Computational Processes in Living Cells: Gene Assembly in Ciliates

Tero Harju[1] and Grzegorz Rozenberg[2]

[1] Department of Mathematics, University of Turku, FIN-20014 Turku Finland. email: harju@utu.fi
[2] Leiden Institute for Advanced Computer Science, Leiden University Niels Bohrweg 1, 2333 CA Leiden, the Netherlands, and Department of Computer Science, University of Colorado at Boulder Boulder, Co 80309-0347, USA. email: rozenber@liacs.nl

Abstract. One of the most complex DNA processing in nature known to us is carried out by ciliates during the sexual reproduction when their micronuclear genome is transformed to the macronuclear genome. This process of gene assembly is intriguing and captivating also from the computational point of view. We investigate here three intramolecular molecular operations (*ld, hi*, and *dlad*) postulated to accomplish gene assembly. The formal models for these operations are formulated on three different abstraction levels: MDS descriptors, legals strings and overlap graphs. In general both legal strings and overlap graphs contain strings and graphs that do not model any micronuclear gene. After a short survey of gene assembly we study the problem of recognizing whether a general legal string or a general overlap graph is a formalization of a micronuclear gene.

"One of the oldest forms of life on Earth has been revealed as a natural born computer programmer."
BBC 10th September 2001

1 Introduction

Ciliates are complex single-cell organisms; around 8,000 species are known by today, but it is generally believed that the actual number of species is much higher than this. Ciliates are about two billion years old, and they live in almost every environment containing water including oceans, lakes, ponds, soils, and rivers. They are characterized by their cilia (Latin *cilium*, eyelash) – tiny hairs used for moving around as well as for moving food (e.g., bacteria or algae) towards mouth opening. Ciliates possess a unique feature of *nuclear dualism*: they have two, functionally different, types of nucleus – the *macronucleus*, which provides for the RNA transcripts needed for the cell to function, and the *micronucleus*, which is mostly dormant – activated only during sexual reproduction.

In *stichotrichs ciliates*, that are considered in this paper, micronucleus and macronucleus differ from each other drastically both on global and local levels.

A micronucleus has about 100 chromosomes; each of them containing very long, hundreds of thousands base pairs (bp), DNA molecule. Micronuclear genes are scattered along these molecules with long stretches of spacer DNA separating them. On the other hand, the macronuclear genome consists of short chromosomes (on average about 2000 base pairs) with the shortest of these about 200 bps – shorter than any other known DNA molecule occurring in nature. As for the local level, the micronuclear genes are not functional and each of them they consists of segments of the macronuclear version of this gene (these segments are called *macronuclear destined segments* or *MDSs*) separated by noncoding segments of DNA (called *internal eliminated segments* or *IESs*).

During the sexual reproduction process a macronucleus develops from a micronucleus. This transformation is the most involved DNA processing known in living cells. During the whole process of gene assembly about 100,000 IESs are excised, and MDSs are ligated to form competent genes. One of the amazing computational features of gene assembly is the fact that ciliates implement this this process using one of the standard data structures of computer science – *linked lists*.

Computational aspects of gene assembly were was first investigated by Landweber and Kari [10, 11]. The focus of their studies is the computational power (in the sense of theoretical computer science) of molecular operations used in gene assembly. A different model was proposed by Ehrenfeucht, Prescott and Rozenberg [9, 19, 20]. Their system of operations for gene assembly is based on three molecular operations: *ld*, *hi*, and *dlad* (see [19]). It was proved by Ehrenfeucht, Petre, Prescott and Rozenberg [7] that every micronuclear gene can be assembled using the three postulated molecular operations. The major difference between the two models for molecular operations is that the model of Landweber and Kari is based on *intermolecular* operations while the other model is based on *intramolecular* operations.

Three abstraction levels of formalizing the three molecular operations *ld*, *hi* and *dlad* were considered in the literature: MDS descriptors, legal strings and overlap graphs. While the MDS descriptors give a rather faithful description of the micronuclear genes, the legal strings and the overlap graphs are more general in nature, that is, there exist legal strings and overlap graphs that do not correspond to any micronuclear gene. In this paper, after surveying the basic notions and results for the operations of gene assembly, we shall study the problem of recognizing when a general legal string, or a general overlap graph, is a formalization of a micronuclear gene.

For problems and results concerning formal aspects of gene assembly we refer to [4] (for characterizations of micronuclear MDS/IES patterns that can be assembled using various subsets of the three molecular operations), [5] (for general graph theoretic method for modelling and analysing gene assembly), and [8] (for invariant properties of the assembly process). For background on DNA molecules, we refer the reader to [12] and [13].

2 Legal strings and overlap graphs

Let Σ be a (finite or infinite) alphabet, and let $\overline{\Sigma} = \{\overline{a} \mid a \in \Sigma\}$ be a copy of Σ such that $\Sigma \cap \overline{\Sigma} = \emptyset$. We also write

$$\Sigma^{\maltese} = (\Sigma \cup \overline{\Sigma})^*$$

for the set of all strings over $\Sigma \cup \overline{\Sigma}$. Each $v \in \Sigma^{\maltese}$ is a *signed string* over Σ. The signing $a \mapsto \overline{a}$ can be extended as follows. For each $a \in \Sigma$, let $\overline{\overline{a}} = a$, and for a nonempty signed string $u = a_1 a_2 \dots a_n \in \Sigma^{\maltese}$, where $a_i \in \Sigma \cup \overline{\Sigma}$ for each i, let the *inversion* of u be $\overline{u} = \overline{a}_n \overline{a}_{n-1} \dots \overline{a}_1 \in \Sigma^{\maltese}$. Let also $u^R = a_n a_{n-1} \dots a_1$ and $u^C = \overline{a}_1 \overline{a}_2 \dots \overline{a}_n$ be the *reversal* and the *complementation* of u, respectively. It is then clear that $\overline{u} = (u^R)^C = (u^C)^R$.

A letter $a \in \Sigma \cup \overline{\Sigma}$ *occurs* in v, if either a or $\overline{a}$ is a substring of v. Let $\mathrm{dom}(v) = \{a \in \Sigma \mid a \text{ occurs in } v\}$ be the *domain* of v.

Let Σ and Γ be two alphabets. A function $\tau\colon \Sigma^{\maltese} \to \Gamma^{\maltese}$ is a *morphism*, if $\tau(uv) = \tau(u)\tau(v)$ for all $u, v \in \Sigma^{\maltese}$, and τ is a *substitution*, if, moreover, $\overline{\tau(u)} = \tau(\overline{u})$ for all $u, v \in \Sigma^{\maltese}$. Note that the images $\tau(a)$ for the letters $a \in \Sigma$ determine the substitution τ. Two strings $u \in \Sigma^{\maltese}$ and $v \in \Gamma^{\maltese}$ are *isomorphic*, if $\tau(u) = v$ for an injective substitution $\tau\colon \Sigma^{\maltese} \to \Gamma^{\maltese}$ such that $\tau(\Sigma) \subseteq \Gamma$. Let ξ be the morphism such that for all $a \in \Sigma$,

$$\xi(a) = a = \xi(\overline{a})\,.$$

A signed string $v \in \Sigma^{\maltese}$ is a *signing* of a string $u \in \Sigma^*$, if $\xi(v) = u$.

Example 1. The signed string $u = 43\overline{3}\overline{5}45 \in \{3,4,5\}^{\maltese}$ is isomorphic to $v = 23\overline{3}\overline{4}24$. The isomorphism τ is defined in this example by $\tau(4) = 2$, $\tau(3) = 3$ and $\tau(5) = 4$. Also, u is a signing of the string 433545. □

Let $v = a_1 a_2 \dots a_n \in \Sigma^*$. Then a string $u \in \Sigma^*$ is a *permutation* of v, if there exists a permutation $(i_1\, i_2 \dots i_n)$ of $\{1, 2, \dots, n\}$ such that $u = a_{i_1} a_{i_2} \dots a_{i_n}$. Moreover, a signing of a permutation of v is said to be a *signed permutation of* v. A string $v \in \Sigma^*$ is a *double occurrence string*, if every letter $a \in \mathrm{dom}(v)$ occurs exactly twice in v. A signing of a nonempty double occurrence string is called a *legal string*. If a legal string $u \in \Sigma^{\maltese}$ contains one occurrence of $a \in \Sigma$ and one occurrence of $\overline{a}$, then a is said to be *positive* in u; otherwise, a is *negative* in u.

Example 2. In the legal string $u = 2\,4\,3\,\overline{2}\,\overline{5}\,3\,4\,5$ letters 2 and 5 are positive while 3 and 4 are negative. The string $w = 2\,4\,3\,\overline{2}\,\overline{5}\,3\,5$ is not legal, since it has only one occurrence of 4. □

Let $u = a_1 a_2 \dots a_n \in \Sigma^{\maltese}$ be a legal string over Σ, where $a_i \in \Sigma \cup \overline{\Sigma}$ for each i. Then for each $a \in \mathrm{dom}(u)$, there are indices i and j with $1 \le i < j \le n$ such that $\xi(a_i) = a = \xi(a_j)$. The substring

$$u_{(a)} = a_i a_{i+1} \dots a_j$$

is called the *a-interval* of u. Two different letters $a, b \in \Sigma$ are said to *overlap in* u, if the a-interval and the b-interval of u overlap: if $u_{(a)} = a_{i_1} \dots a_{j_1}$ and $u_{(b)} = a_{i_2} \dots a_{j_2}$, then either $i_1 < i_2 < j_1 < j_2$ or $i_2 < i_1 < j_2 < j_1$. Moreover, for each letter a, we denote by $O_u(a)$ ($O_u^+(a)$, $O_u^-(a)$, resp.) the set of letters (positive, negative, resp.) overlapping with a in u. For technical reasons, it is convenient to include a in $O_u(a)$: if a is positive in u, then $a \in O_u^+(a)$, and if a is negative in u, then $a \in O_u^-(a)$.

Example 3. The 2-interval $u_{(2)} = 2\,4\,3\,5\,3\,\overline{2}$ of the string $u = 2\,4\,3\,5\,3\,\overline{2}\,\overline{6}\,\overline{5}\,7\,4\,6\,7$ contains only one occurrence of 4 and 5, but two or no occurrences of $3, 6$ and 7. Therefore 2 overlaps with 4 and 5 but not with $3, 6$ and 7, and so $O_u(2) = \{2, 4, 5\}$, $O_u^+(2) = \{2, 5\}$ and $O_u^-(2) = \{4\}$. Similarly,

$$\begin{aligned}
u_{(3)} &= 3\,5\,3 && \text{and } 3 \text{ overlaps with } 5\,,\\
u_{(4)} &= 4\,3\,5\,3\,\overline{2}\,\overline{6}\,\overline{5}\,7\,4 && \text{and } 4 \text{ overlaps with } 2, 6, 7\,,\\
u_{(5)} &= 5\,3\,\overline{2}\,\overline{6}\,\overline{5} && \text{and } 5 \text{ overlaps with } 2, 3, 6\,,\\
u_{(6)} &= \overline{6}\,\overline{5}\,7\,4\,6 && \text{and } 6 \text{ overlaps with } 4, 5, 7\,,\\
u_{(7)} &= 7\,4\,6\,7 && \text{and } 7 \text{ overlaps with } 4, 6\,.
\end{aligned}$$

□

For each letter $a \in \Sigma$, let $\mathbf{a} = \{a, \overline{a}\}$, and for each legal string $v \in \Sigma^{\Phi}$, let $\mathbf{P}_v = \{\mathbf{a} \mid a \in \mathrm{dom}(v)\}$. Define the *overlap graph* $\gamma_v = (\mathbf{P}_v, E, \sigma)$ of v as a graph on the set $\mathbf{P}_v$ of vertices together with the labelling σ of the vertices defined by

$$\sigma(\mathbf{a}) = \begin{cases} +\,, & \text{if } a \in \Sigma \text{ is positive in } v\,,\\ -\,, & \text{if } a \in \Sigma \text{ is negative in } v\,, \end{cases}$$

and

$$\{\mathbf{a}, \mathbf{b}\} \in E \iff a \text{ and } b \text{ overlap in } v\,.$$

The label of a vertex $\mathbf{x}$ in an overlap graph is usually called the *sign* of $\mathbf{x}$. Overlap graphs of double occurrence strings are also known as *circle graphs* (see [1, 2]).

Example 4. The overlap graph of the legal string u of Example 3 is given in Fig. 1, where the sign of each vertex is given as a superscript. □

The mapping $w \mapsto \gamma_w$ of legal strings to overlap graphs is not injective: for each legal string $w = w_1 w_2$, we have

$$\gamma_{w_1 w_2} = \gamma_{w_2 w_1} \quad \text{and} \quad \gamma_w = \gamma_{\varsigma(w)}\,,$$

where ς is any mapping that chooses an element $\varsigma(\mathbf{a})$ from $\mathbf{a} = \{a, \overline{a}\}$ for each a. In particular, all conjugates of a legal string w have the same overlap graph. Also, the reversal w^R and the complementation w^C of a legal string w define the same overlap graph as w does.

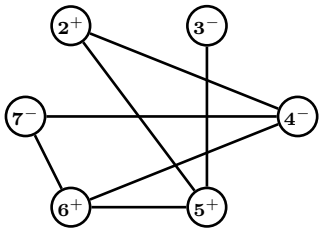

Fig. 1. The overlap graph of $u = 2\,4\,3\,5\,3\,\overline{2}\,\overline{6}\,\overline{5}\,7\,4\,6\,7$

3 Gene assembly in ciliates

The genes in a micronuclear chromosome in ciliates consist of MDSs separated by IESs. Moreover, some of the MDSs may have been inverted. From the viewpoint of gene assembly, the micronuclear gene can be seen as a sequence of MDSs and IESs forming the gene.

Example 5. The actin I gene in *Sterkiella nova* has the following MDS/IES micronuclear structure:

$$M_3 I_1 M_4 I_2 M_6 I_3 M_5 I_4 M_7 I_5 \overline{M}_2 I_7 M_1 I_8 M_8\,. \tag{1}$$

This structure is illustrated in Fig. 2, where each MDS is drawn as a rectangle and the interspacing IESs are represented by lines. Note that the MDS M_2 is inverted. □

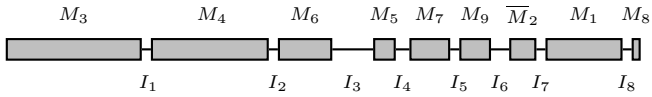

Fig. 2. The micronuclear version of the *actin I* gene in *Sterkiella nova*

In the process of gene assembly each micronuclear gene is translated into a macronuclear gene by excising all IESs and by splicing the MDSs in the orthodox order $M_1, M_2, \ldots, M_\kappa$ (see, [13, 14, 18]). Each MDS M_i has the form

$$M_i = (\pi_i, \mu_i, \pi_{i+1})\,, \quad \text{where } \pi_1 = \frac{b}{\overline{b}},\ \pi_i = \frac{p_i}{\overline{p}_i} (\text{for } 1 < i < \kappa),\ \text{and } \pi_{\kappa+1} = \frac{e}{\overline{e}}\,.$$

Here π_i, μ_i and π_{i+1} correspond to double stranded molecules; π_i and π_{i+1} are the (*incoming* and *outgoing*) *pointers* and μ_i is the *body* of the MDS M_i. The *markers* b and e (and their inversions $\overline{b}$ and $\overline{e}$) designate the locations where the macronuclear DNA gene is to be excised. In the final gene in the macronucleus, the MDSs $M_1, M_2, \ldots, M_\kappa$ are spliced together by "gluing" M_j and M_{j+1} on the common pointer π_{j+1} for each j.

We now describe the operations *ld*, *hi* and *dlad* introduced in [9, 19].

1. The operation *(loop, direct repeat)-excision*, or *ld*, for short, is applied to a molecule with a direct repeat pattern $(- \pi - \pi -)$ of a pointer: either the two occurrences of π are separated by one IES or they are at the two opposite ends of the molecule. The molecule is folded into a loop in such a way that the pointers are aligned and then, the operation proceeds as shown in Fig. 3. The *ld* operation yields two molecules: a linear and a circular one. The circular molecule either contains the whole gene or it contains one IES only.

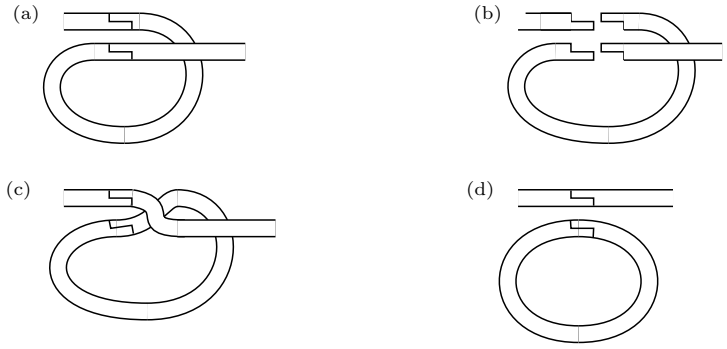

Fig. 3. The *ld* operation

2. The operation *(hairpin, inverted repeat)-excision/reinsertion*, or *hi*, for short, is applied to a molecule with an *inverted repeat pattern* $(- \pi - \overline{\pi} -)$ of a pointer. The molecule is folded into a hairpin in such a way that the pointers are aligned and the operation proceeds as in Fig. 4. The operation *hi* yields only one molecule.

3. The operation *(double loop, alternating direct repeat)-excision/reinsertion*, or *dlad*, for short, is applied to a molecule with an *alternating direct repeat pattern* $(- \pi - \pi' - \pi - \pi' -)$ for two pointers π and π'. The molecule is folded into a double loop in such a way that the pointers π and the pointers π' are aligned and then the operation proceeds as in Fig. 5. The operation *dlad* yields one molecule.

4 MDS descriptors

The process of gene assembly can be seen as a process of assembling MDSs through splicing so that finally the macronuclear gene is obtained (recall that the macronuclear gene is obtained by gluing together, on common pointers, the

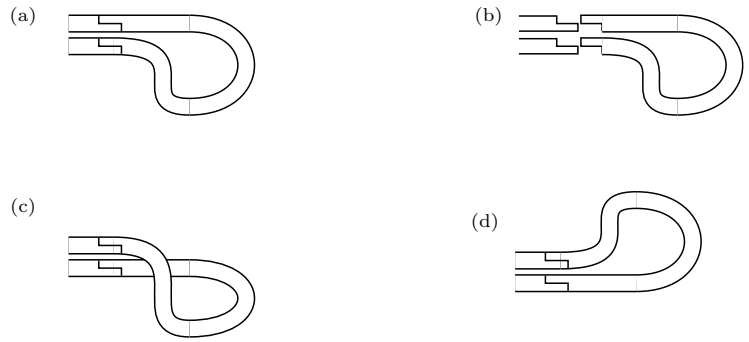

Fig. 4. The operation *hi*

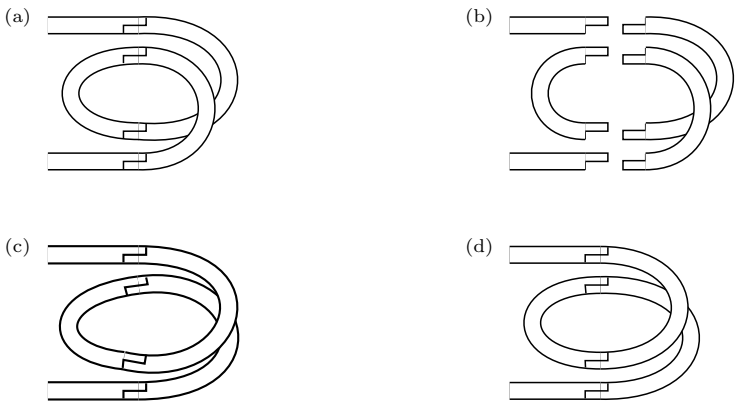

Fig. 5. The operation *dlad*

MDSs arranged in the orthodox order $M_1, M_2, \ldots, M_\kappa$). Therefore the structural information about the micronuclear gene or an intermediate precursor of a macronuclear gene can be given by the sequence of MDSs of the gene. Thus one can represent a macronuclear gene, as well as its micronuclear or an intermediate precursor, by its sequence of MDSs only.

Example 6. The representation of the actin I gene in *Sterkiella nova* from Example 5 has the following simplified representation: $\alpha = M_3M_4M_6M_5M_7\overline{M}_2M_1M_8$ (all the IESs were removed). From the point of view of our considerations has the same information as the structure in (1). □

We shall use the alphabets $\Theta_\kappa = \{ M_{i,j} \mid 1 \le i \le j \le \kappa\}$ to denote the MDSs for all $\kappa \ge 1$. Also, we let

$$\Theta = \bigcup_{\kappa \ge 1} \Theta_\kappa \,.$$

The signed strings in $\Theta^{\maltese}$ are (*MDS*) *arrangements*. Elements $M_{i,i}$ (that denote micronuclear MDSs) are called *elementary MDSs*, and they are often written

simply as M_i. Letters $M_{i,j}$ with $j > i$ denote *composite MDSs* formed during the assembly process by splicing the MDSs $M_i, M_{i+1}, \ldots, M_j$.

We say that an arrangement $\alpha \in {\Theta_\kappa}^{\Join}$ is *orthodox*, if it is of the form

$$\alpha = M_{1,i_2-1} M_{i_2,i_3-1} \ldots M_{i_n,\kappa} \,. \tag{2}$$

Note that an orthodox arrangement does not contain any inverted MDSs. A signed permutation of an orthodox arrangement ($M_1 M_2 \ldots M_\kappa$, resp.) is a *realistic arrangement* (*micronuclear arrangement*, resp.) of *size* κ.

Each orthodox arrangement α such as (2) can be mapped onto the orthodox arrangement $M_1 M_2 \ldots M_{i_n}$ by mapping $M_{i_r,i_{r+1}-1}$ to M_r (with $1 = i_1$). Hence every realistic arrangement is an image of a micronuclear arrangement by such a mapping.

Example 7. The arrangement $M_{1,1} M_{2,5} M_{6,9}$ is orthodox. It is not a micronuclear arrangement, since it contains composite MDSs $M_{2,5}$ and $M_{6,9}$. □

In an assembled gene no pointers are present, because the gene has no IESs. On the other hand, the micronuclear form of a gene has (possibly many) pointers. Thus the gene assembly process can be analysed by

1. representing the micronuclear and each of the intermediate genes by the pattern of pointers present in this gene, and then
2. representing the process by a sequence of such patterns, where each next pattern results by the application of molecular operations to the previous one.

Consequently, we can simplify the formal framework by denoting each MDS by the ordered pair of its pointers and markers only, i.e., $M_{i,j} = (\pi_i, \mu, \pi_j)$ is represented as (p_i, p_j), and its inversion $\overline{M}_{i,j} = (\overline{\pi}_j, \overline{\mu}, \overline{\pi}_i)$ as $(\overline{p}_j, \overline{p}_i)$ for all $i \leq j$.

More formally, let $\Psi = \{b, e, \overline{b}, \overline{e}\}$ denote the set of the *markers* (b stands for "beginning", and e for "end"). For $\kappa \geq 2$, let

$$\Pi_\kappa = \Delta_\kappa \cup \overline{\Delta}_\kappa, \;\; \Pi_{ex,\kappa} = \Pi_\kappa \cup \Psi \;\; \text{where } \Delta_\kappa = \{2, \ldots, \kappa\} \text{ and } \overline{\Delta}_\kappa = \{\overline{2}, \ldots, \overline{\kappa}\} \,.$$

Also, let $\Delta = \{2, 3, \ldots\}$ and $\Pi = \Delta \cup \overline{\Delta}$. We do not use 1, since it represents a begin marker in the encoding defined in a latter section. The letters in Π are called *pointers*, and for each $p \in \Pi$, the pair $\mathbf{p} = \{p, \overline{p}\}$ is the *pointer set* of p (and of $\overline{p}$). Whenever we deal with a specific gene (in any specific species) the number of elementary MDSs in the micronuclear determines the index κ. In order to avoid too involved formalism we shall assume that, unless explicitly stated otherwise, the index κ is clear from the context. Let

$$\begin{aligned} \Gamma_\kappa = {} & \{\, (b,e), (\overline{e}, \overline{b}) \,\} \;\cup\; \{\, (i,j), (\overline{j}, \overline{i}) \;\mid\; 2 \leq i < j \leq \kappa \,\} \cup \\ & \cup\; \{\, (b,i), (\overline{i}, \overline{b}), (i,e), (\overline{e}, \overline{i}) \;\mid\; 2 \leq i \leq \kappa \,\} \,. \end{aligned}$$

A string over Γ_κ is called an *MDS descriptor*.

We define the morphism $\psi_\kappa \colon (\Theta_\kappa)^{\maltese} \to \Gamma_\kappa^*$ as follows:

$$\begin{aligned}
&\psi_\kappa(M_{1,\kappa}) = (b,e) && \text{and} && \psi_\kappa(\overline{M}_{1,\kappa}) &= (\overline{e},\overline{b})\,, && \\
&\psi_\kappa(M_{1,i}) = (b,i+1) && \text{and} && \psi_\kappa(\overline{M}_{1,i}) &= (\overline{i+1},\overline{b}) && (1 \le i < \kappa)\,, \\
&\psi_\kappa(M_{i,\kappa}) = (i,e) && \text{and} && \psi_\kappa(\overline{M}_{i,\kappa}) &= (\overline{e},\overline{i}) && (1 < i \le \kappa)\,, \\
&\psi_\kappa(M_{i,j}) = (i,j+1) && \text{and} && \psi_\kappa(\overline{M}_{i,j}) &= (\overline{j+1},\overline{i}) && (1 < i \le j < \kappa)\,.
\end{aligned}$$

The mapping ψ_κ is bijective, and therefore, for formal purposes, an MDS arrangement α and its image MDS descriptor $\delta = \psi_\kappa(\alpha)$ are equivalent – that is, the structure of these two strings is the same. In particular, the mapping ψ_κ is invertible: if $\delta = \psi_\kappa(\alpha)$ then $\alpha = \psi_\kappa^{-1}(\delta)$ is well defined.

Example 8. For the realistic arrangement $\alpha = M_{3,5}\overline{M}_{9,11}\overline{M}_{1,2}M_{12}\overline{M}_{6,8}$, we obtain the MDS descriptor $\psi_{12}(\alpha) = (3,6)(\overline{12},\overline{9})(\overline{3},\overline{b})(12,e)(\overline{9},\overline{6})$. □

Let $\delta = (x_1,x_2)\ldots(x_{2n-1},x_{2n})$ and $\delta' = (y_1,y_2)\ldots(y_{2n-1},y_{2n})$ be two MDS descriptors. They are *isomorphic*, if

(1) $x_i \in \Delta \iff y_i \in \Delta$ and $x_i \in \Psi \implies y_i = x_i$,
(2) $\xi(x_i) < \xi(x_j) \iff \xi(y_i) < \xi(y_j)$ for $x_i, x_j \notin \Psi$.

Example 9. The MDS descriptors $(4,5)(\overline{8},\overline{6})(b,4)$ and $(2,3)(\overline{5},\overline{4})(b,2)$ satisfy the above requirements, and thus they are isomorphic.

An MDS descriptor δ is said to be *realistic*, if δ is isomorphic with an MDS descriptor $\psi_\kappa(\alpha)$ for some κ and a micronuclear arrangement α. The following lemma is clear from the definition of the mapping ψ.

Lemma 1. *An MDS descriptor δ is realistic if and only if $\delta = \psi_\kappa(\alpha)$ for some κ and a realistic MDS arrangement α of size κ.*

Let $\delta = (x_1,x_2)(x_3,x_4)\ldots(x_{2n-1},x_{2n})$ be a realistic MDS descriptor. Each pointer $p \in \Delta$ either does not occur in δ or it occurs exactly twice in δ. If there are two occurrences, let these be $x_i, x_j \in \mathbf{p} = \{p,\overline{p}\}$ for $1 \le i < j \le 2n$. The *p-interval* of δ is then defined to be the set

$$\delta_{(p)} = \{x_i, x_{i+1}, \ldots, x_j\}\,.$$

If $x_i = x_j$, then p is *negative* in δ; otherwise (i.e., if $x_i = \overline{x}_j$) p is *positive* in δ.

5 The assembly operations

We now formalize the gene assembly operations through formal operations on realistic MDS descriptors. Corresponding to the three molecular operations *ld, hi,* and *dlad*, we have three operations ld, hi, and dlad on MDS descriptors.

1. For each $p \in \mathit{\Pi}_\kappa$, the ld-*rule* for p is defined as follows:

$$\mathsf{ld}_p(\delta_1(q,p)(p,r)\delta_2) = \delta_1(q,r)\delta_2 \tag{l1}$$
$$\mathsf{ld}_p((p,q)\delta_1(r,p)) = (r,q)\delta_1 \tag{l2}$$

where $q, r \in \mathit{\Pi}_{ex,\kappa}$ and $\delta_1, \delta_2 \in (\mathit{\Gamma}_\kappa)^*$.

The case (l1) is called a *simple* ld-*rule*, and it applies to two adjacent occurrences of p separated by one IES only. The case (l2) is called a *boundary* hi-*rule*, and it applies to two occurrences of p at the two ends of the molecule. Both of these cases are illustrated in Fig. 6, where rectangles denote MDSs with their pointers indicated, the zigzag line denotes a segment of a molecule that may contain both MDSs and IESs; a simple straight line represents an IES.

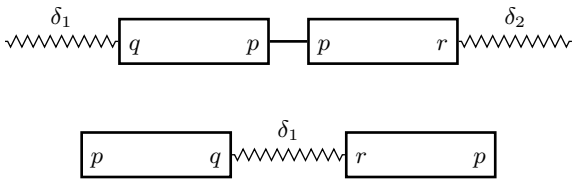

Fig. 6. The MDS/IES structure to which ld_p is applicable

2. For each $p \in \mathit{\Pi}_\kappa$, the hi-*rule* for p is defined as follows:

$$\mathsf{hi}_p(\delta_1(p,q)\delta_2(\overline{p},\overline{r})\delta_3) = \delta_1\overline{\delta}_2(\overline{q},\overline{r})\delta_3 \tag{h1}$$
$$\mathsf{hi}_p(\delta_1(q,p)\delta_2(\overline{r},\overline{p})\delta_3) = \delta_1(q,r)\overline{\delta}_2\delta_3 \tag{h2}$$

where $q, r \in \mathit{\Pi}_{ex,\kappa}$ and $\delta_i \in (\mathit{\Gamma}_\kappa)^*$ for each $i = 1, 2, 3$.

Here one occurrence of p is the incoming pointer and the other occurrence is the outgoing pointer. These cases are illustrated in Fig. 7.

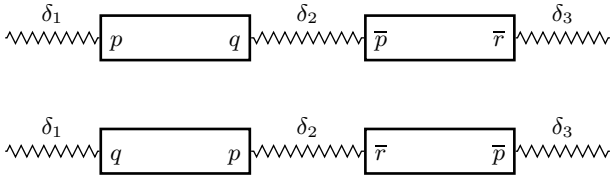

Fig. 7. The MDS/IES structure to which hi_p is applicable

3. For each $p, q \in \Pi_\kappa$, $p \neq q$, the dlad-*rule* for p and q is defined as follows:

$$\mathsf{dlad}_{p,q}(\delta_1(p,r_1)\delta_2(q,r_2)\delta_3(r_3,p)\delta_4(r_4,q)\delta_5) = \delta_1\delta_4(r_4,r_2)\delta_3(r_3,r_1)\delta_2\delta_5 \quad (4a)$$
$$\mathsf{dlad}_{p,q}(\delta_1(p,r_1)\delta_2(r_2,q)\delta_3(r_3,p)\delta_4(q,r_4)\delta_5) = \delta_1\delta_4\delta_3(r_3,r_1)\delta_2(r_2,r_4)\delta_5 \quad (4b)$$
$$\mathsf{dlad}_{p,q}(\delta_1(r_1,p)\delta_2(q,r_2)\delta_3(p,r_3)\delta_4(r_4,q)\delta_5) = \delta_1(r_1,r_3)\delta_4(r_4,r_2)\delta_3\delta_2\delta_5 \quad (4c)$$
$$\mathsf{dlad}_{p,q}(\delta_1(r_1,p)\delta_2(r_2,q)\delta_3(p,r_3)\delta_4(q,r_4)\delta_5) = \delta_1(r_1,r_3)\delta_4\delta_3\delta_2(r_2,r_4)\delta_5 \quad (4d)$$
$$\mathsf{dlad}_{p,q}(\delta_1(p,r_1)\delta_2(q,p)\delta_4(r_4,q)\delta_5) = \delta_1\delta_4(r_4,r_1)\delta_2\delta_5 \quad (3a)$$
$$\mathsf{dlad}_{p,q}(\delta_1(p,q)\delta_3(r_3,p)\delta_4(q,r_4)\delta_5) = \delta_1\delta_4\delta_3(r_3,r_4)\delta_5 \quad (3b)$$
$$\mathsf{dlad}_{p,q}(\delta_1(r_1,p)\delta_2(q,r_2)\delta_3(p,q)\delta_5) = \delta_1(r_1,r_2)\delta_3\delta_2\delta_5 \quad (3c)$$

where $r_i \in \Pi_{ex,\kappa}$ and $\delta_i \in (\Gamma_\kappa)^*$ for each i.

In each of the above instances, the pointer p overlaps with the pointer q. The cases (4a) – (4d) are illustrated in Fig. 8, and the 'short' cases (3a) – (3c) in Fig. 9.

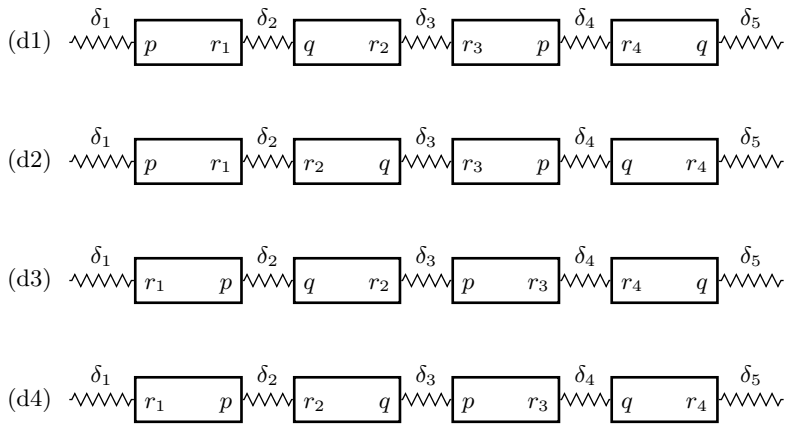

Fig. 8. The main MDS/IES structures to which $\mathsf{dlad}_{p,q}$ is applicable

The following lemma is clear by the form of the operations and the definition of isomorphism of MDS descriptors.

Lemma 2. *Let δ be a realistic MDS descriptor, p and q be pointers in δ, and $\varphi \in \{\mathsf{ld}_p, \mathsf{hi}_p, \mathsf{dlad}_{p,q}\}$. If φ is applicable to δ, then also $\varphi(\delta)$ is realistic.*

Let a composition $\varphi = \varphi_k \dots \varphi_1$ of operations $\varphi_i \in \{\mathsf{ld}_p, \mathsf{hi}_p, \mathsf{dlad}_{p,q} \mid p, q \in \Pi\}$ be applicable to an MDS descriptor δ. In this case, we also say that φ is a *reduction* of δ. Moreover, φ is *successful* for δ, if either $\varphi(\delta) = (b, e)$ or $\varphi(\delta) = (\overline{e}, \overline{b})$.

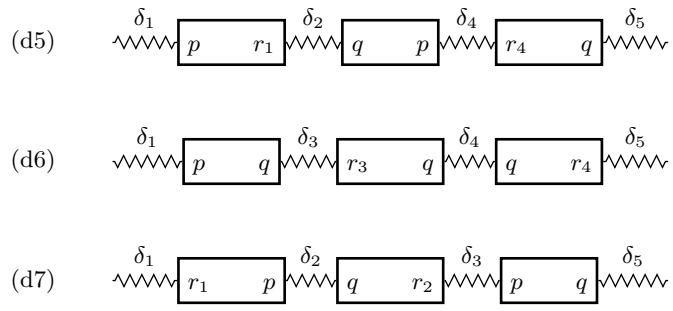

Fig. 9. The special MDS/IES structures to which $\mathsf{dlad}_{p,q}$ is applicable

Example 10. Let $\delta = (4,5)(\overline{2},\overline{b})(5,e)(\overline{4},\overline{3})(\overline{3},\overline{2})$. Now δ is a realistic MDS descriptor, since $\delta = \psi(\alpha)$ for the micronuclear arrangement $\alpha = M_4\overline{M}_1M_5\overline{M}_3\overline{M}_2$. The operations $\mathsf{ld}_{\overline{3}}$, hi_4, and $\mathsf{dlad}_{5,\overline{2}}$ are applicable to δ:

$$\begin{aligned}\mathsf{ld}_{\overline{3}}(\delta) &= (4,5)(\overline{2},\overline{b})(5,e)(\overline{4},\overline{2})\,,\\ \mathsf{hi}_4(\delta) &= (\overline{e},\overline{5})(b,2)(\overline{5},\overline{3})(\overline{3},\overline{2})\,,\\ \mathsf{dlad}_{5,\overline{2}}(\delta) &= (4,e)(\overline{4},\overline{3})(\overline{3},\overline{b})\,.\end{aligned}$$

We also have that $\mathsf{hi}_4\,\mathsf{dlad}_{5,\overline{2}}(\delta) = (\overline{e},\overline{3})(\overline{3},\overline{b})$ and $\mathsf{ld}_{\overline{3}}\,\mathsf{hi}_4\,\mathsf{dlad}_{5,\overline{2}}(\delta) = (\overline{e},\overline{b})$, and therefore $\mathsf{ld}_{\overline{3}}\,\mathsf{hi}_4\,\mathsf{dlad}_{5,\overline{2}}$ is successful for δ. □

A single application of the three operations ld, hi, and dlad shortens the MDS descriptor. The following theorem was first proved in [6]. It shows that the set of our three operations on MDS descriptors, ld, hi, and dlad, is *universal*, i.e., any realistic MDS descriptor δ has a successful reduction for it.

Theorem 1. *Each realistic MDS descriptor δ has a successful reduction.*

Proof. It is sufficient to prove that for every realistic MDS descriptor δ at least one of the three operations is applicable to δ, since a realistic MDS descriptor yields again a realistic MDS descriptor by any one of these operations.

Let then δ be such a descriptor, and assume that neither ld nor hi is applicable for δ. Since hi is not applicable, all pointers in δ must be negative, and since ld is not applicable, δ has no simple direct repeat pattern. Consequently, if $p \in \mathit{\Pi}$ occurs in δ, then the p-interval $\delta_{(p)}$ must contain at least one other pointer.

Let p then be a pointer in δ such that the number of pointers within the p-interval is minimal (no other pointer from δ has less pointers in its interval). Let q be a pointer that has an occurrence within the p-interval. Since all pointers in δ are negative, δ contains two occurrences of q. The other occurrence of q must be outside the p-interval, as otherwise the minimality of p is contradicted. But now either $\mathsf{dlad}_{p,q}$ or $\mathsf{dlad}_{q,p}$ is applicable to δ. □

6 Realizable legal strings

The model of MDS descriptors was greatly simplified in [3] and [7] by considering legal strings instead of MDS descriptors. In order to discuss the reduction of the model of MDS descriptors to the model using legals strings, we need the following definitions.

Let μ be the morphism that removes the parenthesis and the markers from each MDS descriptor. To be more precise, let $\nu\colon (\Delta_{ex,\kappa})^{\maltese} \to \Delta^{\maltese}$ be a substitution defined by $\nu(x) = \Lambda$ if $x \in \Psi$ and $\nu(x) = x$, otherwise. Then let $\mu(x,y) = \nu(x)\nu(y)$ for all $x, y \in \Pi_{ex,\kappa}$. For the MDS arrangements, we define a substitution $\varrho_\kappa\colon (\Theta_\kappa)^{\maltese} \to (\Delta_\kappa)^{\maltese}$ by $\varrho_\kappa = \mu\psi_\kappa$. In particular, for the elementary MDSs, we have

$$\varrho_\kappa(M_1) = 2, \quad \varrho_\kappa(M_\kappa) = \kappa\,, \quad \varrho_\kappa(M_i) = i\,i+1 \quad \text{for } 2 < i < \kappa\,,$$

and $\varrho_\kappa(\overline{M}_i) = \overline{\varrho_\kappa(M_i)}$ for $1 \le i \le \kappa$.

The operations ld, hi, and dlad carry over to legal strings in simplified form. Indeed, as seen from the definition given below instead of the 11 cases for the MDS descriptors, we have only four cases for legal strings.

Let $p, q \in \Pi$.

1. The string operations corresponding to ld_p are the following:

$$\mathsf{ld}_p(uppv) = uv \quad \text{and} \quad \mathsf{ld}_p(pup) = u\,.$$

2. The string operation corresponding to hi_p is the following:

$$\mathsf{hi}_p(upw\overline{p}v) = u\overline{w}v\,.$$

3. The string operation corresponding to $\mathsf{ld}_{p,q}$ is the following:

$$\mathsf{dlad}_{p,q}(u_1pv_1qu_2pv_2qu_3) = u_1v_2u_3v_1u_3\,.$$

For a more detailed discussion concerning the correspondence of the operations for MDS descriptors and legal strings, we refer to [3, 6]

We say that a legal string u is *realistic* if there exists a realistic MDS descriptor δ such that $u = \mu(\delta)$, or equivalently $u = \varrho(\alpha)$ for a signed permutation of an orthodox MDS arrangement α (see Lemma 1).

Example 11. (1) The MDS arrangement $\alpha = M_3M_4M_6M_5M_7M_9\overline{M}_2M_1M_8$ in Example 6 gives the realistic legal string $\varrho(\alpha) = 3\,4\,4\,5\,6\,7\,5\,6\,7\,8\,9\,\overline{3}\,\overline{2}\,2\,8\,9$.

(2) The string $u = 2\,3\,4\,3\,2\,4$ is legal and $\mathrm{dom}(u) = \Delta_\kappa$ for $\kappa = 4$. However, u is not realistic. (It is easy to see that u has no 'realistic parsing'.) □

A legal string $u \in \Sigma^{\maltese}$ is *realizable*, if u is isomorphic to a realistic legal string, that is, if there exists an injective substitution $\tau\colon \Sigma^{\maltese} \to \Delta^{\maltese}$ such that $\tau(\Sigma) \subseteq \Delta$ and $\tau(u)$ is realistic.

Example 12. The legal string $u = 5\,4\,\overline{3}\,2\,\overline{5}\,\overline{2}\,4\,3 \in (\Delta_5)^{\maltese}$ is not realistic, since $4\,\overline{3}\,2$ can never be a substring of a realistic legal string. However, u is realizable: the substitution τ is defined by $\tau(2) = 2$, $\tau(3) = 5$, $\tau(4) = 4$, and $\tau(5) = 3$, and then $\tau(u) = 3\,4\,\overline{5}\,2\,\overline{3}\,\overline{2}\,4\,5$. Now $\tau(u) = \varrho(\alpha)$ for the micronuclear arrangement $\alpha = M_3\overline{M}_5M_1\overline{M}_2M_4$, and thus $\tau(u)$ is realistic.

The following example shows that there are legal strings that are not realizable.

Example 13. The legal string $u = 2\,2\,3\,4\,4\,3\,5\,5$ is not realizable. To see this, assume to the contrary: let v be a realistic legal string and τ an isomorphism such that $u = \tau(v)$. Since the string pp is never an image $\varrho(M_i)$ of any MDS M_i, either one of the following cases (i) or (ii) holds.

(i) $\tau(2) = 2$ and $\tau(5) = 5$. Now also $\tau(3) = 3$, since u begins with 2(23). This yields a contradiction, since now 35 is a substring of v.

(ii) $\tau(2) = 5$ and $\tau(5) = 2$. Now, v should begin with 55, but 55 is never a prefix of any realistic legal string (although $5\overline{5}$ can be). □

We relax now the conditions of realizability by considering realizable signings. Let Σ be an alphabet. Recall that a signed string v over Σ is a signing of a string $u \in \Sigma^*$, if $\xi(v) = u$, where ξ removes the bars from the string. Moreover, if v is realizable, it is a *realizable signing* of u.

Example 14. The legal strings $v_1 = 2\,3\,\overline{2}\,4\,\overline{3}\,5\,4\,5$ and $v_2 = \overline{2}\,3\,2\,\overline{4}\,3\,5\,4\,5$ are both signings of $u = 2\,3\,2\,4\,3\,5\,4\,5$. Note that v_1 is realistic ($v_1 = \varrho(M_2\overline{M}_1\overline{M}_3M_5M_4)$), but u is not. Also, it is easy to see that the signing v_2 of u is not realistic. □

By definition every signing of a double occurrence string is legal. We shall now study the problem which double occurrence strings have realizable signings.

For a double occurrence string $w = a_1a_2 \ldots a_{2\kappa}$, we define an edge-coloured and vertex-labelled (multi)graph A_w on the set $\{1, 2, \ldots, 2\kappa\}$ of vertices as follows: the (undirected) edges e together with their colours $c(e)$ are

$$e_{ij} = \{i, j\} \text{ and } c(e_{ij}) = 1, \text{ if } |i - j| = 1\,,$$
$$e'_{ij} = \{i, j\} \text{ and } c(e'_{ij}) = 0, \text{ if } a_i = a_j\,.$$

The vertex-labelling is defined by

$$\ell(i) = a_i \text{ for } i = 1, 2, \ldots, 2\kappa\,.$$

Note that if both $|i - j| = 1$ and $a_i = a_j$, then there will be two edges between i and j, one edge of each colour. A path $e_1e_2 \ldots e_k$ in the graph A_w is said to be *alternating*, if $c(e_{2r+1}) = 0$ and $c(e_{2(r+1)}) = 1$ for each r. An alternating path is *alternating hamiltonian*, if it visits every vertex of the graph exactly once.

Each edge $e = \{y, z\}$ of the graph A_w has two *orientations*: (y, z) and (z, y). We also denote by

$$y \xrightarrow{c(e)} z$$

the orientation (y, z) together with the colour of the edge. In a drawing of the graph A_w, see Fig. 10, a solid line represents colour 1 and a dashed line colour 0. We also write the values $\ell(x)$ of the vertices beneath them.

Example 15. Let $w = 2\,3\,5\,4\,6\,4\,6\,3\,2\,5$ be a double occurrence string over Δ_6. Then the graph A_w is drawn in Fig. 10. The path

$$1 \xrightarrow{0} 9 \xrightarrow{1} 10 \xrightarrow{0} 3 \xrightarrow{1} 4 \xrightarrow{0} 6 \xrightarrow{1} 5 \xrightarrow{0} 7 \xrightarrow{1} 8 \xrightarrow{0} 2$$

is an alternating hamiltonian path of A_w. □

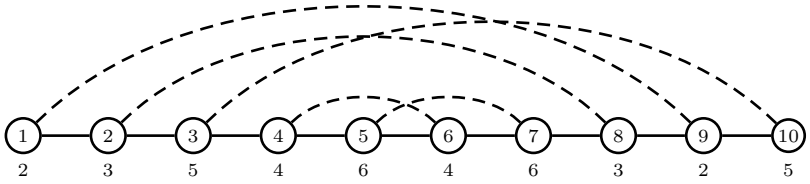

Fig. 10. The graph A_w in Example 15

Theorem 2. *A double occurrence string w has a realizable signing if and only if the graph A_w has an alternating hamiltonian path.*

Proof. (1) We show first by induction on the length of the strings that every double occurrence string w that has a realizable signing w' has an alternating hamiltonian path that starts from the vertex $\varrho(M_1)$ corresponding to the beginning marker and ends in the vertex $\varrho(M_\kappa)$ corresponding to the end marker of the realistic legal string $\tau(w')$, where τ is an isomorphism.

Let w be a double occurrence string of length $2m$ over Σ, and assume that w has a realizable signing $w' \in \Sigma^{\maltese}$. Consequently, there exists a substitution $\tau\colon \Sigma^{\maltese} \to (\Delta_{m+1})^{\maltese}$ such that $\tau(w')$ is realistic. Clearly, the original string w and its image $\tau(w)$ under the isomorphism τ have the same graph, $A_w = A_{\tau(w)}$, except for the labelling of the vertices. Hence, without loss of generality, we can assume that $w = \tau(w)$. In particular, w is over Δ_{m+1}, and w' is a realistic string (and a signing of w).

For each $i \in \Delta_{m+1}$, let $x_{i1}, x_{i2} \in \{1, 2, \ldots, 2m\}$ be the vertices of A_w such that $\ell(x_{i1}) = i = \ell(x_{i2})$ and $x_{i1} < x_{i2}$.

Now $w = w_1(m+1)w_2(m+1)w_3$ for some substrings $w_1, w_2, w_3 \in \Delta_m^*$. The string $v = w_1w_2w_3$ is a signing of the realistic string v', which is obtained from the signing w' of w by removing the two occurrences of the letters from $\{m+1, \overline{m+1}\}$. By the induction hypothesis, v' has an alternating hamiltonian path from x_{2t} to x_{mr}, where $t, r \in \{1, 2\}$, x_{2t} is the position of the beginning marker $\varrho(M_1)$ and x_{mr} is the position of the end marker $\varrho(M_m)$ (of v'). Since w' is realistic, the occurrence of this $m = \varrho(M_m)$ is a part of the substring $m(m+1)$ or $(\overline{m+1})\,\overline{m}$ in w'. It is now obvious that we have an alternating hamiltonian path

$$x_{2t} \xrightarrow{0} \ldots \xrightarrow{0} x_{mr} \xrightarrow{1} x_{(m+1)r'} \xrightarrow{0} x_{(m+1)r''}$$

in A_w, where $\{r', r''\} = \{1, 2\}$ and $x_{(m+1)r''}$ is the position of the end marker $m + 1 = \varrho(M_{m+1})$ in w'. This proves the claim in one direction.

(2) In the other direction, suppose that w is a double occurrence string such that the graph A_w does have an alternating hamiltonian path. Let the hamiltonian path be

$$x_{11} \xrightarrow{0} x_{12} \xrightarrow{1} \ldots \xrightarrow{1} x_{i1} \xrightarrow{0} x_{i2} \xrightarrow{1} \cdots \to x_{m1} \xrightarrow{0} x_{m2}\,,$$

where again $\ell(x_{i1}) = \ell(x_{i2})$, for each $i = 1, 2, \ldots, m$, so that the edge $\{x_{i1}, x_{i2}\}$ has colour 0. Here $\{x_{i1}, x_{i2} \mid i = 1, 2, \ldots, m\} = \{1, 2, \ldots, 2m\}$. Define the substitution τ by $\tau(\ell(x_{i1})) = i + 1$, and then define the signing as follows: if $x_{i2} > x_{(i+1)1}$ (and so $x_{i2} = x_{(i+1)1} + 1$) in the edge $\{x_{i2}, x_{(i+1)1}\}$ of colour 1, then sign both $\tau(\ell(x_{i2}))$ and $\tau(\ell(x_{(i+1)1}))$; otherwise $x_{i2} = x_{(i+1)1} - 1$, and in this case, $\tau(\ell(x_{i2}))$ and $\tau(\ell(x_{(i+1)1}))$ are left unsigned. By the construction, the so obtained string is realistic, and the claim follows from this. □

Example 16. We illustrate the previous theorem and its proof by continuing Example 15. In Fig. 11 we have drawn the alternating hamiltonian path

$$1 \xrightarrow{0} 9 \xrightarrow{1} 10 \xrightarrow{0} 3 \xrightarrow{1} 4 \xrightarrow{0} 6 \xrightarrow{1} 5 \xrightarrow{0} 7 \xrightarrow{1} 8 \xrightarrow{0} 2$$

of A_w. The labels beneath the vertices are now obtained by following the hamiltonian path as in the proof above. The signing given by the proof produces the realistic legal string $2\,6\,3\,4\,\overline{5}\,\overline{4}\,5\,6\,2\,3$. □

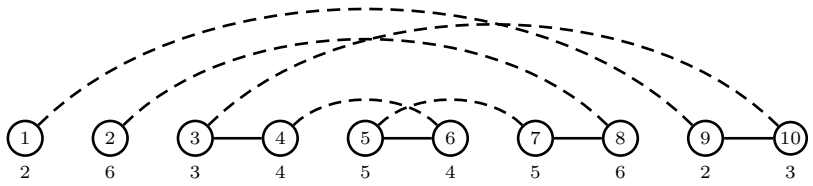

Fig. 11. Alternating hamiltonian path of Example 16

From the proof of Theorem 2 we have the following procedure to determine whether a given legal string is realizable. Let v be a legal string of length $2m$ over an alphabet Σ. We consider the graph $A_v = A_{v'}$. Let x be a vertex of A_v. Define an *induced alternating path* $\mathrm{alt}_v(x)$ *of* v *from* x as follows:

(1) Start with the edge $x_1 \xrightarrow{0} x_2$, where $x_1 = x$ and $\ell(x_2) = \ell(x_1)$. Then $\mathrm{alt}(w)$ is the maximum path obtained by iterating the step (2):

(2) Suppose the alternating path $x_1 \xrightarrow{0} x_2 \xrightarrow{1} \ldots \xrightarrow{1} x_{i-1} \xrightarrow{0} x_i$ has been constructed so that the colour of the edge $\{x_{i-1}, x_i\}$ is 0. If the label of x_i is signed in w, then the next edges are $\{x_i, x_i - 1\}$ (of colour 1) and $\{x_i - 1, x_{i+1}\}$ (of colour 0), where $\ell(x_{i+1}) = \ell(x_i)$.

Notice that after the vertex x is chosen, the induced alternating path $\mathrm{alt}_v(x)$ is well defined, that is, in every step the next edge is uniquely determined.

The following result is a corollary to Theorem 2.

Theorem 3. *Let v be a legal string. Then v is realizable if and only if there exists a vertex x of A_v such that the induced alternating path $\mathrm{alt}(v)$ is an alternating hamiltonian path of A_v.*

7 Nonrealizable strings and graphs

The model of MDS descriptors and (realistic) legal strings can be further abstracted by considering overlap graphs of legal strings. This line of research was initiated in [3] and [7].

By Example 13, there are legal strings that are not realizable. We prove a stronger result in this section: there exist overlap graphs that cannot be 'realized' by any micronuclear arrangement.

For the next two proofs we adopt the following notation: let $A = \{x_1, x_2, x_3\}$ be an alphabet, and let $[x_i] = x_{i1}x_{i2}x_{i3}x_{i0}x_{i1}x_{i2}x_{i3}$.

Lemma 3. *The cyclic conjugates of the double occurrence string*

$$w' = x_{10}x_{20}x_{30}[x_1][x_2][x_3]$$

do not have realizable signings.

Proof. We let the letters y and y_i be variables, and let

$$[y] = y_1y_2y_3y_0y_1y_2y_3\,. \tag{3}$$

Let u be a realizable double occurrence string such that either $u = v_1[y]v_2$ or $u = w_2vw_1$, where $[y] = w_1w_2$. The specified substring $[y]$ or the scattered cyclic conjugate w_2w_1 of $[y]$ in the above is called a $[y]$-*block* of u. By Theorem 2, there exists an alternating hamiltonian path of the graph A_u. Let H be any such path.

It is straightforward to show by case analysis that if the path H does not start at a position inside the $[y]$-block, then H ends at a position inside the $[y]$-block, and symmetrically, if H does not end at a position inside the $[y]$-block, then H begins at a position inside the $[y]$-block. In Fig. 12 we have illustrated one possibility to travel along the substring $[y]$ in H for the case $u = v_1[y]v_2$. (In the figure, the vertices of A_u are simply represented by their labels.) There the path H visits outside the string $[y]$ between the last y_3 and the next $z = y_0$. The path ends in y_0 in the middle of $[y]$.

Therefore H either starts or ends at a vertex corresponding to an occurrence of a pointer in the $[y]$-block. It follows that any cyclic conjugate of a realizable double occurrence string can contain at most two different blocks, a $[y]$-block and a $[y']$-block, of the form (3) such that these do not share letters. In particular, the string w' of the claim does not have conjugates with realizable signings. □

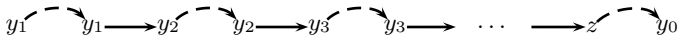

Fig. 12. A part of the path H

As we have seen the mapping $w \mapsto \gamma_w$ from the legal strings to the overlap graphs is not injective. Therefore Lemma 3 leaves unanswered the question whether there are overlap graphs γ_w that are not 'realizable'. In the following theorem we answer this question by constructing an (unsigned) overlap graph that is not realizable.

Theorem 4. *There exists an overlap graph of a double occurrence string that has no signing of the vertices such that the result is an overlap graph of a realistic legal string.*

Proof. Let $w' = x_{10}x_{20}x_{30}[x_1][x_2][x_3]$ be as stated in Lemma 4. The overlap graph $\gamma = \gamma_{w'}$ of w' is given in Fig. 13. We show now that this graph is different from γ_w for all realizable double occurrence strings w.

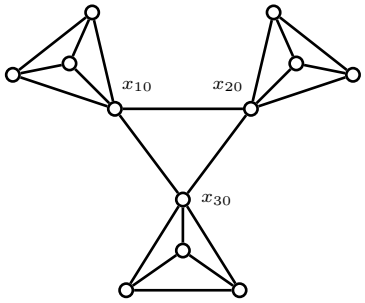

Fig. 13. The overlap graph from the proof of Theorem 4

Assume to the contrary that there exists a realizable double occurrence string w such that $\gamma_w = \gamma$. In particular, w has the same pairs of overlapping letters as w'. For each i, the vertices corresponding to x_{i1}, x_{i2} and x_{i3} are adjacent to each other in γ, and they have the same neighbourhood in the rest of γ (the vertex corresponding to x_{i0}). Hence, we can assume that they occur in w in the given order: $w = v_{i0}x_{i1}v_{i1}x_{i2}v_{i2}x_{i3}v_{i3}x_{i1}v_{i4}x_{i2}v_{i5}x_{i3}v_{i6}$ for some substrings v_{ij} for each i.

Since the letters $x_{j0}, \ldots, x_{j3}$ overlap with each other in w and they do not overlap with x_{i1}, x_{i2}, x_{i3} for $i \neq j$, we conclude that all occurrences of x_{jk} for each $0 \leq k \leq 3$ are either (a) in one substring v_{it} for some $1 \leq t \leq 5$, or (b) they are all in $v_{i0}v_{i6}$. We show now that there are indices i and j such that the case (a) holds. Indeed, there is an index i such that $[x_i]$ is not a substring

of w, since otherwise there would be three disjoint substrings in w of the form (3), which is not possible by the proof of Lemma 3. Now if the occurrences of $x_{j0}, x_{j1}, x_{j2}, x_{j3}$, for both indices j with $j \neq i$, are in $v_{i0}v_{i6}$, then an occurrence of x_{i0} must also be in $v_{i0}v_{i6}$ (since x_{i0} overlaps with x_{j0}), and therefore the second occurrence of x_{i0} must be in v_{i3} (since x_{i0} overlaps with x_{i1}, x_{i2} and x_{i3}). This means, however, that $[x_i]$ is a substring of w; a contradiction.

Moreover, exactly one occurrence of x_{i0} is in v_{it}, since x_{i0} overlaps with x_{j0}. This occurrence must be in $v_{j0}v_{j6}$, since the other occurrence of x_{i0} is not in v_{it}.

Now, by the overlapping properties of x_{j0}, we have for $m = j$ that

$$[x_m]x_{i0}x_{m0} \quad \text{or} \quad x_{m0}x_{i0}[x_m] \text{ is scattered substring in } v_{it}. \tag{4}$$

Since for the remaining index $k \notin \{i, j\}$, x_{k0} overlaps with x_{j0}, an occurrence of x_{k0} lies in the interval of the letter x_{j0}, and thus it is in v_{it}. Similarly to the above, we can show that (4) also holds for $m = k$.

By the above, x_{k0} does not occur in v_{jr} for any $1 \leq r \leq 5$, because otherwise both occurrences of x_{i0} are in v_{jr}. Therefore the index i is unique with respect to the property (a), and then it follows that $[x_m]$ is a substring of v_{it} for both m with $m \neq i$. This implies that a conjugate of w has three disjoint substrings of the form $[y]$, which yields, by the proof of Lemma 3, a contradiction. □

Example 17. The simplest form of the counter example discussed in the above proof is the string 2 3 4 5 6 7 2 5 6 7 8 9 10 3 8 9 10 11 12 13 4 11 12 13 . □

Acknowledgements. G. Rozenberg gratefully acknowledges partial support by NSF grant 0121422.

References

1. Bouchet, A., Circle graphs. *Combinatorica* **7** (1987), 243 – 254.
2. Bouchet, A., Circle graph obstructions. *J. Combin. Theory Ser B* **60** (1994), 107 – 144.
3. Ehrenfeucht, A., T. Harju, I. Petre, D. M. Prescott, and G. Rozenberg, Formal systems for gene assembly in ciliates. *Theoret. Comput. Sci.* **292** (2003), 199 – 219.
4. Ehrenfeucht, A., T. Harju, I. Petre, and G. Rozenberg, Characterizing the micronuclear gene patterns in ciliates. *Theory of Computation and Systems* **35** (2002), 501 – 519.
5. Ehrenfeucht, A., T. Harju, and G. Rozenberg, Gene assembly through cyclic graph decomposition. *Theoretic Comput. Syst.* **281** (2002), 325 – 349.
6. Ehrenfeucht, A., I. Petre, D. M. Prescott, and G. Rozenberg, Universal and simple operations for gene assembly in ciliates. In *Words, Sequences, Languages: Where computer science, biology and linguistics come across*, V. Mitrana, C. Martin-Vide (eds.), Kluwer Academic Publishers, Dortrecht/Boston, 329 – 342, (2001).
7. Ehrenfeucht, A., I. Petre, D. M. Prescott, and G. Rozenberg, String and graph reduction systems for gene assembly in ciliates. *Math. Structures Comput. Sci.* **12** (2001), 113 – 134.

8. Ehrenfeucht, A., I. Petre, D. M. Prescott, and G. Rozenberg, Circularity and other invariants of gene assembly in cliates. In *Words, semigroups, and transductions*, M. Ito, Gh. Păun, S. Yu (eds.), World Scientific, Singapore, 81 – 97, 2001.
9. Ehrenfeucht, A., D. M. Prescott, and G. Rozenberg, Computational aspects of gene (un)scrambling in ciliates. In *Evolution as Computation*, L. Landweber, E. Winfree (eds.), 45 – 86, Springer-Verlag, Berlin, Heidelberg, 2001.
10. Landweber, L. F., and L. Kari, 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, 3 – 15 (1998).
11. Landweber, L. F., and L. Kari, Universal molecular computation in ciliates. In *Evolution as Computation*, L. Landweber, E. Winfree (eds.), Springer-Verlag, Berlin, Heidelberg, 2002.
12. Păun, Gh., G. Rozenberg, and A. Salomaa, *DNA Computing*. Springer-Verlag, Berlin, Heidelberg, 1998.
13. Prescott, D. M., Cutting, splicing, reordering, and elimination of DNA sequences in Hypotrichous ciliates. *BioEssays* **14** (1992), 317 – 324.
14. Prescott, D. M., The unusual organization and processing of genomic DNA in Hypotrichous ciliates. *Trends in Genet.* **8** (1992), 439 – 445.
15. Prescott, D. M., The DNA of ciliated protozoa. *Microbiol Rev.* **58**(2) (1994), 233 – 267.
16. Prescott, D. M., The evolutionary scrambling and developmental unscabling of germlike genes in hypotrichous ciliates. *Nucl. Acids Res.* **27** (1999), 1243 – 1250.
17. Prescott, D. M., Genome gymnastics: unique modes of DNA evolution and processing in ciliates. *Nat Rev Genet.* 1(3) (2000), 191 – 198.
18. Prescott, D. M., and M. DuBois, Internal eliminated segments (IESs) of Oxytrichidae. *J. Eukariot. Microbiol.* **43** (1996), 432 – 441.
19. Prescott, D. M., A. Ehrenfeucht, and G. Rozenberg, Molecular operations for DNA processing in hypotrichous ciliates. *European Journal of Protistology* **37** (2001), 241 – 260.
20. Prescott, D. M., and G. Rozenberg, How ciliates manipulate their own DNA – A splendid example of natural computing. *Natural Computing* **1** (2002), 165 – 183.