Solution of a Problem in DNA Computing

Eric Anderson * Marek Chrobak † John Noga ‡ Jiří Sgall §
Gerhard J. Woeginger ¶

Abstract

We answer a question of Rozenberg and Salomaa arising from a problem in DNA computing. This problem was posed at the ICALP conference in July 1999 in Prague.

1 Introduction

Let Σ be a finite alphabet. By a *word* w we will mean a sequence of letters from Σ in which each occurrence of each letter a is assigned an orientation, $\overleftarrow{a}$ or $\overrightarrow{a}$. A word w is *well-formed*, if each letter of w occurs exactly twice. For example, for $\Sigma = \{\mathtt{A}, \mathtt{B}, \mathtt{C}, \mathtt{D}\}$, among the words

$$\overrightarrow{\mathtt{B}}\,\overleftarrow{\mathtt{A}}\,\overrightarrow{\mathtt{B}}\,\overleftarrow{\mathtt{C}}\,\overleftarrow{\mathtt{C}}\,\overrightarrow{\mathtt{A}},\qquad \overrightarrow{\mathtt{A}}\,\overrightarrow{\mathtt{A}}\,\overrightarrow{\mathtt{B}}\,\overrightarrow{\mathtt{B}}\,\overleftarrow{\mathtt{D}}\,\overrightarrow{\mathtt{D}},\qquad \overrightarrow{\mathtt{A}}\,\overrightarrow{\mathtt{A}}\,\overrightarrow{\mathtt{A}}\,\overrightarrow{\mathtt{E}}\,\overrightarrow{\mathtt{E}}\,\overrightarrow{\mathtt{B}}\,\overrightarrow{\mathtt{B}}\,\overrightarrow{\mathtt{C}},\qquad \overrightarrow{\mathtt{A}}\,\overrightarrow{\mathtt{A}}\,\overleftarrow{\mathtt{A}}\,\overleftarrow{\mathtt{A}},\qquad \overleftarrow{\mathtt{A}}\,\overleftarrow{\mathtt{C}}\,\overrightarrow{\mathtt{B}}\,\overrightarrow{\mathtt{B}}$$

the first two are well-formed, and the last three are not. We stress the fact that the empty word λ is well-formed.

We define two operations on well-formed words. If $w = xaa'z$, with a, a' being two occurrences of the same letter with arbitrary orientations, then the *a-erase* transforms w into xz. If $w = xaya'z$, with a, a' being two occurrences of the same letter with *opposite* orientation, then the *a-flip* transforms w into $u = xy^R z$, where y^R is obtained from y by reversing the order of the letters in y as well as their orientations. Both operations preserve well-formedness.

The example below shows two sequences of operations applied to the same initial word:

$$\overrightarrow{\mathtt{A}}\overrightarrow{\mathtt{B}}\overleftarrow{\mathtt{A}}\overrightarrow{\mathtt{C}}\overleftarrow{\mathtt{B}}\overrightarrow{\mathtt{C}}\overrightarrow{\mathtt{D}}\overrightarrow{\mathtt{D}} \xrightarrow{\mathtt{B}\text{-}\mathit{flip}} \overrightarrow{\mathtt{A}}\overleftarrow{\mathtt{C}}\overrightarrow{\mathtt{A}}\overrightarrow{\mathtt{C}}\overrightarrow{\mathtt{D}}\overrightarrow{\mathtt{D}} \xrightarrow{\mathtt{C}\text{-}\mathit{flip}} \overrightarrow{\mathtt{A}}\overleftarrow{\mathtt{A}}\overrightarrow{\mathtt{D}}\overrightarrow{\mathtt{D}} \xrightarrow{\mathtt{A}\text{-}\mathit{erase}} \overrightarrow{\mathtt{D}}\overrightarrow{\mathtt{D}} \xrightarrow{\mathtt{D}\text{-}\mathit{erase}} \lambda$$

$$\overrightarrow{\mathtt{A}}\overrightarrow{\mathtt{B}}\overleftarrow{\mathtt{A}}\overrightarrow{\mathtt{C}}\overleftarrow{\mathtt{B}}\overrightarrow{\mathtt{C}}\overrightarrow{\mathtt{D}}\overrightarrow{\mathtt{D}} \xrightarrow{\mathtt{A}\text{-}\mathit{flip}} \overleftarrow{\mathtt{B}}\overrightarrow{\mathtt{C}}\overleftarrow{\mathtt{B}}\overrightarrow{\mathtt{C}}\overrightarrow{\mathtt{D}}\overrightarrow{\mathtt{D}} \xrightarrow{\mathtt{D}\text{-}\mathit{erase}} \overleftarrow{\mathtt{B}}\overrightarrow{\mathtt{C}}\overleftarrow{\mathtt{B}}\overrightarrow{\mathtt{C}}$$

*`eric@cs.washington.edu`. Dept. of Computer Science, University of Washington, Seattle, WA 98195, USA. Supported by an IBM Research Fellowship.

†`marek@cs.ucr.edu`. Dept. of Computer Science, University of California, Riverside, CA 92521, USA. Research supported by NSF grant CCR-9988360, and an NRC COBASE grant (NSF contract INT-9522667).

‡`noga@opt.math.tu-graz.ac.at`. TU Graz, Institut für Mathematik B, Steyrergasse 30, A-8010 Graz, Austria. Supported by the START program Y43-MAT of the Austrian Ministry of Science.

§`sgall@math.cas.cz`. Mathematical Institute, AS CR, Žitná 25, CZ-11567 Praha 1, Czech Republic. Partially supported by grant A1019901 of GA AV ČR, postdoctoral grant 201/97/P038 of GA ČR, cooperative research grant INT-9600919/ME-103 from the NSF and MŠMT CR, and project LN00A056 of MŠMT CR.

¶`gwoegi@opt.math.tu-graz.ac.at`. TU Graz, Institut für Mathematik B, Steyrergasse 30, A-8010 Graz, Austria. Supported by the START program Y43-MAT of the Austrian Ministry of Science.

Rozenberg & Salomaa [5] considered the following problem: Given a well-formed word w, is there a sequence of flip and erase operations that transform w into the empty word λ? As the example above shows, some sequences of operations can reduce w to λ, while other could lead to a dead-end, in which no operation can be applied.

The above question arises in research on DNA computing in vivo, in particular in the study of gene assembly in ciliates. In ciliates, the genetic information is stored in a scrambled form in one of its two nuclei. During the replication process, the stored sequence of genes needs to be reassembled in the order determined by the locations of pairs of DNA markers. In our notation, these markers are represented by the letters in Σ. The task of reassembly is achieved by a sequence of transformations that resemble the erase and flip operations defined above. We refer the reader to [4] for general information on DNA computing, and to [2, 1, 3] for recent research on mathematical modeling of gene assembly in ciliates.

In this short note, we answer the question posed in [5], by giving a simple characterization of well-formed words w that are reducible to λ. This characterization is easier to formulate in terms of a certain labelled graph G_w called the *overlap graph of* w, as described by Rozenberg in [5].

The vertex set of G_w is the alphabet Σ. For any letter a that occurs in w, if its two occurrences in w have opposite orientations, then vertex a in G_w is labeled with $+$, otherwise it's labeled with $-$. (If a does not occur in w, the labeling is arbitrary.) The edges are determined as follows: For any letter a in w define the interval $I_a = [i, j]$, where $i < j$ are the two locations of a in w. Two intervals I_a, I_b are said to *overlap* if they intersect, but neither contains the other as a subset. Then vertices a and b are adjacent in G_w if and only if I_a and I_b overlap. As a consequence, if a letter a does not occur in w then it is an isolated vertex in G_w.

Note that G_w does not uniquely characterize w, since different well-formed words may have the same overlap graph. Nevertheless, G_w captures enough structure of w to determine whether w is reducible to λ or not. We prove the following characterization.

Theorem 1 *A well-formed word w is reducible to λ if and only if each non-singleton connected component of G_w has a positive node.*

The reduction of the word problem to the graph problem uses a transformation almost identical to those presented by Rozenberg in [5]. Thus our main contribution is solving the graph problem, namely proving Lemma 2.

2 Proof of Theorem 1

For the sake of completeness, we now explain the transformation from [5], showing how the reducibility problem on strings can be transformed into a problem about overlap graphs. We settle this problem for general graphs below in Lemma 2.

The first observation is that any sequence of operations can be replaced by another that consists of a sequence of flips followed by a sequence of erases. For suppose that we have an a-erase followed by a b-flip. Since flips do not move apart adjacent letters, we can exchange the order of these two operations without affecting the resulting word. By repeating these exchanges, we can transform any sequence of operations into one that has the desired form.

Next, note that w can be reduced to λ by a sequence of erases if and only if G_w has no edges. For if (a, b) is an edge, then I_a and I_b overlap, so we can never apply either the a-erase or the b-erase. If G_w does not have an edge, pick a for which I_a is shortest. The two occurrences of a in w must be consecutive, so we can execute the a-erase and proceed recursively.

Finally, we want to determine how flip operations affect G_w. Clearly, after the a-flip vertex a is isolated in G_w. The other changes are as follows:

- For $b \neq a$, if I_b overlaps I_a then the a-flip will change the orientation of the occurrence of b inside I_a; thus the sign of b in G_w will get reversed. If I_b is disjoint from I_a, wholly contains I_a, or is wholly contained in I_a, then the a-flip either changes the orientations of both occurrences of b or none, so it does not change the sign of vertex b.

- Let $b, c \neq a$. Suppose first that one of I_b, I_c, say I_c does not overlap I_a. I_b overlaps I_c if and only if I_c has exactly one occurrence of b inside, and the a-flip does not change the contents of I_c, only the ordering and orientations. Therefore, if one of b, c is not adjacent to a, then the a-flip does not affect the adjacency of b and c.

  Suppose now that both I_b and I_c overlap I_a, but do not overlap each other. I_a contains one occurrence of each of b and c. We have two cases: one when I_b and I_c are disjoint, and the other when one of them contains the other. In each case, the order of the occurrences of b and c inside I_a changes and, after the a-flip, I_b and I_c will overlap. Since the reversal operation is its own inverse, we obtain that if both b, c are adjacent to a, then the a-flip reverses the adjacency of b, c.

For a vertex v of G_w, by $N(v)$ we denote the set containing all the neighbors of v in G and the vertex v itself. Summarizing what we said above, the a-flip has the following effect on G_w: (a) The signs of the vertices in $N(a)$ are switched from $+$ to $-$ and vice versa, and (b) The edges and non-edges inside $N(a)$ are switched, that is, every edge becomes a non-edge and vice versa (in particular, vertex a becomes isolated and remains isolated for the rest of the game, since flips do not affect signs or edges in any other component). There are no other changes in G_w. For simplicity, we will refer to the graph operation consisting of (a) and (b) above as the *a-flip*, same as the operation on words.

The three observations above imply the following fact from [5]: w is reducible to λ if and only if there is a sequence of vertex flips that transforms G_w into a graph without edges.

Throughout the rest of the proof, we let G be an arbitrary graph whose vertices are labelled with symbols $+$ and $-$. To complete the proof of Theorem 1, it is sufficient to prove the lemma below.

Lemma 2 *There exists a sequence of vertex flips that eliminates all edges from G if and only if each non-singleton connected component of G has a positive vertex.*

Proof. Implication ($\Rightarrow$) is easy, since if G has a connected component C with at least two vertices, and all vertices in C are negative, then flips applied to other components do not affect the signs in C and we cannot execute any flips for vertices in C.

We now prove ($\Leftarrow$). It is sufficient to show that if G has at least one edge, we can select a vertex v to be flipped, so that the graph after the flip satisfies the condition from the lemma. Since the v-flip isolates v, by repeating this process, we can eliminate all edges from G.

By $N^+(x)$ and $N^-(x)$ we denote the elements of $N(x)$ labeled by $+$ and $-$, respectively. Let $B \subseteq V$ be the set of those positive vertices x for which $|N^-(x)|$ is maximized. We choose v to be the vertex in B for which $|N^+(v)|$ is minimized.

Components of G that do not contain v are unchanged by the v-flip. The component containing v can become disconnected after the flip; however, since only the edges connecting vertices in $N(v)$ are changed, every resulting sub-component will contain some element from $N(v)$. Thus it is sufficient to prove that, after the flip, for each vertex $u \in N(v) - \{v\}$, either u will be positive, or will be isolated, or will have a positive neighbor.

If $u \in N^-(v)$, then u is positive after the flip. If $N^-(v) - N^-(u) \neq \emptyset$ then any $s \in N^-(v) - N^-(u)$ will become a positive neighbor of u after the flip. If $N^+(u) - N^+(v) \neq \emptyset$ then any $s \in N^+(u) - N^+(v)$ will remain a positive neighbor of u after the flip.

The last case is when $u \in N^+(v)$, $N^-(v) \subseteq N^-(u)$ and $N^+(u) \subseteq N^+(v)$. For such u, by the definition of B and the fact that $v \in B$, we have $N^-(u) = N^-(v)$. Hence $u \in B$ as well. By the choice of v within B, we also get that $N^+(u) = N^+(v)$. We conclude that $N(v) = N(u)$ and, since v is isolated after the flip, so is u. ■

3 Final Comments

Note that many graphs are not interval overlap graphs. Thus the problem we solve in Lemma 2 is more general than the original question from [5], and we believe it is of its own interest.

Our proof provides a simple and efficient algorithm that, given any well-formed word w, determines whether w is reducible to λ, and if it is, it produces the sequence of flip and erase operations that transform w into λ. We first build the overlap graph G_w. If G_w has a non-singleton connected component whose all vertices are negative, the algorithm reports that w cannot be reduced to λ. Otherwise, we determine the sequence of flips as explained in the proof of Lemma 2, and after transforming G_w into a graph without edges, we apply erase operations to convert the resulting word to λ. The most time-consuming part – determining the flip sequence – can be easily implemented in time $O(n^3)$.

We know of no efficient method for solving a related problem: given a well-formed word w of length $2n$, do all sequence of n flip and erase operations reduce w to λ? In other words, is it possible to reach a dead-end starting from w?

References

[1] A. Ehrenfeucht, I. Petre, D.M. Prescott, 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, 2000, to appear.

[2] A. Ehrenfeucht, D.M. Prescott and G. Rozenberg. Computational aspects of gene scrambling in ciliates. In: *Evolution as Computation*, L. Landweber and E. Winfree, eds. Springer-Verlag, 2000, to appear.

[3] L. KARI, J. KARI, L. LANDWEBER. Reversible molecular computation in ciliates. In: *Jewels are Forever*, J. Karhumaki, H. Maurer, G. Paun, G. Rozenberg, eds. Springer-Verlag, 1999, pp.353-363.

[4] G. PAUN, G. ROZENBERG AND A. SALOMAA. *DNA Computing: New Computing Paradigms.* Springer Verlag, Berlin, Heidelberg, New York, September 1998.

[5] G. ROZENBERG AND A. SALOMAA. DNA Computing: New ideas and paradigms. Invited talk at the *26th International Colloquium on Automata, Languages and Programming (ICALP'99)*, Prague, Czech Republic, July 1999.