Sepinoud Azimi | Tero Harju | Miika Langille | Ion Petre

Simple gene assembly as a rewriting of directed overlap-inclusion graphs

Turku Centre *for* Computer Science

TUCS Technical Report
No 1030, December 2011

Simple gene assembly as a rewriting of directed overlap-inclusion graphs

Sepinoud Azimi
Åbo Akademi University, Department of Information Techology
Joukahaisenkatu 3-5 A, Turku 20520 Finland
sepinoud.azimi@abo.fi
Tero Harju
University of Turku, Department of Mathematics
Turku 20014 Finland
tero.harju@utu.fi
Miika Langille
Åbo Akademi University, Department of Information Techology
Joukahaisenkatu 3-5 A, Turku 20520 Finland
miika.langille@gmail.com
Ion Petre
Åbo Akademi University, Department of Information Techology
Joukahaisenkatu 3-5 A, Turku 20520 Finland
ion.petre@abo.fi

TUCS Technical Report
No 1030, December 2011

Abstract

The simple intramolecular model for gene assembly in ciliates consists of three molecular operations, simple ld, simple hi and simple dlad. Mathematical models in terms of signed permutations and signed strings proved limited in capturing some of the combinatorial details of the simple gene assembly process. Brijder and Hoogeboom introduced a new model in terms of overlap-inclusion graphs which could describe two of the three operations of the model and their combinatorial properties. To capture the third operation, we extended their framework to directed overlap-inclusion (*DOI*) graphs in [1]. In this paper we introduce DOI graph-based rewriting rules that capture all three operations of the simple gene assembly model and prove that they are equivalent to the string-based formalization of the model.

Keywords: Simple gene assembly, Directed overlap inclusion graphs, Graph-based simple operations

TUCS Laboratory
Computational Biomodelling Laboratory

1 Introduction

Ciliates, an old group of unicellular eukaryotes, part of the phylum Ciliophora, have developed into a wide variety of lineages [7]. Ciliates possess two types of nuclei, each present in multiple copies, depending on the species: macronuclei (abbreviated as *MAC*) and micronuclei (*MIC*) [7, 2]. The ciliate genome is very differently organized in the two types of nuclei. The macronuclear genes are presented as contiguous sequence of nucleotides. In contrast, the micronuclear genes are split into blocks (macronuclear destined sequences, or MDSs) that are separated by noncoding sequences (internally eliminated sequences, or IESs), and may even be inverted. At some stage during their life cycle, ciliates destroy all their macronuclei and rebuild their macronuclear genes by assembling in the orthodox order all MDSs from their micronuclear genes in a process called gene assembly. The process is enabled by some short, specific nucleotide sequences called *pointers* that are repeated at the end of an MDS and at the beginning of the MDS that should follow it in the orthodox order. We refer to [7, 12] and references therein for details.

The gene assembly of ciliates has been investigated through several different molecular models, see [2]. One of them is the simple intramolecular model, originally introduced in [8], consisting of three molecular operations: the ld, the simple hi, and the simple dlad operations. This model could successfully predict the assembly of all currently known genes, see [10]. The simple model was modeled mathematically as a sorting of signed permutations in [11], a string rewriting system in [4, 5] and as signed overlap-inclusion graphs in [3]. All these models were limited in capturing the details of the local interactions postulated by the simple model, especially in capturing the details of the simple dlad operations. To address this difficulty we extended in [1] the framework of [3] and defined the *directed* overlap-inclusion (in short, DOI) graphs as a model for ciliate genes. The extension is in fact quite small with respect to the overlap-inclusion graphs of [3]: we simply replace their undirected overlap edges with directed overlap edges where we additionally represent the order in which they occur in the string. In this way we replace a 'hybrid' directed/undirected graph with a directed graph that turns out to capture enough information to be able to represent all three operations of the simple model.

2 Preliminaries

We introduce here some of the notions and notations we use throughout the paper. For more details we refer to [7].

2.1 The simple gene assembly model

We focus in this paper on the intramolecular model for gene assembly in ciliates, introduced in [9, 13]. The model consists of three molecular operations, all conjecturing the folding of the gene in a specific pattern (a loop, a hairpin, or a double loop) so that a pair of pointers (two pairs in the case of dlad) are aligned. Recombination on those pointers is thus enabled and as a result, two (or, in some special cases of dlad, even three) MDSs get spliced together to form a longer MDS. The gene assembly process is thus modeled as a computational process in which the MDSs get longer in each step, to eventually yield as an output the assembled gene. We refer to [7] for details on the model.

The simple version of the intramolecular model, introduced in [10], assumes that the folds involved in each of the three operations are as simple as possible: in-between the aligned pointers there is a minimal number of other pointers (zero in the case of simple ld and simple dlad, one in the case of simple hi). The resulting model was shown in [10] to be capable of explaining the assembly of all currently known ciliate genes in [6]. We refer to [7] and [2] for more details, including figures of the folds of the three molecular operations of the model.

2.2 Legal strings

Let $\Delta_k = \{2, 3, \ldots, k\}$ be an alphabet the elements of which are called *pointers* and let $\Sigma_k = \Delta_k \cup \{m\}$, for some $k \geq 1$ and $\overline{\Sigma_k} = \overline{\Delta_k} \cup \overline{m}$ be a signed copy of Σ_k, where letter m refers to a marker. We make no distinction on this level between the beginning and the ending markers – we simply treat them as being two occurrences of the special symbol m.

Let Σ_k^* be the set of all strings over Σ_k and let $\Sigma_k^{\maltese} = (\Sigma_k \cup \overline{\Sigma_k})^*$ where for each $p \in \Sigma_k$, $\overline{\overline{p}} = p$.

We say that a string u in $\Sigma_k^{\maltese}$ is *legal* if for any $p \in \Sigma_k$, u contains either 0 or 2 occurrences from the set $\{p, \overline{p}\}$. If u contains occurrences from the set $\{p, \overline{p}\}$, for some $p \in \Sigma_k$, then we say that p occurs in u and denote it by $p \in u$. We define the *domain* of u as $\mathrm{dom}(u) = \{p \in \Sigma_k \mid p \in u\}$. The i^{th} occurrence in u (when scanned from left to right) of a pointer/maker from $\{p, \overline{p}\}$ is denoted by p^i, where $i = 1$ or $i = 2$. We say that u is *sorted* if $\mathrm{dom}(u) = \{m\}$; in other words, u is sorted if it only contains the two markers and no pointers.

Let $p \in \Sigma_k \cup \overline{\Sigma_k}$ and let $u \in \Sigma_k^{\maltese}$ be a legal string. If u contains both substrings p and $\overline{p}$ then p is said to be *positive* in u; otherwise, it is said to be *negative*. If $u = u_1 p^1 u_2 p^2 u_3$, with $p^1, p^2 \in \{p, \overline{p}\}$, then the *$p$-interval* of u is the substring u_2.

For any distinct $p, q \in \mathrm{dom}(u)$, p and q have one of the following relations:

- p and q *overlap* if exactly one occurrence from $\{p, \overline{p}\}$ can be found in the q-interval of u. We denote the overlapping relation by $p \Rightarrow_u q$, if the first occurrence from $\{p, \overline{p}\}$, i.e. if the order of occurrence is $q^1 \cdots p^1 \cdots q^2 \cdots p^2$, and we denote it by $q \Rightarrow_u p$ otherwise;

- q is *included* in p if the two occurrences from $\{q, \overline{q}\}$ are found within the p-interval. This relation is denoted by $p \rightarrow_u q$;
- p and q are *disjoint* if they do not overlap and neither is included in the other in u.

2.3 Overlap inclusion graphs

Overlap-inclusion graphs have been introduced first in [3]. Using our notation, for a legal string u its overlap-inclusion graph $G = (V, \sigma, E)$ is defined as follows: $V = \text{dom}(u)$, $\sigma : V \rightarrow \{+, -\}$ is the signing of its vertices such that for each $p \in V$, $\sigma(p) = +$ if p is a positive pointer in u and $\sigma(p) = -$ otherwise and $E = \{\{p, q\} \mid p \Rightarrow_u q \text{ or } q \Rightarrow_u p)\} \bigcup \{(p, q) \mid q \rightarrow_u p\}$. In this way, for any pair of overlapping pointers $\{p, q\}$ in u there is an undirected edge in G between p and q, and for any pointer p whose interval is included in the interval of some pointer q, G has the edge $p \rightarrow_G q$ from p to q.

Example 1. The overlap-inclusion graph corresponding to $u = 255m\overline{2}343m\overline{4}$ is shown in Figure 1(a).

2.4 Directed overlap-inclusion graphs

We recall here the notion of directed overlap-inclusion *(DOI)* graphs introduced in [1] and recall some of their properties. We first recall that a directed graph G is called *connected* if for any distinct vertices u and v of G, there is either a (directed) path from u to v, or a (directed) path from v to u. We also recall that for a set of vertices U of G we denote by $G \setminus U$ the graph obtained from G by removing all vertices in U and all edges incident to them.

Definition 1 ([1])**.** Let u be a legal string over some Σ_k. The *directed overlap-inclusion (DOI) graph* $G_u = (V, \sigma, E_o, E_i)$ corresponding to u is defined as follows: $V = \text{dom}(u)$ is the set of vertices, $\sigma : V \rightarrow \{+, -\}$ is the signing of its vertices such that for each $p \in V$, $\sigma(p) = +$ if p is a positive pointer in u and $\sigma(p) = -$ otherwise. E_o and E_i are sets of its directed edges, $E_o = \{(p, q) \mid p \Rightarrow_u q\}$ and $E_i = \{(p, q) \mid p \rightarrow_u q\}$. For a DOI graph G and any string u such that $G = G_u$ we say that u corresponds to G.

Example 2. The DOI graph corresponding to string $u = 255m\overline{2}343m\overline{4}$ is shown in Figure 1(b).

Definition 2. Let G be a directed, vertex- and edge-labeled graph. We say that G is *forbidden* if there is no string u such that G is isomorphic to the *DOI* graph corresponding to u.

Figure 1: (a) The overlap-inclusion graph and (b) the directed overlap-inclusion graph corresponding to string $u = 255m\overline{2}343m\overline{4}$.

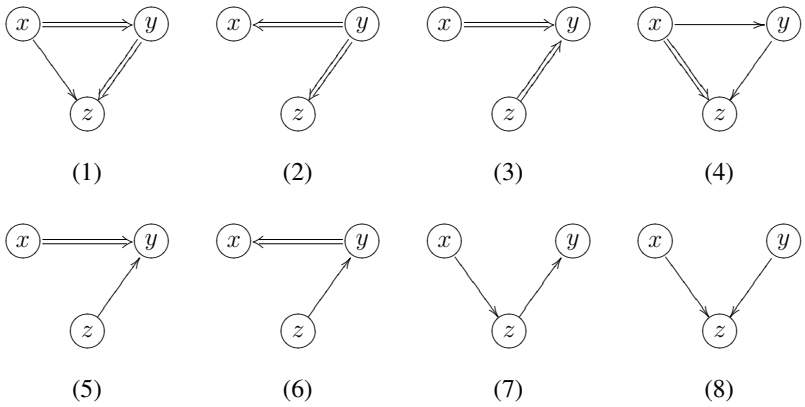

Figure 2: All 3-vertex, acyclic, forbidden graphs. Inclusion edges are illustrated as simple arrows and overlap edges as double arrows.

The following results have been proved in [1].

Theorem 1 ([1])**.** *Let G be a directed labeled graph with $\{+, -\}$ as vertex labels and with two edge colors as edge labels. If G is a 3-vertex, acyclic graph, then G is* forbidden *if and only if it is isomorphic to one of the graphs in Figure 2.*

Theorem 2 ([1])**.** *Any DOI graph G is a directed acyclic graph.*

Example 3. We notice that there can exist more than one string corresponding to a DOI graph. For example, the strings 245566324377, 246655324377, 772455663243 and 772466553243 have the same DOI graph as shown in Figure 3.

3 The strings-to-DOI graphs mapping

We discuss in this section the injectivity of the strings-to-DOI graphs mapping. While in general the same DOI graph can correspond to several strings, we show

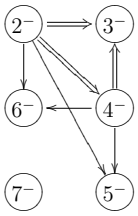

Figure 3: A graph with more than one string corresponding to it.

that this is not true for the realistic strings and for the overlap-only connected DOI graphs.

Lemma 3. *Let G be a connected DOI graph consisting of overlap edges only, i.e. without inclusion edges. Then there is a unique string u_G corresponding to G.*

Proof. It follows from Theorem 2 that in any connected component G with at least two vertices there is a vertex with indegree 0. Since G is connected, there is only one such vertex; we denote it by p. Similarly, there exists a unique vertex, say q with outdegree 0.

We prove now that there is a directed path from p to q visiting all vertices of G. Since G is connected, there is a path P from p to q, say $p = p_0 \Rightarrow p_1 \Rightarrow \ldots \Rightarrow p_n = q$, for some $n \geq 1$. Assume there is a vertex r not visited by this path. A path from p to r exists as well: $p = p_0 \Rightarrow p_1 \Rightarrow \ldots \Rightarrow p_i \Rightarrow r_0 \Rightarrow r_1 \Rightarrow \ldots \Rightarrow r_m = r$, for some $0 \leq i < n$, with r_0 not visited by path P. Thus, by replacing r with r_0, we can assume without loss of generality that r has an edge from a vertex p_i in path P, for some $0 \leq i < n$. Moreover, we choose the largest i with this property.

Note that there must be an overlap edge between p_{i+1} and r since otherwise the subgraph induced by p_i, p_{i+1} and r would be forbidden, being isomorphic with the graph in Figure 2(2). Based on the maximality of i, there cannot be an edge from p_{i+1} to r. Consequently, there is an edge from r to p_{i+1} and so, we can replace path P with the path $p = p_0 \Rightarrow p_1 \Rightarrow \ldots \Rightarrow p_i \Rightarrow r \Rightarrow p_{i+1} \Rightarrow \ldots \Rightarrow p_n = q$, which visits r, as well as all vertices visited by P. By iterating this argument, we build a path from p to q visiting all vertices of G.

We claim now that G consists of a single path from p to q. Assume that this is not true and that there is an edge $p_i \Rightarrow p_j$, for some $0 \leq i, j \leq n$, $i + 1 < j$. We choose the largest such i. In order for the subgraph induced by p_i, p_{i+1} and p_j not to induce the forbidden subgraph in Figure 2(2), there must be an overlap edge between p_{i+1} and p_j. Based on the maximality of i we obtain that $p_j \Rightarrow p_{i+1}$ is an edge in G. But then we obtain that $p_{i+1}, p_{i+2}, \ldots, p_j$ induce a cycle in G, a contradiction.

It is easy to see now that there is a unique string corresponding to G i.e. $p_0p_1p_0p_2p_1p_3 \cdots p_ip_{i+2} \cdots p_{n-2}p_np_{n-1}p_n$. □

Note that the analogous result for inclusion-only graphs does not hold. For example, $pqqrrp$ and $prrqqp$ have the same DOI graph.

We discuss next the case of the so-called realistic strings and we show that the string-to-DOI graphs mapping is injective for this type of strings. Realistic legal strings correspond to the concrete IES strings describing the genome structure of ciliates; see [7].

In the following, for clarity, we identify the marker m with the pointer 1, and we consider the pointers modulo k. In this case $k+1 \equiv 1 \pmod{k}$, i.e., m follows k. For this reason, let $\Gamma_k = \{1, 2, \ldots, k\}$, and let $\Gamma_k^{\maltese}$ denote the strings over the alphabet $\Gamma \cup \overline{\Gamma}_k$ in the natural manner.

A legal string $u \in \Gamma_k^{\maltese}$ is called *realistic*, if $u = \varphi(w)$, where

- $w \in \Gamma_k^{\maltese}$ is a signed permutation, i.e., each symbol $p \in \Gamma_k$ occurs in w exactly once - either in the form p or $\overline{p}$;
- $\varphi\colon \Gamma_k^{\maltese} \to \Gamma_k^{\maltese}$ is a homomorphism that satisfies the substitution laws

$$\varphi(p) = p(p+1), \qquad \varphi(\overline{p}) = (\overline{p+1})\overline{p}\,.$$

By our convention, we have $\varphi(k) = k\,1$ and $\varphi(\overline{k}) = \overline{1}\,\overline{k}$. For example, if $k = 4$ then $w = 3\overline{1}\overline{4}2$ is a signed permutation that gives the realistic string $\varphi(w) = 3412\overline{1}\overline{4}23$.

Theorem 4. *If u and v are different realistic strings, then $G_u \neq G_v$.*

Proof. Let u and v be two different realistic strings corresponding to a common DOI graph G. Therefore there are permutations w and w' such that $u = \varphi(w)$ and $v = \varphi(w')$. If $w = qs$ and $w' = qs'$ for a common prefix symbol $q \in \Gamma_k \cup \overline{\Gamma_k}$ and strings s and s', then also $G_{\varphi(sq)} = G_{\varphi(s'q)}$. Hence we can assume that the permutations w and w' do not have any nonempty common prefix, i.e., their first symbols are different. It follows, by the definition of φ, that also the first pointers in u and v are different. Let then

$$u = pp_2 \ldots p_n \text{ and } v = qq_2 \ldots q_n\,;$$

where $n = 2k, p, q, p_i, q_i \in \Gamma_k \cup \overline{\Gamma_k}$ for each i, and $p \neq q$.

Since $G_u = G_v$, the p-intervals $I_u(p)$ of u and $I_v(p)$ of v contain the same number of each pointer from Γ_k.

Suppose first that $q = \overline{p}$. By symmetry, we may assume that $p \in \Gamma_k$. It follows that $u = p(p+1)\alpha$ and $v = \overline{p}(\overline{p-1})\beta$ for some strings α and β. Since the sign of p is the same in G_u and G_v, we have either

$$\begin{aligned} u &= p(p+1)\cdots(p-1)p\sigma_u, \\ v &= \overline{p}(\overline{p-1})\cdots(\overline{p+1})\overline{p}\sigma_v\,, \end{aligned} \tag{1}$$

or

$$\begin{aligned} u &= p(p+1)\cdots\overline{p}(\overline{p-1})\sigma_u, \\ v &= \overline{p}(\overline{p-1})\cdots p(p+1)\sigma_v\,, \end{aligned} \tag{2}$$

because u and v are realistic.

In Case (1), the pointer $p+1$ occurs once in the p-interval $I_u(p)$ of u, and hence it occurs also once in the p-interval $I_v(p)$ of v, i.e., $\varphi(p+1) = (p+1)(p+2)$ is in $I_v(p)$. Moreover, $p+2$ occurs only once in $I_v(p)$, since $p+1$ does so in the interval $I_u(p)$ of u. We proceed by induction to obtain that the intervals $I_u(p)$ and $I_v(p)$ contain a unique occurrence of every pointer $\Gamma_k \setminus \{p\}$.

Now, the second occurrence of $p+1$ in u and v lies in the suffix σ_u and σ_v, respectively.

Let then t with $t \neq p, p+1$ be a pointer. From the form of the string u, we deduce that $p+1 \Rightarrow t$ or $p+1 \to t$, since one occurrence of t belongs to the p-interval. The form of the string v yields that the second possibility cannot hold. Thus $p+1 \Rightarrow t$ for all pointers $t \neq p, p+1$. But clearly, in the string v this does not hold for $t = p-1$, because the first occurrence of $p-1$ comes before than the first occurrence of $p+1$ in v. This contradiction shows that Case (1) does not hold.

In Case (2), the pointer $p-1$ occurs once in the p-interval $I_u(p)$, and an analogous reasoning to the previous gives that each pointer different from p has a unique occurrence in the p-intervals of u and v. As in the case for (1), we derive a contradiction.

We now assume that $q \notin \{p, \overline{p}\}$.

Note that we do not have any overlap edge in G between the pointers p and q: Indeed, if $p \Rightarrow q$, then $p \Rightarrow q$ in the string u and $q \Rightarrow p$ in the string v. Note also that the intervals $I_u(p)$ and $I_v(q)$ cannot be included in the other one.

We can now write

$$\begin{aligned} u &= p^1\delta_1 p^2\delta_2 q^1\delta_3 q^2\delta_4 \\ v &= q^1\delta_1' q^2\delta_2' p^1\delta_3' p^2\delta_4'. \end{aligned}$$

We define $u_1 = p^1\delta_1 p^2\delta_2$, $u_2 = q^1\delta_3 q^2\delta_4$, $v_1 = q^1\delta_1' q^2\delta_2'$, $p^1\delta_3' p^2\delta_4'$. We claim that u_1 and u_2 do not contain common pointers, and neither do v_1 and v_2.

Let t be a pointer such that $t^1 \in u_1$ and $t^2 \in u_2$. If $t^2 \in \delta_3$ then, $t \Rightarrow q$, which is a contradiction since in v the pointer q starts the word. Similarly, if $t^2 \in \delta_4$ then $t \to q$, which is again a contradiction. A similar argument gives that v_1 an v_2 are disjoint of pointers.

By our assumptions, we can again assume that u is either of the forms

$$\begin{aligned} u &= p(p+1)\cdots(p-1)p\delta_2 q^1\delta_3 q^2\delta_4, \\ u &= p(p+1)\cdots\overline{p}(\overline{p-1})\delta_2 q^1\delta_3 q^2\delta_4. \end{aligned}$$

Since u_1 and u_2 are disjoint of pointers. If a pointer t occurs in u_1, then both occurrences of t are in u_1, and either $t = q - 1$ or also $t + 1$ (modulo k) is in u_1, since $\varphi(t) = t(t+1)$, i.e., $t+1$ leans on t inside u_1. Hence, we obtain inductively that all pointers $p, p+1, \cdots, q-1$ (modulo k) are in u_1. Similarly, by going in the other direction down from p, we have that the pointers $p, p-1, p-2, \cdots, q+1$ (modulo k) are all in u_1. However, $u_2 \neq q^1 q^2$ by the form of $u = \varphi(w_u)$. This contradiction proves the claim. □

4 Simple gene assembly in ciliates

We introduce in this section our DOI-graph-based model for simple gene assembly in ciliates. We first recall the formulation of the model in terms of legal strings.

4.1 Simple gene assembly on legal strings

A gene can be represented as a legal string by simply denoting it as a sequence of pointers and markers. The three molecular operations are defined on legal strings as follows.

Definition 3. The string pointer reduction system is formalized as follows. In each case $p, q \in \Delta_k$ are distinct pointers and $u_1, u_2, u_3 \in \Sigma^{\maltese}$.

i. The *simple string negative* rule ssn_p is defined as follows:

$$\mathsf{ssn}_p(u_1\tilde{p}\tilde{p}u_2) = u_1u_2, \qquad \mathsf{ssn}_p(\tilde{p}u_3\tilde{p}) = u_3, ,$$

where $\tilde{p} \in \{p, \overline{p}\}$ and u_3 contains only markers (boundary case). We denote $\mathsf{Ssn} = \{\mathsf{ssn}_p \mid p \in \Delta_k, k \geq 2\}$.

ii. The *simple string positive* rule ssp_p for a pointer p is defined as follows:

$$\mathsf{ssp}_p(u_1\tilde{p}q\overline{\tilde{p}}u_3) = u_1\overline{q}u_3,$$

where $\tilde{p} \in \{p, \overline{p}\}$ and $\tilde{q} \in \{q, \overline{q}\}$. We denote $\mathsf{Ssp} = \{\mathsf{ssp}_p \mid p \in \Delta_k, k \geq 2\}$.

iii. The *simple string double* rule $\mathsf{ssd}_{p,q}$ for pointers p and q is defined as follows:

$$\mathsf{ssd}_{p,q}(u_1\tilde{p}\tilde{q}u_3\tilde{p}\tilde{q}u_5) = u_1u_3u_5,$$

where $\tilde{p}\tilde{q} \in \{pq, \overline{pq}\}$. We denote $\mathsf{Ssd} = \{\mathsf{ssd}_{p,q} \mid p, q \in \Delta_k, k \geq 2\}$.

The string-based versions of the simple operations are based on "locality": only pointers that are at a minimal distance from each other can be removed by such an operation. This concept of locality is however lost when going from a legal string to its associated overlap graph since overlap graphs do not contain the information about the order of pointers in their corresponding strings. By adding the concept of "inclusion" to overlap graphs it is possible to capture the information about the domains of p – intervals for each pointer p.

4.2 Simple gene assembly on DOI graphs

Definition 4. Let G be a DOI graph and p an arbitrary vertex of G. We introduce the following terms:

i. *Incoming inclusion edges*: we denote by $inSet_i(p)$ the set of all vertices q such that $q \to p$ is an (inclusion) edge in G. Also, $inDeg_i(p)$ is the number of vertices in $inSet_i(p)$.

ii. *Outgoing inclusion edges*: we denote by $outSet_i(p)$ the set of all vertices q such that $p \to q$ is an (inclusion) edge in G. Also, $outDeg_i(p)$ is the number of vertices in $outSet_i(p)$.

iii. *Incoming overlap edges*: we denote by $inSet_o(p)$ the set of all vertices q such that $q \Rightarrow p$ is an (overlap) edge in G. Moreover, $inDeg_o(p)$ is the number of vertices in $inSet_o(p)$.

iv. *Outgoing overlap edges*: we denote by $outSet_o(p)$ the set of all vertices q such that $p \Rightarrow q$ is an (overlap) edge in G. Also, $outDeg_o(p)$ is the number of vertices in $outSet_o(p)$.

We define now the DOI graphs-based version of our gene assembly operations: *simple graph negative rule* sgn, *simple graph positive rule* sgp, and *simple graph double rule* sgd.

Definition 5. Let $G = (V, \sigma, E_o, E_i)$ be a DOI graph be a directed, vertex- and edge-labeled graph. For any distinct vertices $p, q \in V \setminus \{m\}$, the graph operations sgn_p, sgp_p and $\mathsf{sgd}_{p,q}$ are defined on G as follows:

(i) *The simple graph negative rule* sgn for p, denoted sgn_p, is applicable to G if $\sigma(p) = -$ and $inDeg_o(p) = outDeg_o(p) = outDeg_i(p) = 0$. In this case, $\mathsf{sgn}_p(G) = G \setminus \{p\}$.

We denote $\mathsf{Sgn} = \{\mathsf{sgn}_p \mid p \in \Delta_k, k \geq 2\}$. We say that sgn_p *corresponds* to a string-rewriting rule ssn_p.

(ii) *The simple graph positive rule* sgp for p, denoted sgp_p, is applicable to G if $\sigma(p) = +$, $inDeg_o(p) + outDeg_o(p) = 1$, and $outDeg_i(p) = 0$. Let q be the vertex with the property $inSet_o(p) \cup outSet_o(p) = \{q\}$. In this case, $\mathsf{sgp}_p(G)$ is the graph obtained from $G \setminus \{p\}$ by switching the label of q: q is negative in $\mathsf{sgp}_p(G)$ if and only if it is positive in G.

We denote $\mathsf{Sgp} = \{\mathsf{sgp}_p \mid p \in \Delta_k, k \geq 2\}$. We say that sgp_p *corresponds* to a string-rewriting rule ssp_p.

(iii) *The simple graph double rule* sgd for p, q, denoted $\mathsf{sgd}_{p,q}$, is applicable to G if:

- $\sigma(p) = \sigma(q) = -$,

- $q \in outSet_o(p)$,
- $inSet_o(p) \cup p = inSet_o(q)$,
- $outSet_o(p) = outSet_o(q) \cup q$,
- $inSet_i(p) = inSet_i(q)$ and
- $outSet_i(p) = outSet_i(q)$.

In this case, $\mathsf{sgd}_{p,q}(G) = G \setminus \{p, q\}$.

We denote $\mathsf{Sgd} = \{\mathsf{sgd}_{p,q} \mid p, q \in \Delta_k, k \geq 2\}$. We say that $\mathsf{sgd}_{p,q}$ corresponds to a string-rewriting rule $\mathsf{ssd}_{p,q}$.

Definition 6. Let G be a DOI graph, G is said to be *reduced* by a composition ϕ of operations from $\mathsf{Sgn} \cup \mathsf{Sgp} \cup \mathsf{Sgd}$ if $\phi(G)$ consists of the isolated negative vertex m only.

Example 4. Consider the DOI graph G corresponding to $u = m234566m\overline{32}45$, see Figure 4(a). Clearly, sgn_6 is applicable to G since vertex 6 is negative and $inDeg_o(6) = outDeg_o(6) = outDeg_i(6) = 0$. The result is the graph G' illustrated in Figure 4(b): vertex 6 is removed with all its incident edges. We can apply to G' operation $\mathsf{sgd}_{4,5}$, since both 4 and 5 are negative, edge $4 \Rightarrow 5$ is present in G' and there are no other paths from 4 to 5, sets of incoming overlap edges into 4 and 5 are the same with the exception of edge $4 \Rightarrow 5$, sets of outgoing overlap edges from 4 and 5 are the same with the exception of $4 \Rightarrow 5$, and sets of incoming and outgoing include edges of 4 and 5 are equal (empty). As a result of the application of $\mathsf{sgd}_{4,5}$ we obtain graph G'' depicted in Figure 4(c): vertices 4 and 5 are removed together with all their incident edges. Then, we can apply sgp_3 to G'', since 3 is positive in G'', it has one incoming overlap edge and no outgoing edges (both overlap and inclusion). Consequently we obtain graph G''' (see Figure 4(d)) with vertex 3 removed together with its incident edges and vertex m changed its sign from *negative* to *positive*. Then, we can apply sgp_2 to G''' and finally obtain single isolated negative vertex m.

Next we prove that the string-based model for simple gene assembly and the DOI graph-based one are equivalent.

Theorem 5. *Let u be a legal string, G_u its DOI graph, and $\psi \in \mathsf{Ssn} \cup \mathsf{Ssp} \cup \mathsf{Ssd}$. Let $\phi \in \mathsf{Sgn} \cup \mathsf{Sgp} \cup \mathsf{Sgd}$ be the DOI graph operation corresponding to ψ. Then ϕ is applicable to G_u and $G_{\psi(u)} = \phi(G_u)$.*

Proof. We prove the claim inductively.

Case 1: $\psi = \mathsf{ssn}_p$**.** In this case $u = u_1 p p u_2$, for some strings u_1, u_2. Thus there are no overlap edge incident to p in G_u no inclusion edges starting from p.

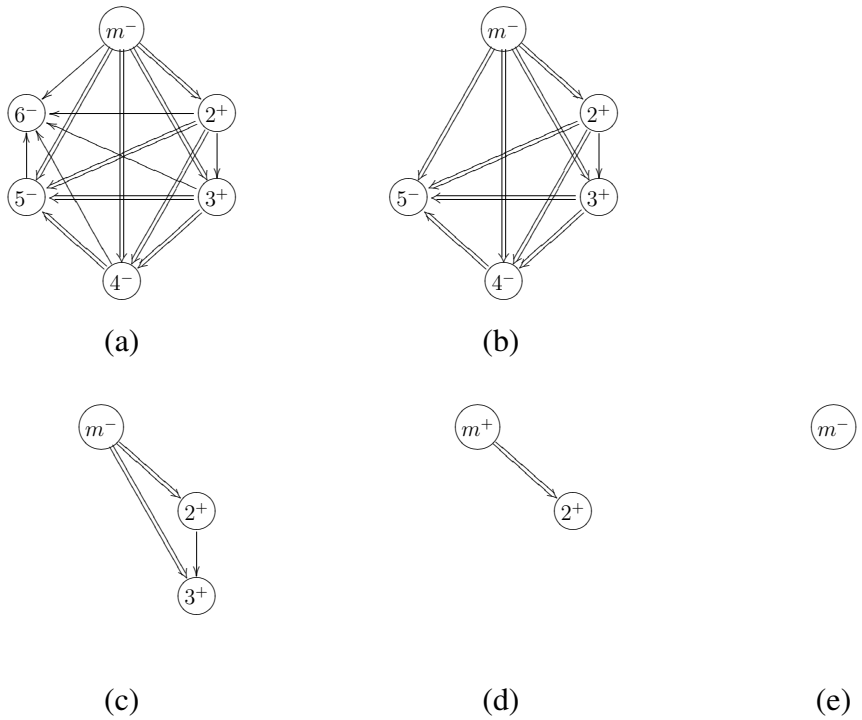

Figure 4: Reducing overlap-inclusion graph: (a) G is the DOI graph corresponding to $u = m234566m\overline{32}45$, (b) $G' = \mathsf{sgn}_6(G)$ corresponds to $u' = \mathsf{ssn}_6(u) = m2345m\overline{32}45$, (c) $G'' = \mathsf{sgd}_{4,5}(G')$ corresponds to $u'' = \mathsf{sdr}_{4,5}(u') = m23m\overline{32}$, (d) $G''' = \mathsf{sgp}_3(G'')$ corresponds to $u''' = \mathsf{spr}_3(u'') = m2\overline{m}\overline{2}$, (e) $G^{iv} = \mathsf{sgp}_2(G''')$ corresponds to $u^{iv} = \mathsf{spr}_2(u''') = mm$.

Thus, $\mathsf{sgn}_p(u)$ is applicable to G_u; $\mathsf{sgn}_p(u)$ removes the pointer p and any possible incoming inclusion relations, yielding the graph corresponding to $\mathsf{sgn}_p(G_u)$.

Case 2: $\psi = \mathsf{ssp}_p$**.** In this case $u = u_1pu_2pu_3$, for some strings u_1, u_2, u_3, where $|u_2| = 1$. Thus p must have a single (outgoing or incoming) overlap edge, and cannot have any outgoing inclusion edges. Operation $\mathsf{ssp}_p(u)$ removes the pointer p and all inclusion and overlap relationships connected to it, in addition to reversing the sign of the pointer found in u_2. The result follows from the definition of $\mathsf{sgp}_p(G_u)$.

Case 3: $\psi = \mathsf{ssd}_{p,q}$**.** In this case $u = u_1pqu_2pqu_3$. Thus, p, q overlap in u and as they are adjacent to each other in the string, the corresponding vertices in G_u have the same overlap and inclusion neighbours, except for each other. Operation $\mathsf{ssd}_{p,q}(u)$ removes both pointers, thus yielding the DOI graph corresponding to $\mathsf{sgd}_{p,q}(G_u)$. □

Theorem 6. *Let G be a DOI graph corresponding to a legal string u. Let $G' = \phi(G)$, where $\phi \in \mathsf{Sgn} \cup \mathsf{Sgp} \cup \mathsf{Sgd}$ and ψ is the string-based operation corre-*

sponding to ϕ. Then ψ is applicable to u and $\psi(u)$ is a string corresponding to $\phi(G)$.

Proof. We discuss the following three cases.

Case 1: $\phi = \mathsf{sgn}_p(G)$. Vertex p has no overlap and no outgoing inclusion edge, therefore, any corresponding string must have the form $u = u_1 p p u_2$. The claim follows since $\mathsf{ssn}_p(u) = u_1 u_2$.

Case 2: $\phi = \mathsf{sgp}_p(G)$. Vertex p has only one overlap edge, either outgoing or incoming and no outgoing inclusion edge, therefore, any corresponding string must have the form $u = u_1 p q \overline{p} u_2$, or $u = u_1 \overline{p} q p u_2$. To obtain G' from G, vertex p is removed and the signing of the vertex which is overlap-adjacent to p will be reversed. The claim follows since $\mathsf{ssp}_p(u) = u_1 \overline{q} u_2$.

Case 3: $\phi = \mathsf{sgd}_{p,q}(G)$. By the definition of $\mathsf{sgd}_{p,q}$, we have that $\sigma(p) = \sigma(q) = -$, $q \in outSet_o(p)$ and $inSet_o(p) \cup p = inSet_o(q)$, $outSet_o(p) = outSet_o(q) \cup q$, $inSet_i(p) = inSet_i(q)$ and $outSet_i(p) = outSet_i(q)$. It follows then that $u = u_1 p \alpha q u_2 p \beta q u_3$, for some string $u_1, u_2, u_3, \alpha, \beta$. We claim that α and β are both empty. If α were not empty, then let r be a pointer occurring in α. Depending on where the other occurrence from the pointer set $\{r, \overline{r}\}$, we get a contradiction with the relationships above on the neighborhoods of p and$q$. A similar argument holds for β. Thus, $u = u_1 p q u_2 p q u_3$. If $G' = \mathsf{sgd}_{p,q}(G)$, then in G', p, q are removed, as well as all the edges incident to them, yielding the DOI graph corresponding to $\mathsf{ssd}_{p,q}(u) = u_1 u_2 u_3$. □

Corollary 7. *Let G be a DOI graph and $\phi \in \mathsf{Sgn} \cup \mathsf{Sgp} \cup \mathsf{Sgd}$ applicable to G. Then $\phi(G)$ is also a DOI graph.*

Example 5. Consider string $u = b234566e\overline{32}45$ from Example 4. Its overlap-inclusion graph G is presented in Figure 4(a). If we apply ssn_6 to u, we obtain string $u' = b2345e\overline{32}45$ to which graph $G' = \mathsf{sgn}_6(G)$ from Figure 4(b) corresponds. If we apply to u' operation $\mathsf{ssd}_{4,5}$, then we obtain string $u'' = b23e\overline{32}$ to which graph $G'' = \mathsf{sgd}_{4,5}(G')$ from Figure 4(c) corresponds. By applying ssp_3 to u'' we obtain string $u''' = b2\overline{e}\overline{2}$ to which graph $G''' = \mathsf{sgp}_3(G'')$ from Figure 4(d) corresponds. Finally, by applying ssp_2 to u''' we obtain string be which corresponds to graph $\mathsf{sgp}_2(G''')$ consisting of a single negative vertex m, see Figure 4(e).

5 Discussion

In this paper we introduced a graph-based formalization of the simple gene assembly operations. The formalization is based on a new type of graphs – DOI graphs – as a model for the pointer structure of ciliate genes. The DOI graphs extend a closely related type of graph, overlap-inclusion graphs, introduced in [3]

for the same purpose. While the overlap graphs were used successfully in [3] to represent two of the three simple gene assembly operations, they could not be used for defining the third operation, the so-called simple double rule, called also the simple double-loop alternating direct repeat rule in [7]. We proved here that our DOI graph-based rules are equivalent with the string-based rewriting system for simple gene assembly, thus capturing successfully all three operations of the model, answering to the open problem formulated in [3].

A number of interesting problems remain open in connection to the DOI graphs and the graph-based simple operations; we only mention here one, concerned with the characterization of the reduction power of the simple operations: which graphs can be reduced by using only negative rules, only positive rules, only double rules, only negative and positive rules, etc. The power of the negative and the positive rules (both by themselves and in combination with each other) was characterized in [3] in terms of overlap-inclusion graphs and the results can be extended to DOI graphs as well; the counterparts of these results where double rules are involved remain open.

References

[1] S. Azimi, T. Harju, M. Langille, I. Petre, and V. Rogojin. Graph-based modeling of simple gene assembly in ciliates. *Fundamenta Informaticae*, 110, 2011.

[2] R. Brijder, T. Harju, N. Jonoska, I. Petre, and G. Rozenberg. Gene assembly in ciliates. *Handbook of Natural Computing,*, to appear, 2011.

[3] R. Brijder and H.J. Hoogeboom. Combining overlap and containment for gene assembly in ciliates. *Theoretical Computer Science*, 411(6):897–905, 2010.

[4] R. Brijder, M. Langille, and I. Petre. A string-based model for simple gene assembly. In *Fundamentals of Computation Theory*, pages 161–172. Springer, 2007.

[5] R. Brijder, M. Langille, and I. Petre. Extended strings and graphs for simple gene assembly. *Theoretical Computer Science*, 411(4-5):730–738, 2010.

[6] A.R.O. Cavalcanti, T.H. Clarke, and L.F. Landweber. Mds_ies_db: a database of macronuclear and micronuclear genes in spirotrichous ciliates. *Nucleic acids research*, 33(suppl 1):D396, 2005.

[7] A. Ehrenfeucht, T. Harju, I. Petre, D.M. Prescott, and G. Rozenberg. *Computation in living cells: gene assembly in ciliates*. Springer, 2004.

[8] A. Ehrenfeucht, I. Petre, D.M. Prescott, and G. Rozenberg. Universal and simple operations for gene assembly in ciliates. *Where mathematics, computer science, linguistics, and biology meet: essays in honour of Gheorghe Păun*, page 329, 2001.

[9] A. Ehrenfeucht, D.M. Prescott, and G. Rozenberg. Computational aspects of gene (un) scrambling in ciliates. *Springer-Verlag, Berlin, Heidelberg*, pages 216–256, 2001.

[10] T. Harju, I. Petre, V. Rogojin, and G. Rozenberg. Patterns of simple gene assembly in ciliates. *Discrete Applied Mathematics*, 156(14):2581–2597, 2008.

[11] T. Harju and G. Rozenberg. Computational processes in living cells: Gene assembly in ciliates. *Lecture Notes in Computer Science*, 2450:1–20, 2003.

[12] I. Petre and G. Rozenberg. Gene assembly in ciliates. *Scholarpedia*, 5(1):9269, 2010.

[13] D.M. Prescott, A. Ehrenfeucht, and G. Rozenberg. Molecular operations for dna processing in hypotrichous ciliates. *European Journal of Protistology*, 37(3):241–260, 2001.

Joukahaisenkatu 3-5 B, 20520 Turku, Finland | www.tucs.fi

University of Turku
- Department of Information Technology
- Department of Mathematics

Åbo Akademi University
- Department of Information Technologies

Turku School of Economics
- Institute of Information Systems Sciences

ISBN 978-952-12-2688-5
ISSN 1239-1891