TUCS

Tero Harju | Ion Petre | Vladimir Rogojin | Grzegorz Rozenberg

Simple Operations for Gene Assembly

Turku Centre *for* Computer Science

TUCS Technical Report
No 757, August 2006

TUCS

Simple Operations for Gene Assembly

Tero Harju
Department of Mathematics, University of Turku
Turku 20014 Finland
harju@utu.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
Vladimir Rogojin
Turku Centre for Computer Science
Department of Computer Science, Åbo Akademi University
Turku 20520 Finland,
vrogojin@abo.fi
Grzegorz Rozenberg
Leiden Institute for Advanced Computer Science
Niels Bohrweg 1, 2333 CA Leiden, the Netherlands,
rozenber@liacs.nl

TUCS Technical Report
No 757, August 2006

Abstract

The intramolecular model for gene assembly in ciliates considers three operations, ld, hi, and dlad that can assemble any gene pattern through folding and recombination: the molecule is folded so that two occurrences of a pointer (short nucleotide sequence) get aligned and then the sequence is rearranged through recombination of pointers. In general, the sequence rearranged by one operation can be arbitrarily long and consist of many coding and non-coding blocks. We consider in this paper simple variants of the three operations, where only one coding block is rearranged at a time. We characterize in this paper the gene patterns that can be assembled through these variants. Our characterization is in terms of signed permutations and dependency graphs. Interestingly, we show that simple assemblies possess rather involved properties: a gene pattern may have both successful and unsuccessful assemblies and also more than one successful assembling strategy.

Keywords: Gene assembly, simple operations, signed permutations, sorting

TUCS Laboratory
Computational Biomodelling Laboratory

1 Introduction

Ciliates are very old eukaryotic organisms that have developed a very unusual way of organizing their genomic sequences. In the macronucleus, the somatic nucleus of the cell, each gene is a contiguous DNA sequence. Genes are generally placed on their own very short DNA molecules. In the micronucleus, the germline nucleus of the cell, the same gene is broken into pieces called MDSs (macronuclear destined sequences) that are separated by noncoding blocks called IESs (internally eliminated sequences). Moreover, the order of MDSs is shuffled, with some of the MDSs being inverted. The structure is particularly complex in a family of ciliates called *Stichotrichs* – we concentrate in this paper on this family. During the process of sexual reproduction, ciliates destroy the old macronuclei and transform a micronucleus into a new macronucleus. In this process, ciliates must assemble all genes by placing in the orthodox order all MDSs. The complexity of the gene assembly process is given by the fundamentally different organization of the micronuclear and the macronuclear genomes.

The macronuclear genes are very short molecules, ranging in the Sterkiella nova organisms between 200bp and 3700bp, with an average of 2200 bp in length, see [23, 19, 4, 5]. Incidentally, these are the shortest DNA molecules known in Nature, even shorter than those of viruses, see [21]. On the other hand, the micronuclear genome is organized on very long chromosomes (about 120 chromosomes, each with about 10^7 bp in S.nova, see [19]), with coding sequences occupying as little as 2 - 5% of the genome, see, e.g., [4]. Ciliates thus have to identify precisely the genetic material and splice it out from the chromosomes. The real intricacy however is revealed when looking into the gene structure in micro- and macronucleus. The macronuclear gene is a contiguous sequence of nucleotides. The same gene in the micronucleus is broken into blocks called MDSs (macronuclear-destined sequences), separated by non-coding blocks called IESs (internally-eliminated sequences). Moreover, the order of the MDSs is shuffled and some of them may even be inverted. Here is where the challenge (and the beauty) of gene assembly lies: ciliates have to identify correctly more than 100 000 MDSs in their genome, see [21], assemble them together in the orthodox order, and eliminate all IESs. We refer to [12, 19, 25] for more details on ciliates and gene assembly.

A hint on how ciliates achieve gene assembly is given by the structure of MDSs. It turns out that ciliates have developed a very ingenious way of organizing their genomic data as linked lists in the style used in computer science, see [19]. A short sequence in the end of each MDS is repeated identically in the beginning of the MDS that should follow it in the orthodox order, thus serving as a computer science-like pointer. Moreover, the first MDS starts with a special beginning marker, while the last MDS ends with a special ending marker. It is currently believed that ciliates splice together their MDSs on the common pointers to assemble the gene. There are two main models for gene assembly, see [16, 17] and [8, 22], that both agree on this generic mechanism.

The intramolecular model for gene assembly, introduced in [8] and [22] consists of three operations: ld, hi, and dlad. In each of these operations, the molecule folds on itself so that two or more pointers get aligned and through recombination two or more MDSs get combined into a bigger composite MDS. The process continues until all MDSs have been assembled. For details related to ciliates and gene assembly we refer to [12], [19], [20] and for details related to the intramolecular model and its mathematical formalizations we refer to [6]. For a different intermolecular model we refer to [14], [16], [17].

In general there are no restrictions on the number of nucleotides between the two pointers that should be aligned in a certain fold. However, all available experimental data is consistent with restricted versions of our operations, in which between two aligned pointers there is never more than one MDS, see [6] and [7]. We propose in this paper a mathematical model for simple variants of ld, hi, and dlad. The model, in terms of signed permutations, is used to answer the following question: which gene patterns can be assembled by the simple operations? As it turns out, the question is difficult: the simple assembly is a non-deterministic process, with more than one strategy possible for certain patterns and in some cases, with both successful and unsuccessful assemblies. We completely answer the question in terms of sorting signed permutations. Here, a signed permutation represents the sequence of MDSs in a gene pattern, including their orientation.

There is rich literature on sorting (signed and unsigned) permutations, both in connection to their applications to computational biology in topics such as genomic rearrangements or genomic distances, but also as a classical topic in discrete mathematics, see, e.g., [1], [2], [9], [13].

A preliminary version of this paper has been published in [10]. We present here full constructions, complete proofs, and new examples. We also correct some errors in [10], in connection with defining the notion of dependency graph.

2 Preliminaries

For an alphabet Σ we denote by Σ^* the set of all finite strings over Σ. For a string u we denote $\mathsf{dom}(u)$ the set of letters occurring in u. We denote by Λ the empty string. For strings u, v over Σ, we say that u is a *substring* of v, denoted $u \leq v$, if $v = xuy$, for some strings x, y. We say that u is a *subsequence* of v, denoted $u \leq_s v$, if $u = a_1a_2 \ldots a_m$, $a_i \in \Sigma$ and $v = v_0a_1v_1a_2 \ldots a_mv_m$, for some strings v_i, $0 \leq i \leq m$, over Σ. For some $A \subseteq \Sigma$ we define the morphism $\phi_A : \Sigma^* \to A^*$ as follows: $\phi_A(a_i) = a_i$, if $a_i \in A$ and $\phi_A(a_i) = \Lambda$ if $a_i \in \Sigma \setminus A$. For any $u \in \Sigma^*$, we denote $u|_A = \phi_A(u)$. We say that the *relative positions* of letters from set $A \subseteq \Sigma$ are the same in strings $u, v \in \Sigma^*$ if and only if $u|_A = v|_A$.

Let $\Sigma_n = \{1, 2, \ldots, n\}$ and let $\overline{\Sigma}_n = \{\overline{1}, \overline{2}, \ldots, \overline{n}\}$ be a *signed copy* of Σ_n. For any $p \in \Sigma_n$ we say that p is a *unsigned letter*, while $\overline{p}$ is a *signed letter*. We call the *identity mapping* and denote it by id the automorphism on $(\Sigma_n \cup \overline{\Sigma}_n)^*$ such that $\mathsf{id}(u) = u$ for any string u over $(\Sigma_n \cup \overline{\Sigma}_n)$. Let $\|.\|$ be the morphism from $(\Sigma_n \cup \overline{\Sigma}_n)^*$ to Σ_n^* that unsigns the letters: for all $a \in \Sigma_n$, $\|\overline{a}\| = \|a\| = a$. For a string u over $\Sigma_n \cup \overline{\Sigma}_n$, $u = a_1a_2 \ldots a_m$, $a_i \in \Sigma_n \cup \overline{\Sigma}_n$, for all $1 \leq i \leq m$, we denote its *inversion* by $\overline{u} = \overline{a}_m \ldots \overline{a}_2\overline{a}_1$, where $\overline{\overline{a}} = a$, for all $a \in \Sigma_n$.

Consider a *bijective mapping* (called *permutation*) $\pi : \Delta \to \Delta$ over an alphabet $\Delta = \{a_1, a_2, \ldots, a_l\}$ with the order relation $a_i \leq a_j$ for all $i \leq j$. We often identify π with the string $\pi(a_1)\pi(a_2)\ldots\pi(a_l)$. The domain of π, denoted $\mathsf{dom}(\pi)$, is Δ. We say that π is *(cyclically) sorted* if $\pi = a_k\, a_{k+1} \ldots a_l\, a_1\, a_2\, \ldots\, a_{k-1}$, for some $1 \leq k \leq l$.

A *signed permutation* over Δ is a string ψ over $\Delta \cup \overline{\Delta}$ such that $\|\psi\|$ is a permutation over Δ. We say that ψ is *(cyclically) sorted* if $\psi = a_k\, a_{k+1} \ldots a_l\, a_1\, a_2\, \ldots$ $\ldots\, a_{k-1}$ or $\psi = \overline{a}_{k-1} \ldots \overline{a}_2\, \overline{a}_1\, \overline{a}_l \ldots \overline{a}_{k+1}\, \overline{a}_k$, for some $1 \leq k \leq l$. Equivalently, ψ is sorted if either ψ, or $\overline{\psi}$ is a sorted unsigned permutation. In the former case we say that ψ is sorted in the *orthodox order* or that ψ is a *sorted orthodox permutation*, while in the latter case we say that ψ is sorted in the *inverted order* or that ψ is a *sorted inverted permutation*.

For basic notions and results on graph theory we refer to [24].

3 The Simple Intramolecular Model

The micronuclear gene structure may be abstracted (by ignoring the non-coding blocks) as a shuffled sequence of coding blocks called MDSs. During gene assembly, the MDSs are sorted in the orthodox order to yield the assembled macronuclear gene. This rearrangement is facilitated by the special structure of the MDSs: each MDS M ends with a short nucleotide sequence that is repeated in the beginning of the MDS following M in the assembled gene. Thus, each MDS M starts with an *incoming pointer*, “pointing” to the MDS preceding M in the assembled gene, and it ends with an *outgoing pointer*, “pointing” to the MDS succeeding M in the assembled gene. Exceptions are the first and the last MDSs from the assembled gene: the first MDS has a *beginning marker* rather than an incoming pointer and the last MDS has an *ending marker* rather than an outgoing pointer.

Three molecular operations, ld, hi and dlad where conjectured in [8] and [22] for gene assembly, see [6] for a detailed presentation. We consider in this paper the simple versions of these molecular operations, defined bellow, and investigate the gene patterns they can assemble. It is important to note that, as observed in [11], all available experimental data, see [3], is consistent with applications of the simple operations, although they are not complete: there are signed permutations (sequences of MDSs) that they cannot sort (assemble).

The effect of the ld operation is to combine two consecutive MDSs $M_i\, M_{i+1}$ into a bigger composite MDS $M_{i,i+1}$ by eliminating the non-coding sequences between them. In this paper however, we do not consider the non-coding sequences separating the MDSs and in this way, assembling the gene simply becomes sorting the MDSs in the orthodox order. Consequently, in this abstraction, we will effectively ignore the ld operation.

The simple hi operation is applicable to an MDS sequence δ, if in δ there are two consecutive MDSs M and N, both containing one copy of a pointer p, one being inverted with respect to the other. The operation changes δ as illustrated in Figure 1: depending on the incoming/outgoing position of p, either M or N is inverted.

The simple dlad operation is applicable to an MDS sequence δ if in δ there is

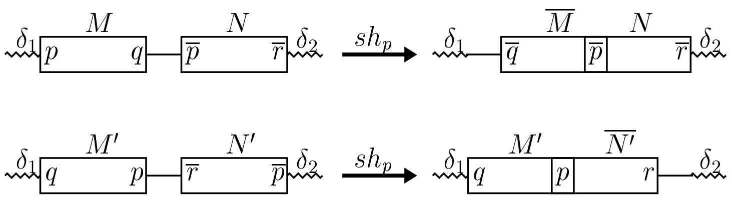

Figure 1: The MDS structures where the simple hi-rule is applicable: the two occurrences of pointer p, one inverted, are placed on consecutive MDSs. The MDS sequence is changed as illustrated in the figure. A rectangle denotes one MDS, with its two pointers indicated, a straight line indicates that no MDSs occur in that area, while a jigged line denotes an arbitrary sequence of MDSs; $\overline{M}$ denotes the inverse of MDS M and $\overline{N'}$ denotes the inverse of N'.

an MDS M flanked by some pointers p and q, where there is no MDS occurring in δ between the second occurrence of p and the second occurrence of q. The operation changes δ as illustrated in Figure 2: MDS M is moved between the second occurrence of p and the second occurrence of q.

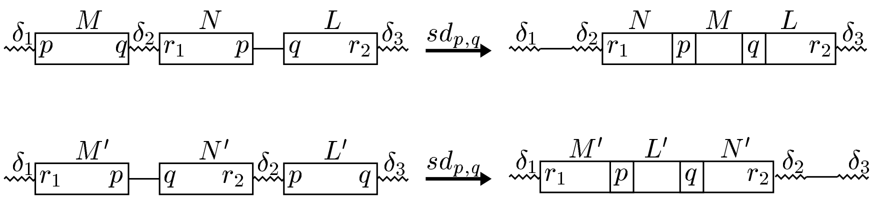

Figure 2: The MDS structures where the simple dlad-rule is applicable: one pair of pointers p and q is placed on the same MDS, while in between the other pair of p and q there is no MDS. The MDS sequence is changed as illustrated in the figure. A rectangle denotes one MDS, with its two pointers indicated, a straight line indicates that no MDSs occur in that area, while a jigged line denotes an arbitrary sequence of MDSs.

For a detailed presentation of the molecular transformations conjectured to take place in simple hi and simple dlad, including folding of the DNA molecules and various recombinations, we refer to [11].

In this paper we consider restricted versions of the simple operations. We consider such simple hi and dlad that rearrange parts of the molecule containing only non-composite MDSs. For a study on non-restricted simple operations we refer to [18].

4 Gene Assembly as a Sorting of Signed Permutations

In this paper we represent each MDS M_p by symbol p and its inversion $\overline{M}_p$ by symbol $\overline{p}$. In this way, a sequence of MDSs is represented by a signed permutation. In this paper we choose to ignore the ld operation observing that once such an operation becomes applicable to a gene pattern, it can be applied at any later step of the assembly, see [6] for a formal proof. In particular, we can assume that all ld operations are applied in the last stage of the assembly, once all MDSs are sorted in the correct order. In this way, the process of gene assembly can indeed be described as a process of sorting the associated signed permutation, i.e., arranging the MDSs in the proper order, be that orthodox or inverted.

The simple hi is formalized on permutations through operation sh. For each $p \geq 1$, sh_p is defined as follows:

$$\begin{aligned} &\mathsf{sh}_p(x\,p\,\overline{(p+1)}\,y) = x\,p\,(p+1)\,y, && \mathsf{sh}_p(x\,\overline{(p+1)}\,p\,y) = x\,\overline{(p+1)}\,\overline{p}\,y,\\ &\mathsf{sh}_p(x\,\overline{p}\,(p+1)\,y) = x\,p\,(p+1)\,y, && \mathsf{sh}_p(x\,(p+1)\,\overline{p}\,y) = x\,\overline{(p+1)}\,\overline{p}\,y, \end{aligned}$$

where x, y are signed strings over Σ_n. We denote $\mathsf{Sh} = \{\mathsf{sh}_p \mid 1 \leq p \leq n\}$.

The simple dlad is formalized on permutations through operation sd. For each p, $2 \leq p \leq n-1$, sd_p is defined as follows:

$$\begin{aligned} \mathsf{sd}_p(x\,p\,y\,(p-1)\,(p+1)\,z) &= x\,y\,(p-1)\,p\,(p+1)\,z,\\ \mathsf{sd}_p(x\,(p-1)\,(p+1)\,y\,p\,z) &= x\,(p-1)\,p\,(p+1)\,y\,z,\\ \mathsf{sd}_p(x\,\overline{(p+1)}\,\overline{(p-1)}\,y\,\overline{p}\,z) &= x\,\overline{(p+1)}\,\overline{p}\,\overline{(p-1)}\,y\,z,\\ \mathsf{sd}_p(x\,\overline{p}\,y\,\overline{(p+1)}\,\overline{(p-1)}\,z) &= x\,y\,\overline{(p+1)}\,\overline{p}\,\overline{(p-1)}\,z, \end{aligned}$$

where x, y, z are signed strings over Σ_n. We denote $\mathsf{Sd} = \{\mathsf{sd}_p \mid 1 \leq p \leq n\}$.

Definition 1. *We define orthodox and inverted operations as follows:*

- *Operations sh_p transforming strings $u\,\overline{p}\,(p+1)\,v$ and $u\,p\,\overline{(p+1)}\,v$ to $u\,p\,(p+1)\,v$ we will call orthodox Sh operations;*

- *Operations sh_p transforming strings $u\,\overline{(p+1)}\,p\,v$ and $u\,(p+1)\,\overline{p}\,v$ to $u\,\overline{(p+1)}\,\overline{p}\,v$ we will call inverted Sh operations;*

- *Operations sd_p transforming strings $u\,p\,v\,(p-1)\,(p+1)\,w$ and $u\,(p-1)\,(p+1)\,v\,p\,w$ to $u\,v\,(p-1)\,p\,(p+1)\,w$ and to $u\,(p-1)\,p\,(p+1)\,v\,w$ respectively we will call orthodox Sd operations;*

- *Operations sd_p transforming strings $u\,\overline{p}\,v\,\overline{(p+1)}\,\overline{(p-1)}\,w$ and $u\,\overline{(p+1)}\,\overline{(p-1)}\,v\,\overline{p}\,w$ to $u\,v\,\overline{(p+1)}\,\overline{p}\,\overline{(p-1)}\,w$ and to $u\,\overline{(p+1)}\,\overline{p}\,\overline{(p-1)}\,v\,w$ respectively we will call inverted Sd operations.*

For a composition of operations $\Phi = \phi_k \circ \ldots \circ \phi_1$ we write $\phi_i \in \Phi$ for all $1 \leq i \leq k$ and we say that ϕ_i is *used* in Φ *before* ϕ_j for all $1 \leq i < j \leq k$.

We say that a signed permutation π over the set of integers Σ_n is *sortable* if there is a composition $\Phi = \phi_k \circ \ldots \circ \phi_1$ such that $\Phi(\pi)$ is a (cyclically) sorted signed permutation. In this case we say that Φ *sorts* π and also, that it is a *sorting composition* for π. Permutation π is Sh-*sortable* if $\phi_1, \ldots, \phi_k \in \mathsf{Sh}$ and π is Sd-*sortable* if $\phi_1, \ldots, \phi_k \in \mathsf{Sd}$.

Example 1. *(i) Permutation $\pi_1 = 3\,\overline{4}\,\overline{5}\,6\,\overline{1}\,2$ is sortable and a sorting composition is $\mathsf{sh}_1(\mathsf{sh}_5(\mathsf{sh}_3(\pi_1))) = 3\,4\,5\,6\,1\,2$. Permutation $\pi'_1 = 3\,4\,5\,6\,\overline{1}\,\overline{2}$ is unsortable. Indeed, no Sh operations and no Sd operation is applicable to π'_1.*

(ii) Permutation $\pi_2 = 1\,3\,4\,2\,\overline{5}$ is sortable and has only one sorting composition: $\mathsf{sh}_4(\mathsf{sd}_2(\pi_2)) = 1\,2\,3\,4\,5$.

(iii) There exist permutations with several sorting compositions, even leading to different sorted permutations. One such permutation is $\pi_3 = 3\,5\,1\,2\,4$. Indeed, $\mathsf{sd}_3(\pi_3) = 5\,1\,2\,3\,4$. At the same time, $\mathsf{sd}_4(\pi_3) = 3\,4\,5\,1\,2$.

(iv) The simple operations yield a nondeterministic process: there are permutations having both sorting compositions and non-sorting compositions leading to unsortable permutations. One such permutation is $\pi_4 = 1\,3\,5\,7\,9\,2\,4\,6\,8$. Note that $\mathsf{sd}_3(\mathsf{sd}_5(\mathsf{sd}_7(\pi_4))) = 1\,9\,2\,3\,4\,5\,6\,7\,8$ is a unsortable permutation. However, π_4 can be sorted, e.g., by the following composition: $\mathsf{sd}_2(\mathsf{sd}_4(\mathsf{sd}_6(\mathsf{sd}_8(\pi_4)))) = 1\,2\,3\,4\,5\,6\,7\,8\,9$.

(v) Permutation $\pi_5 = 1\,3\,5\,2\,4$ has both sorting and non-sorting compositions. Indeed, $\mathsf{sd}_3(\pi_5) = 1\,5\,2\,3\,4$, a unsortable permutation. However, $\mathsf{sd}_2(\mathsf{sd}_4(\pi_5)) = 1\,2\,3\,4\,5$ is sorted.

(vi) Applying a cyclic shift to a permutation may render it unsortable. Indeed, permutation $2\,1\,4\,3\,5$ is sortable, while $5\,2\,1\,4\,3$ is not.

(vii) Consider the signed permutation $\pi_7 = 1\,11\,3\,9\,5\,7\,2\,4\,13\,6\,15\,8\,10\,12\,14\,16$. Operation sd may be applied to π_7 on elements 3, 6, 9, 11, 13, and 15. Doing that however leads to a unsortable permutation:

$$\mathsf{sd}_3(\mathsf{sd}_6(\mathsf{sd}_9(\mathsf{sd}_{11}(\mathsf{sd}_{13}(\mathsf{sd}_{15}(\pi_7)))))) = 1\,5\,6\,7\,2\,3\,4\,8\,9\,10\,11\,12\,13\,14\,15\,16.$$

However, omitting sd_3 from the above composition leads to a sorting composition for π_7: let

$$\pi'_7 = \mathsf{sd}_6(\mathsf{sd}_9(\mathsf{sd}_{11}(\mathsf{sd}_{13}(\mathsf{sd}_{15}(\pi_7))))) = 1\,3\,5\,6\,7\,2\,4\,8\,9\,10\,11\,12\,13\,14\,15\,16.$$

Then $\mathsf{sd}_2(\mathsf{sd}_4(\pi'_7))$ is a sorted permutation.

The following lemma follows directly from the definition of sd and sh.

Lemma 1. *Let π be a signed permutation over Σ_n and $p \in \Sigma_n$. Then we have the following properties:*

(i) sd_p is applicable to π if and only if sd_p is applicable to $\overline{\pi}$ and in this case, $\overline{\mathsf{sd}_p(\pi)} = \mathsf{sd}_p(\overline{\pi})$;

(ii) sh_p is applicable to π if and only if sh_p is applicable to $\overline{\pi}$ and in this case, $\overline{\mathsf{sh}_p(\pi)} = \mathsf{sh}_p(\overline{\pi})$;

(iii) $\|\,\mathsf{sh}_p(\pi)\| = \|\pi\|$;

(iv) If $p(p+1) \leq \pi$, then for any composition Φ of Sh and Sd operations applicable to π, $p(p+1) \leq \Phi(\pi)$;

(v) If $p(p+1) \leq \pi$, then sd_p, sd_{p+1}, and sh_p cannot be used in any composition applicable to π.

Lemma 2. *Let π be a signed permutation over Σ_n and Φ a composition applicable to π. Then, Φ is applicable to $\overline{\pi}$ as well and we have that $\overline{\Phi(\pi)} = \Phi(\overline{\pi})$.*

Proof. We prove this by induction on the number of operations in Φ. The case when $|\Phi| = 1$ follows from Lemma 1. Now, assume for any composition Φ of length k applicable to π we have that Φ is also applicable to $\overline{\pi}$ and $\overline{\Phi(\pi)} = \Phi(\overline{\pi})$. Consider composition $\Phi' = \phi \circ \Phi$, where ϕ is either an Sh or an Sd operation. Consider the permutation $\pi' = \Phi(\pi)$. Clearly, ϕ can be applied to π' and $\overline{\phi(\pi')} = \phi(\overline{\pi'})$ by Lemma 1. But, $\overline{\phi(\pi')} = \overline{\phi(\Phi(\pi))} = \overline{\Phi'(\pi)}$ and $\phi(\overline{\pi'}) = \phi(\overline{\Phi(\pi)}) = \phi(\Phi(\overline{\pi})) = \Phi'(\overline{\pi})$. In this way $\overline{\Phi'(\pi)} = \Phi'(\overline{\pi})$ and so, the lemma is proved. ☐

The following result follows from Lemma 1(iv), (v) and the definition of the operations sh and sd.

Lemma 3. *Let π be a signed permutation over Σ_n and $p \in \Sigma_n$.*

(i) sd_{p-1} and sd_p cannot be used in the same composition applicable to π.

(ii) sh_{p-1} and sd_p cannot be used in the same composition applicable to π.

(iii) sd_p can be used at most once in a composition applicable to π.

(iv) sh_p can be used at most once in a composition applicable to π.

(v) sh_p and sd_p cannot be used in the same composition applicable to π.

(vi) sd_1 and sd_n are not applicable in any composition.

(vii) sh_n cannot be used in any of compositions.

Theorem 4. *No permutation π can be sorted both to an orthodox permutation and to an inverted one.*

Proof. Assume that there is a permutation π that can be sorted both to an orthodox permutation and to an inverted one. We have two cases: either $1\,n \leq_s \|\pi\|$, or $n\,1 \leq_s \|\pi\|$. Assume the first case, as the second one can be reduced to the first one by Lemma 2. Then there are two sorting compositions Φ_o and Φ_i for π such that $\Phi_o(\pi) = 1\,2 \dots n$ and $\Phi_i(\pi) = (\overline{k-1}) \dots \overline{2}\,\overline{1}\,\overline{n} \dots (\overline{k+1})\,\overline{k}$, for some $k \geq 2$. We have now the following two cases:

(i) 1 is unsigned in π. Then $\mathsf{sh}_1 \in \Phi_i$ and so, $k \geq 3$. Also, it follows by Lemma 3 that $\mathsf{sd}_1, \mathsf{sd}_2 \notin \Phi_i$ and so, the relative position of 1 and 2 does not change in π: $2\,1 \leq_s \|\pi\|$.

Since $\Phi_o(\pi) = 1\,2 \dots n$, it follows that $\mathsf{sd}_2 \in \Phi_o$ and so, by Lemma 3, $\mathsf{sd}_3 \notin \Phi_o$. Then $2\,1\,3 \leq_s \|\pi\|$.

If 2 is unsigned in π, then $\mathsf{sh}_2 \in \Phi_i$, but for sh_2 to be applicable, sd_3 has to be applied in Φ_i before sh_2, contradicting Lemma 3.

If 2 is signed in π, then either $\mathsf{sh}_1 \in \Phi_o$, or $\mathsf{sh}_2 \in \Phi_o$. Since $\mathsf{sd}_2 \in \Phi_o$, this contradicts Lemma 3.

(ii) 1 is signed in π. Then $\mathsf{sh}_1 \in \Phi_o$ and so, $\mathsf{sd}_2 \notin \Phi_o$, i.e., the relative position of 1 and 2 does not change through applying Φ_o: $1\,2 \leq_s \|\pi\|$. We have now two cases as follows:

(ii.1) $k \geq 3$: $\Phi_i(\pi) = (\overline{k-1}) \ldots \overline{1}\,\overline{n} \ldots \overline{k}$. In this case, $\mathsf{sd}_2 \in \Phi_i$ and so, $\mathsf{sd}_3 \notin \Phi_i$, i.e., $3\,1\,2 \leq_s \|\pi\|$.

If 2 is unsigned in π, i.e., $\overline{1}\,2 \leq_s \pi$, then $\mathsf{sh}_1 \in \Phi_i$ or $\mathsf{sh}_2 \in \Phi_i$, a contradiction by Lemma 3 since $\mathsf{sd}_2 \in \Phi_i$.

If 2 is signed in π, i.e., $\overline{1}\,\overline{2} \leq_s \pi$, then $\mathsf{sh}_2 \in \Phi_o$ and so, to become applicable, sd_3 must be used in Φ_o before sh_2, contradicting Lemma 3.

(ii.2) $k = 2$: $\Phi_i(\pi) = \overline{1}\,\overline{n} \ldots \overline{2}$.

If 2 is unsigned in π, i.e., $\overline{1}\,2 \leq_s \pi$, then either $\mathsf{sh}_1 \in \Phi_i$ or $\mathsf{sh}_2 \in \Phi_i$ and by Lemma 3, $\mathsf{sd}_1, \mathsf{sd}_2, \mathsf{sd}_n \notin \Phi_i$. Thus, $1, 2, n$ do not change their relative position through Φ_i and so, $1\,n\,2 \leq_s \|\pi\|$. Consequently, $\mathsf{sd}_2 \in \Phi_o$, a contradiction by Lemma 3 since $\mathsf{sh}_1 \in \Phi_o$.

If 2 is signed in π, i.e., $\overline{1}\,\overline{2} \leq_s \pi$, then $\mathsf{sh}_2 \in \Phi_o$ and so, by Lemma 3, $\mathsf{sd}_2, \mathsf{sd}_3 \notin \Phi_o$. Thus, $1, 2, 3$ do not change their relative position through Φ_o and so, $1\,2\,3 \leq_s \|\pi\|$. But then, either $\mathsf{sd}_2 \in \Phi_i$, or $\mathsf{sd}_3 \in \Phi_i$, but not both. Thus, either $\mathsf{sd}_3 \notin \Phi_i$, or $\mathsf{sd}_2 \notin \Phi_i$, i.e., either 1, 3, n or 1, 2, n do not change their relative positions through Φ_i, i.e., either $1\,n\,3 \leq_s \|\pi\|$ or $1\,n\,2 \leq_s \|\pi\|$. But then, either $\mathsf{sd}_3 \in \Phi_o$, or $\mathsf{sd}_2 \in \Phi_o$, a contradiction by Lemma 3 since $\mathsf{sh}_2 \in \phi_o$.

□

Lemma 5. *Let π be a signed permutation.*

(a) π cannot be sorted to an orthodox order if there exists p such that:

(i) $(p+1)\,\overline{p} \leq \pi$, or

(ii) $(\overline{p+1})\,p \leq \pi$, or

(iii) $(\overline{p+1})\,(\overline{p-1}) \leq \pi$.

(b) π cannot be sorted to an inverted order if there exists q such that:

(iv) $q\,(\overline{q+1}) \leq \pi$, or

(v) $\overline{q}\,(q+1) \leq \pi$, or

(vi) $(q-1)\,(q+1) \leq \pi$.

Proof. We only prove here part (a) of the result, since part (b) is symmetric with respect to inversion.

To prove (a.i), assume that $(p+1)\,\overline{p} \leq \pi$ and π may be sorted to an orthodox order through a composition Φ of Sh and Sd operations. Then either $\mathsf{sh}_{p-1} \in \Phi$ or $\mathsf{sh}_p \in \Phi$ and so, by Lemma 3, $\mathsf{sd}_p \notin \Phi$. But then, $\mathsf{sd}_{p+1} \in \Phi$ and so, $\mathsf{sh}_p \notin \Phi$. Thus, $\mathsf{sh}_{p-1} \in \Phi$ and $\mathsf{sd}_{p+1} \in \Phi$. The contradiction comes from the fact that sh_{p-1} must be applied before sd_{p+1} which in its turn, must be applied before sh_{p-1} and an operation may only be used once in a composition, by Lemma 3.

Claim (a.ii) follows similarly as (a.i).

To prove (a.iii), assume as above that $(\overline{p+1})\,(\overline{p-1}) \leq \pi$ and π is sorted to an orthodox order by Φ. Since an orthodox sorted permutation has no signed letters, it follows that $\mathsf{sh}_p, \mathsf{sh}_{p-1} \in \Phi$. Consequently, throughout the assembly, we must obtain both $p\,(\overline{p+1})$ and $(\overline{p-1})\,p$ as substrings. Thus, $\mathsf{sd}_p \in \Phi$, a contradiction by Lemma 3 since $\mathsf{sh}_p \in \Phi$.

□

The following result follows from Lemma 5.

Lemma 6. *Let π be a signed permutation. If an orthodox operation on p is applicable to π, then there is no composition applicable to π containing an inverted rule on p. Similarly, if an inverted operation on p is applicable to π, then there is no composition applicable to π containing an orthodox rule on p.*

Lemma 7. *If both orthodox and inverted operations are applicable to π, then π cannot be sorted.*

Proof. Assume ϕ_p is an orthodox and ϕ_q is an inverted operation applicable on π. For inverted ϕ_q we have either

(i) $(q+1)\,\overline{q} \leq \pi$, or

(ii) $\overline{q}\,(q-1) \leq \pi$, or

(iii) $(\overline{q+1})\,(\overline{q-1}) \leq \pi$ and q is signed in π.

By Lemma 5 we cannot sort any of (i)–(iii) to an orthodox order. Thus, π cannot be sorted to an orthodox permutation.

For orthodox ϕ_p we have either

(iv) $(p-1)\,\overline{p} \leq_s \pi$, or

(v) $\overline{p}\,(p+1) \leq_s \pi$, or

(vi) $(p-1)\,(p+1) \leq \pi$ and p is unsigned in π.

By Lemma 5 we cannot sort any of (iv)–(vi) to an inverted order and so, we cannot sort π to an inverted order.

In this way, π cannot be sorted.

□

Corollary 8. *Permutation π is sortable to an orthodox order if and only if π is sortable and no inverted rule is applicable to π.*

Proof. Consider π a permutation sortable to an orthodox order and let ϕ be an operation applicable to π. If ϕ is an inverted rule, then by definition, there is p such that either $\overline{(p+1)}\, p \leq \pi$, or $(p+1)\,\overline{p} \leq \pi$, or $\overline{(p+1)}\ \overline{(p-1)} \leq \pi$. It follows then by Lemma 5 that π cannot be sorted to an orthodox permutation, a contradiction.

The reverse implication follows based on similar arguments.

□

Example 2. *(i) Consider the permutation $\pi_1 = 1\,\overline{3}\,4\,2\,\overline{5}$. This permutation is sorted to an orthodox order. Indeed, we have $\mathsf{sh}_4 \circ \mathsf{sd}_2 \circ \mathsf{sh}_3(\pi_1) = 1\,2\,3\,4\,5$. Moreover, by Lemma 1 $\overline{\pi_1}$ should be sorted to an inverted order. By Lemma 2 composition $\mathsf{sh}_4 \circ \mathsf{sd}_2 \circ \mathsf{sh}_3$ is applicable to $\overline{\pi_1}$ as well and it should sort it to an inverted order. Indeed, $\mathsf{sh}_4 \circ \mathsf{sd}_2 \circ \mathsf{sh}_3(\overline{\pi_1}) = \mathsf{sh}_4 \circ \mathsf{sd}_2 \circ \mathsf{sh}_3(5\,\overline{2}\,\overline{4}\,3\,\overline{1}) = \overline{5}\,\overline{4}\,\overline{3}\,\overline{2}\,\overline{1} = \overline{\mathsf{sh}_4 \circ \mathsf{sd}_2 \circ \mathsf{sh}_3(\pi_1)}$.*

(ii) Consider the permutations $\pi_2' = 1\,3\,2\,5\,\overline{4}$ and $\pi_2'' = 1\,3\,2\,\overline{6}\,\overline{4}\,\overline{5}$. Orthodox sd_2 and inverted sh_4 are applicable to π_2', but it is easy to see, that we can sort π_2' neither to an orthodox order nor to an inverted order. Orthodox sd_2 and inverted sd_5 are applicable to π_2'', but π_2'' can be sorted neither to an orthodox order nor to an inverted order.

(iii) Consider the permutation $\pi_3' = 1\,3\,\overline{4}\,\overline{6}\,2\,5$. Orthodox operations sd_2 and sh_3 are applicable to π_3'. By Corollary 8 if π_3' is sortable, it should be sorted to an orthodox order. Let's try to sort it: $\mathsf{sh}_3 \circ \mathsf{sd}_2(\pi_3') = 1\,2\,3\,4\,\overline{6}\,5$. Now inverted sh_5 became applicable. By using it we get permutation $1\,2\,3\,4\,\overline{6}\,\overline{5}$. This permutation cannot be sorted. Since there are no other compositions applicable to π_3', it cannot be sorted as well.

Consider the permutation $\pi_3'' = 1\,3\,\overline{4}\,5\,2\,\overline{6}$. Orthodox sd_2 and sh_3 can be applied to π_3''. We can sort π_3''. Indeed, $\mathsf{sh}_5 \circ \mathsf{sh}_3 \circ \mathsf{sd}_2(\pi_3'') = 1\,2\,3\,4\,5\,6$. Thus, π_3'' is sortable, orthodox operations are applicable and π_3'' is sorted to an orthodox order.

5 Sh-sortable permutations

We characterize in this section all signed permutations that can be sorted using only Sh operations. As it turns out, they are easy to describe since the Sh operations do not change the relative positions of the letters in the permutation.

The following result characterizes all Sh-sortable signed permutations.

Theorem 9. *A signed permutation π over Σ_n is Sh-sortable if and only if*

(i) $\|\pi\| = p\,(p+1)\ldots n\,1\ldots(p-1)$, for some $1 \leq p \leq n$ and there are r, t, $1 \leq r \leq p-1$, $p \leq t \leq n$ such that r and t are unsigned letters, or

(ii) $\|\pi\| = (p-1)\ldots 1\,n\ldots(p+1)\,p$, for some $1 \leq p \leq n$ and there are r, t, $1 \leq r \leq p-1$, $p \leq t \leq n$ such that r and t are signed letters.

In Case (i), π sorts to $p\,(p+1)\dots n\,1\dots(p-1)$, while in Case (ii), π sorts to $\overline{(p-1)}\dots\overline{1}\,\overline{n}\dots\overline{(p+1)}\,\overline{p}$.

Proof. The conditions of the theorem are clearly sufficient. Consider now a Sh-sortable permutation π. Thus, there is a composition Φ of operations in Sh such that $\Phi(\pi) = p\,(p+1)\dots n\,1\dots(p-1)$ for some $1 \le p \le n$, or $\Phi(\pi) = \overline{(p-1)}\dots\overline{1}\,\overline{n}\dots\overline{(p+1)}\,\overline{p}$. Consider the first case – the second one is symmetric with respect to inversion.

Note, that an Sh operation does not change the relative order of letters in π, but only changes one sign. Thus, it follows that $\|\pi\| = p\,(p+1)\dots n\,1\dots(p-1)$ for some $1 \le p \le n$. It is easy to see that to sort a permutation to an orthodox order by only Sh operations, it is necessary to have at least one unsigned letter in $\{p, p+1, \dots, n\}$ and at least one unsigned letter in $\{1, 2, \dots, p-1\}$. $\square$

Example 3. *(i) The permutation $\pi_1 = \overline{5}\,\overline{6}\,7\overline{8}\,\overline{1}\,2\overline{3}\,\overline{4}$ is Sh sortable and an Sh-sorting for π_1 is $\mathsf{sh}_3(\mathsf{sh}_2(\mathsf{sh}_1(\mathsf{sh}_7(\mathsf{sh}_5(\mathsf{sh}_6(\pi_1)))))) = 5\,6\,7\,8\,1\,2\,3\,4$. Note that sh_5 can be used only after sh_6 and also, sh_3 can be used only after sh_2.*

(ii) The permutation $\pi_2 = \overline{5}\,\overline{6}\,7\overline{8}\,\overline{1}\,\overline{2}\,\overline{3}\,\overline{4}$ is unsortable, since we cannot unsign 1, 2, 3 and 4.

6 Sd-Sortable Permutations

We characterize in this section the Sd-sortable permutations. Since Sd operations do not change the sign of elements, we consider only unsigned permutations. The case when all elements are signed is symmetric with respect to inversion. A crucial role in our result is played by the dependency graph of a permutation.

6.1 The dependency graph

The dependency graph describes for a unsigned permutation π the order in which orthodox Sd operations can be used in a composition applicable to π. It is in general a directed graph with self-loops.

Definition 2. *For a permutation π over Σ_n we define its dependency graph as the directed graph $G_\pi = (\Sigma_n, E)$, where*

$$\begin{aligned} E = &\{(p,q) \mid (q-1)p(q+1) \le_s \pi, 1 \le p \le n, 2 \le q \le n-1\} \cup \\ &\{(q,q) \mid (q+1)(q-1) \le_s \pi \text{ or } q = 1 \text{ or } q = n\}. \end{aligned}$$

Intuitively, an edge (p,q) in the dependency graph of a permutation says that sd_q may be used in a composition for π only after sd_p was used. A loop (q,q) means that sd_q can never be used in a composition for π. Note that G_π may also have a loop on node q if $(q-1)q(q+1) \le_s \pi$.

Example 4. *(i) The graph associated to the permutation $\pi_1 = 1\,4\,3\,6\,5\,7\,2$ is shown in Figure 3(a). It can be seen, e.g., that sd_3 can never be used in a composition applicable to π, neither can sd_5 because of the edge $(3,5)$.*

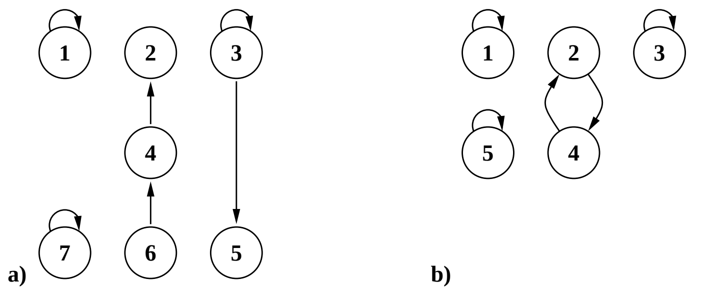

Figure 3: Dependency graphs: (a) associated to $\pi_1 = 1\,4\,3\,6\,5\,7\,2$ and (b) associated to $\pi_2 = 1\,4\,3\,2\,5$.

Also, the graph suggests that sd_6 *should be used before* sd_4 *and this one before* sd_2*. Indeed,* $\mathsf{sd}_2(\mathsf{sd}_4(\mathsf{sd}_6(\pi))) = 1\,2\,3\,4\,5\,6\,7$.

(ii) The graph associated to the permutation $\pi_2 = 1\,4\,3\,2\,5$ *is shown in Figure 3(b). Thus, the graph has a cycle with nodes* 2 *and* 4*. Indeed, to use* sd_2 *in a composition for* π_2*,* sd_4 *should be used first and the other way around.*

Lemma 10. *For any signed permutation over* Σ_n *and any* $p \in \Sigma_n$*, if* $(p+1)$ $(p-1) \leq_s \pi$*, then* sd_p *cannot be used in a composition applicable to* π*.*

Proof. Indeed, to use sd_p we need to obtain the substring $(p-1)\,(p+1)$ first. But, for this we need to use either sd_{p-1} or sd_{p+1}. However, by Lemma 3 we cannot use sd_p afterwards.

□

Lemma 11. *Let* π *be a unsigned permutation over* Σ_n *and* $G_\pi = (\Sigma_n, E)$ *its dependency graph.*

(i) If there is a path from p *to* q *in* G_π*, then in any composition where* sd_q *is used,* sd_p *is used before* sd_q*.*

(ii) If G_π *has a cycle containing* $p \in \Sigma_n$*, then* sd_p *cannot be used in any composition applicable to* π*.*

Proof. We prove claim (i) by induction along the length of paths from p to q. For a path of length 1, note that if we have an edge (p,q) in G_π, with $p \neq q$, then $(q-1)p(q+1) \leq_s \pi$. Now, sd_q can be used only after $(q-1)\,(q+1)$ is obtained and so, sd_p has to be applied before sd_q in any composition applicable to π. Assume now that the path is of length k and is presented by the sequence $(p\,p_1\,p_2\,\ldots\,p_{k-1}\,q)$. Clearly $\mathsf{sd}_{p_{k-1}}$ is applied before sd_q and by the induction hypothesis we have that sd_p is applied before $\mathsf{sd}_{p_{k-1}}$.

Claim (ii) follows from (i) and from Lemma 3. Indeed, if there is a nonempty path from p to itself, then we have either:

(a) $(p+1)\,(p-1) \leq_s \pi$, or

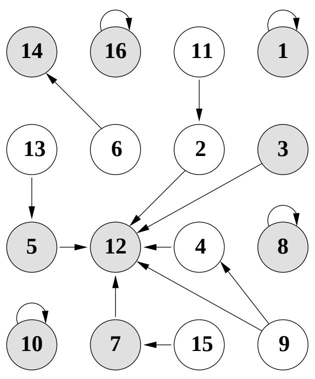

Figure 4: The dependency graph associated to $\pi = 1\,11\,3\,9\,5\,7\,2\,4\,13\,6\,15\,8\,10\,12\,14\,16$. The nodes indicated by white background are used in a sorting composition for π.

(b) $(p-1)\,p\,(p+1) \leq_s \pi$, or

(c) Neither $(p+1)\,(p-1) \leq_s \pi$ nor $(p-1)\,p\,(p+1) \leq_s \pi$, but there is a path of a length greater than 1 from p to itself in the graph.

In case (b) and (c) it follows that sd_p should be used twice in a composition applicable to π, which is impossible by Lemma 3(iii). Case (a) is proved by Lemma 10.

□

6.2 The Characterization

We characterize in this subsection the Sd-sortable permutations. We first give an example.

Example 5. *Consider the dependency graph G_π for $\pi = 1\,11\,3\,9\,5\,7\,2\,4\,13\,6\,15\,8\,10\,12\,14\,16$, shown in Figure 4. Based on Lemmas 3 and 11 we build a sorting composition Φ for π. We label all nodes p for which sd_p is used in Φ by D and the other nodes by U. Nodes labelled by D are shown with a white background in Figure 4, while nodes labelled by U are shown with a gray background.*

By Lemmas 3 and 11, operations sd_1, sd_8, sd_{10} and sd_{16} cannot be used in any composition applicable to π. Thus, $1, 8, 10, 16 \in U$. Now, if we want to use sd_2, the operation sd_{11} should be used first, since the edge $(11, 2)$ is in G_π. Thus, $2, 11 \in D$. According to Lemma 3 we cannot use sd_2 and sd_3 in the same composition, thus we label 3 by U. If we want to use sd_4, we need to use sd_9 first because of the edge $(9, 4)$. Thus, we label both 4 and 9 by D. It follows then by Lemma 3 that $5 \in U$. Then 6 can be labelled by D and then, necessarily, $7 \in U$. Note now, that if $12 \in D$, since $(3, 12)$ is an edge in G_π, then by Lemma 11(ii), $3 \in D$, which contradicts our labelling of 3. Thus, $12 \in U$. Then 13 can be labelled by D and necessarily, $14 \in U$. Also, 15 can now be labelled by D.

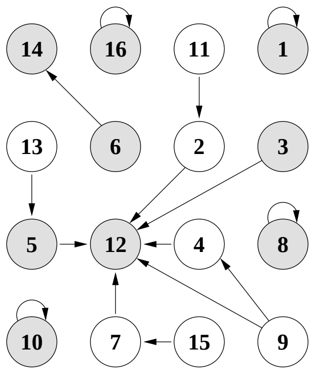

Figure 5: The dependency graph associated to $\pi = 1\,11\,3\,9\,5\,7\,2\,4\,13\,6\,15\,8\,10\,12\,14\,16$ and partition $U = \{1, 3, 5, 6, 8, 10, 12, 14, 16\}$, $D = \{2, 4, 7, 9, 11, 13, 15\}$.

In this way, we obtain $D = \{2, 4, 6, 9, 11, 13, 15\}$ and $U = \{1, 3, 5, 7, 8, 10, 12, 14, 16\}$. Note that, since elements in U do not change their relative positions in the composition Φ we are building, $\pi|_U$ has to be sorted: $\pi|_U = 1\,3\,5\,7\,8\,10\,12\,14\,16$.

Φ is a composition of the operations sd_p, with $p \in D$. The dependency graph shows the order in which these operations should be used, i.e., sd_2 can be used only after sd_{11}, sd_4 can be used only after sd_9. In this way, we can sort π by using the following sorting composition: $(\mathsf{sd}_2 \circ \mathsf{sd}_4 \circ \mathsf{sd}_6 \circ \mathsf{sd}_{15} \circ \mathsf{sd}_{13} \circ \mathsf{sd}_{11} \circ \mathsf{sd}_9)(\pi) = 1\,2\,3\,4\,5\,6\,7\,8\,9\,10\,11\,12\,13\,14\,15\,16$. Clearly, our choice of D and U is not unique. For instance, we may have chosen $D = \{2, 4, 7, 9, 11, 13, 15\}$ and $U = \{1, 3, 5, 6, 8, 10, 12, 14, 16\}$ as shown in Figure 5. Then the sorting composition is $\mathsf{sd}_2 \circ \mathsf{sd}_4 \circ \circ \mathsf{sd}_7 \circ \mathsf{sd}_{15} \circ \mathsf{sd}_{13} \circ \circ \mathsf{sd}_{11} \circ \mathsf{sd}_9$.

The following result characterizes all Sd-sortable permutations.

Theorem 12. *Let π be a unsigned permutation. Then π is Sd-sortable if and only if there exists a partition $\{1, 2, \ldots, n\} = D \cup U$, such that the following conditions are satisfied:*

(i) $\pi|_U$ is sorted;

(ii) The subgraph induced by D in G_π is acyclic;

(iii) If $(p, q) \in G_\pi$ with $q \in D$, then $p \in D$;

(iv) For any $p \in D$, $(p-1)(p+1) \leq_s \pi$;

(v) For any $p \in D$, $(p-1), (p+1) \in U$.

Proof. Consider a sortable unsigned permutation π and let $\Phi = \mathsf{sd}_{p_k} \circ \ldots \circ \mathsf{sd}_{p_1}$ be a sorting composition for π, $D = \{p_1, \ldots, p_k\}$ and $U = \Sigma_n \setminus D$.

The relative positions of integers in U are not changed throughout the sorting and so (i) follows. Since Φ can be applied to π, (ii), (iii), (v) follow from Lemmas 3 and 11. To prove (iv), consider now $p \in D$. Then $(p-1), (p+1) \in U$ and so, their relative position does not change throughout the sorting. Since $(p-1)(p+1)$ is a substring of the permutation when sd_p becomes applicable, it follows that $(p-1)(p+1) \leq_s \pi$, proving *(iv)*.

We prove the converse implication by induction on $|D|$. If $|D| = 0$, then the claim follows by (i). Let $|D| > 0$.

By (ii), D induces a directed forest in the dependency graph; let p be a source of this forest. By (iv), $(p-1)(p+1) \leq_s \pi$. If $(p-1)(p+1)$ is not a substring of π, then there is a q such that $(p-1)q(p+1) \leq_s \pi$. But then $(q,p) \in G_\pi$ and so, by (iii), $q \in D$, contradicting the choice of p as a root. Consequently, $(p-1)(p+1) \leq \pi$ and so, sd_p is applicable to π. Let $\pi' = \mathsf{sd}_p(\pi)$: then $(p-1)\,p\,(p+1) \leq \pi'$. Consider the partition $D' = D \setminus \{p\}$, $U' = U \cup \{p\}$. We claim that D' and U' satisfy conditions *(i) - (v)* for the permutation π'.

It is easy to see that $\pi'|_{U'}$ is sorted because $\pi|_U$ is sorted and $(p-1)(p+1)$ is a substring of π, proving (i).

Assume now that (iii) does not hold, i.e., there is a dependency $(r,t) \in G'_\pi$ ($(t-1)r(t+1) \leq_s \pi'$) with $r \in U'$, $t \in D'$. We claim that $(r,t) \in G_\pi$. Indeed, if this is not the case, then either $r = p$, or $t-1 = p$, or $t+1 = p$.

If $t-1 = p$, then $t = p+1 \in U \subseteq U'$ and so, $t \in D' \cap U'$; a contradiction. The case when $t+1 = p$ is analogous. Now, if $r = p$, then either $t-1 = p-1$ and thus $t = p$, or $(t-1)\,(p-1)\,(t+1) \leq_s \pi'$. The case $t = p$ is impossible since $t \in D' = D \setminus \{p\}$. Consider then the case $(t-1)(p-1)(t+1) \leq_s \pi'$ and $t-1, t+1 \neq p$. Consequently, $(t-1)(p-1)(t+1) \leq_s \pi$, i.e., $(p-1,t) \in G_\pi$. It follows from Condition (iii) for π that $p-1 \in D$, which contradicts Condition (v) for π, since $p \in D$. Consequently, (iii) holds for π'.

To prove (iv) consider $r \in D'$. Thus, $r \in D$ and so $(r-1)(r+1) \leq_s \pi$. If $r-1 = p$ ($r+1 = p$, resp.), then $r = p+1$ ($r = p-1$, resp.), i.e., $r \in U \subset U'$, which is impossible. Thus, $r-1 \neq p$ and $r+1 \neq p$ and so, $(r-1)(r+1) \leq_s \pi'$, i.e., (iv) holds.

Using a similar argument it is easy to show that for $r,t \in D'$, $(r,t) \in G'_\pi$ if and only if $(r,t) \in G_\pi$, thus proving (ii).

Condition (v) follows since $D' \subseteq D$ and $U \subseteq U'$.

Consequently, since $|D'| < |D|$, it follows by induction that π' is sortable. Then, since $\pi' = \mathsf{sd}_p(\pi)$, π is also sortable, concluding the proof. $\square$

Example 6. *Consider the permutation* $\pi = 1\,3\,8\,10\,5\,7\,2\,9\,11\,4\,6\,12$. *Its dependency graph is shown in Figure 6. Based only on this graph and using Theorem 12 we deduce a sorting composition for* π.

It follows by property (ii) that $1, 7, 11, 12 \in U$. *Then, it follows from property (iii) that* $3 \in U$. *Since* $1, 3 \in U$, *it follows from property (i) that* $2 \in D$. *Also, since* $3, 7, 11 \in U$, *it follows from property (i) that* $4, 6, 8, 10 \in D$ *and so, by property (v),* $5, 9 \in U$. *We have now a complete labelling for* G_π:

$$D = \{2,4,6,8,10\},\ U = \{1,3,5,7,9,11,12\}.$$

We represent the elements from D *as white vertices and elements from* U *as gray vertices.*

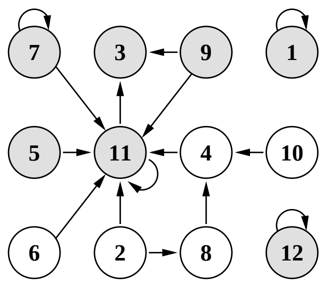

Figure 6: The dependency graph associated to $\pi = 1\,3\,8\,10\,5\,7\,2\,9\,11\,4\,6\,12$, $U = \{1, 3, 5, 7, 9, 11, 12\}$, $D = \{2, 4, 6, 8, 10\}$.

The permutation π may be sorted now by a composition of operations sd_p with $p \in D$. The dependency graph imposes the following order of operations: sd_4 after sd_8 and sd_{10}, sd_8 after sd_2. The other operations can be used in any order. For instance, we can sort π in the following way:

$$(\mathsf{sd}_4 \circ \mathsf{sd}_8 \circ \mathsf{sd}_2 \circ \mathsf{sd}_{10} \circ \mathsf{sd}_6)(\pi) = 1\,2\,3\,4\,5\,6\,7\,8\,9\,10\,11\,12,$$

but also,

$$(\mathsf{sd}_6 \circ \mathsf{sd}_4 \circ \mathsf{sd}_8 \circ \mathsf{sd}_2 \circ \mathsf{sd}_{10})(\pi) = 1\,2\,3\,4\,5\,6\,7\,8\,9\,10\,11\,12.$$

7 $\{\mathsf{Sd}, \mathsf{Sh}\}$-Sortable Permutations

We characterize in this section all signed permutations that can be sorted using our operations in $\mathsf{Sd} \cup \mathsf{Sh}$. First we give some examples.

Example 7. *(i) The signed permutations $\pi_1 = 2\,1\,4\,\overline{3}\,5$ and $\pi_2 = 1\,5\,\overline{2}\,4\,3\,6$ are not $\{\mathsf{Sd}, \mathsf{Sh}\}$-sortable. Indeed, only sh_3 can be applied to π_1, but it does not sort it, and no operation can be applied to π_2.*

(ii) The signed permutations $\pi_3 = 9\,2\,\overline{10}\,\overline{11}\,1\,5\,3\,7\,\overline{4}\,6\,8$ and $\pi_4 = 5\,4\,\overline{3}\,\overline{8}\,2\,1\,\overline{9}\,7\,\overline{6}$ are $\{\mathsf{Sd}, \mathsf{Sh}\}$-sortable:

$$(\mathsf{sh}_{10} \circ \mathsf{sh}_9 \circ \mathsf{sd}_2 \circ \mathsf{sd}_5 \circ \mathsf{sh}_3 \circ \mathsf{sd}_7)(\pi_3) = 9\,10\,11\,1\,2\,3\,4\,5\,6\,7\,8$$

and

$$(\mathsf{sh}_4 \circ \mathsf{sh}_3 \circ \mathsf{sh}_1 \circ \mathsf{sh}_2 \circ \mathsf{sd}_8 \circ \mathsf{sh}_6)(\pi_4) = \overline{5}\,\overline{4}\,\overline{3}\,\overline{2}\,\overline{1}\,\overline{9}\,\overline{8}\,\overline{7}\,\overline{6}.$$

Definition 3. *Consider a permutation π. Let $H, D \subseteq \{1, 2, \ldots, n\}$, $H \cap D = \emptyset$. The (orthodox) dependency graph $\Gamma_{\pi,H,D}$ generated by π, H and D has Σ_n as its set of vertices, while its edges are defined as follows:*

i. For $q \in D$ and some $p \in \Sigma_n$,

– *if $(q-1)\,p\,(q+1) \leq_s \|\pi\|$, then $(p, q) \in \Gamma_{\pi,H,D}$;*

– *if* $(q+1)\,(q-1) \leq_s \|\pi\|$, *then* $(q,q) \in \Gamma_{\pi,H,D}$;
– *if* $q-1, q$ *have different signs in* π, *then* $(q-2,q) \in \Gamma_{\pi,H,D}$;
– *if* $q, q+1$ *have different signs in* π, *then* $(q+1,q) \in \Gamma_{\pi,H,D}$.

ii. For $q \in H$ *and some* $p \in \Sigma_n$,

– *if* $q\,p\,(q+1) \leq_s \|\pi\|$, *then* $(p,q) \in \Gamma_{\pi,H,D}$;
– *if* $(q+1)\,q \leq_s \|\pi\|$ *or* $q\,(q+1) \leq_s \pi$, *then* $(q,q) \in \Gamma_{\pi,H,D}$;
– *if* $\overline{q}(\overline{q+1}) \leq_s \pi$, *then*

- *if* $q-1$ *is not in* H *or* $(q,q-1)$ *is an edge, then* $(q+1,q) \in \Gamma_{\pi,H,D}$,
- *else* $(q-1,q) \in \Gamma_{\pi,H,D}$.

For a composition Φ *applicable to* π, *we denote* $H_\Phi = \{p \in \Sigma_n \mid \mathsf{sh}_p \in \Phi\}$ *and* $D_\Phi = \{p \mid \mathsf{sd}_p \in \Phi\}$. *Also, we denote* $\Gamma_{\pi,\Phi} = \Gamma_{\pi,H_\Phi,D_\Phi}$.

Example 8. *Consider* $\pi = 6\,8\,10\,1\,9\,\overline{3}\,7\,4\,2\,\overline{5}$ *and let* $H = \{3,4\}$ *and* $D = \{2,7,9\}$. *The dependency graph* $G = \Gamma_{\pi,\Phi_H,\Phi_D}$, *shown in Figure 7, is built as follows.*

We represent the elements from H *as slanted vertices, the elements from* D *as white vertices and the rest of the elements as gray vertices. For each vertex* q *from* G *we have the following edges* (p,q):

– *Node* 1: *we do not have edges* $(p,1)$, *since* $1 \notin H$ *and* $1 \notin D$;

– *Node* 2: $2 \in D$, *elements* 2 *and* 3 *have different signs in* π, *thus,* $(3,2) \in G$. *Moreover, we have substring* $1\,9\,3 \leq_s \|\pi\|$ *and thus,* $(9,2) \in G$;

– *Node* 3: $3 \in H$, $3\,7\,4 \leq_s \|\pi\|$, *thus* $(7,3) \in G$;

– *Node* 4: $4 \in H$, *we have substring* $4\,2\,5 \leq_s \|\pi\|$, *then* $(2,4) \in G$;

– *Node* 5: *no predecessors for element* 5, *since* $5 \notin H$ *and* $5 \notin D$;

– *Node* 6: *no predecessors for element* 6, *since* $6 \notin H$ *and* $6 \notin D$;

– *Node* 7: $7 \in D$, $6\,8 \leq \pi$, *thus we have no edges* $(p,7)$;

– *Node* 8: $8 \notin H$ *and* $8 \notin D$, *thus we have no edges* $(p,8)$;

– *Node* 9: $9 \in D$, $8\,10 \leq \pi$, *thus we have no edges* $(p,9)$;

– *Node* 10: *no predecessors for element* 10, *since* $10 \notin H$ *and* $10 \notin D$.

Lemma 13. *If in a permutation* π *the elements* p *and* $p+1$ *are signed, for some* $p \geq 2$, *then the orthodox* Sh *operations* sh_{p-1}, sh_p *and* sh_{p+1} *are applicable in the same composition only in one of the following orders: either in* sh_{p-1}, sh_p, sh_{p+1}, *or in* sh_{p+1}, sh_p, sh_{p-1}, *where* $p-1$ *is signed.*

Proof. Consider all possible orders:

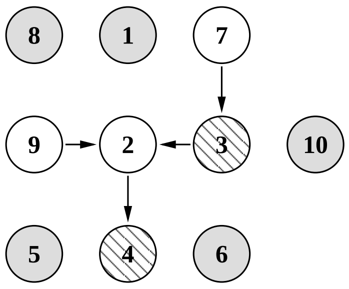

Figure 7: The dependency graph associated to $\pi = 6\,8\,10\,1\,9\,\overline{3}\,7\,4\,2\,\overline{5}$, $H = \{3, 4\}$ and $D = \{2, 7, 9\}$.

- sh_{p-1}, sh_{p+1}, sh_p or sh_{p+1}, sh_{p-1}, sh_p: By the definition of orthodox Sh operations, after application of sh_{p-1} and sh_{p+1} we get substrings $(p-1)\,p$ and $(p+1)\,(p+2)$, i.e., p and $(p+1)$ are unsigned. In order to use sh_p we should sign either p or $(p+1)$ first. This can be done either by inverted sh_{p-1} or sh_p or sh_{p+1}, i.e., some of these operations should be used twice in the same composition, which is not possible;

- sh_p, sh_{p-1}, sh_{p+1} or sh_p, sh_{p+1}, sh_{p-1}: Since p and $p+1$ are signed, sh_p cannot be used first;

- sh_{p-1}, sh_p, sh_{p+1}: After sh_{p-1} was used, we get substring $(p-1)\,p$. Then, we can use sh_p, thus element $(p+1)$ becomes unsigned, and then, we can use sh_{p+1} if $p+2$ is signed;

- sh_{p+1}, sh_p, sh_{p-1}: The operation sh_{p+1} unsigns $p+1$, then sh_p can be used and unsigns p, then sh_{p-1} can be used if $p-1$ is signed.

□

Lemma 14. *Let π be a signed permutation over Σ_n and Φ a composition applicable to π where only orthodox operations are used. Let $\Gamma_{\pi,\Phi}$ be the orthodox dependency graph associated to π and Φ. Then:*

(i) If there is a path from p to q in $\Gamma_{\pi,\Phi}$, $p \neq q$, then $\phi_p \in \Phi$ and ϕ_p is used before ϕ_q in Φ;

(ii) The dependency graph $\Gamma_{\pi,\Phi}$ is acyclic.

Proof. Let H be the set of all elements to which Sh operations are applied in Φ and let D be the set of all elements to which Sd operations are applied in Φ.

We will prove the first claim by induction on the length of the paths. For the beginning we consider paths of length 1, i.e., edges (p, q) for some $p, q \in \Sigma_n$, where $p \neq q$. By the definition of the dependency graph we have here two cases: either $q \in D$ (i.e., $\phi_q = \mathsf{sd}_q \in \Phi$) or $q \in H$ (i.e., $\phi_q = \mathsf{sh}_q \in \Phi$).

Consider $\phi_q = \mathsf{sd}_q$. Here we have one of the following subcases:

- $(q-1)\,p\,(q+1) \leq_s \|\pi\|$;
- elements $q-1$, q have different signs in π and $p = q-2$;
- elements $q+1$, q have different signs in π and $p = q+1$;
- $(q+1)\,(q-1) \leq_s \|\pi\|$ and $p = q$. This subcase is impossible since we assumed $p \neq q$;

If $(q-1)\,p\,(q+1) \leq_s \pi$, then we need to obtain the substring $(q-1)\,(q+1)$ first. Clearly, in order to obtain $(q-1)(q+1)$ we need to use sd_p first.

If $q-1$ and q are of a different sign in π, then to use sd_q, we should obtain $q-1$ and q of the same sign first. This can be done either by sh_{q-2} or by sh_{q-1} or by sh_q. By Lemma 3 the operations sh_{q-1} and sd_q or sh_q and sd_q cannot be used in the same composition. Thus, sh_{q-2} is used in Φ before sd_q.

If $q+1$ and q are of a different sign in π, then to use sd_q, we should obtain q and $q+1$ of the same sign first. This can be done either by sh_{q-1} or by sh_q or by sh_{q+1}. By Lemma 3 the operations sh_{q-1} or sh_q cannot be used with sd_q in the same composition. In this way, sh_{q+1} is used before sd_q in composition Φ.

Consider now $\phi_q = \mathsf{sh}_q$. By the definition, we have the following subcases:

- $q\,p\,(q+1) \leq_s \|\pi\|$;
- $\overline{q}\,(\overline{q+1}) \leq_s \pi$, $p = q+1$, $q-1 \notin H$ or $(q, q-1) \in \Gamma_{\pi,\Phi}$;
- $\overline{q}\,(\overline{q+1}) \leq_s \pi$, $p = q-1$, $q-1 \in H$ and $(q, q-1) \notin \Gamma_{\pi,\Phi}$;
- $(q+1)\,q \in \|\pi\|$ and $p = q$. This subcase is impossible since we assumed $p \neq q$.

If $q\,p\,(q+1) \leq_s \|\pi\|$, then we should use sd_p first to obtain either substring $q\,(\overline{q+1})$ or $\overline{q}\,(q+1)$.

If $\overline{q}\,(\overline{q+1}) \leq_s \pi$, then to use sh_q either q or $q+1$ should be unsigned first. This can be done either by sh_{q-1} or by sh_{q+1}. We will prove by induction that, if $p = q+1$, with $q-1 \notin H$ or $(q, q-1) \in \Gamma_{\pi,\Phi}$, then sh_{q+1} is used before sh_q in Φ.

Indeed, if $q-1 \notin H$, then we can obtain the substring $\overline{q}\,(q+1)$ only after sh_{q+1} is used. Now, assume that $q-k-1 \notin H$, $q-k, q-k+1, \ldots, q-1, q \in H$ and $(q, q-1), (q-1, q-2), \ldots, (q-k+1, q-k) \in \Gamma_{\pi,\Phi}$. Then, sh_{q-k} is used after sh_{q-k+1}, sh_{q-k+1} is used after $\mathsf{sh}_{q-k+2}, \ldots, \mathsf{sh}_{q-2}$ is used after sh_{q-1} and sh_{q-1} is used after sh_q. We show that sh_q can be used only after sh_{q+1}. Indeed, since sh_{q-1} is used after sh_q, we cannot unsign q before sh_q is used. Then, we unsign $q+1$ by sh_{q+1} first, thus, $p = q+1$.

If $q-1 \in H$, $p = q-1$ and $(q, q-1) \notin \Gamma_{\pi,\Phi}$, then we will show that sh_{q-1} is used first. Here we have one of two cases: either $(q-1)$ is unsigned or $(q-1)$ is signed. If $(q-1)$ is unsigned, it is clear, that sh_{q-1} can be used before sh_q. Moreover, if $(q-1)$ is unsigned, sh_q cannot be used before sh_{q-1}. Now, we will prove, that if $(q-1)$ is signed, $q-1 \in H$ and $(q, q-1) \notin \Gamma_{\pi,\Phi}$, then sh_{q-1} is used before sh_q. We will prove this by induction. Assume, we have subsequence $(q-2)\,(\overline{q-1})\,\overline{q}$. By Lemma 13 sh_{q-2} is used before sh_{q-1} and sh_{q-1} is used before sh_q, no other orders are possible. Now, assume we have subsequence $(q-k)\,(\overline{q-k+1})\,(\overline{q-k+2})\,\ldots\,(\overline{q-1})\,\overline{q}$, edges $(q-k, q-k+1), (q-k+1, q-k+2), \ldots, (q-2, q-1), (q-1, q) \in \Gamma_{\pi,\Phi}$ and sh_{q-k} is used before sh_{q-k+1}, sh_{q-k+1} is used before sh_{q-k+2}, ..., sh_{q-2} is used before sh_{q-1}, no other orders are applicable. We will show, that sh_{q-1} is used before sh_q and not viceversa. By Lemma 13 we can use Sh operations either in order sh_{q-2}, sh_{q-1}, sh_q or in order sh_q, sh_{q-1}, sh_{q-2}. Since sh_{q-1} cannot be used before sh_{q-2} by our assumption, sh_q is used after sh_{q-1} in Φ.

Assume now, that if we have a path from p to q' in graph $\Gamma_{\pi,\Phi}$ of a length at most n, then ϕ_p is used before $\phi_{q'}$ in a composition Φ. Assume, we have a path from p to q via element q' and $(q', q) \in \Gamma_{\pi,\Phi}$. As we have shown above, ϕ_q can be used only after $\phi_{q'}$. Since, by our assumption $\phi_{q'}$ can be used only after ϕ_p, then ϕ_q is used after ϕ_p.

To prove the second claim of the lemma, assume on the contrary that $\Gamma_{\pi,\Phi}$ has a cycle.

If the cycle has length at least two, then the claim follows from part (i) and Lemma 3.

Assume now that $\Gamma_{\pi,\Phi}$ has a loop: $(p, p) \in \Gamma_{\pi,\Phi}$, for some p. We have the following two cases:

(a) $p \in D_\Phi$, i.e., $\mathsf{sd}_p \in \Phi$. Then by definition, $(p-1)\,p\,(p+1) \leq_s \|\pi\|$, or $(p+1)\,(p-1) \leq_s \|\pi\|$. It is easy to see that in the first case, sd_p cannot be used through ϕ, a contradiction. In the second case, for sd_p to become applicable, we need to obtain the substring $(p-1)\,(p+1)$, i.e., either sd_{p-1}, or sd_{p+1} should be used in Φ before sd_p, a contradiction by Lemma 3.

(b) $p \in H_\Phi$, i.e., $\mathsf{sh}_p \in \Phi$. Then it follows from the definition that $(p+1)\,p \leq_s \|\pi\|$, or $p\,(p+1) \leq_s \pi$. In the first case, in order to use (orthodox) sh_p, we first must obtain substring $\overline{p}\,(p+1)$ or $p\,(\overline{p+1})$, i.e., we need to use either sd_p, or sd_{p+1} before sh_p. This is impossible, see Lemma 3. In the second case, either p, or $p+1$ needs to be signed before we can apply sh_p. Thus, sh_{p-1}, sh_p, or sh_{p+1} should be used in Φ before sh_p. This is impossible by Lemma 3.

□

Lemma 15. *Let π be a signed permutation, $\Phi = \phi_k \circ \ldots \circ \phi_1$ a composition applicable to π where all operations are orthodox. Let also $\Phi' = \phi'_k \circ \ldots \circ \phi'_1$, where $\phi'_i = \phi_i$, if $\phi_i \in SD$ and $\phi'_i = \mathsf{id}$ otherwise. Then $\|\Phi(\pi)\| = \Phi'(\|\pi\|)$.*

Proof. If $k = 1$, then either $\Phi = \mathsf{sd}_p$ or $\Phi = \mathsf{sh}_p$. In the former case $\Phi' = \mathsf{sd}_p$. Clearly, $\|\mathsf{sd}_p(\pi)\| = \mathsf{sd}_p(\|\pi\|)$, since p, $p-1$ and $p+1$ are not signed in the permutation and $\|.\|$ does not change the relative positions of letters. In the case when $\Phi = \mathsf{sh}_p$, $\Phi' = \mathsf{id}$. Then $\|\Phi(\pi)\| = \Phi'(\|\pi\|)$.

If $k > 1$, then $\pi' = (\phi_{k-1} \circ \ldots \circ \phi_1)(\pi)$ and by inductive assumption $\|\pi'\| = (\phi'_{k-1} \circ \ldots \circ \phi'_1)(\|\pi\|)$. Now, if $\phi_k = \mathsf{sd}_p$, then $\phi'_k = \mathsf{sd}_p$, if $\phi_k = \mathsf{sh}_p$, then $\phi'_k = \mathsf{id}$.

In both cases we have that $\|\Phi(\pi)\| = \|\phi_k(\pi')\| = \phi'_k(\|\pi'\|) = \phi'_k(\|(\phi_{k-1} \circ \ldots \circ \phi_1)(\pi)\|) = \phi'_k((\phi'_{k-1} \circ \ldots \circ \phi'_1)(\|\pi\|)) = \Phi'(\|\pi\|)$.

□

The following theorem gives the main result of this section.

Theorem 16. *A permutation π is $\{\mathsf{Sh}, \mathsf{Sd}\}$-sortable to an orthodox order if and only if there is a partition $\{1, 2, \ldots, n\} = D \cup H \cup U$ such that the following conditions are satisfied:*

(i) For any $p \in D$, p is unsigned in π;

(ii) H sorts $\pi\ |_{H\cup U}$ to an orthodox order;

(iii) D sorts $\|\pi\|$;

(iv) The subgraph of $\Gamma_{\pi,H,D}$ induced by $H \cup D$ is acyclic.

Proof. We prove first that the conditions of the theorem are necessary. Let π be a signed permutation sorted by the composition Φ to an orthodox order. Let $H = H_\Phi = \{p \mid \mathsf{sh}_p \in \Phi\}$, $D = D_\Phi = \{p \mid \mathsf{sd}_p \in \Phi\}$ and $U = U_\Phi = \Sigma_n \setminus (H \cup D)$. Then (i) follows from the fact, that p can be unsigned either by sh_{p-1} or by sh_p, but by Lemma 3 sd_p cannot be used in the same composition either with sh_{p-1} or with sh_p. Property (iii) follows from Lemma 15, and (iv) follows by Lemma 14.

Condition (ii) we will prove by induction $|D|$. If $|D| = 0$ then the claim follows directly from the fact that π is sorted by Φ. Assume now, that π is sorted by a composition Φ where $|D| = k$. Consider a permutation π' to which sd_p is applicable for some $p \in U$ and $\pi = \mathsf{sd}_p(\pi')$. Then, π' is sorted by the composition $\Phi' = \Phi \circ \mathsf{sd}_p$. Clearly, $H' = H_{\Phi'} = H$, $D' = D_{\Phi'} = D \cup \{p\}$ and $U' = U_{\Phi'} = U \setminus \{p\}$. We claim that H' sorts $\pi'\ |_{H'\cup U'}$. Indeed, since $\mathsf{sd}_p \in \Phi'$, then $\mathsf{sh}_{p-1} \notin \Phi'$ by Lemma 3 and of course $\mathsf{sh}_{p-1} \notin \Phi$. Since neither sh_{p-1} nor sh_p are used in Φ, the element p is not needed by Sh operations from H and so, $\pi|_{H\cup U\setminus\{p\}}$ is sorted by H. But, $H \cup U \setminus \{p\} = H' \cup U'$ and thus, $\pi|_{H\cup U\setminus\{p\}} = \pi'_{H'\cup U'}$. Since $H' = H$, H' sorts $\pi'_{H'\cup U'}$.

To prove the reverse implication, consider now a permutation π and a partition $\{1, 2, \ldots, n\} = D \cup H \cup U$ satisfying conditions (i)-(iv) of the theorem. We prove the claim by induction on $|D \cup H|$. If $|D \cup H| = 0$, then the claim follows from (ii). We will show that, if conditions (i)–(iv) are satisfied for a partition $H \cup D \cup U$ such that $|D \cup H| \geq 1$, then we can always apply at least an operation ψ to π and we can choose a new partition $H' \cup D' \cup U'$ satisfying (i)–(iv) for permutation $\pi' = \psi(\pi)$ and graph $\Gamma_{\pi',H',D'}$.

Condition (iv) implies that the subgraph of Γ_{H_Φ,D_Φ} induced by $H \cup D$ is a directed forest. Let p be a source of it. By the definition of the dependency graph we have either substring $\overline{p}\,(p+1)$ or $p\,(\overline{p+1})$ or $(p-1)\,(p+1)$ and p is unsigned in π, i.e., either sh_p or sd_p is applicable to π. Consider both cases:

(a) $\mathsf{sh}_\mathbf{p} \in \mathbf{\Phi}$. Then, by the definition of the dependency graph $p \in H$ and $\pi' = \mathsf{sh}_p(\pi) = \pi_1\, p\, (p+1)\, \pi_2$. Let $H' = H \setminus \{p\}$, $D' = D$, $U' = U \cup \{p\}$.

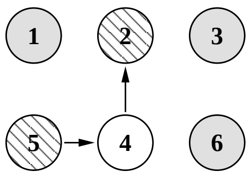

Figure 8: The dependency graph $\Gamma_{\mathsf{sh}_2 \circ \mathsf{sh}_5, \mathsf{sd}_4}$ associated to $\pi = \overline{2}\,4\,3\,\overline{5}\,6\,1$.

We claim that H', D', U' satisfy conditions (i)-(iv) of the theorem for permutation π'.

Indeed, (i) and (iii) are obvious. To prove (ii), note that $\pi|_{H \cup U}$ can be sorted by a composition of operations sh_q, $q \in H$, where sh_p can be used first. Then (ii) follows since $\pi'|_{H' \cup U'} = \mathsf{sh}_p(\pi|_{H \cup U})$. Condition (iv) also follows since $\Gamma_{\pi', H', D'}$ is a subgraph of $\Gamma_{\pi, H, D}$ and $H' \cup D' \subseteq H \cup D$.

(b) $\mathsf{sd}_\mathbf{p} \in \boldsymbol{\Phi}$. Then, by the definition of the dependency graph $p \in D$ and $\pi = \pi_1(p-1)(p+1)\pi_2 p \pi_3$ or $\pi = \pi_1 p \pi_2 (p-1)(p+1)\pi_3$. Let $H' = H$, $D' = D \setminus \{p\}$, $U' = U \cup \{p\}$. We claim that H', D', U' satisfy condition (i)-(iv) for π'.

Conditions (i) and (ii) are obvious. Condition (iii) follows by noting that $\|\pi\|$ can be sorted by a composition of operations sd_q, $q \in D$, where sd_p can be used first. Condition (iv) follows since graph $\Gamma_{\pi', H', D'}$ is a subgraph of $\Gamma_{\pi, H, D}$ and $H' \cup D' \subseteq H \cup D$.

In this way we proved the reverse implication and the theorem follows.

□

Example 9. *Let $\pi = \overline{2}\,4\,3\,\overline{5}\,6\,1$. We build a sorting composition for π based on Theorem 16. Consider $H = \{2, 5\}$. Clearly, $\|\pi\| = 2\,4\,3\,5\,6\,1$ is sorted by using sd_4. Then let $D = \{4\}$ and $U = \{1, 3, 6\}$. We verify now conditions of Theorem 16. Consider $\pi\ |_{H \cup U} = \overline{2}\,3\,\overline{5}\,6\,1$. Then $\mathsf{sh}_2(\mathsf{sh}_5(\pi\ |_{H \cup U})) = 2\,3\,5\,6\,1$, a (circularly) sorted string. Graph $\Gamma_{\mathsf{sh}_2 \circ \mathsf{sh}_5, \mathsf{sd}_4}$ is shown in Figure 8, where elements from H are shown as slanted vertices, elements from D are shown as white vertices and elements from U are shown as gray vertices. Clearly, $H \cup D$ induces an acyclic subgraph in $\Gamma_{\mathsf{sh}_2 \circ \mathsf{sh}_5, \mathsf{sd}_4}$. Thus, by Theorem 16, π is sortable and a sorting composition should be obtained by combining $\mathsf{sh}_2 \circ \mathsf{sh}_5$ and sd_4 as indicated by the graph. Since $(4, 2)$ is an edge in the graph, it follows that sd_4 must be used before sh_2. Also, since $(5, 4)$ is an edge, it follows that sh_5 must be used before sd_4. Consequently, $\mathsf{sh}_2 \circ \mathsf{sd}_4 \circ \mathsf{sh}_5$ must be a sorting composition for π. Indeed, $\mathsf{sh}_2(\mathsf{sd}_4(\mathsf{sh}_5(\pi))) = 2\,3\,4\,5\,6\,1$, a sorted permutation.*

Example 10. *Let $\pi = 2\,1\,4\,3\,7\,\overline{5}\,9\,6\,8\,\overline{10}\,11$. We build a sorting composition for π based on Theorem 16. Consider $H = \{5, 10\}$. The unsigned permutation $\|\pi\| = 2\,1\,4\,3\,7\,5\,9\,6\,8\,10\,11$ can be sorted by $\mathsf{sd}_2 \circ \mathsf{sd}_4 \circ \mathsf{sd}_9 \circ \mathsf{sd}_7$, thus $D = \{2, 4, 7, 9\}$. Set $U = \{1, 3, 6, 8, 11\}$. The dependency graph G associated to π and $H \cup D$ is shown in Figure 9. Clearly, permutation $\pi|_{H \cup U} = 1\,3\,\overline{5}\,6\,8\,\overline{10}\,11$ can be sorted to*

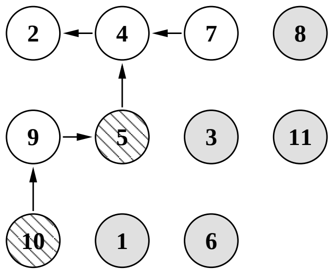

Figure 9: The dependency graph associated to $\pi = 2\,1\,4\,3\,7\,\overline{5}\,9\,6\,8\,\overline{10}\,11$ and $H = \{5, 10\}$, $D = \{2, 4, 7, 9\}$.

cyclically sorted permutation $1\,3\,5\,6\,8\,10\,11$ *by using* sh_5 *and* sh_{10}. *Also,* $H \cup D$ *induces an acyclic subgraph in* G. *It follows then that* π *is sortable. Indeed, a sorting composition, as suggested by* G, *is* $\mathsf{sd}_2 \circ \mathsf{sd}_4 \circ \mathsf{sd}_7 \circ \mathsf{sh}_5 \circ \mathsf{sd}_9 \circ \mathsf{sh}_{10}$. *Another sorting composition is* $\mathsf{sd}_2 \circ \mathsf{sd}_4 \circ \mathsf{sh}_5 \circ \mathsf{sd}_9 \circ \mathsf{sd}_7 \circ \mathsf{sh}_{10}$.

Example 11. *Let* $\pi = \overline{3}\,\overline{1}\,\overline{2}$. *Clearly, the only sorting composition for* π *is* $\mathsf{sd}_2(\pi) = \overline{3}\,\overline{2}\,\overline{1}$. *Note that the (orthodox) graph* $\Gamma_{\pi,\emptyset,\{2\}}$ *has a loop on node* 2 *but this does not contradict Theorem 16, since* π *sorts to an inverted order. However,* $\overline{\pi} = 2\,1\,3$ *and* $\Gamma_{\overline{\pi},\emptyset,\{2\}}$ *is a discrete graph. According to Theorem 16,* $\overline{\pi}$ *is sortable to an orthodox permutation.*

8 Discussion

We considered in this paper a mathematical model for the so-called simple operations for gene assembly in ciliates. The simple operations were defined so that the DNA sequence that they manipulate is minimal: only one MDS is affected. We considered in this paper only the case where this MDS is always micronuclear. Recall however that the ld-operation (that we ignored in our abstraction) may combine two consecutive MDSs $M_p\ M_{p+1}$ into a bigger composite MDS $M_{p,p+1}$. Consequently, if the MDS affected by simple hi or simple dlad is allowed to be composite, then the mathematical model needs to be slightly reformulated. This approach has been considered in [18] and somewhat surprisingly, it leads to very different results. While in the approach presented in the paper, there are permutations with both sorting compositions and non-sorting compositions leading to unsortable permutations, see Example 1(iv) and 1(v), it turns out that in the framework of [18], a permutation has only sorting or only non-sorting compositions. Moreover, all those compositions have essentially the same "structure".

The gene structure and the gene assembly may be studied on three levels of abstraction: as (sorting of) signed permutations, as (reductions of) signed double occurrence strings, and as (reduction of) signed overlap graphs, see [6]. The molecular model of simple operations illustrated in Figures 1 and 2 has been formulated in [11] both on the level of permutations, and on that of strings. Translating the model to overlap graphs seems difficult: the overlap graphs do not represent the linear distance between pointers, which is the main ingredient in the molecular model of simple hi and dlad. We suggest however that defining a minimal graph reduction model is possible: consider the graph-based hi operation (often called gpr, see [6]) applicable only to vertices with at most one neighbour in the graph, as well as the graph-based dlad operation (often called gdr, see [6]) applicable only to adjacent vertices having the same neighbourhood. It is unclear how this "simple" graph-based model relates to the other two abstractions of the simple model for gene assembly.

Acknowledgments The authors gratefully acknowledge support by Academy of Finland (Tero Harju – project 39802, Ion Petre – project 108421, Vladimir Rogojin – project 203667) and NSF (Grzegorz Rozenberg – grant 0121422).

Vladimir Rogojin is on leave of absence from Institute of Mathematics and Computer Science of Academy of Sciences of Moldova, Chisinau MD-2028 Moldova.

References

[1] Berman, P., and Hannenhalli, S., Fast sorting by reversals. *Combinatorial Pattern Matching, Lecture Notes in Comput. Sci.* **1072** (1996) 168 – 185.

[2] Caprara, A., Sorting by reversals is difficult. In S. Istrail, P. Pevzner and M. Waterman (eds.) *Proceedings of the 1st Annual International Conference on Computational Molecular Biology* (1997) 75 – 83.

[3] Cavalcanti, A., Clarke, T.H., Landweber, L., MDS_IES_DB: a database of macronuclear and micronuclear genes in spirotrichous ciliates. *Nucleic Acids Research* **33** (2005) 396 – 398.

[4] Chang, W.J., Bryson, P.D., Liang, H., Shin, M.K., Landweber, L., The evolutionary origin of a complex scrambled gene. PNAS 102(42) (2005) 15149 - 15154.

[5] Chang, W.J., Kuo, S., Landweber, L., A new scrambled gene in the ciliate Uroleptus. Gene (2006), to appear.

[6] Ehrenfeucht, A., Harju, T., Petre, I., Prescott, D. M., and Rozenberg, G., *Computation in Living Cells: Gene Assembly in Ciliates*, Springer (2003).

[7] Ehrenfeucht, A., Petre, I., Prescott, D. M., and Rozenberg, G., Universal and simple operations for gene assembly in ciliates. In: V. Mitrana and C. Martin-Vide (eds.) *Words, Sequences, Languages: Where Computer Science, Biology and Linguistics Meet*, Kluwer Academic, Dortrecht, (2001) 329 – 342.

[8] Ehrenfeucht, A., Prescott, D. M., and Rozenberg, G., Computational aspects of gene (un)scrambling in ciliates. In: L. F. Landweber, E. Winfree (eds.) *Evolution as Computation*, Springer, Berlin, Heidelberg, New York (2001) 216 – 256.

[9] Hannenhalli, S., and Pevzner, P. A., Transforming cabbage into turnip (Polynomial algorithm for sorting signed permutations by reversals). In: *Proceedings of the 27th Annual ACM Symposium on Theory of Computing* (1995) 178 – 189.

[10] Harju, T., Petre, I., Rogojin, V., and Rozenberg, G., Simple operations for gene assembly. In: A. Carbone, N. A. Pierce (eds.) *DNA Computing: 11th International Workshop on DNA Computing*, Lecture Notes in Comput. Sci. **3892** (2006), 96 – 111.

[11] Harju, T., Petre, I., and Rozenberg, G., Modelling simple operations for gene assembly, In: Junghuei Chen, Natasha Jonoska, Grzegorz Rozenberg (Eds), *Nanotechnology: Science and Computation*, 361 – 376, Springer, (2006).

[12] Jahn, C. L., and Klobutcher, L. A., Genome remodeilng in ciliated protozoa. *Ann. Rev. Microbiol.* **56** (2000), 489 – 520.

[13] Kaplan, H., Shamir, R., and Tarjan, R. E., A faster and simpler algorithm for sorting signed permutations by reversals. *SIAM J. Comput.* **29** (1999) 880 – 892.

[14] Kari, L., and Landweber, L. F., Computational power of gene rearrangement. In: E. Winfree and D. K. Gifford (eds.) *Proceedings of DNA Bases Computers, V* American Mathematical Society (1999) 207 – 216.

[15] Landweber, L.F., Kuo, T.-C., Curtis, E.A., Evolution and assembly of an extremely scrambled gene. Proc. Natl. Acad. Sci. 97(7) (2000), 3298 - 3303.

[16] Landweber, L. F., and Kari, L., The evolution of cellular computing: Nature's solution to a computational problem. In: *Proceedings of the 4th DIMACS Meeting on DNA-Based Computers*, Philadelphia, PA (1998) 3 – 15.

[17] Landweber, L. F., and Kari, L., Universal molecular computation in ciliates. In: L. F. Landweber and E. Winfree (eds.) *Evolution as Computation*, Springer, Berlin Heidelberg New York (2002).

[18] Langille, M., and Petre, I., Simple gene assembly is deterministic. *Fundamenta Informaticae*, IOS Press (2006), to appear.

[19] Prescott, D. M., The DNA of ciliated protozoa. *Microbiol. Rev.* **58**(2) (1994) 233 – 267.

[20] Prescott, D. M., Genome gymnastics: unique modes of DNA evolution and processing in ciliates. *Nat. Rev. Genet.* 1(3) (2000) 191 – 198.

[21] Prescott, D.M., DNA manipulations in ciliates. In: W.Brauer, H.Ehrig, J.Jarhumäki, A.Salomaa (Eds.) *Formal and Natural Computing*, LNCS 2300, Springer (2002) 394 - 417.

[22] Prescott, D. M., Ehrenfeucht, A., and Rozenberg, G., Molecular operations for DNA processing in hypotrichous ciliates. *Europ. J. Protistology* **37** (2001) 241 – 260.

[23] Swanton, M.T., Heumann, J.M., Prescott, D.M., Genesized DNA molecules of the macronuclei in three species of hypotrichs: size distribution and absence of nicks. Chromosoma 77 (1980) 217 - 227.

[24] West, D. B., *Introduction to Graph Theory*, Prentice Hall, Upper Saddle River, NJ (1996)

[25] Yao, M.C., Fuller, P., Xi, X., Programmed DNA Deletion As an RNA-Guided System of Genome Defense, Science 300 (2003) 1581 - 1584.

ISBN 952-12-1701-4
ISSN 1239-1891