applicable to \( \pi, p \in \Sigma_n \), then

\[
\pi \in \{x(p + 1)y, x(p + 1)py, x(p + 1)py, x(p + 1)py \mid x, y \in \Sigma_n^* \},
\]

(1)
Also, if $sd_p$ is applicable to $\pi$, $p \in \Sigma_n$, then

$$\pi \in \{ x(p-1)(p+1)y pz, xpy(p-1)(p+1)z \mid x, y, z \in \Sigma_n^* \}, \quad (2)$$

Similarly, if $sd_q$ is applicable to $\pi$, then

$$\pi \in \{ xpy(p+1)(p-1)z, x(p+1)(p-1)y pz \mid x, y, z \in \Sigma_n^* \}. \quad (3)$$

The next lemma shows that as far as the sorting strategies are concerned, the signed permutation $\pi$ and its structure $\sigma(\pi)$ are in a sense equivalent. Its proof is straightforward.

**Lemma 2.** For any signed permutation $\pi$ over $\Sigma_n$ and any $\phi \in SH \cup SD$, $\phi$ is applicable to $\pi$ if and only if $\sigma(\phi)$ is applicable to $\sigma(\pi)$. Moreover, $\phi(u)$ and $\sigma(\phi)(\sigma(\pi))$ have the same structure.

**Corollary 3.** For any signed permutation $\pi$, $\pi$ has a successful (unsuccessful) strategy if and only if $\sigma(\pi)$ has a successful (unsuccessful, resp.) strategy.

### 5.2 Confluent strategies

We prove in this section that the sorting strategies for a given signed permutation $\pi$ are confluent: they are either all successful, or all unsuccessful and moreover, they lead to final results having the same structure.

We first prove a useful result for strategies using only $sh$ operations.

**Lemma 4.** Let $\pi$ be a simple signed permutation over $\Sigma_n$ and $\phi, \psi \in SH$ be two operations applicable to $\pi$. Then either $\phi(\pi) = \psi(\pi)$, or $\phi(\psi(\pi)) = \psi(\phi(\pi))$.

**Proof.** Let $\phi = sh_p$ and $\psi = sh_q$, for some $p \neq q$. It follows from (1) that if $|p - q| > 1$, then $\phi(\psi(\pi)) = \psi(\phi(u))$.

Now, if $|p - q| = 1$, say $q = p + 1$, then

$$\pi \in \{ x\overline{p}(p+1)(p+2)y, x(p+2)(p+1)\overline{p}y, xp(p+1)(p+2)y, x(p+2)(p+1)p y \},$$

for some $x, y$ signed words over $\Sigma_n$. Clearly, in this case, $\phi(\pi) = \psi(\pi)$.

The same result holds also for strategies using only $sd$ operations.

**Lemma 5.** Let $\pi$ be a simple signed permutation over $\Sigma_n$ and $\phi, \psi \in SD$ be two operations applicable to $\pi$. Then either $\phi(\psi(\pi)) = \psi(\phi(\pi))$, or $\phi(\pi)$ and $\psi(\pi)$ have the same structure.

**Proof.** Let $\phi = sd_p$ and $\psi = sd_q$, for some $p \neq q$. Clearly, if one of them is signed and the other is unsigned, then $\phi(\psi(\pi)) = \psi(\phi(\pi))$. Assume then that $p, q \in \Sigma_n$. The case $p, q \in \Sigma_n$ is completely similar.

It follows from (2) that if $|p - q| > 1$, then $\phi(\psi(\pi)) = \psi(\phi(\pi))$. If $|p - q| = 1$, say $q = p + 1$, it follows from (2) that $\pi = x(p-1)(p+1)yp(p+2)z$ or $\pi = xp(p+2)y(p-1)(p+1)z$, for some $x, y, z$. In the former case, $\phi(\pi) = x(p-1)p(p+1)y(p+2)z$ and $\psi(\pi) = x(p-1)y(p+1)(p+2)z$ have the same structure. The latter case is completely similar.

We can now extend the results above for strategies using both $sh$ and $sd$ operations.
Lemma 6. Let $\pi$ be a simple signed permutation over $\Sigma_n$ and $\phi, \psi \in \text{SH} \cup \text{SD}$ be two operations applicable to $\pi$. Then either $\phi(\psi(\pi)) = \psi(\phi(\pi))$, or $\phi(\pi)$ and $\psi(\pi)$ have the same structure.

Proof. Based on the previous two lemmata, we only need to prove the claim in the case $\phi \in \text{SH}$, $\psi \in \text{SD}$. Now, if $\phi = \text{sh}_p$ and $\psi = \text{sd}_q$, it follows from (1), (2), (3) that if $|p-q| > 1$, then $\phi(\psi(\pi)) = \psi(\phi(\pi))$.

Assume now that $|p-q| = 1$. Without loss of generality, assume that $q \in \Sigma_n$.

If $q = p + 1$, then, since $\text{sh}_p$ is applicable to $\pi$, $p$ and $p+1$ necessarily have different signatures. On the other hand, since $\text{sd}_{p+1}$ is applicable to $\pi$, $p$ and $p+1$ must have the same signature, a contradiction.

If $q = p - 1$, then

$$\pi \in \{x(p-2)p(p+1)y(p-1)z, x(p-1)y(p-2)p(p+1)z \mid x, y, z \in \Sigma_n^\Phi\}.$$ 

Then clearly, $\phi(\psi(\pi)) = \psi(\phi(\pi))$. □

Lemma 6 can be now extended easily to any signed permutations.

Theorem 7. Let $\pi$ be a signed permutation over $\Sigma_n$ and $\phi, \psi \in \text{SH} \cup \text{SD}$ be two operations applicable to $\pi$. Then

i. either $\phi(\pi)$ and $\psi(\pi)$ have the same structure,

ii. or $\phi \circ \psi$, $\psi \circ \phi$ are both applicable to $\pi$ and $\phi \circ \psi(\pi)$, $\psi \circ \phi(\pi)$ have the same structure.

Proof. Let $\pi' = \sigma(\pi)$. If $\sigma(\phi)(\pi')$ and $\sigma(\psi)(\pi')$ have the same structure, then it follows by Lemma 2 that $\phi(\pi)$ and $\psi(\pi)$ have the same structure. If that is not the case, it follows by Lemma 6 that $\sigma(\phi \circ \sigma(\psi))(\pi') = \sigma(\psi \circ \sigma(\phi))(\pi')$. But then ii. follows by Lemma 2. □

Example 5. Let $\pi = 135246$.

i. Both $\text{sd}_3$ and $\text{sd}_5$ are applicable to $\pi$. Moreover, $\text{sd}_2 \circ \text{sd}_5$ and $\text{sd}_5 \circ \text{sd}_2$ are also applicable to $\pi$ and $\text{sd}_2 \circ \text{sd}_5(\pi) = \text{sd}_5 \circ \text{sd}_2(\pi) = 123456$.

ii. Both $\text{sd}_2$ and $\text{sd}_3$ are applicable to $\pi$. Moreover, $\text{sd}_2(\pi) = 123546$ and $\text{sd}_3(\pi) = 152346$ have the same structure: $\sigma(\text{sd}_2(\pi)) = \sigma(\text{sd}_3(\pi)) = 1324$.

We are ready now to prove the main result of the paper, proving that simple gene assembly is deterministic.

Theorem 8. Let $\pi$ be a signed permutation over $\Sigma_n$ and $\phi, \psi$ be two strategies for $\pi$. Then either $\phi$ and $\psi$ are both sorting strategies for $\pi$, or they are both unsuccessful strategies. Moreover, $\phi(\pi)$ and $\psi(\pi)$ have the same structure.

Proof. Assume that the claim of the theorem is not true and consider a signed permutation $\pi$ of minimal length such that $\phi = \phi_1 \ldots \phi_k$ is a successful strategy for $\pi$, while $\psi = \psi_1 \ldots \psi_l$ is an unsuccessful one, $\phi_i, \psi_j \in \text{SH} \cup \text{SD}$, $1 \leq i \leq k$, $1 \leq j \leq l$.

It follows from Theorem 7 that either $\phi_k(\pi)$ and $\psi_l(\pi)$ have the same structure, or $\phi_k(\psi_l(\pi)) = \psi_l(\phi_k(\pi))$. In the former case however we obtain by Lemma 2 that $\sigma(\phi_k(\pi))$ is a smaller counterexample than $\pi$ contradicting the minimality of $\pi$. In the latter case note that due to the minimality of $\pi$, it
follows that \( \phi_k(\pi) \) has only successful strategies and \( \psi_l(\pi) \) has only unsuccessful strategies. Consequently, \( \psi_l(\phi_k(\pi)) \) has both successful and unsuccessful strategies, contradicting the minimality of \( \pi \).

**Corollary 9.** For any signed permutation \( \pi \) over \( \Sigma_n \) it is decidable in quadratic time whether \( \pi \) is SH-, SD-, or SH∪SD-sortable or not.

**Proof.** Based on the determinism of simple operations, as proved in Theorem 8, we can simply apply to \( \pi \) the first applicable operation sh\(_6\) or sd\(_6\) and repeat the procedure with the new permutation until it leads either to a sorted permutation or to a blocked one.

**Example 6.** The following examples demonstrate the confluence of assembly strategies. In each case we show that although different strategies lead to different results, the results have the same structure.

i. The permutation \( \pi_1 = 4\overline{671235} \) has several sorting strategies. Some of them are shown below.

\[
\begin{align*}
\pi_1^{(1)} &= sd_5 \circ sh_6 \circ sh_1(\pi_1) = 4567123, & \sigma(\pi_1^{(1)}) = 21, \\
\pi_1^{(2)} &= sd_5 \circ sh_6 \circ sh_2(\pi_1) = 4567123, & \sigma(\pi_1^{(2)}) = 21, \\
\pi_1^{(3)} &= sd_4 \circ sh_6 \circ sh_2(\pi_1) = 6712345, & \sigma(\pi_1^{(3)}) = 21, \\
\pi_1^{(4)} &= sh_6 \circ sh_1 \circ sd_4(\pi_1) = 6712345, & \sigma(\pi_1^{(4)}) = 21.
\end{align*}
\]

ii. The permutation \( \pi_2 = 13685724 \) has several unsuccessful strategies. Some of them are shown below.

\[
\begin{align*}
\pi_2^{(1)} &= sd_2 \circ sd_7(\pi_2) = 12367854 & \sigma(\pi_2^{(1)}) = 1432 \\
\pi_2^{(2)} &= sd_2 \circ sd_6(\pi_2) = 12385674 & \sigma(\pi_2^{(2)}) = 1432 \\
\pi_2^{(3)} &= sd_3 \circ sd_7(\pi_2) = 18567234 & \sigma(\pi_2^{(3)}) = 1432 \\
\pi_2^{(4)} &= sd_3 \circ sd_6(\pi_2) = 18567234 & \sigma(\pi_2^{(4)}) = 1432.
\end{align*}
\]

**Acknowledgment**

The work of Ion Petre is supported by Academy of Finland, projects 110815 and 203667.

**References**


Last updated December 9, 2005. Miika Langille, Ion Petre


