Some Properties of Ciliate Bio-operations*

Mark Daley and Lila Kari

Department of Computer Science,
University of Western Ontario,
London, ON, N6A 5B7 Canada,
{lila,daley}@csd.uwo.ca

Abstract. The process of gene unscrambling in ciliates (a type of unicellular protozoa), which accomplishes the difficult task of re-arranging gene segments in the correct order and deleting non-coding sequences from an "encrypted" version of a DNA strand, has been modeled and studied so far from the point of view of the computational power of the DNA bio-operations involved. Here we concentrate on a different aspect of the process, by considering only the linear version of the bio-operations, that do not involve thus any circular strands, and by studying the resulting formal operations from a purely language theoretic point of view. We namely investigate closure operations of language families under the mentioned bio-operations and study language equations involving them. Among the problems addressed, we study the decidability of existence of solutions to equations of the form $L \diamond Y = R$, $X \diamond L = R$ where L and R are given languages, X and Y are unknowns, and $\diamond$ signifies one of the defined bio-operations.

1 Introduction

The stichotrichous ciliates are a group of primitive single-celled organisms which have generated a great deal of interest due to their unique genetic mechanisms. Most organisms store their genomic DNA in a linear sequence consisting of coding regions interspersed with non-coding regions. Several ciliate genes, however, are stored in a scrambled form. For example, if a functional copy of a gene consists of the coding regions arranged in the order 1-2-3-4-5, it may appear in the order 3-5-4-1-2 in the genome. This presents an interesting problem for the organism, who must somehow descramble these genes in order to generate functional proteins required for its continued existence.

The details of the biological mechanism underlying the unscrambling process are still unknown. For further information on the biology of the descrambling process in ciliates the reader is referred to [12,13,14]. The two existing formal models for gene unscrambling, by Kari and Landweber [5,10], respectively Ehrenfeucht, Harju, Petre, Prescott and Rozenberg [11,3,15] are consistent with the existing

* Research supported by Natural Sciences and Engineering Council of Canada Grants. All correspondence to L.K.

M. Ito and M. Toyama (Eds.): DLT 2002, LNCS 2450, pp. 116–127, 2003.

biological data. Each proposes a set of two, respectively three, atomic operations the combination of which can lead to the unscrambling of an arbitrarily scrambled gene. The bio-operations proposed by the first model are circular insertions and deletions, i.e. insertions and deletions of circular strands into/from linear strands, guided by the presence of certain pointers [5,10]. The second model focuses more on the properties of pointers and proposes three operations: hi(hairpin loop with inverted pointers) which reverses a substring between two inverted pointer sequences, ld(loop with direct pointers)-excision which deletes a substring between two pointers and dlad(double loop with alternating direct pointers)-excision/reinsertion which swaps two substrings marked by pointer-pairs. In both cases, the operations presented are based on real biological events that can occur and change a DNA molecule.

This paper does not address the biological aspects and implications of the proposed operations. Instead, we continue in the style of Dassow et. al's work on properties of operations inspired by general DNA recombination events [2] and focus on some of their properties as word/language operations. We namely consider, in Section 2, closure properties of languages in the Chomsky hierarchy under the defined operations. Moreover, in Section 3 we consider language equations of the type $L \diamond Y = R$, $X \diamond L = R$ where L and R are given languages and X and Y are the unknowns. We study the decidability of the existence of solutions to these equations as well as the existence of singleton solutions.

The notations used in the paper are summarized as follows. An alphabet Σ is a finite non-empty set. A word w over Σ is an element of the free semigroup (denoted Σ^+) generated by the letters of Σ and the catenation operation. The length of a word, written $|w|$ is equal to the number of letters in the word. In the free monoid Σ^* we also allow the empty word λ where $|\lambda| = 0$. A language L is a, possibly infinite, set of words over a given alphabet. The complement of a language L is written L^c and is defined as $L^c = \Sigma^* \setminus L$.

We will consider here the classic families of the Chomsky Hierarchy, that is, the families of: regular languages (REG), context-free languages (CF), context-sensitive languages (CS) and recursively enumerable languages (RE). For further details on basic formal language theory, the reader is referred to [16].

2 Closure Properties

This section will define the two bio-operations of the [5,10] model and a generalization of an operation in the [11,3,15] model and investigate closure properties of families in the Chomsky hierarchy under them.

2.1 Synchronized Insertion and Deletion

Two basic operations have been defined in [5,10] that model the processes of intramolecular respectively intermolecular DNA recombinations that are thought to accomplish gene unscrambling.

The operation modeling the intermolecular recombination accomplishes the insertion of a circular sequence vx in a linear string uxw, resulting in a string

$uxvxw$. If we ignore the fact that the inserted string is circular, the operation, called *synchronized insertion* (the word "synchronized" points out that insertion can only happen if the sequence x is present in both input strands), is formally defined as follows.

Definition 1. *Let α, β be two nonempty words in Σ^*. The synchronized insertion of β into α is defined as: $\alpha \oplus \beta = \{uxvxw | \alpha = uxw, \beta = vx, x \in \Sigma^+, u, v, w \in \Sigma^*\}$.*

The operation modeling intramolecular recombination accomplishes the deletion of a sequence vx from the original strand $uxvxw$, in the form of a circular strand. Ignoring the differences between the linear and circular strands, the resulting operation, that of *synchronized deletion*, is defined as follows.

Definition 2. *Let α, β be two nonempty words in Σ^*. The synchronized deletion of β from α is defined as: $\alpha \ominus \beta = \{uxw | \alpha = uxvxw, \beta = vx, x \in \Sigma^+, u, v, w \in \Sigma^*\}$.*

The operations differ from the original ones defined in [10] and [5] in that no circular strands are present here. The above two definitions can be extended to languages in the natural way. This section examines the closure properties of the families of regular, context-free, context-sensitive and recursively enumerable languages under the synchronized insertion and deletion.

We begin by recognizing that we can consider, without loss of generality, only one-symbol contexts instead of arbitrarily-sized contexts.

Lemma 1. *For any $\alpha, \beta \in \Sigma^+$, $\alpha \oplus \beta = \{u'av'aw' | \alpha = u'aw', \beta = v'a, a \in \Sigma, u', v', w' \in \Sigma^*\}$.*

Using the preceding result we can now show that the synchronized insertion can be expressed in terms of the controlled sequential insertion, defined in [9] as follows. Let $L \subseteq \Sigma^+$ and, for each $a \in \Sigma$, let $\Delta(a) \subseteq \Sigma^*$. The controlled sequential insertion into L, according to Δ, is defined as $L \longleftarrow \Delta = \cup_{u \in L}(u \longleftarrow \Delta)$, where $u \longleftarrow \Delta = \{u_1 a v_a u_2 |\ u = u_1 a u_2, v_a \in \Delta(a)\}$.

Proposition 1. *REG, CF, CS and RE are closed under synchronized insertion.*

Proof. We claim that $L_1 \oplus L_2 = L_1 \longleftarrow \Delta$ where $\Delta(a) = (L_2\{a\}^{-1})\ \{a\}$, and the right quotient of two languages in Σ^*, denoted by $L_1 L_2^{-1}$, is defined as

$$L_1 L_2^{-1} = \{u |\ uv \in L_1, v \in L_2\}.$$

"$\subseteq$" Given $\gamma \in L_1 \oplus L_2$, by Lemma 1 γ is of the form $uavaw$ where $uaw \in L_1$, $va \in L_2$. As $va \in L_2$, $v \in L_2\{a\}^{-1}$ and therefore $va \in (L_2\{a\}^{-1})\{a\}$. Consequently, $\gamma = uavaw \in L_1 \longleftarrow \Delta$.

"$\supseteq$" Suppose $\gamma \in L_1 \longleftarrow \Delta$. Then γ is of the form $uav_a w$ where $uaw \in L_1$ and $v_a \in \Delta(a)$. Since $\Delta(a) = (L_2\{a\}^{-1})\{a\}$, $v_a = va$, $v \in L_2/\{a\}$ and therefore $va \in L_2$. Consequently, $\gamma \in uaw \oplus va$, $uaw \in L_1, va \in L_2$, $a \in \Sigma^+, u, v, w \in \Sigma^*$.

Since REG, CF, CS, RE are closed under controlled sequential insertion [9], they are closed under synchronized insertion as well.

The closure of the family of regular languages under synchronized deletion can be similarly ascertained by expressing synchronized deletion in terms of the controlled sequential deletion, [9]. The controlled sequential deletion is defined similarly to the controlled sequential insertion, with the difference that

$$u \longrightarrow \Delta = \{u_1 a u_2 |\ u = u_1 a v_a u_2, u_1, u_2 \in \Sigma^*, a \in \Sigma, v_a \in \Delta(a)\}.$$

By [9], REG are closed under controlled sequential deletion, while CF is not (but it is closed under controlled sequential deletion with regular languages) and CS is not even closed under controlled sequential deletion with regular languages (but it is closed under controlled sequential deletion with singleton languages). The first of these closure properties leads to the next closure result, preceded by a lemma that simplifies synchronized deletion similarly to the synchronized insertion.

Lemma 2. *For any* $\alpha, \beta \in \Sigma^+$ $\alpha \ominus \beta = \{u'aw | \alpha = u'av'aw, \beta = v'a, a \in \Sigma, u', v', w \in \Sigma^*\}$.

We are now ready to address the closure properties of the families in the Chomsky hierarchy under synchronized deletion.

Proposition 2. *REG and RE are closed under synchronized deletion.*

Proof. We claim that $L_1 \ominus L_2 = L_1 \longrightarrow \Delta$ where $\Delta(a) = (L_2\{a\}^{-1})\{a\}$.

"$\subseteq$" Given $\gamma \in L_1 \ominus L_2$, by Lemma 2, we have $\gamma = uaw$ where $uavaw \in L_1$ and $va \in L_2$. Given that $\Delta(a) = (L_2\{a\}^{-1})\{a\}$, we have $va \in \Delta(a)$ and thus $\gamma = uaw \in L_1 \longrightarrow \Delta$.

"$\supseteq$" Suppose $\gamma \in L_1 \longrightarrow \Delta$ where $\Delta(a) = (L_2\{a\}^{-1})\{a\}$. Then γ is of the form uaw where $uavaw \in L_1$ and $va \in (L_2\{a\}^{-1})\{a\}$, thus $\gamma \in L_1 \ominus L_2$. The result follows as REG and RE are closed under controlled sequential deletion [9].

Proposition 3. *CF is not closed under synchronized deletion.*

Proof. Let L_1, L_2 be the context free languages:

$$L_1 = \#(\{a^i b^{2i} | i > 0\}^* \amalg \{\#\}),$$

$$L_2 = a\{b^i a^i | i > 0\}^* \#,$$

where the shuffle operation is defined as

$$u \amalg v = \{u_1 v_1 \dots u_n v_n |\ u = u_1 \dots u_n, v = v_1 \dots v_n, u_i, v_i \in \Sigma^*, 1 \leq i \leq n\}.$$

(Similar languages were used in [4] to show that CF is not closed under left quotient).

The language

$$(L_1 \ominus L_2) \cap \#b^* = \{b^{2^n} | n > 0\}$$

is not context-free and the result follows.

Proposition 4. *CS is not closed under synchronized deletion with regular languages.*

Proof. Let $L \subseteq \Sigma^*$, with $a, b \notin \Sigma$, be a recursively enumerable language which is not context-sensitive.

There exists a context-sensitive language L_1 such that L_1 consists of words of the form $a^i b\alpha$ where $i \geq 0$ and $\alpha \in L$. Furthermore, for all $\alpha \in L$ there exists some $i \geq 0$ such that $a^i b\alpha \in L_1$ [16].

Suppose $\#$ is a symbol not in $\Sigma \cup \{a, b\}$. Consider the language

$$\#(L_1 \amalg \{\#\}) \ominus a^*b\#.$$

Clearly, $a^*b\#$ is regular and moreover, $\#(L_1 \amalg \{\#\}) \ominus a^*b\# = \#L$ which, by the definition of L, is not context-sensitive.

Even though CS is not closed under synchronized deletion with regular languages, it is closed under synchronized deletion with singleton languages. Indeed, this follows directly from Proposition 2 and the closure of context-sensitive languages under controlled sequential deletion with singleton languages [9].

2.2 Hairpin Inversion

We next consider a generalization of the hairpin inversion operation *hi* defined in [11]. The name of the operation reflects the fact that it models the process of a DNA strand forming a hairpin, having the end of the hairpin cleaved and then re-attached with the sticky ends switched. This results in the sequence of the cleaved and re-attached region now being the mirror image of what it was prior to the operation.

If $w = a_1 a_2 \dots a_n$, $a_i \in \Sigma$, $1 \leq i \leq n$ is a word in Σ^+ the *reverse* or *mirror image* of w is denoted by $\tilde{w}$ and defined as $\tilde{w} = a_n \dots a_2 a_1$.

Definition 3. *Let α be a word in Σ^+. The hairpin inverse of α, denoted by $hi(\alpha)$ is defined as $hi(\alpha) = \{xp\tilde{y}\tilde{p}z | \alpha = xpypz$ and $x, y, z \in \Sigma^*, p \in \Sigma^+\}$.*

This definition can be extended to languages in Σ^+ in the natural way. Similarly to Lemma 1, we first show that it is enough to consider pointers of length one only.

Lemma 3. *If $\alpha \in \Sigma^+$, $hi(\alpha) = \{xa\tilde{y}az | \alpha = xayaz, a \in \Sigma, x, y, z \in \Sigma^*\}$.*

The mirror image operation has the property that $\tilde{\tilde{L}} = L$. The hairpin inversion operation is a variation of the mirror image operation in that it inverts subwords inside words of a language. The following lemma answers the question of whether or not applying hairpin inversion twice to a language yields the original language. As it turns out, while the hairpin invertible words of a language L are included in $hi(hi(L))$, the reverse does not hold.

Lemma 4. *If $L \subseteq \Sigma^+$, then for all $a \in \Sigma$, $L \cap \Sigma^* a \Sigma^* a \Sigma^* \subseteq hi(hi(L))$ while $hi(hi(L))$ is not included in $L \cap \Sigma^* a \Sigma^* a \Sigma^*$.*

The following propositions address the closure properties of families in the Chomsky hierarchy under the operation of hairpin inversion.

Proposition 5. *REG is closed under hairpin inversion.*

Proof. Let L be the regular language accepted by an automaton $A = (\Sigma, S, s_o, s_f, P)$, where Σ is the alphabet, S is the set of states, s_0 is the initial state, s_f is the final state, and P is the set of productions of the form $sa \longrightarrow s'$, $s, s' \in S$, $a \in \Sigma$. For every two states $s_i, s_j \in S$, define $L_{s_i,s_j} = \{w \in \Sigma^* | s_i w \Rightarrow^* s_j\}$. In other words, L_{s_i,s_j} consists of those words which will cause the automaton A to move from state s_i to s_j when read. We claim that

$$hi(L) = \bigcup_{s_i,s_j,s_k,s_l \in S} \bigcup_{a \in L_{s_i,s_j} \cap L_{s_k,s_l} \cap \Sigma} L_{s_0,s_i}\{a\}\widetilde{L_{s_j,s_k}}\{a\}L_{s_l,s_f}.$$

"$\subseteq$" Let $\alpha \in hi(L)$. Then there exists $xayaz \in L, a \in \Sigma, x, y, z \in \Sigma^*$ such that $\alpha = xa\tilde{y}az$. As $xayaz \in L(A) = L$, there exists a derivation $s_0 xayaz \Rightarrow^* s_f$, *i.e.* there exist $s_i, s_j, s_k, s_l \in S$ such that $s_0 xayaz \Rightarrow^* s_i ayaz \Rightarrow^* s_j yaz \Rightarrow^* s_k az \Rightarrow^* s_l z \Rightarrow^* s_f$.

This implies $x \in L_{s_0,s_i}, a \in L_{s_i,s_j}, y \in L_{s_j,s_k}, a \in L_{s_k,s_l}, z \in Ls_e, s_f$.

We then have $\tilde{y} \in \widetilde{L_{s_j,s_k}}$ and $\alpha \in L_{s_0,s_i} a \widetilde{L_{s_j,s_k}} a L_{s_l,s_f} \subseteq$ RHS.

"$\supseteq$" Consider $\alpha \in$ RHS. Take $x \in L_{s_0,s_i}$, $\tilde{y} \in \widetilde{L_{s_j,s_k}}$, $z \in L_{s_l,s_f}$ and $a = a$. This means $xayaz \in L_{s_0,s_i} L_{s_i,s_j} L_{s_j,s_k} L_{s_k,s_e} L_{s_l,s_f}$ which means there exists a derivation $s_0 xayaz \Rightarrow^* s_f$ which implies $xayaz \in L$ and $xa\tilde{y}az \in hi(L)$. The proposition now follows as REG is closed under finite union, catenation and mirror image.

Proposition 6. *CF is not closed under hairpin inversion.*

Proof. Consider the language $L = \{wpc\tilde{w}dp | w \in \{a, b\}^*, p, c, d \in \Sigma\}$ where $\Sigma = \{a, b, p, c, d\}$ and $\tilde{w}$ denotes the mirror image of w.

Clearly L is context-free. However,

$$hi(L) \cap \Sigma^* pd\Sigma^* cp = \{wpdwcp | w \in \{a, b\}^*, p, c, d \in \Sigma\}$$

which is a classic example of a non context-free language.

Proposition 7. *CS is closed under hairpin inversion.*

Proof. Let $L = L(G)$ where $G = (N, T, S, P)$ is generated by a context-sensitive grammar in a normal form where every production in P containing letters of T is of the form $X \to a$ where $X \in N^*, a \in T$, [16].

We construct a context-sensitive grammar $G' = (N', T', S', P')$ as follows: $N' = N \cup \{X_a, Y_a, Z_a, B_a, a'', a' | a \in T\} \cup \{S', \$, C, C', C'', B, B'\}$, $P' = \{A \to X_a | A \to a \in P\} \cup \{A \to Y_a | A \to a \in P\} \cup (P \setminus \{A \to a | A \to a \in P\}) \cup \{X_a \to a, Y_a \to a, a' \to a\}$, $T' = T \cup \{\#\}$.

In addition, for all $a, d \in \Sigma$, P' contains the following rules:

1. $S' \to \#BS\#$
2. $Ba' \to a'B$ (B will have to check if the sentential form is in the correct form. Here B traverses x' in a sentential form $\#Bx'X_ay'Y_az'\#$ corresponding to a word $xayaz \in L(G)$).
3. $BX_a \to X_aB_a$ (B meets X_a, this change is registered by its transformation in B_a which stores knowledge about a in its subscript.)
4. $B_ac' \to c'B_a$ (B_a reads y')
5. $B_aY_a \to Y_aB'$ (B_a reads Y_a if Y_a has same index with X_a)
6. $B'a' \to a'B'$ (B' reads z')
7. $B'\# \to C\#$ (B' reaches the end of the sentential form and changes to C)
8. $\alpha C \to C\alpha$, $\alpha \in \Sigma' \cup \{Y_a\}$ (C moves left until it reaches X_a, when it will start reversing y')
9. $X_aC \to X_aC'$ (C' starts to reverse the word y')
10. $C'd' \to Z_dC'$ (Store d' in Z_d, where d' is the first letter of y')
11. $Z_dC'b' \to b'Z_dC'$ (Z_dC' moves right until it reaches Y_a)
12. $Z_dC'Y_a \to d''C''Y_a$ (the move of the first letter d of y' to the end of the word has been completed)
13. $\alpha C'' \to C''\alpha, \alpha \in \{a'', a'|a \in \Sigma\}$ (C'' moves left until it reaches X_a)
14. $X_aC'' \to X_aC'$ (start again)
15. $Z_dC'b'' \to b''Z_dC'$ (Z_dC' should be able to move also over b'')
16. $X_aC'd'' \to X_a\$d''$ (When there are no letters in y' to invert anymore, transform C' into \$).
17. $\$\alpha \to \alpha\$$, $\alpha \in \{a', a'', Y_a|a \in \Sigma\}$, $\$\# \to \#\#$ (the dollar sign moves to the left until it reaches # when it changes into #).
18. $X_a \to a, Y_a \to a, a' \to a, a'' \to a$.

We claim that $L(G') = \#hi(L)\#\#$. Indeed, a derivation according to G' can only proceed as follows.

$S' \overset{1}{\Rightarrow} \#BS\# \overset{P}{\Rightarrow}{}^* \#Bx'X_ay'Y_az'\# \overset{2}{\Rightarrow}{}^* \#x'BX_ay'Y_az'\# \overset{3}{\Rightarrow}$
$\#x'X_aB_ay'Y_az'\# \overset{4}{\Rightarrow}{}^* \#x'X_ay'B_aY_az'\# \overset{5}{\Rightarrow} \#x'X_ay'Y_aB'z'\# \overset{6}{\Rightarrow}{}^*$
$\#x'X_ay'Y_az'B'\# \overset{7}{\Rightarrow} \#x'X_ay'Y_az'C\# \overset{8}{\Rightarrow}{}^* \#x'X_aCy'Y_az'\# \overset{9}{\Rightarrow}$
$\#x'X_aC'y'Y_az'\# = \#x'X_aC'd'y'_1Y_az'\# \overset{10}{\Rightarrow} \#x'X_aZ_dC'y'_1Y_az'\# \overset{11}{\Rightarrow}{}^*$
$\#x'X_ay'_1Z_dC'Y_az'\# \overset{12}{\Rightarrow} \#x'X_ay'_1d''C''Yaz'\# \overset{13}{\Rightarrow}{}^* \#x'X_aC''y'_1d''Y_az'\# \overset{14}{\Rightarrow}$
$\#x'X_aC'y'_1d''Y_aZ'\# \overset{8-14}{\Rightarrow}{}^* \#x'X_aC'\tilde{y''}Y_az'\# \overset{16}{\Rightarrow} \#x'X_a\$\tilde{y''}Y_az'\# \overset{17}{\Rightarrow} \#x'\tilde{y''}Y_az'\$$
$\# \overset{17}{\Rightarrow} \#x'X_a\tilde{y''}Y_az''\#\# \overset{18}{\Rightarrow}{}^* \#xa\tilde{y}az\#\#$. The proposition now follows as $\#hi(L)\#\#$ is a context-sensitive language and therefore $hi(L)$ is a context-sensitive language as it is the image of $\#hi(L)\#\#$ through a homomorphism that erases # and leaves all other letters unchanged.

3 Language Equations

We begin this section by investigating equations of the type $hi(X) = R$, where R is a given language and X is the unknown.

Proposition 8. *Let $R \subseteq \Sigma^*$ be regular language. If there exists a language $L \subseteq \Sigma^*$ such that $hi(L) = R$ then there exists a regular language $R', L \subseteq R' \subseteq \Sigma^*$, with the same property.*

Proof. Construct the language $R' = [hi(R^c)]^c$.

(i) We show that $hi(R') \subseteq R$ by way of contradiction. Assume there exists some $u \in hi(R')$ such that $u \notin R$. Since $u \notin R$, it must be the case that $u \in R^c$. As $u \in hi(R')$ it must be of the form $u = x\tilde{a}yaz$ where $xayaz \in R'$. However, $u = x\tilde{a}yaz$ implies $xayaz \in hi(u) \subseteq hi(R^c)$ a contradiction since $R' = [hi(R^c)]^c$.

(ii) We show now that every language $L \subseteq \Sigma^*$ such that $hi(L) \subseteq R$ is included in R'. Indeed, assume there exists $L \subseteq \Sigma^*$ with $hi(L) \subseteq R$ but $L \not\subseteq R'$. Then there must exist a word $u \in L \setminus R'$. As $u \notin R', u \in hi(R^c)$ implies that $u = x\tilde{a}yaz$ with $xayaz \in R^c$. However, as $u \in L$, by the definition of hairpin inversion, $xayaz \in hi(L) \subseteq R$ a contradiction.

Return now to the proof of the proposition. If there exists L with $hi(L) = R$ then, by (ii), $L \subseteq R'$. By (i) we have that $R = hi(L) \subseteq hi(R') \subseteq R$ which means $hi(R') = R$. By Proposition 5, and closure of REG under complement, R' is regular. Moreover, it follows from the proof that R' can be effectively constructed.

The preceding proposition aids us in deciding whether an equation $hi(X) = R$ has a solution X in case R is a regular language.

Proposition 9. *If $R \subseteq \Sigma^*$ is a regular language, the problem of whether or not the equation $hi(X) = R$ has a solution $X \subseteq \Sigma^*$ is decidable.*

Proof. Construct $R' = [hi(R^c)]^c$. If $hi(X) = R$ has a solution then R' is also, by Proposition 8, a solution. An algorithm for deciding our problem will consist in effectively constructing R' and then checking whether or not $hi(R') = R$. The problem is thus decidable as the equality of regular languages is decidable.

We now investigate equations of the form $X \diamond L = R$, $L \diamond Y = R$ where L and R are given languages, X and Y unknowns, and $\diamond$ signifies the synchronized insertion or deletion operation. To find their solutions, we proceed similarly to solving algebraic equations $x + a = b$. Namely, we must employ an operation "inverse" to addition (in this case subtraction) to determine the solution $x = b - a$. As, unlike addition, the operations of synchronized insertion and deletion are not commutative, we will need to define two separate notions: the notion of a left inverse for solving equations $X \diamond L = R$, and of right inverse for solving equations of the form $L \diamond Y = R$. In the interest of space, we will omit proofs in the sequel.

Definition 4. *Let $\diamond$, $*$ be two binary word operations. The operation $*$ is said to be the left-inverse of the operation $\diamond$ if, for all words u,v,w over the alphabet Σ, the following relation holds:*

$$w \in (u \diamond v) \text{ iff } u \in (w * v).$$

In other words, the operation $*$ is the left-inverse of the operation $\diamond$ if, given a word w in $u \diamond v$, the left operand u belongs to the set obtained from w and the other operand v, by using the operation $*$. The relation "is the left-inverse of" is symmetric.

Proposition 10. *The left-inverse of the operation $\oplus$ of synchronized insertion is the operation $\ominus$ of synchronized deletion.*

We can now use Proposition 10 and the following theorem, [8], to investigate solutions of language equations of the type $X \oplus L = R$ where L and R are given languages in Σ^* and X is the unknown.

Theorem 1. *Let L, R be languages over an alphabet Σ and $\diamond, *$ be two binary word (language) operations, left-inverses to each other. If the equation $X \diamond L = R$ has a solution $X \subseteq \Sigma^*$, then also the language $R' = (R^c * L)^c$ is a solution. Moreover, R' includes all the other solutions of the equation (set inclusion).*

Corollary 1. *If the equation $X \oplus L = R$ (respectively $X \ominus L = R$) has a solution, then $R' = (R^c \ominus L)^c$ (respectively $R' = (R^c \oplus L)^c$) is a maximal solution to the equation.*

We shall use the above results to investigate the decidability of the following problems: Given languages L and R over Σ, R regular, *Does there exist a solution X to the equation $X \oplus L = R$?*, and *Does there exist a singleton solution $X = \{w\}$ to the equation $X \oplus L = R$?*

Proposition 11. *The problem "Does there exist a solution X to the equation $X \oplus L = R$?," is decidable for regular languages L and R.*

Proposition 12. *The problem "Does there exist a singleton solution $X = \{w\}$ to the equation $X \oplus L = R$?"is decidable for regular languages L and R.*

The study of the existence of solutions to the equation $X \oplus L = R$, when R is regular, is completed by the following undecidability results.

Proposition 13. *The problem "Does there exist a solution X to the equation $X \oplus L = R$?"is undecidable for context-free languages L and regular languages R.*

If L is a language over an alphabet Σ, the word $x \in \Sigma^+$ is called *left-useful* with respect to $\ominus$ and L (shortly, left-useful) if there exists a $y \in L$ such that $x \ominus y \neq \emptyset$. A language X is called left-useful with respect to $\ominus$ and L (shortly, left-useful), if it consists only of left-useful words. From the above definitions it follows that the problem "Does there exist a solution X to the equation $X \ominus L = R$?" and its singleton version are equivalent to the corresponding problems where the existence of a left-useful language or word are investigated. Therefore, in the sequel, we will mean a left-useful language when referring to a language or word whose existence is sought.

An argument similar to Proposition 11, and based on the effectiveness of the proofs of closure of REG under $\oplus$ and $\ominus$, shows that the problem "Does there exist a solution X to the equation $X \ominus L = R$?" is decidable for regular languages L and R. For the context-free case the following result holds.

Proposition 14. *The problem "Does there exist a language X such that $X \ominus L = R$"is undecidable for context-free languages L and regular languages R.*

The following decidability result is basically a consequence of the fact that the result of a synchronized deletion from a word is a finite set.

Proposition 15. *The problem "Does there exist a word w such that $w \ominus L = R$?"is decidable for regular languages L and R.*

To investigate symmetric equations of the type $L \oplus Y = R$ and $L \ominus Y = R$ where L and R are given languages and Y is an unknown language, we shall make use of the following result from [8], keeping in mind that, in the case of synchronized deletion, we are actually investigating the existence of right-useful solutions (the notion is defined similarly to that of left-useful solutions).

Theorem 2. *Let L, R be languages over Σ and $\diamond, *$ be two binary word (language) operations right-inverses to each other. If the equation $L \diamond Y = R$ has a solution Y, the language $R' = (L * R^c)^c$ is a maximal solution.*

The notion of right-inverse in the preceding theorem, similar to the notion of left-inverse, is formally defined in [8] as follows.

Definition 5. *Let $\diamond, *$ be two binary word operations. The operation $*$ is said to be right-inverse of the operation $\diamond$ if, for all words u, v, w in Σ^* the following relation holds: $w \in (u \diamond v)$ iff $v \in (u * w)$.*

By using Theorem 2 we could find solutions to equations of the form $L \oplus Y = R, L \ominus Y = R$ if we found the right inverses of $\oplus$ and $\ominus$.

Definition 6. *Let $u, v \in \Sigma^+$. The synchronized bi-deletion of v from u is defined as*

$$u \boxminus v = \{w \mid u = xayaz, v = xaz, w = ya, a \in \Sigma,\ x, y, z \in \Sigma^*\}.$$

Definition 7. *Let $\diamond$ be a binary operation. The word operation $\diamond^r$ defined by $u \diamond^r v = v \diamond u$ is called reversed $\diamond$.*

We can now find out the right inverses of synchronized insertion and deletion and thus solutions to our language equations.

Proposition 16. *The right-inverse of synchronized deletion $\ominus$ is synchronized bi-deletion. The right-inverse of synchronized insertion is reversed synchronized bi-deletion.*

Corollary 2. *If the equation* $L \oplus Y = R$ *(respectively* $(L \ominus Y = R)$*) has a solution, then* $R' = (R^c \boxminus L)^c$ *(respectively* $(L \boxminus R^c)^c$*) is a maximal solution.*

Before solving our decidability problems, we need to first determine the closure properties of the families in the Chomsky hierarchy under synchronized bi-deletion.

Proposition 17. *The family of regular languages is closed under synchronized bi-deletion while the family of context-free language is not closed under synchronized bi-deletion and the family of context-sensitive languages is not closed under synchronized bi-deletion with regular languages.*

The preceding results on synchronized bi-deletion lead to the following proposition.

Proposition 18. *The problem of whether or not there exists a solution* Y *to the equations* $L \oplus Y = R$, $L \ominus Y = R$ *is decidable for regular languages* L *and* R.

Proposition 19. *The existence of a solution* Y *to the equations* $L \oplus Y = R$ *and* $L \ominus Y = R$ *is undecidable for regular languages* R *and context-free languages* L.

4 Conclusion

We have considered the properties of three operations used in the modeling of the ciliate gene descrambling process: synchronized insertion, synchronized deletion and hairpin inversion. We found that all the families of the Chomsky hierarchy are closed under synchronized insertion while only the families of regular and recursively enumerable languages are closed under synchronized deletion. Additionally we showed that only the family of context-free languages was not closed under hairpin inversion. In order to consider language equations involving each of the three operations we have also defined the operation of synchronized bi-deletion (the right-inverse of synchronized deletion) and showed only the families of regular and recursively enumerable languages to be closed under this operation.

We demonstrated that the existence of a solution X to the equation $hi(X) = R$, where R is a regular language is decidable. Additionally, the existence of a solution was shown to be decidable for equations of the form $L \diamond Y = R$ and $X \diamond L = R$ where $\diamond$ is one of synchronized insertion or synchronized deletion operations and L, R are regular languages. The same problems are undecidable in the case that L is a context-free language.

By investigating the properties of these formal operations, we have provided some insight into the nature of the bio-operations that must be present in the ciliate gene descrambling mechanism. Continued theoretical study of the gene descrambling problem combined with improved biological results will hopefully lead to a better understanding of this fascinating process.

References

1. J.M. Autebert, J. Berstel, L. Boasson 1997. Context-free languages and Pushdown Automata. *Handbook of formal languages*, 1: 111–174, Springer, Berlin.
2. J. Dassow, V. Mitrana, A. Salomaa. 2002. Operations and language generating devices suggested by the genome evolution. *Theoretical Computer Science*, 270: 701-738.
3. A. Ehrenfeucht, D.M. Prescott, G. Rozenberg. 2001. Computational aspects of gene (un)scrambling is ciliates. In *Evolution as Computation* (L.F. Landweber, E. Winfree eds.) Springer-Verlag, Berlin, Heidelberg, 45-86.
4. S. Ginsburg. 1975. *Algebraic and Automata-Theoretic Properties of Formal Languages.* North-Holland, Amsterdam.
5. L. Kari, L.F. Landweber. 2000. Computational power of gene rearrangment In *DNA5, DIMACS series in Discrete Mathematics and Theoretical Computer Science*)(E. Winfree, D. Gifford eds.), American Mathematical Society, 54: 207-216.
6. L. Kari, G. Thierrin. 1996. Contextual insertions/deletions and computability. *Information and Computation*, 131: 47–61.
7. L. Kari. 1992. Insertion and deletion of words: determinism and reversibility. *Lecture Notes in Computer Science*, 629:315-327.
8. L. Kari. 1994. On language equations with invertible operations. *Theoretical Computer Science*, 132: 129-150.
9. L. Kari. 1991. *On insertions and deletions in formal languages.* PhD thesis, University of Turku, Finland.
10. L.F. Landweber, L. Kari. 1999. The evolution of cellular computing: nature's solutions to a computational problem. *DNA4 Biosystems* (L. Kari, H. Rubin, D.H. Wood eds.), Elsevier, 52(1-3):3-13.
11. I. Petre, A. Ehrenfeucht, T. Harju and G. Rozenberg. 2002. Patterns of micronuclear genes in cilliates. In *DNA7, Lecture Notes in Computer Science* (N. Jonoska, N. Seeman eds.), Springer-Verlag, 2340: 279-289.
12. D.M. Prescott. 1992. Cutting, splicing, reordering, and elimination of DNA sequences in hypotrichous ciliates. *BioEssays*, 14(5): 317-324.
13. D.M. Prescott. 1992. The unusual organization and processing of genomic DNA in hypotrichous ciliates. *Trends in Genet.*, 8:439-445.
14. D.M. Prescott. 2000. Genome gymnastics: Unique modes of DNA evolution and processing in ciliates. *Nature Reviews Genetics*, 1:191-198.
15. D.M. Prescott, A. Ehrenfeucht, G. Rozenberg. 2001. Molecular operations for DNA processing in hypotrichous ciliates. To appear in *European Journal of Protistology.*
16. A. Salomaa. 1973. *Formal languages*, Academic Press, New York.